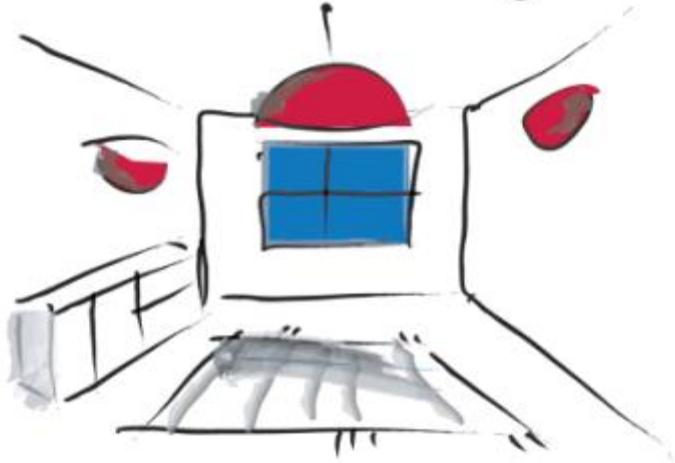


User Manual

Maestro Designer



V1.61 6/2018

Contents

1. [License](#)
2. [Installing Maestro Designer Software](#)
3. [Main Menu](#)
4. [Windows](#)
 - 4.1. [Windows and docking system](#)
 - 4.2. [Projects window](#)
 - 4.3. [Graphic Editor](#)
 - 4.4. [Linkers and Devices](#)
 - 4.4.1. [Device editor](#)
 - 4.4.2. [Group \(Address\) view](#)
 - 4.4.3. [Topology View](#)
 - 4.4.4. [Variables View](#)
 - 4.4.5. [External devices](#)
 - 4.4.6. [IR Remotes](#)
 - 4.4.7. [Serial Comm](#)
 - 4.4.8. [Modbus](#)
 - 4.4.9. [MQTT](#)
5. [Tool bar](#)
6. [Automatic data conversion](#)
7. [Persistent storage](#)
8. [Device Set up page.](#)
9. [Buttons](#)
10. [Function Blocks controller](#)
 - 10.1. [Accum \(Accumulator\) Function-Block](#)
 - 10.2. [Add Function-Block](#)
 - 10.3. [Amp Function-Block](#)
 - 10.4. [And, AndN Function-Blocks](#)
 - 10.5. [Clock Function Block](#)
 - 10.6. [Comp \(Comparator\) Function-Block](#)
 - 10.7. [Const1, Const2, ConstN Function-Blocks](#)
 - 10.8. [Counter Function-Block](#)
 - 10.9. [Div \(Divide\) Function-Block](#)
 - 10.10. [Event2, EventN Function-Blocks](#)
 - 10.11. [Expression function block](#)
 - 10.12. [Inv \(Inverter\) Function-Block](#)
 - 10.13. [Macro Function Block](#)
 - 10.14. [Mail function block](#)
 - 10.15. [Mul \(Multiply\) Function-Block](#)
 - 10.16. [NAnd, NAndN Function-Blocks](#)
 - 10.17. [Neg \(Negative\) Function-Block](#)
 - 10.18. [Not Function-Block](#)
 - 10.19. [NOr, NOrN Function-Blocks](#)
 - 10.20. [OpAmp Function-Block](#)
 - 10.21. [Or, OrN Function-BlocksPoller Function-Block](#)
 - 10.22. [Poller Function-Block](#)
 - 10.23. [Pos \(Positive\) Function-Block](#)
 - 10.24. [PWM Function-Block](#)

- 10.25. [Random Function-Block](#)
 - 10.26. [Range Function-Block](#)
 - 10.27. [Rout Function-Block](#)
 - 10.28. [Save1, Save2, SaveN Function-Blocks](#)
 - 10.29. [Sel2, SelN \(Select\) Function-Blocks](#)
 - 10.30. [SIP function block](#)
 - 10.31. [Sub \(Subtract\) Function-Block](#)
 - 10.32. [Switch2, SwitchN Function-Blocks](#)
 - 10.33. [Timer Function-Block](#)
 - 10.34. [XOr, XOrN Function-Blocks](#)
-
- 11. [Web control](#)
 - 12. [logger](#)
 - 13. [Maestro script language](#)
 - 14. [List of IP ports](#)
 - 15. [List of http web pages](#)
 - 16. [Remote Shell and using Maestro as simple ASCII-KNX gateway.](#)
 - 17. [LAN web page](#)
 - 18. [Maestro KNX – App for iOS](#)
 - 19. [Maestro Graphs and data logger](#)
 - 20. [MQTT](#)
 - 21. [DMX](#)
 - 22. [Amazon echo \(Alexa\)](#)
 - 23. [Frequently asked questions](#)

Glossary:

- **Block:**
Any **Object** that can be placed on a **Controller Page** . There are three types of **Blocks** :
 - o **Function Block** – This **Block** type has an internal algorithm, and has no direct representation on the **Graphic Pages**. (e.g. *And, Or, Counter Function Blocks*).
 - o **Button Block** - For every **Button**, placed on a **Graphic Page** (e.g. “Table light *Toggle* button”),



Table light Toggle Button

the system automatically produces a corresponding **Button Block** and places it on the *Buttons* tab on the *Controller Palette* window.



Use the Buttons tab to select Button Blocks and place them on the controller sheets

Button Blocks enable you to link the **Inputs** and **Outputs** of **Graphic Objects** to the controller's **Function Blocks**. They also allow you to link the inputs and outputs of **Graphic Objects** to other **Graphic Objects**.

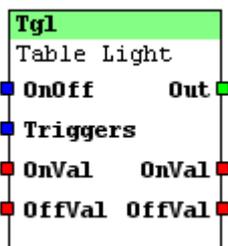
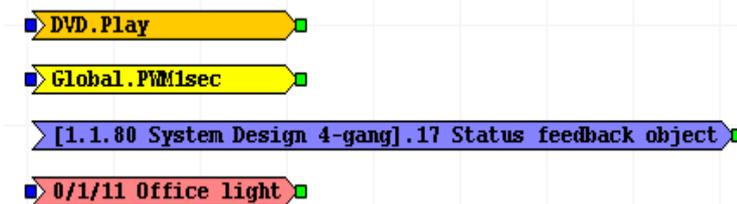


Table light button block

A **Button block** can be placed in only one location for the whole project

- o **Linker Block** – This **Block** type represents Linker in the **Controller Page**.

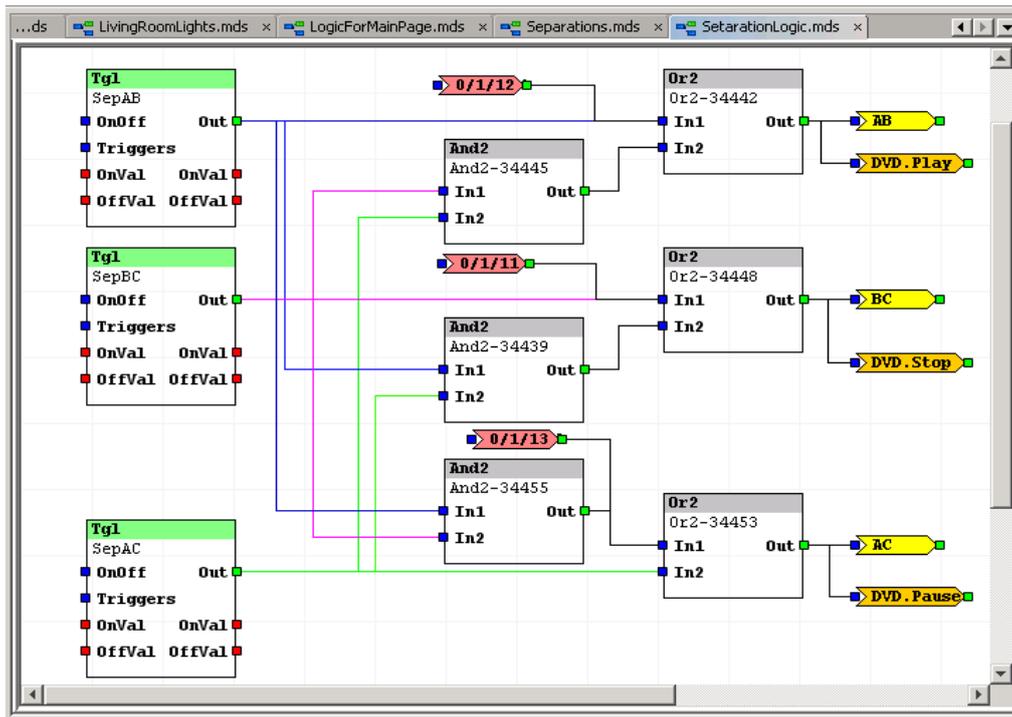


Linker Block is generated by drag and drop of a **Linker** to a **Controller Page**.

- **Controller Pages:**

Controller Pages (*/Controller Sheets*) are the graphic screens that display the controller's block diagrams.

Controller Pages are designed by the **System integrator**, enabling him to implement complex logics, math, and control functions. Block diagrams **are not** part of the graphics presented to the end user on **The Device** screen.



Controller Page

- **Button:**

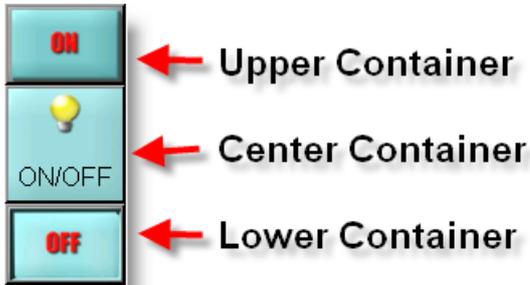


Dimmer bar + inc/dec button

A **BUTTON** is a dynamic, graphic component that operates as a user interface.

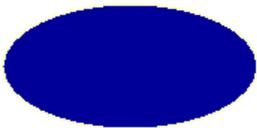
- **Buttons** can be pushed by the **User**. They are the means for the **User** to launch His instruction to **The Device**.
- **Buttons** have more then one graphic state, so they can be used to provide Feedback to the **User**.
- **Buttons** have **Inputs** and **Outputs** that can be linked directly to Group Addresses.
- Every **button** has a corresponding **Button Block**. You can use the **Button block** on the **Controller Page**, of the function block controller. In this way you can link the **button** to other **buttons** and **function blocks**.

- **Container:**
An area of a **Button** that has uniform graphical characteristics. **Buttons** can have one or more **Containers**, for example the following *On/Off Button* includes three **Containers**: Upper, Central and Lower.



On/Off button - composed of 3 containers

- **Event:**
Fresh data arriving to an **Input** or Fresh Data outgoing from an **Output**.
☺ An EIB telegram transmission is an **Event**
- **Graphic Object :**
Any **Object** that can be placed in a **Graphic Page**.



Graphic object: Oval



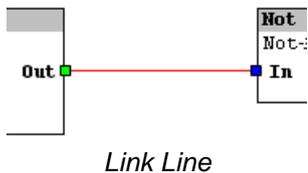
Graphic Object: Dimmer Bar

- **Graphic Pages:**
The graphic screens that are displayed on **The Device**. **Graphic pages** are designed by the **System integrator** and are the interface between **The Device** and the **End User**.

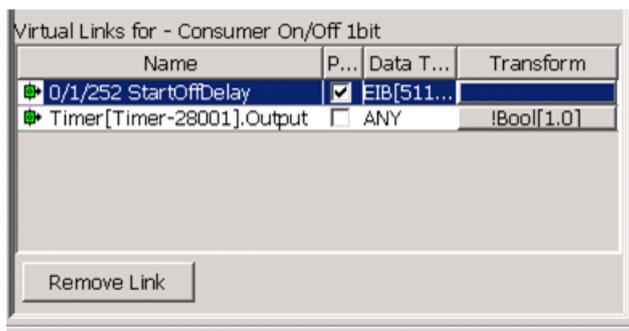


Graphic page

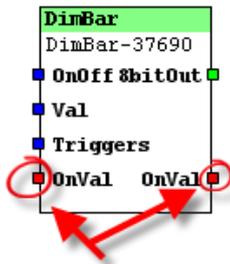
- **Input :**
A **Point** in an **Object** that receives data.
- **System integrator:**
The person who uses the *Maestro Designer* software and prepares the Maestro application.
- *Italic* – Terms written in *italic* font are terms used by the Maestro Editor software. The terms are part of the different user interface texts, such as menu commands, windows and tabs names, and properties...
- **Link:**
A connection, made by the **System integrator**, that defines a data rout between: two **Points** or from **Point** to Group Address. Links between points of **Blocks** are displayed as lines:



All **links** made to a **Point** are presented with details, on the Links window, when the **Point** is selected



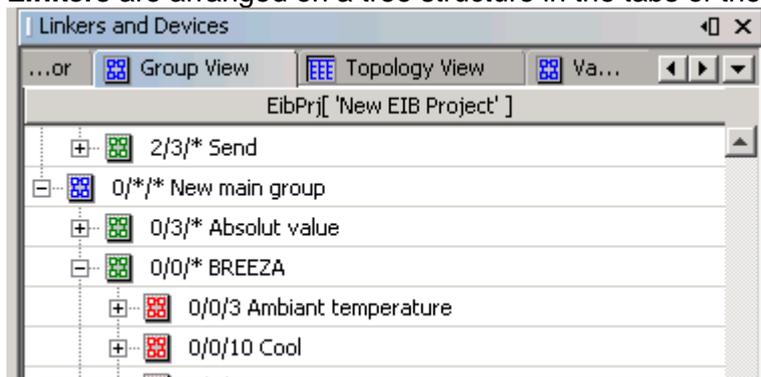
- **Linkable Parameters:**
The values of these parameters are used by the **Object's** algorithm. A **Linkable Parameter** has an **Input Point** and an **Output Point**. Linking the **Input Point** of the **Linkable Parameter** enables the control of its value. Linking the **Output** point enables the monitoring of its value. When used in the controller Linkable Parameters link points are marked in red color.



Linkble parameter

- **Linker:**
The following **objects** are **linkers**: Serial and IR commands, Variables, KNX Objects and Group Addresses:

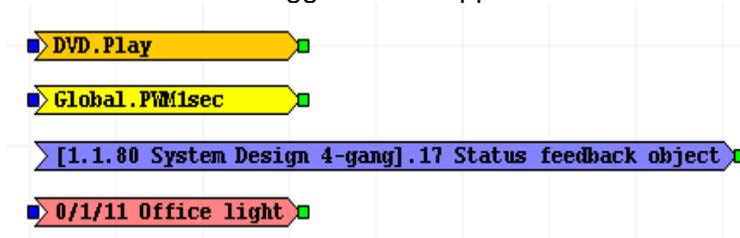
Linkers are arranged on a tree structure in the tabs of the *Linkers and Devices* window:



KNX Group Addresses are linkers

Linkers can be dragged and dropped in to **Controller Pages**, to the Links window, and to the *Linkers* tab in Macro and schedule editor.

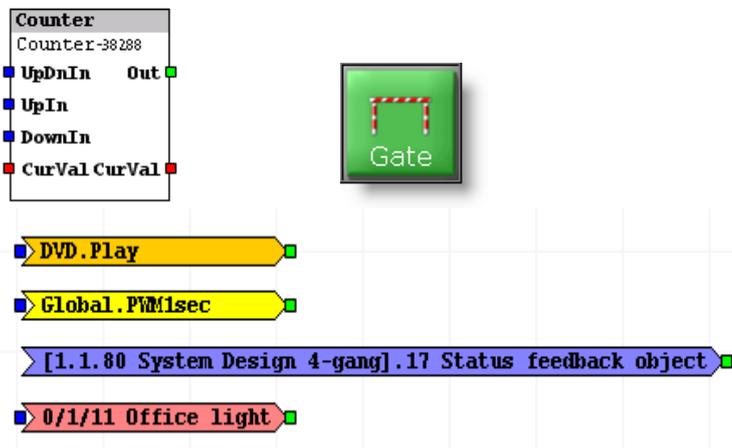
When a **Linker** is dragged and dropped in to **Controller Page** it creates a **linker block**



Different types of Linker Blocks

- **Object:**

Any element that can be placed on a **Graphic Page** or **Controller Page** .



Counter function block, Push/Release button and linkers are all **Objects**

- **Output:**

A **Point** in an **Object** that generates **Events**.

- **Point:**

An element of an **Object** that can be used to receive (**Input Point**) or transmit (**Output Point**) data.

- **Specialized buttons:**

Buttons made to control specific loads. Group Addresses linked to **Specialized Buttons** must have the Same Data Type as the button **Input/Output**.

There are 3 types of **Specialized buttons**: *Dimmer*, *Shutter* and *On/Off buttons*.

Specialized buttons are designed to increase the efficiency of the **System integrator**'s work, and to prevent association errors.

All other **Inputs** and **Outputs** of all other **Button** Types can be linked to Group Addresses of any Data Type).

- **The Device:**

The hardware device running the Maestro software application, e.g. "Maestro Touch Server 10.1", "Maestro server".

- **User/End User:**

A person who uses the Maestro device.

- **Unlinkable Parameters:**

Are used as parameters by the **Object** algorithm. Their values are fixed and are set (only) by the Maestro Editor software. They are used by the **Object** internally and cannot be changed or linked.

- **Variables** – internal variables

☺ : This symbol indicates Tips and useful information.

Notes/License

1.1. General remarks:

Maestro Designer User Manual by CD Innovation Ltd.

This manual is copyright protected. All rights reserved. No part of this manual may be copied or reproduced in any form.

The Manual and any examples and remarks contained herein are provided "as is". CD Innovation Ltd. shall not be liable for errors and omissions made in this manual.

CDI Innovation Ltd makes no warranty of any kind with regards to this manual.

The information in this manual may be subject to change without prior notice.

All brand or product names are trademarks or registered trademarks of their respective companies.

1.2. Software License Agreement *Maestro Designer* Software:

IMPORTANT - READ CAREFULLY BEFORE INSTALLING THE SOFTWARE.

The following Software License Agreement ("License") for one licensed copy of the Maestro Touch Software (the "Software") applies to you. This is a legal agreement between you ("You") and CD Innovation LTD ("CDI"). Use of this Software is subject to the terms and conditions of this License.

You should read all of the terms of this License carefully. **BY INSTALLING AND/OR BY USING THE SOFTWARE YOU AGREE TO BE BOUND BY THE TERMS OF THIS LICENSE.** If you do not agree to be bound by all the terms of this License, CDI is unwilling to grant you any rights to use the Software, and you **ARE NOT PERMITTED TO install OR USE** the Software. If you do not agree to the terms of this License, promptly erase all Maestro Editor Files from your computer and any data storing device and return. Notwithstanding the foregoing, installing or otherwise using the software indicates your acceptance of the terms of this license.

1) License. The Software is licensed pursuant to the terms of this License and not sold. CDI grants to you a personal, nonexclusive license to Use the Software only on a computer that is directly or indirectly coupled to, and only for operation in conjunction with, one or more CDI Maestro devices or Maestro Editor training. "Use" means storing, loading, installing, executing or displaying the Software. You may not modify the Software or disable any licensing or control features of the Software. You are obtaining no rights in the Software except those given in this limited license.

2) Ownership. The Software, together with all intellectual property rights embodied therein, is owned by CDI and its copyright protected according to international laws. Your license confers no title or ownership in the Software. CDI may protect its rights in the Event of any violation of this License.

3) Copies. You may make one copy of the Software for backup or archival purposes, or when copying is an essential step in the authorized Use of the Software. You must reproduce all copyright notices in the original Software on all copies. You may not copy the Software onto any bulletin board or similar system, nor can you copy the user documentation provided with the Software except for your own authorized use.

4) Restrictions on Use. The Software contains copyrighted material, trade secrets, and other proprietary material of CDI. You are not permitted to, nor are you permitted to cause or permit others to: (i) modify, adapt, alter, translate, decompile, disassemble or otherwise reverse engineer the Software or reduce the Software to human-readable form by any means whatsoever; ii) remove any identification, copyright or other notices from the Software; (iii) create a derivative work of any part of the Software; or (iv) rent, lease, loan or distribute the Software in whole or in part. In addition to the restrictions set forth above, you are expressly not permitted to Use the Software for use in conjunction with the operation of nuclear facilities, aircraft navigation, aircraft communication, aircraft flight control, aircraft air traffic control systems, weapons devices or systems, or in any devices or systems in which a malfunction (including, without limitation, software related delay or failure) would result in foreseeable risk of injury or death to the operator of the device or system, or to others. If EC law is applicable, the restrictions in Section 4(i) are limited so that they prohibit such activity only to the maximum extent such activity may be prohibited without violating the EC Directive on the Legal Protection of Computer Programs.

5) NO Warranty: CDI is licensing the Software to you "as is." Neither CDI nor its suppliers warrant that the Software will meet your requirements or that the operation of the Software will be uninterrupted or that the Software will be error-free or virus-free. CDI MAKES NO WARRANTY, CONDITION OR OTHER CONTRACTUAL TERM WHATSOEVER WITH RESPECT TO THE SOFTWARE AND HEREBY DISCLAIMS, TO THE MAXIMUM PERMITTED BY LAW, ANY AND ALL OTHER WARRANTIES OR CONDITIONS, WHETHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTY OR CONDITION OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON INFRINGEMENT OF THIRD PARTY RIGHTS..



2. Installing *Maestro Designer* Software.

2.1. Before installing:

2.1.1. License:

BY INSTALLING AND/OR BY USING THE SOFTWARE YOU AGREE TO BE BOUND BY THE TERMS OF LICENSE FOR THE SOFTWARE AS DESCRIBED IN CHAPTER 1 POINT 2 ABOVE.

2.1.2. Hardware requirements:

Maestro Designer will run on windows10, windows7, windows2000, windows XP, and Vista.

2.1.3. About the Manuel: The *Maestro Designer* software is designed for ease of use. Nevertheless, some sophisticated functions are not self explanatory. We recommend reading the manual completely. After installing the software, we recommend opening Demo projects from CDI's web site and becoming familiar with them.

2.2. Installing the *Maestro Designer* software:

2.2.1. Install Sun Microsystems Java on you PC. Java is free and downloadable from Sun Microsystems web site.

2.2.2. Run *MaestroInstallerSetUp.exe* this will start the installation wizard.

2.2.3. Follow the wizard's instructions.



Important remarks:

Editing Maestro project:

The only way to create and edit a project/page is by using *Maestro Designer*.

It is strictly prohibited to manipulate any of the *Maestro Designer* project files directly from Windows!

Images:

- For images you can use .PNG or .JPG files in addition you can use 9patch images, in this cast the file name has to set as *file_name.9.png* .
- For image file name, use English letters only (spaces and other characters are not allowed).
- Every image you use should have its unique file name.

Graphic and controller pages:

- For page name use English letters only (spaces and other characters are not allowed).
- Once a page is created it is impossible to change its name.

Appearance:

Maestro Server is capable of presenting 24bit color. PCs are usually, capable of presenting more colors. Therefore, it is possible that the graphic screen on the Maestro device will appear slightly different from how it appears on the Maestro Designer on the PC.

3. Main Menu:

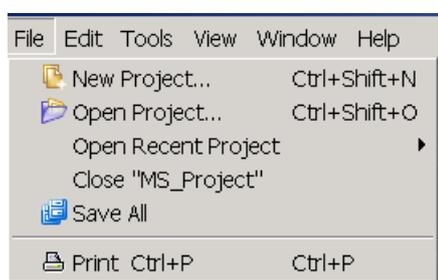
The main menu is located at the upper part of the software space.



Main Menu

3.1. File menu:

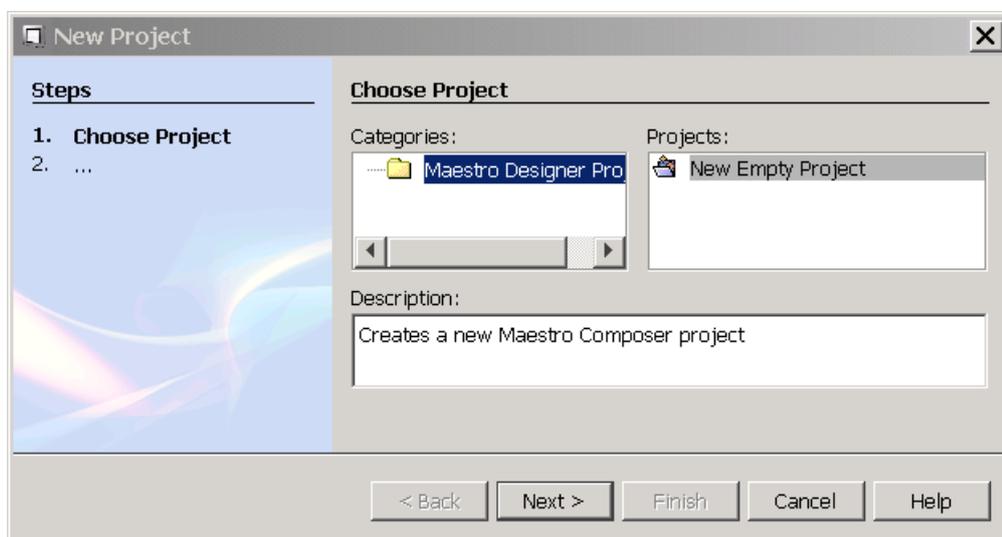
Here is the list of function available from *File* menu:



File menu

3.1.1. New Project:

Use this command to start the wizard that creates new projects.



Ne

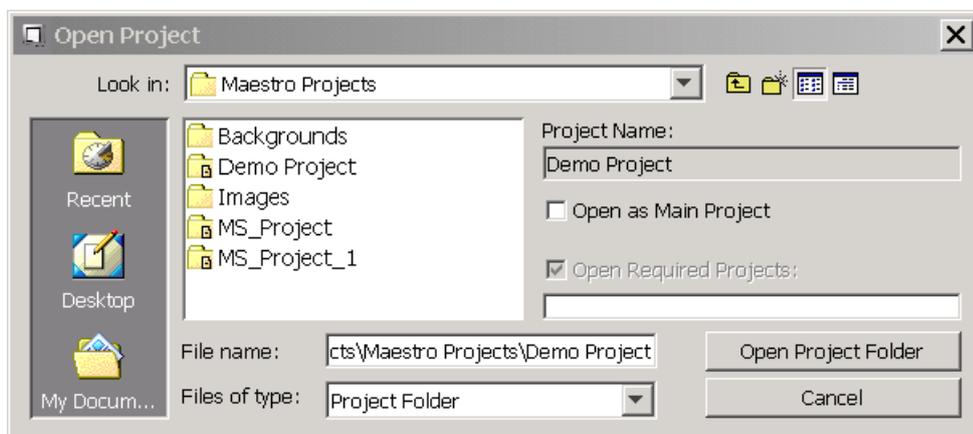
3.1.2. new Project wizard

You can use "<Back" and "Next>" buttons to go back and forth through the wizard or use "Cancel" to quit the operation at any time.

Follow the wizard's instructions. When you are finished press the "Finish >" button to create the new project.

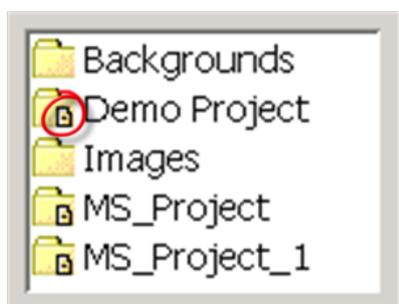
3.1.3. Open project.

Use this function to open an existing *Maestro Designer* project folder.



Open project explorer

Any time you open a directory, the Maestro software will scan it for valid *Maestro Designer* project folders. Once the analysis is completed the software will mark the *Maestro Designer* project folders:

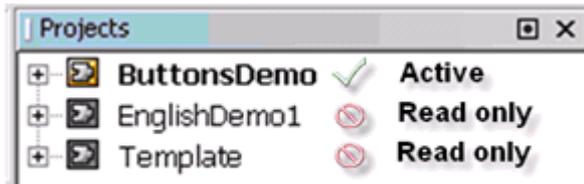


"DemoProject" folder - is marked as potential *Maestro Designer* project

"images" folder - is not a valid *Maestro Designer* folder and therefore it is not marked.

© Searching for valid Maestro project folders consumes time. The more projects you store in the same directory – the more time it will take the software to search for Maestro folders

The first project you open will be the active project. You will be able to edit it. The following projects you open will be "read only" projects. You can view their graphic pages, copy buttons from them, and paste the buttons to the active project but, on the read only project, you will not be able to edit them. You may open multiple "read only" projects simultaneously, but only the first project opened will be active



Only the first opened project is an active project

The active project icon in the Projects window is Orange: 

The "read only" projects icon in the Projects window is gray: 

Remark: When you copy and paste a button – its graphic properties are copied, but its functional properties are set to the default value.

3.1.4. *Open Recent Projects..*

Use this function to open recently edited projects.

3.1.5. *Close "[Project name]" Project..*

Use this function to close an opened project. When you close the active project all the "read only", projects will be closed as well. When you close a "read only" project, the other projects will not be affected.

3.1.6. *Save All:*

Use this function to save all changes made since the last time the project was saved.

3.1.7. *Print:*

When you select this function the software will print the graphical pages, followed by controller sheets, followed by a technical description dump of the complete project.

3.2. *Edit menu:*

Here is the list of functions available from *Edit* menu:



3.2.1. *Undo*:

Undo contains a list of the last 100 performed functions for every graphic page. Previous actions are not saved and can not be canceled by the undo button. Use this function to cancel the previously made action on the active page. Undo is only applicable for functions made on *graphical* tab properties. Other actions are not saved and cannot be canceled by the undo. Undo does not apply to Delete Object functions.

3.2.2. *Redo*:

If your most recent actions on the active page where *Undo* commands, *Redo* will cancel them one by one.

3.2.3. *Cut*:

Use this function to delete selected objects from their current position and copy them to the Maestro Editor clipboard.

3.2.4. *Copy*:

Use this function to copy the selected graphic **Objects** to the Maestro Editor clipboard.

3.2.5. *Paste*:

Use this function to paste graphic **Objects** from the Maestro Editor clipboard to the active page.

3.2.6. *Delete*:

Use this function to delete the selected graphic **Objects** from the Maestro active page.

3.2.7. *Select All*:

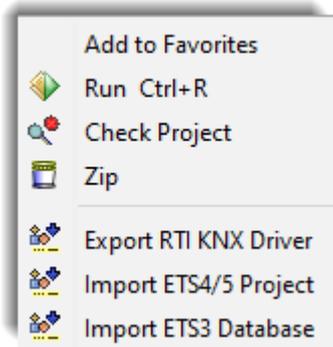
Use this function to select all objects of the active page.

- ☺ You can use Ctrl+Ins and Ctrl+C to copy alphanumeric values from properties fields of **objects** and to paste the values to other **objects** using Shift+Ins or Ctrl+V e.g. copy color properties of one **Button** and paste them to another **Button**:



3.3. *Tools* menu:

Here is the list of functions available from *Tools* menu:



3.3.1. *Run*:

Use this function to start the simulation software. This will open a pop-up window. The software will simulate **The Device's** graphic dynamics and functionality.

The *Run* function simulates all **Device's** functionality with the following exceptions:

- It does not send or receive EIB/KNX telegrams from the EIB bus.
- It is impossible to flip to SetUp page.
- It is impossible to access the macro editor.
- It is impossible to access the Scheduler.
- It uses PC native colors range (that is normally greater than Maestro touch screen screen).

As long as you do not close the Maestro Editor - persistent data is saved on your PC. Rerunning the simulation is similar to rebooting the device, and the persistent data of the last *Run* operation (e.g Last state of a buttons) will be reused.

3.3.2. *Check Project*:

Use this function to check your application for potential errors The result will be presented on the Output window.

3.3.3. *Zip*:

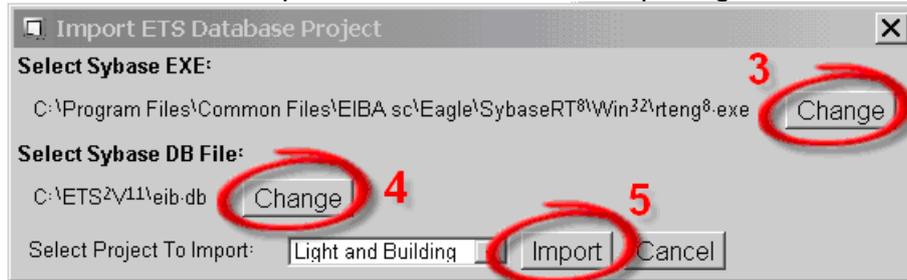
Zip command will prepare a downloadable Zip file of the project. The file will be stored in the "dist" directory of the project and its name will be [project name].Zip. The zip file includes all the directories and files of the project, excluding unused resources, for example image placed on the Images directory of the project but not used in any graphic page – will not be included in the Zip file.

3.3.4. Export RTI KNX Driver:

use this function to create and save a RTI to KNX driver, by importing this driver to RTI you will import all KNX group addresses of the project to RTI and you will be able to control and monitor KNX from RTI using Maestro's port 7000.

3.3.5. Import ETS3 Database:

Use this function to open the window used for importing ETS 2/3 database:



Import ETS database window

Follow the next 5 steps:

- 1) Make sure ETS is installed on your PC.
- 2) If ETS software is running – close it.
- 3) Next to "Select Sybase EXE" use the "Change" button to browse your PC and point to the location of the "rteng8.exe" file. This file is included with, and installed by, your ETS application.
- 4) Next to "Select Sybase DB file" use the "Change" button to browse your PC and point to the location of the "xxx.db" file holding the desired ETS project (e.g. eib.db)
- 5) On "Select project to import" use the drop down menu to select the ETS project you want to import data from and press "Import".

If you have made changes on your ETS DB file you can reimport the file – the new data will add to existing data and changes will overwrite the existing data. Exception to this it KNX data type that will not be overwritten.

During import process *Maestro Designer* tries to find the Data type of the imported Group address. If the data type is not defined in the ETS, *Maestro Designer* will look for the length of the Data type and mach a data type automatically to the Group Address according to the following defaults:

Data Length	Type	
1 BIT	1.****	1 BIT
2 BIT	2.****	1 BIT CONTROLLED
4 BIT	3.****	3 BIT CONTROLLED
8 BIT	5.****	1BYTE UNSIGNED VALUE
2 BYTE	9.****	2 BYTE FLOAT VALUE
3 BYTE	10.****	3 BYTE TIME OF DAY
4 BYTE	14.****	4 BYTE FLOAT VALUE
14 BYTE	16.****	14 BYTE STRING

when, during import, Maestro editor finds that it already has more accurate definition of data type – it will ignore the data type imported from ETS and keep its local definition.

3.4. Import ETS 4/5 project

use this function to import the data from ETS4/5 .knxpoj files.

3.5. View menu:

Here is the list of function available from View menu:

3.5.1. Toolbars sub-menu:

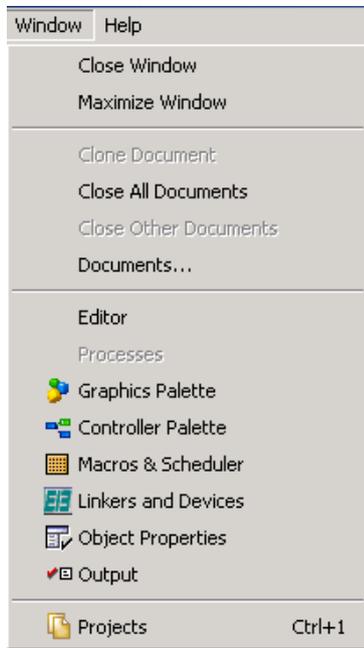


Toolbars sub-menu

Use this Sub-menu to:

- Show or hide shortcut button-sets from the toolbar.
- Select the size of the shortcut buttons (small or normal).

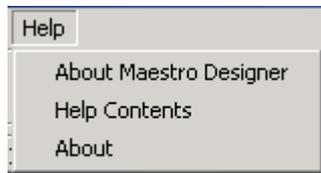
3.5.2. Window menu:



Window menu

Use this menu to open/close the software's windows.

3.6. *Help Menu:*



Help menu

3.6.1. *About Maestro Designer:*

Opens the About window where you can find information about the software version.



About Maestro Designer Suit versions

3.6.2. *Help Contents:*

Opens the Maestro Editors documentation window.

3.6.3. About:



About software environment Details

Opens the About window where you can find information about the Java software version and used directories.

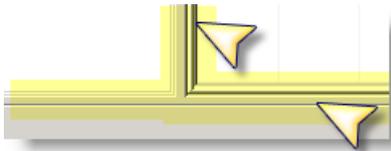
4. Windows

4.1. Maestro Editor windows and docking system:

The Maestro Editor working space is dynamic and you can arrange it in a way convenient to you. The working space is built in an hierarchal structure:

4.1.1. Area:

The working space is divided to Areas: Areas are separated from each other by a separation stripe:



Separation stripe

Using your mouse you can drag the separation line and resize the Area according to your needs.

It is possible to toggle between a normal view and a view where one Area is maximized and uses all the working space. To do so you have to double click the upper bar of a Window.



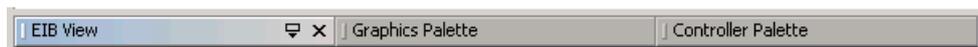
To maximize - double click the upper bar

An Area can contain one or more Windows. By dragging a Window you can move it from one Area to another or create a new Area. If you drag all the Windows out of an Area – this area will be canceled.

There is one exception – the Graphic Editor Area: this Area can not be canceled.

4.1.2. Window:

Every Area can have a few Windows.



Windows upper bar

Windows can be arranged in very flexible way. Drag the upper bar of a window to change its position. While you drag the Window, a silhouette will show you the suggested new location for the Window.

A Window can be in one of the following states:

4.1.2.1. Active Window:

When the window is in an active state, the upper bar is colored blue



Active window

the Window is visible, it is on top of the other Windows, and it is operational. At any given time the *Maestro Designer* software will have one active Window. The Window's upper bar shows the name of the Window on its left side and has functional buttons on its right.

Here is the description for the functionality of those buttons:



- Close the Window: When you use this function the Window will be closed. To reopen the Window you will have to click its name from the main menu.



- Minimize the Window: when you use this function the Window will be minimized and a button representing it will be placed on the perimeter of the working space.



- Restore the Window: when you use this function the Window will change its state from Pop Up to Active.

Windows can be arranged in a flexible way. Drag the upper bar of a Window to change its position.

4.1.2.2. Opened Window:

When the window is in an opened state, its upper bar is colored light gray,



Opened window

the Window is visible, it is in top of the other Windows sharing the same Area but it is not active. At any given time, one Window is Opened (OR Active) In every area.

4.1.2.3. Hidden Window:

Window is behind an opened/active Window.



Hidden window

Only its upper bar is visible and colored dark gray. Click on the Window's upper bar (or select its name from the *Window* menu) to make it the active Window.

4.1.2.4. Minimized Window:

The Window is minimized and represented by a button on the perimeter of the working space.



Minimized window button

4.1.2.5. Pop Up Window:

When the mouse moves over the button of the minimized Window - the Window will Pop Up on top of other components and the Window will become active. It will remain this way as long as the mouse is over the Popped Up window area.

When you press a minimized-window-button – the Window will temporarily pop up. It will be minimized automatically only when you use another Window. The Window will be minimized also if you press its minimized-window-button again.

4.1.2.6. Closed Window:

No part of the window is visible.

4.1.3. Tabs:

A window can contain Tabs.



The Controller Palette Window includes 4 tabs

A Tab can be in one of the following states:

4.1.3.1. Active Tab:

When a Tab is in the active state, the upper bar is colored blue

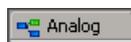


Active tab

the Tab is visible, it is on top of the other Tabs, and it is active. If a Window holds Tabs - only one of the Tabs will be in Active state at any given time.

4.1.3.2. Hidden Tab:

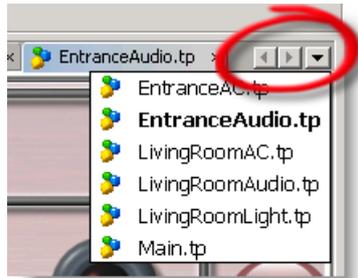
When a Tab is in a Hidden state, the upper bar is colored gray,



Hidden tab

and it is behind the active Tab. Only its upper bar is visible. Click on the Tab's upper bar to make it the Active Tab.

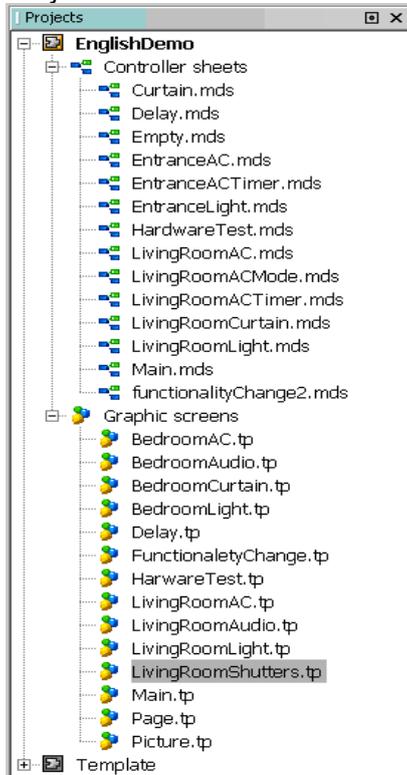
The arrow buttons on the upper right corner of the window allow you to navigate between the tabs of the Window:



Select tab from drop down menu

Switch to the neighboring Tabs by clicking on left/right arrow button or click on the down arrow and select the name of the desired Tab from a dropdown list.

4.2. Projects window:

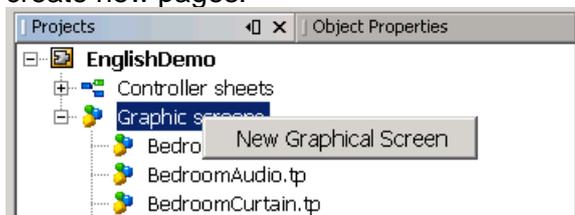


Projects window

A *Projects* window presents the list of all opened projects, their **Graphic Pages**, and their **Controller Pages** (*Controller Sheets*). The page names within the main project are arranged in a tree view having two branches:

- *Graphic pages* branch – Used for presenting the list of **Graphic Pages**.
- *Controller sheets* branch - Used for presenting the list of **Controller Pages**.

When you select the root of the branch, you can use the right mouse button menu to create new pages:



Use right mouse button menu to create new pages

When you select a page, the following right mouse button menu commands are available:

- *Open*: opens the selected page file and presents it as the active page on the graphic editor area.
- *Delete*: deletes the selected page.
- *Rename*: renames the page file.

4.3. Graphic Editor:

This window is the main window of the Maestro software. Its Area is static - you can't close or move it. The window holds project files as tabs and enables you to see and edit them. There are few types of files you can edit using the Graphic Editor. The Editor automatically, fits its functionality according to the edited file type:

4.3.1. Editing **Graphic Page**:

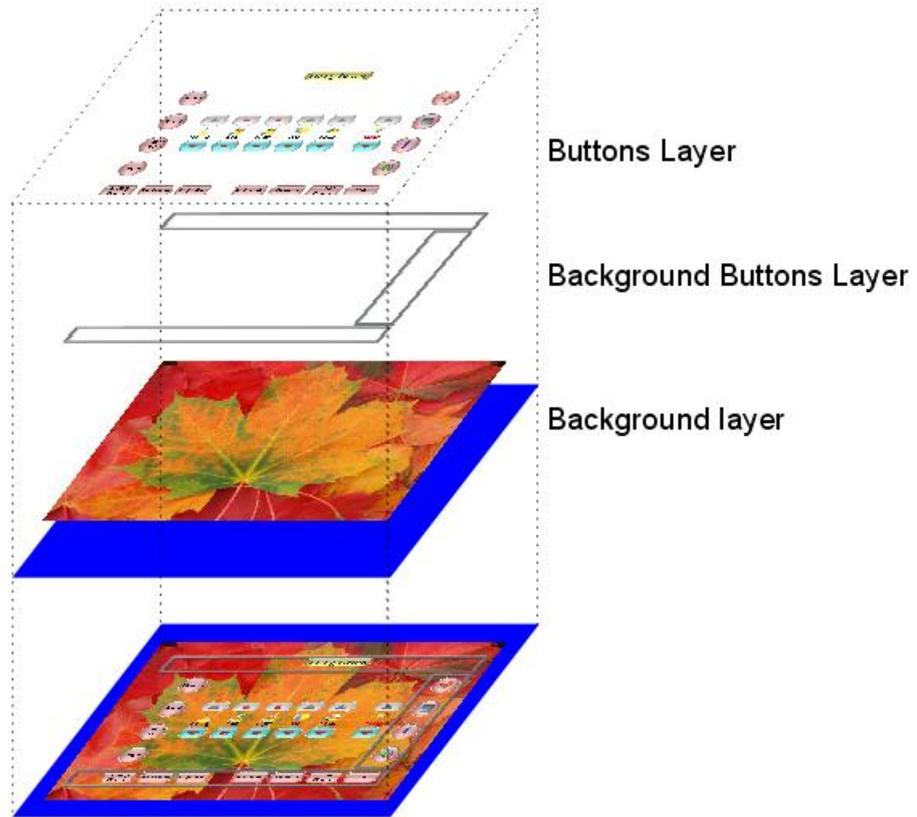


Graphic page

The Graphic Editor enables you to display and edit **Graphic Pages**. Here is a description of the Graphic Editor's functionality during **Graphic Page** design.

4.3.1.1. Layers:

The graphic **Objects** are arranged in 3 layers, one on top of the other.



Graphic Page Layers

It is impossible to move **Graphic Objects** from one layer to the other.

Here is the description of the layers:

- Background layer – This is the lowest layer. Use it to set the background color and to place a picture on top of the background color.
- Background Buttons layer – This is the second layer and placed on top of the background layer. Background buttons are placed in this layer. All background button types are placed on *graphic palette* window > *Background* Tab:



Typical uses of this layer are static elements like text headlines and simple shapes breaking the screen area into a few functional areas.



Background buttons

- Buttons layer – This is the upper layer, Buttons are placed on this layer.

4.3.1.2. Editing the Background layer:

Click in an area free of any other **Graphic Object** OR click the page on the gray area outside the **Graphic Page**. This way the *Object Properties* window > *Graphic* tab will switch to present the background properties allowing you to edit them.

4.3.1.3. Selecting **Graphic Object**:

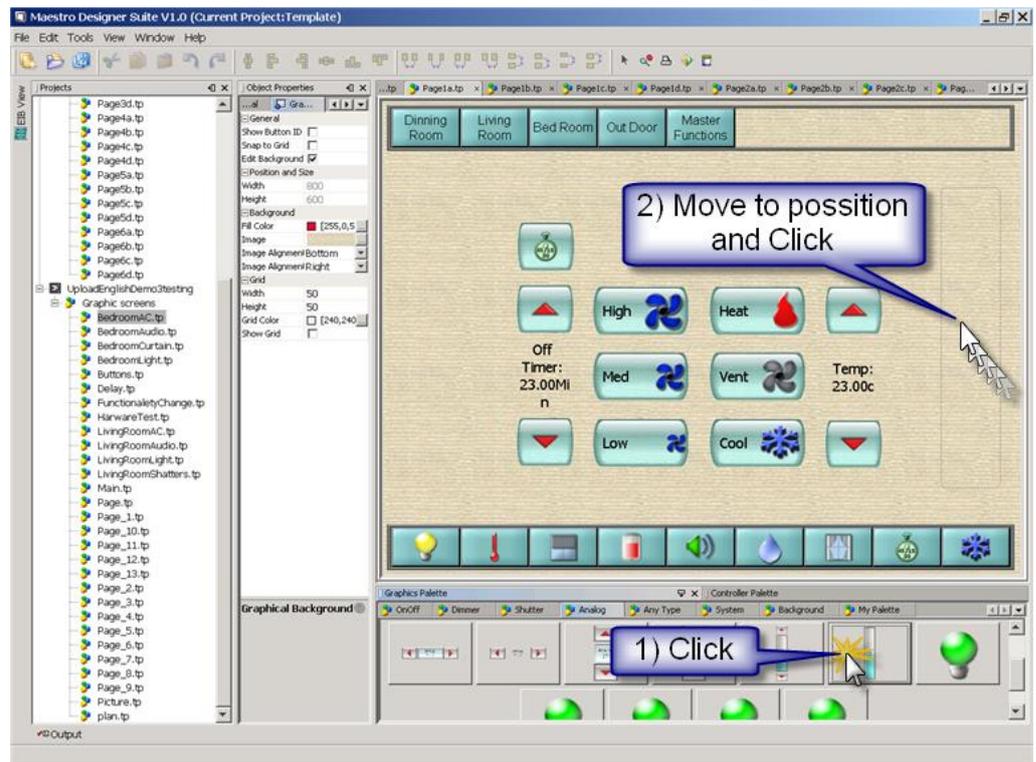
You can Select a **Graphic Object** by clicking on it, or by dragging your mouse over an area that surrounds the complete area of the **Object**. Once an **Object** is selected 8 yellow points will appear on its perimeter:



A selected button

4.3.1.4. Adding a **Graphic Object**:

Open the *Graphics Palette* window and select the relevant tab. Find the **Graphic Object** you want to add, and click on it. Move the mouse to the **Graphic Page**. Once the mouse is over the **Graphic Page**, you will notice that it is dragging the **Object's** silhouette, (a dashed rectangle at the size of the **Object**). Move the silhouette to the place you want to place the **Object**, and click to complete the procedure.



Adding new Graphic Object

4.3.1.5. Delete **Graphic Object**:

To delete a **Graphic Object**, first you have to select it and then:

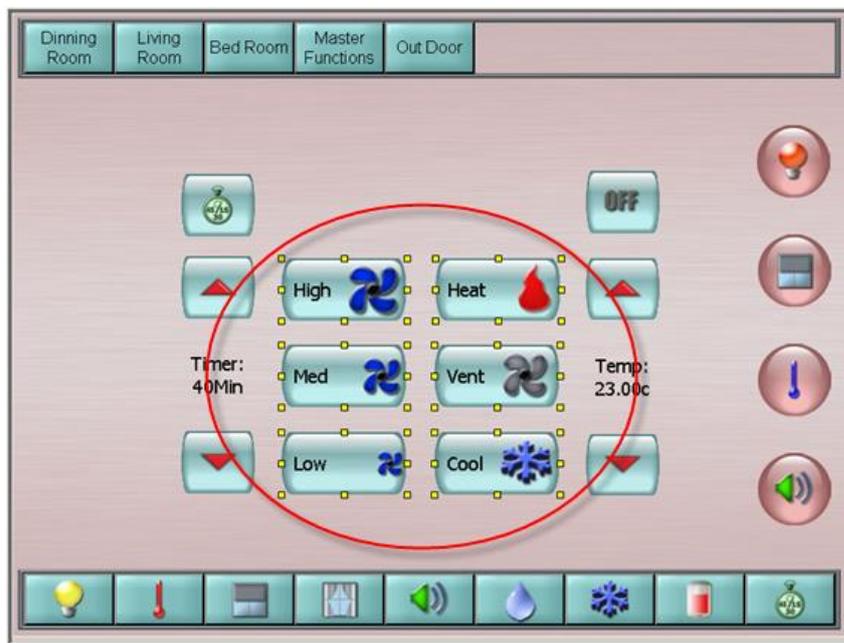
- Push the Delete key of your key board, Or
- Use *Edit* menu > *Delete* command, Or
- Right click the mouse to bring up the content drop down menu and then select the *Delete* command.

4.3.1.6. Multiple selection:

You can select a few **Objects** by Pressing and dragging the mouse over an area that surrounds them. Once you release the mouse every selected **Object** will have 8 yellow points surrounding it.

Now you can *Move, Cut, Delete, Copy, Store, Bring to front, Send to back, or Group* all selected **Objects**.

You can also edit their common Graphic properties using the *Graphical* tab. When a selected **Object** has few states (e.g. On state and Off state) - editing the properties will affect only the current state of the Objects.



Multiple selection

4.3.1.7. Adding and removing **Objects** from the selection:

Hold the Ctrl key and click on **Graphic Objects** you want to add or remove from the selection.

4.3.1.8. Resizing:

Select the **Object** and then drag the yellow points on the perimeter of the **Object** to resize it.

4.3.1.9. Moving a **Graphic Object/Objects**:

Select the **Object/s** and then drag the mouse OR Select the **Object** and hit the arrow keys on your keyboard. Every stroke will move the **Object** by one pixel.

4.3.1.10. Moving a **Graphic Object/Objects** in a straight horizontal or vertical line:

Select the **Object/s**. First press Ctrl+Alt and then click the mouse and drag the Object. The Object will move in a straight horizontal or vertical line depending on the position of the mouse.

4.3.1.11. *Copy:*

To copy an **Object**, Select it and then use:

Edit menu > *Copy* command, Or

Right click the mouse to bring up the content drop down menu and then select the *Copy* command.

4.3.1.12. *Paste:*

Once you have copied an **Object/Objects** to the software's clipboard you can paste it. To do so:

use *Edit* menu > *paste* command, Or

Right click the mouse to bring up the content drop down menu and then select the *paste* command.

4.3.1.13. Duplicate (Copy + Paste):

To duplicate an **Object**, click and hold the right mouse button and then drag it. Release the mouse where you want the new **Object** to be placed.

4.3.1.14. *Bring to front:*

To place a graphic **Object** on top of all other **Objects** in its layer - right click the mouse to bring up the content drop down menu, and then select the *Bring To Front* command.

4.3.1.15. *Send to Back:*

To place a graphic **Object** behind all other **Objects** in its layer - Right click the mouse to bring up the content drop down menu, and then select the *Send To Back* command.

4.3.1.16. *Align:*

Once you have made a multiple selection of **Objects** you can use the Tool bar shortcut buttons to align the selected buttons.



Align Tool bar buttons

4.3.1.17. *Distribute:*

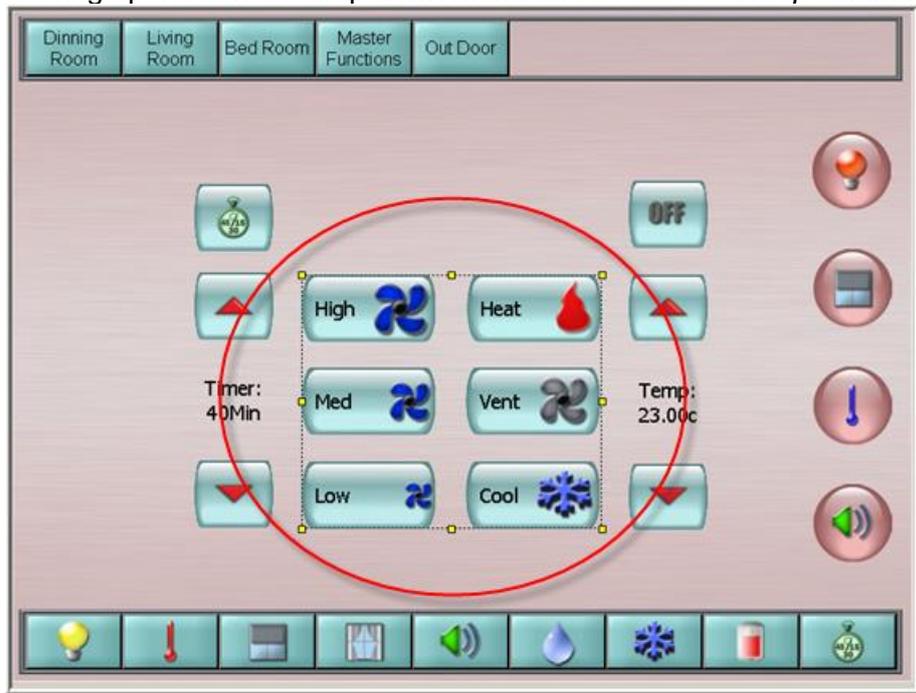
Once you have made a multiple selection of **Objects** you can use the Tool bar shortcut buttons to distribute the selected **Buttons** with even spacing.



Distribute Tool bar buttons

4.3.1.18. Group:

A Group is few **Objects** that behave as one unit. Operations you perform on a Group apply equally to each of its **Objects**. To Group a few graphic **Objects**, first make a multiple selection of a few **Objects**. Then right click the mouse to bring up the content drop down menu and select the *Group* command.



Grouped Objects

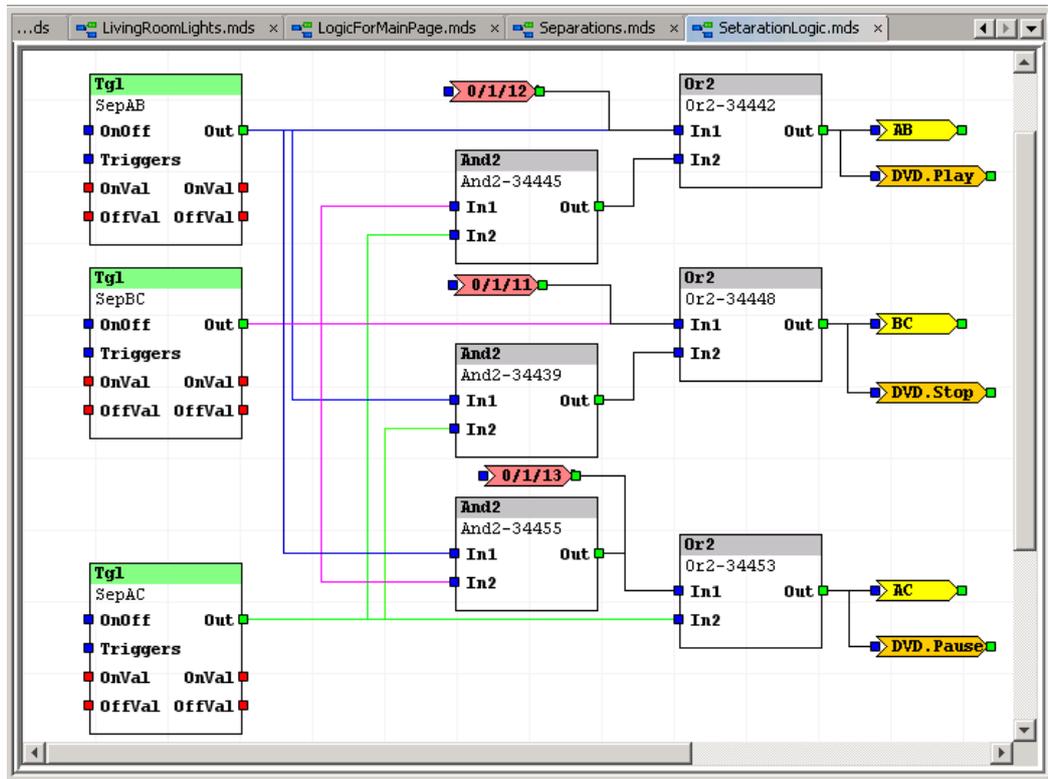
4.3.1.19. Ungrouping:

To ungroup the Group, select it and then right click the mouse to bring up the content drop down menu. Then select the *Ungroup* command.

☺ More manipulations of Graphic **Objects** can be performed by using functions from the *Object Properties* window > *graphic* tab. See description in the [Object Properties](#) chapter of this manual.

4.3.2. Editing **Controller Page**:

The Graphic editor enables you to view and design Block Diagram for the **Device's** internal virtual controller.



Block diagram

Operation of the **Controller Page** editor is similar to operation of the Graphic Editor with some exceptions:

4.3.2.1. Page size:

You can resize the **Controller Page** according to your needs. To set the page Width and Height use *Object Properties* window => *Graphical* Tab => *Page Width/Height*.

4.3.2.2. Layers:

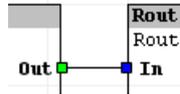
Controller Page has only one layer – *Function-Blocks* layer.

4.3.2.3. Objects:

The only **Objects** you can place on the **Controller Page** are items from the *Controller palette* window and **Linkers**

4.3.2.4. Link

Function-Blocks Objects have **Inputs** marked in Blue and **Outputs** marked in Green.

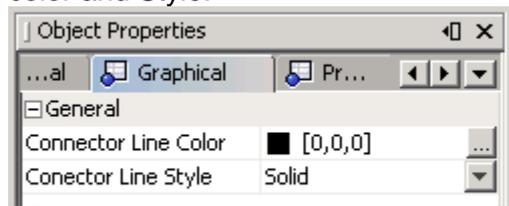


A link

To link **Output** of one **Object** to **Input** of another **Object**:

- 1) Hover with the mouse over the starting **Point**.
- 2) When the mouse symbol will change from to a cross, click and drag the mouse to the target **Point**. During this operation the software will draw a dashed line from the starting point to the position of the mouse.
- 3) When the mouse reaches the target, the mouse symbol will change to a cross. Release the mouse. Once a **Link** is made the software will draw a line between the linked **Points**.

It is possible to set the look of the link line by using its graphical properties: color and Style.



Link line graphical properties

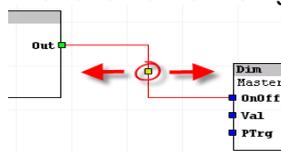
- ☉ It is possible to link a few inputs (including group addresses!) to one output.
- ☉ It is possible to link a few Outputs to one input – in this case the input will have the value of the last incoming data.
- ☉ you can use internal variables to pass values between different **Controller Pages**

4.3.2.5. Select a **Link**:

To select a **Link**, place the mouse over the line that connects two **Points** and then click. The selected **Link** line will change its color to red.

4.3.2.6. Change **Link** line route:

Select the **Link** Line, and then drag the yellow handle on the center of the **Link** line. By dragging the handle you can move the central leg of the **Link** line to left or to the right.



You can move the central leg of the **Link** line to left or to the right

4.3.2.7. Breaking a **Link**:

To break a **Link**:

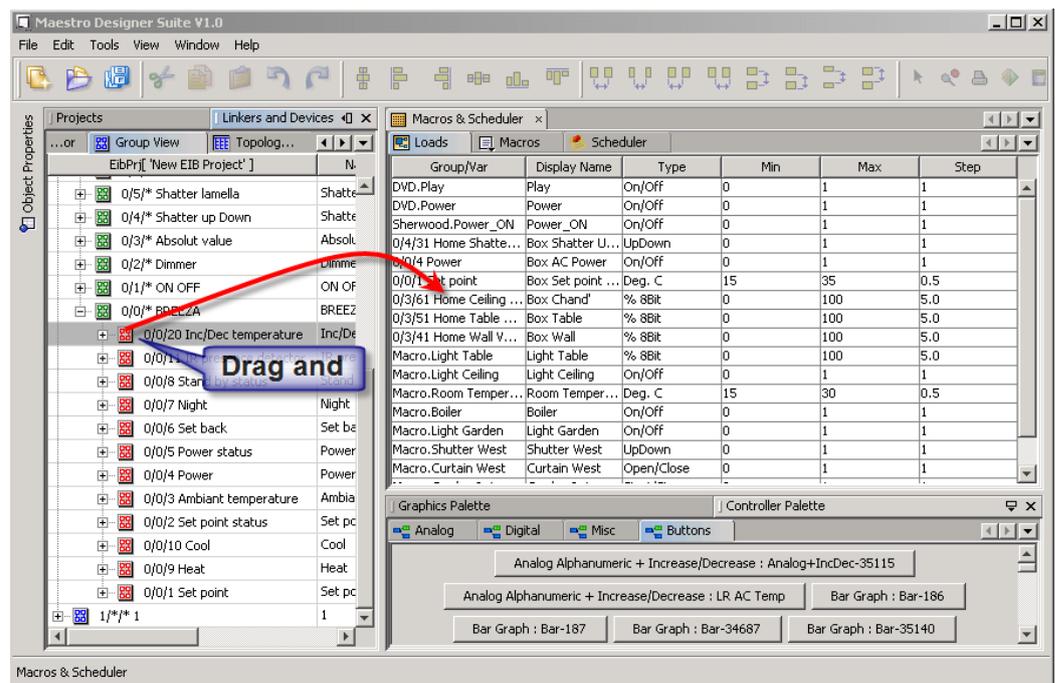
- Select the **Link** and press the Delete button on the keyboard, Or
- Select the **Link** and Use Edit menu => *Delete* command, Or
- Right click the mouse to bring up the content drop down menu, and then select the *Delete* command.

4.3.3. Macros and Scheduler:

Macros & Schedules window is presented on in the Graphic editor Area. To open the *Macros & Schedules* window, use the Main menu command *Window > Macros & Schedules*. The *Macros & Schedules* window includes 3 tabs:

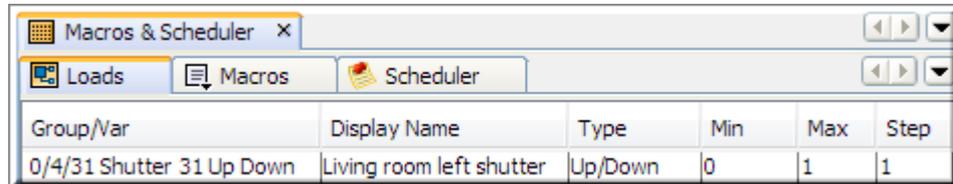
4.3.3.1. *Loads* tab:

This tab shows the table of **linkers** that can be used by the **End User** in Macros, Scheduler, Graphs, Alexa control.... To add **Linker** to the list, drag and drop the Linker from the *Linkers and Devices* window to the *Loads* tab. KNX objects cannot be used in *loads* table.



To add Linker to the list, drag and drop the Linker from the *Linkers and Devices* window to the *Loads* tab

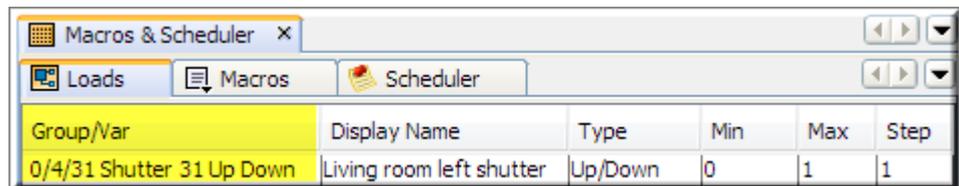
The Loads Table contains 6 columns:



Group/Var	Display Name	Type	Min	Max	Step
0/4/31 Shutter 31 Up Down	Living room left shutter	Up/Down	0	1	1

4.3.3.1.1. *Group/Var:*

Presents the **linker's** full name (in case of KNX Group Address it will present the Group Address and name). This is "read only" field that can not be edited.

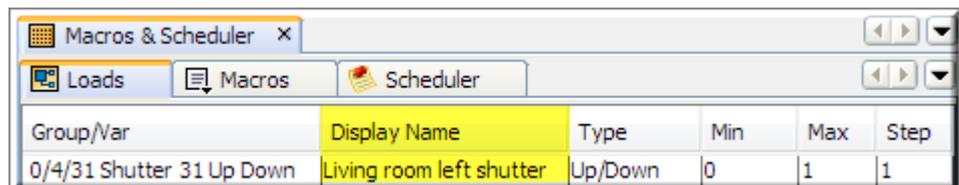


Group/Var	Display Name	Type	Min	Max	Step
0/4/31 Shutter 31 Up Down	Living room left shutter	Up/Down	0	1	1

"Group/Var (linkers)" column

4.3.3.1.2. *Display Name:*

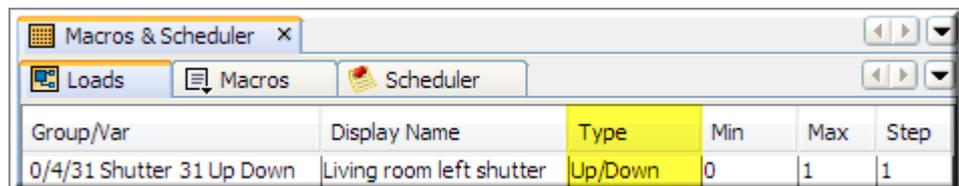
Presents a name used by the End user's interface to describe the **linker** for the **User**. By default the software will use the **Linker's** name but you can edit this field to assign better descriptions.



Group/Var	Display Name	Type	Min	Max	Step
0/4/31 Shutter 31 Up Down	Living room left shutter	Up/Down	0	1	1

"Display Name" column

4.3.3.1.3. *Type Column:*



Group/Var	Display Name	Type	Min	Max	Step
0/4/31 Shutter 31 Up Down	Living room left shutter	Up/Down	0	1	1

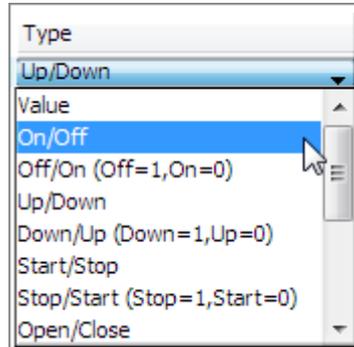
"Type" column

Use this field to choose the user interface type for this Linker. User interface type defines how values of this load will be presented on Macro and Scheduler editors. The purpose of this function is to enable a clear and friendly interface to the **User**.

Following is the table of available types and the way system converters user interface values to the value that is assigned to the Group Address:

User interface type	User interface format	Description	Linker value
Value	X.Y	General control	The Linker will get exactly the value entered by the user .
On/Off	"On" or "Off"	Control of On & Off loads	"On" = 1 , "Off" = 0
Off/On	"On" or "Off"	Control of On & Off loads	"Off" = 1 , "On" = 0
Up/Down	"Up" or "Down"	Control of Up & Down loads	"Up" = 1 , "Down" = 0
Down/Up	"Up" or "Down"	Control of Up & Down loads	"Down" = 1 , "Up" = 0
%	X.Y%	Control of % loads	"X%" = X (Input and Output range is the same and its default is 0 to 100)
%8bit	X.Y%	Control of 8 bit scaling % loads	"X%" = Xx255/100 (Input range is 0-100 linearly transformed to Output range 0-255)
Deg. C	"XC"	Control of degrees Celsius loads	"XC" = X
Start/Stop	"Start" or "Stop"	Start & Stop control	"Start" = 1 , "Stop" = 0
Stop/Start	"Start" or "Stop"	Start & Stop control	"Stop" = 1 , " Start " = 0
Open/Close	"Open" or "Close"	Control of On & Off loads	"Open" = 1 , "Close" = 0
Close/Open	"Open" or "Close"	Control of On & Off loads	"Close" = 1 , " Open " = 0
Enable/Disable	"Enable" or "Disable"	Enable & Disable control	"Enable" = 1 , " Disable " = 0
Disable\Enable	"Enable" or "Disable"	Enable & Disable control	"Disable" = 1 , " Enable " = 0
Trigger (Const)	No interface	Sends predefined (fixed) value	Fixed value
String	ASCII Keyboard	Sends, end user defined, string.	string
Color RGB	Color picker	RGB color setting	3 byte RGB
Color RGBW	Color picker	RGBW color setting	4 byte RGBW

To change the type –
Click on the type cell and choose the desired type from the drop-down menu.



"Type" drop down menu

4.3.3.1.4. *Min and Max:*

Min	Max
0	100

"Min" and "Max" columns

The Macro and Schedule editors will allow the **User** to only enter values within the range defined by the minimum and maximum values presented in these columns.

The range of some types is fixed by the system while the range of other types can be set by the **System integrator**:

Ranges fixed by the system: On/Off, Up/Down, Start/Stop, Open/Close

Ranges set by the **System integrator**: %, %8bit, Deg. C.

To change Min and Max values – type the new values in the desired cell.

4.3.3.1.5. *Step:*

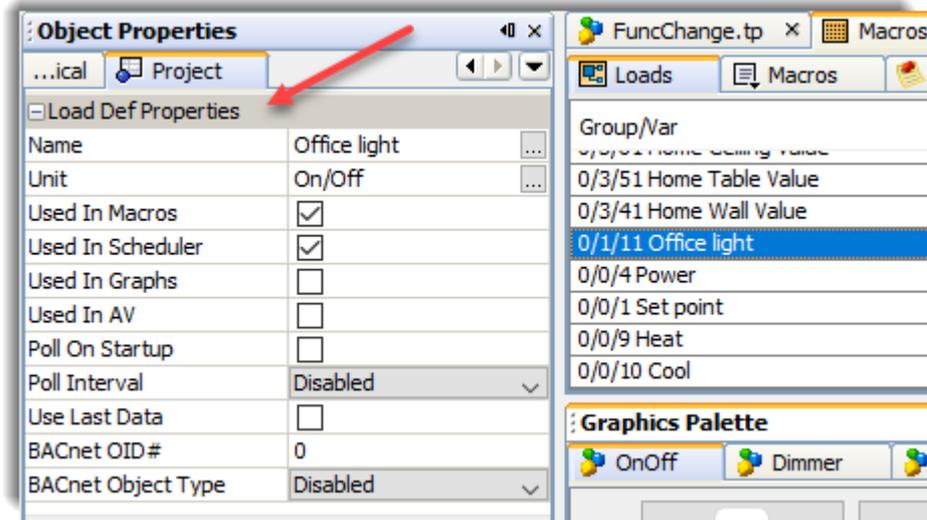
Step
5.0

Step column

This column presents the size of a single step used by the macro and schedule editors user interface, for increases or decreases of the load value. Step size is variable and can be changed by the **System integrator**.

To change step size – type the new value in the desired cell.

once you select a linker (/load) from the table you can edit it's following parameters:



Name - Presents a name used by the End user's interface to describe the **linker** for the **User**. By default, the software will use the **Linker's** name but you can edit this field to assign better descriptions.

Unit - Presents a name used by the End user's interface to describe the **linker** units for the **User** (for example on graph) . By default, the software will use the **Linker's** type, but you can edit this field to assign better descriptions.

Used In Macros – if checked, the **User** will be able to use this in it's Macro Editor.

Used In Scheduler – if checked, the **User** will be able to use this on the scheduler editor.

Used in Graphs - if checked, Maestro will log the values of this Linker present them om the graphs manager and let the **User** use it on his custom graphs.

Used in AV -

Poll On Startup – Applicable for load of the type Group Address only: if checked and it's a group address – Maestro will poll this load on start up.

Poll Interval - Applicable for load of the type Group Address only: select one of the options of this drop down menu to poll the load cyclically (can be useful when you need, for a graph, more readings then the sensor provide without using cyclic polling).

Use Last Data – Maestro's data base stores values on fixed predefined interval. Use Last Data let you select what Maestro should do when time comes to store a new value in cases where, since it's last store, no fresh data arrived to Maestro.

When unchecked – maestro will leave the current storage location empty and

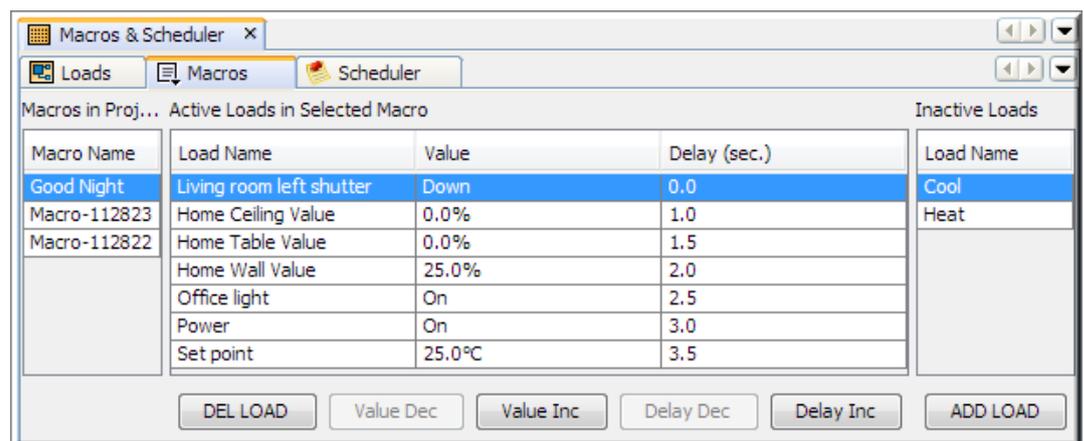
when the graph is drawn – at this point time, the graph will be cleared. In other words - Maestro will draw the graph only between real data points. When unchecked – maestro will duplicate the last saved value and store the duplicate value on current storage, when the graph is drawn – at this point time the graph will be continued as flat line having the last stored value. In other words - Maestro will use last stored values in order to draw continues graph without gaps.

BACnet OID# -
BACnet object type -

4.3.3.2. *Macros tab:*

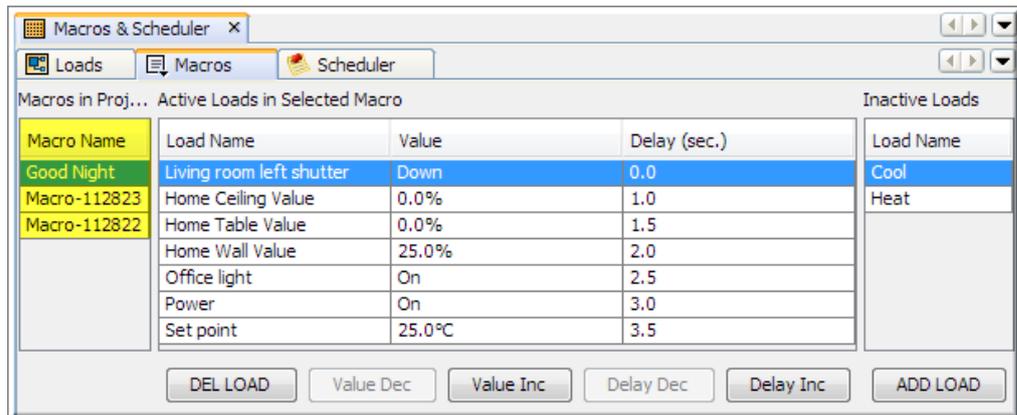
Is a Macro editor, it displays all the Macros used in the project and allows you to edit them and update the Maestro Device.

On MS4/MTS4 and later models - the "Values" and "Delays" set on the Macro editor will be used by the Maestro **Device** only if the device has no values on it's persistent storage, in other words: values set directly to the device are in higher priority than values set from Maestro designer.



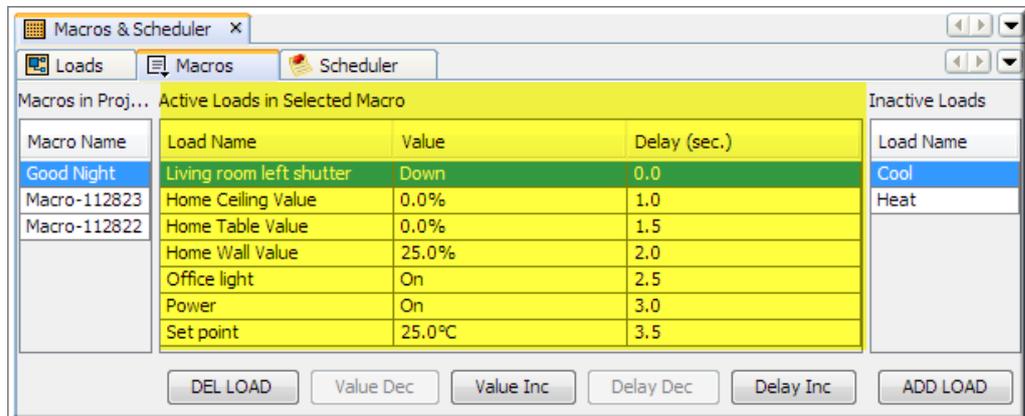
Macro Editor

Under the title *Macros in Project* on the left part of this tab is a list of all Macros used in the project. The Macros names in the list are the ID names of the Macro **Buttons** placed on the **Graphic Pages** or the ID names of Macro **Function Blocks** placed on the **Controller Pages**.
To view and edit a Macro, select it from the list using the mouse.



Macros list

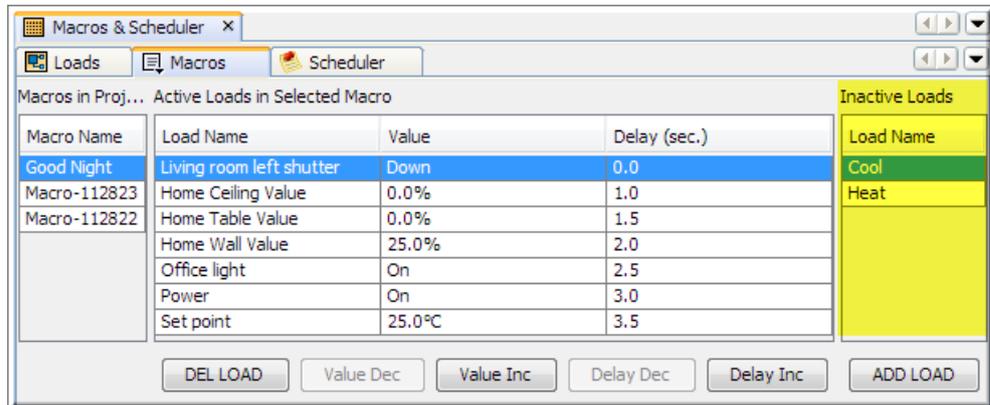
Under the title *Active Loads in Selected Macro* on the central part of Macro tab are 3 columns showing information regarding the setup for **Linkers** assigned to the selected Macro:



Active loads

The left column shows the **Linker** name as you entered it on the *Linkers* tab. The central part shows the value the **linker** will have when the Macro is activated. The right part shows the delay time (sec) from the moment the Macro is activated to the moment the **Linker** will receive the value set on the Macro.

Under the title *Inactive loads* on the right part of the tab, are the list of **Linkers** that can be added to the current Macro. The **Linker** names used on this list are the names you entered on the *Loads* tab.

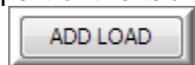


Inactive loads

Once you select a Macro from the macro list on the left part of the tab you can perform the following actions:

Add a **Linker** to a macro:

Select the **Linker** you want to add from the list of *Inactive loads* on the right part of the tab and press the *ADD LOAD* button:



Remove **Linker** from a macro:

Select the **Linker** you want to remove from the list of loads on the central part of the tab, (*Active Loads*) and press the *DEL LOAD* button:



Increase/Decrease value:

Select the **Linker** from the list of loads on the central part of the tab, (*Active Loads*). To set the desired value - use the increase/decrease buttons:



The value will be graduated by steps. The step size will be equal to the value you set for this Linker on the *Step* column of the *Loads* tab.

Increase/Decrease Delay:

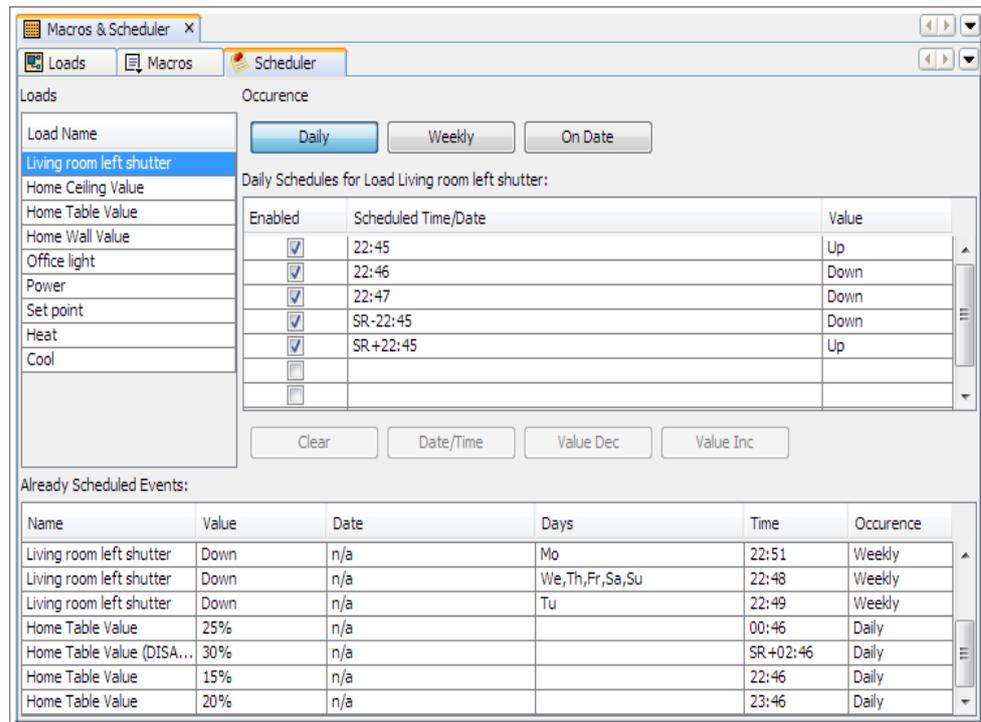
Select the **Linker** from the list of loads on the central part of the tab (*Active Loads*). To set the desired delay - use the increase/decrease buttons:



The value will be graduated by 0.5 second steps. Maximum delay is limited to 60 seconds.

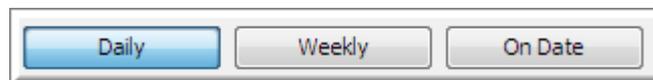
4.3.3.3. Scheduler tab:

Is a Scheduled events editor. It displays all the time dependent events used in the project and allows you to add, remove, and edit them. On MS4/MTS4 and later models - The "Values" and "Scheduled Time/Date" set on the Scheduled events editor **will be used by the Maestro Device only if the device has no values on it's persistent storage**, in other words: values set directly to the device are in higher priority than values set from Maestro designer.



Scheduled events editor

On the upper left part of the tab, under the title *Occurrence* , are three buttons:



Occurrence buttons

use them to define what type of event you want to view/edit:

Daily – these events will be triggered every day.

Weekly – these events will be triggered every week only on the specified weekdays.

On Date – these events will be triggered once on the specified date.

You can assign up to 6 scheduled events of each type to every load.

On the upper left part of the tab, under the title *Loads*, is a list of all **Linkers** available for schedule as you entered it on the *Loads* tab.

On the central part of the tab, under the title *Scheduled Time* is a list of scheduled events.

Daily Schedules for Load Living room left shutter:

Enabled	Scheduled Time/Date	Value
<input checked="" type="checkbox"/>	07:45	Up
<input checked="" type="checkbox"/>	23:00	Down
<input checked="" type="checkbox"/>	00:00	Down
<input checked="" type="checkbox"/>	SR-00:45	Down
<input checked="" type="checkbox"/>	SR+00:50	Up
<input type="checkbox"/>		
<input type="checkbox"/>		

Clear Date/Time Value Dec Value Inc

Scheduled Time list

The check box on the left displays, and allows you to define, if a scheduled event is enabled (will be a generated event) or disabled (will be ignored) when its scheduled time arrives.

☹ Uncheck a scheduled event when you want to temporarily skip the event but you want to maintain its setup for future use.

Next to the checkbox is a cell presenting the Time schedule for the event: Daily events are presented by event time. Weekly events are presented by time and weekdays. Date events are presented by scheduled time and date. Sunrise related events are presented as "SR+hh:mm" Or "SR-hh:mm" offset time. Sunset related events are presented as "SS+hh:mm" Or "SS-hh:mm" offset time, for example:

SR-01:30

which translates to: One hour and 30 minutes before sunrise.

On the upper right side of the tab, under the title *Value*, are cells presenting the value that will be assigned to the **Linker** when the scheduled event is triggered.

Value
Up
Down
Down
Down
Up

Value list

The way values are presented (e.g. "%", "On/Off, Up/Down...") is according to the user interface type defined on the *Linkers* tab.

On the lower part of the tab is the *already scheduled events* list

Already Scheduled Events:					
Name	Value	Date	Days	Time	Occurrence
Living room left shutter	Down	n/a	Mo	22:51	Weekly
Living room left shutter	Down	n/a	We,Th,Fr,Sa,Su	22:48	Weekly
Living room left shutter	Down	n/a	Tu	22:49	Weekly
Home Table Value	25%	n/a		00:46	Daily
Home Table Value (DISA...	30%	n/a		SR+02:46	Daily
Home Table Value	15%	n/a		22:46	Daily
Home Table Value	20%	n/a		23:46	Daily

Already scheduled events list

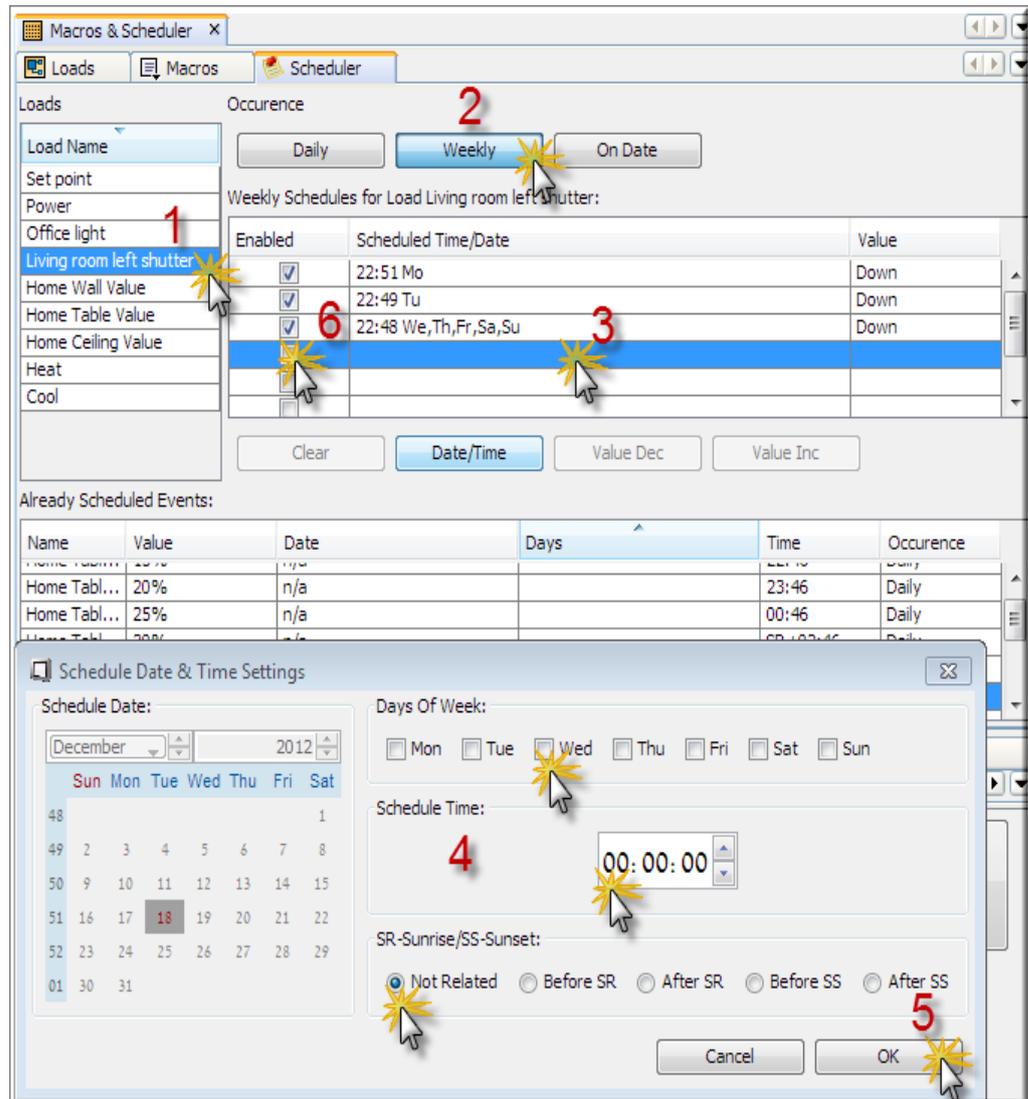
The list presents all the active events. The list can be sorted by any of its columns. To sort the list, click on the headline of desired column.

Priorities on the scheduler:

When an event is set on a specific day by the *On Date* scheduler – on this specific date, all other weekly and daily events on the same load will be ignored.

When an event is set for specific week days on the *weekly* schedule - the daily events for the same load will be ignored on the relevant days.

Adding an event to the scheduler:



Adding event to the scheduler

Adding event to the scheduler:

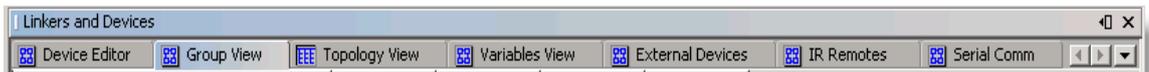
- 1) Select the **Linker** name from the *Loads* list.
- 2) Click on the desired *Occurrence* button, *Daily*, *Weekly* Or *On Date*.
- 3) Click on one of the empty *Schedule Time* cells - A dialog window will open.
- 4) Use the dialog window to set the event time
- 5) press the *OK* button.
- 5) Check the checkbox on the left of the time.

Deleting event from scheduler:

- 1) Click on the desired *Occurrence* button, *Daily*, *Weekly* Or *On Date*.
- 2) Select the **Linker** name from the *Loads* list.
- 3) Click on the *Schedule Time* cell you want to delete.
- 4) Press the *Clear* button.

4.4. Linkers and Devices window:

This window displays and allows you to edit and use the different types of **Linkers**. It also enables you to set the communication parameters of Maestro and external devices.

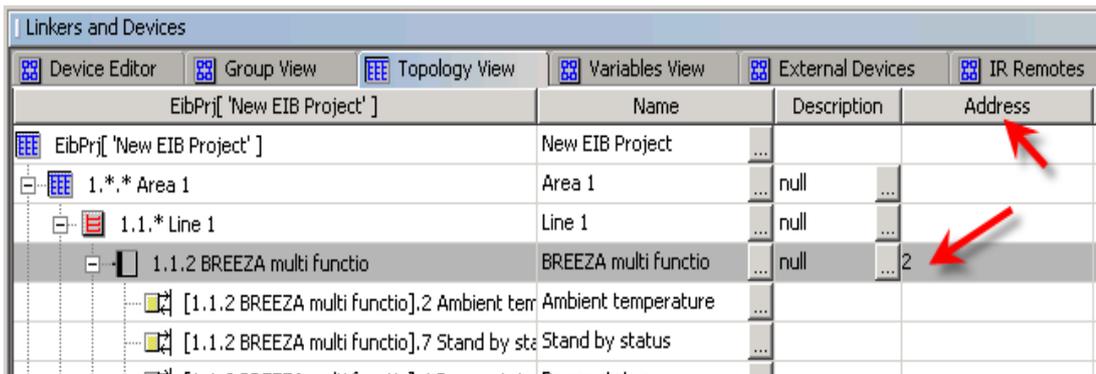


linkers and devices tab

The items on the tabs of this window are arranged in hierarchal tree structure, and their data is arranged in a table.

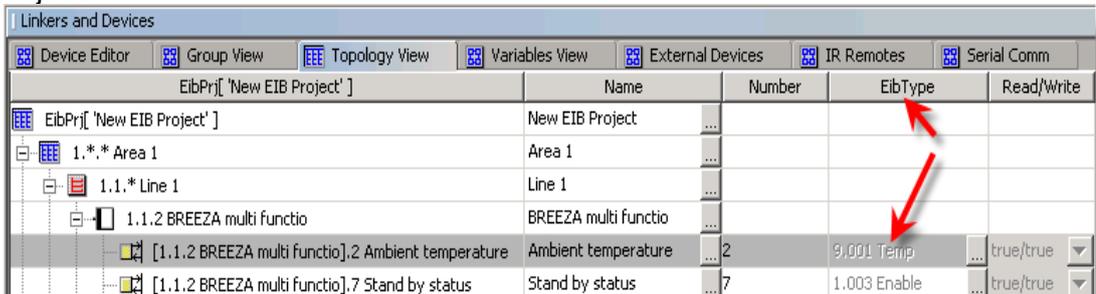
The tree structure is presented on the most left column; every level of the tree presents different type of data. For example, in Topology View, KNX areas are presented on the first level, KNX lines are presented on the next level and KNX devices are presented on the last level ...

According to your selection of tree level items - the columns placed to the right of the tree will dynamically change the data type they present. For example, when you select KNX device - the table's fourth column will present the device's address:



Forth column presents the device's address

but if you select KNX object - the forth column will present the KNX Data type of the object:

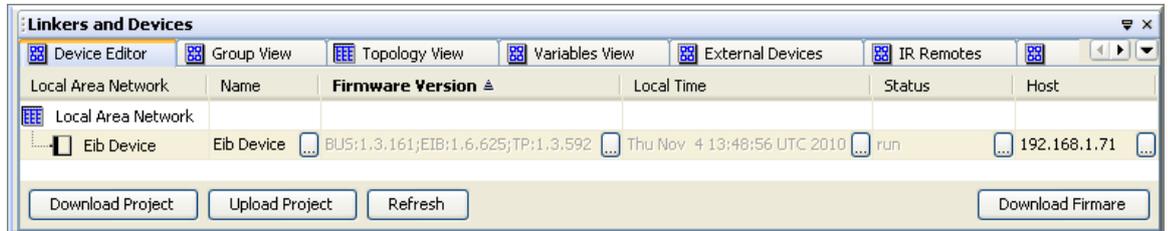


Forth column presents KNX Data type

Click on the header of the columns to sort the table.

Linkers and Devices window includes the following tabs:

4.4.1. Device editor tab



Device Editor tab

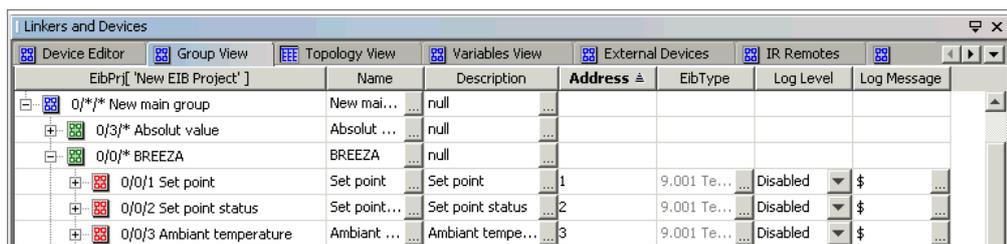
This tab is used for viewing and setting of communication to The **Device**. It contains the following fields:

- *Name* – use it to receive and set the **Device**'s name (not implemented yet).
- *Firmware Version* – after pushing the *Refresh* button it will present the firmware version read from the **Device**.
- *Local Time* - after pushing the *Refresh* button it will present the time and date read from the **Device**.
- *Status* - after pushing the *Refresh* button it will present the working state of the controller.

Host: enter the IP address of The **Device** you want to communicate with. the default port is 22.

4.4.2. Group view tab:

This tab displays, and enables you to edit and Link, the Group Addresses used in the project. It's tabs: Name, Description, Address and KNX Type are matching the ETS group view format.



and it has additional two tabs:

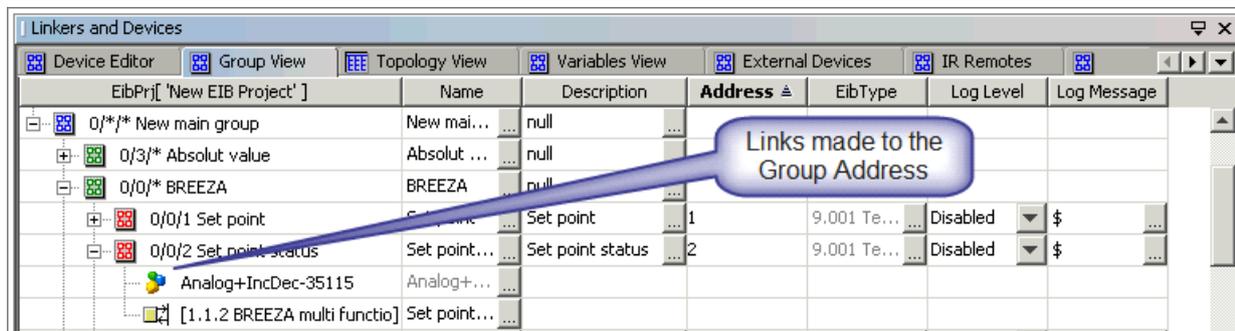
Log Level – set the log level or disable logging of the selected Group Address.

Log Message – set the text messes to be logged on a Group Address event.

Anywhere you place a \$ sign, the \$ will be replaced by the Group Address values.

In addition it displays Links made to the group address in the Maestro and ETS

projects.



Group view tab

Use it to:

- Edit Group Address properties: *Name*, *Description*, *Address* and *KNX (Data) Type*.

- Create **Links** by dragging Group Addresses and Dropping them on *Link for (Input/Output)* window and on **Controller Pages**.

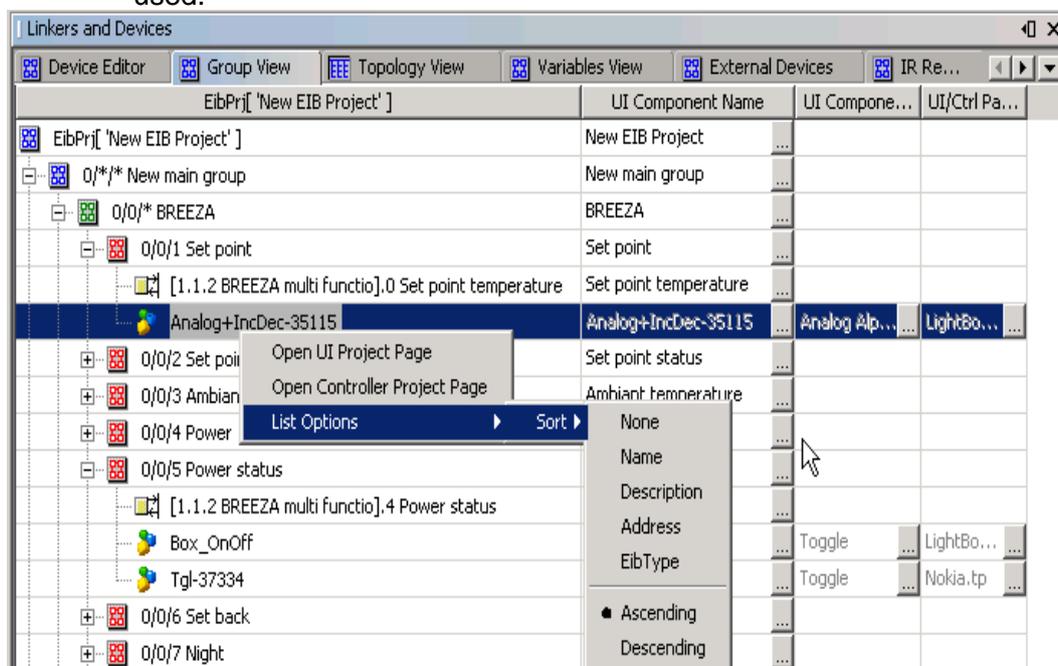
- View the list of Maestro **Objects Linked** to Group Address

- Using right mouse menu:

- Add and remove Group Addresses.

- Sort the table

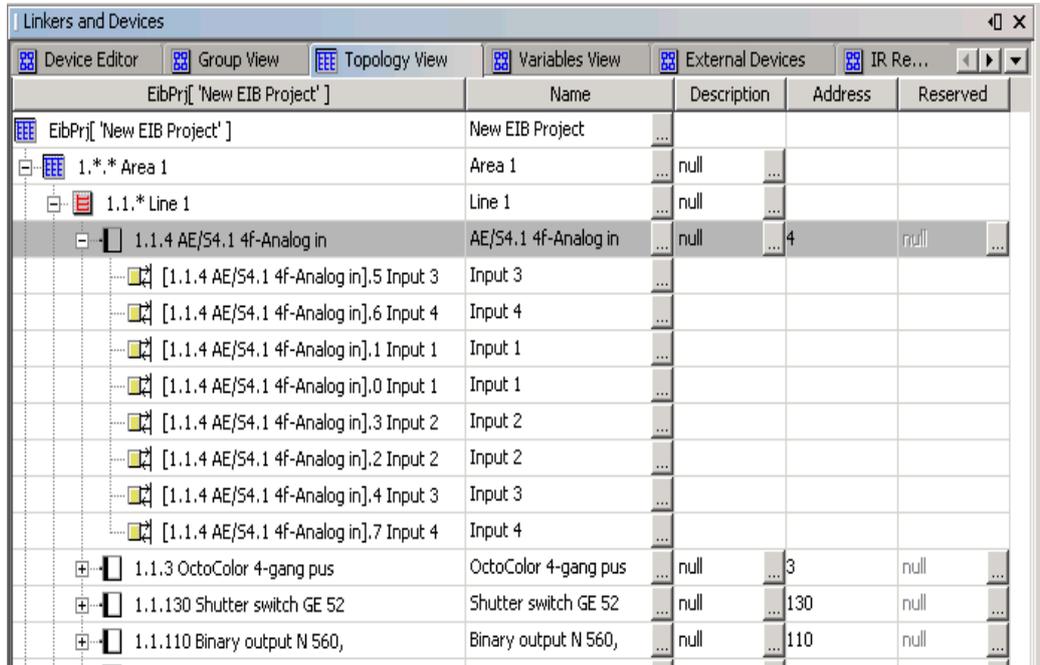
- Select Maestro **Objects Linked** to Group Address; using right mouse button menu, locate and open the project pages where these objects are used.



Right mouse menu

4.4.3. Topology View tab:

This view displays, and enables you to edit and **Link**, the KNX objects used in the project. It matches the ETS Topology format.



EibPrj['New EIB Project']		Name	Description	Address	Reserved
EibPrj['New EIB Project']		New EIB Project	...		
1.*.* Area 1		Area 1	...	null	...
1.1.* Line 1		Line 1	...	null	...
1.1.4 AE/S4.1 4f-Analog in		AE/S4.1 4f-Analog in	...	null	4
[1.1.4 AE/S4.1 4f-Analog in].5 Input 3		Input 3	...		
[1.1.4 AE/S4.1 4f-Analog in].6 Input 4		Input 4	...		
[1.1.4 AE/S4.1 4f-Analog in].1 Input 1		Input 1	...		
[1.1.4 AE/S4.1 4f-Analog in].0 Input 1		Input 1	...		
[1.1.4 AE/S4.1 4f-Analog in].3 Input 2		Input 2	...		
[1.1.4 AE/S4.1 4f-Analog in].2 Input 2		Input 2	...		
[1.1.4 AE/S4.1 4f-Analog in].4 Input 3		Input 3	...		
[1.1.4 AE/S4.1 4f-Analog in].7 Input 4		Input 4	...		
1.1.3 OctoColor 4-gang pus		OctoColor 4-gang pus	...	null	3
1.1.130 Shutter switch GE 52		Shutter switch GE 52	...	null	130
1.1.110 Binary output N 560,		Binary output N 560,	...	null	110

Topology view tab

When you drag and drop a KNX object to the *Virtual Links For* window – the *Maestro Designer* will search for all KNX Group Addresses associated with this KNX object and **Link** these Group Address to the point. This action is exactly the same as linking, one by one, all the group addresses associated with the KNX object, to the point.

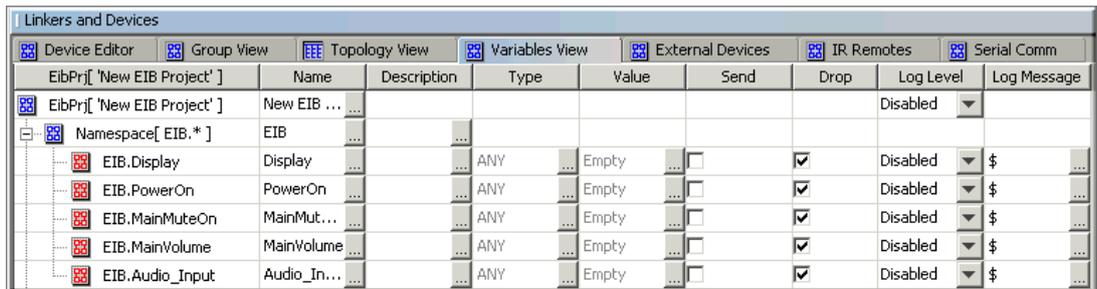
When you drag and drop a KNX object to a **Controller Page** it crates a **Linker Block** for the KNX object:

[1.1.80 System Design 4-gang].17 Status feedback object

This block has no **inputs** and One **Output**, the output will transmit its value any time that that it receives fresh data from one of the Group Address associated with this KNX object. Any time you reimport the ETS data base - *Maestro Designer* will update the list of associated Group Addresses of the **Linker Block**.

4.4.4. Variables View tab:

This view displays, and enables you to edit and Link, the variables used in the project.



Linkers and Devices									
EibPrj['New EIB Project']									
	Name	Description	Type	Value	Send	Drop	Log Level	Log Message	
EibPrj['New EIB Project']	New EIB ...						Disabled		
Namespace[EIB.*]	EIB								
EIB.Display	Display		ANY	Empty	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Disabled		\$
EIB.PowerOn	PowerOn		ANY	Empty	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Disabled		\$
EIB.MainMuteOn	MainMut...		ANY	Empty	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Disabled		\$
EIB.MainVolume	MainVolume		ANY	Empty	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Disabled		\$
EIB.Audio_Input	Audio_In...		ANY	Empty	<input type="checkbox"/>	<input checked="" type="checkbox"/>	Disabled		\$

Variables tab

The columns in this view are:

Name – Variable name (or space name when you select Namespace branch).

Description - Variable description (or space name description).

Type – variable data type.

Value – value of the variable immediately after initialization.

Send – when this box is checked, the variable will transmit its initial value immediately after start up. When initial value is not set (it is “Empty”) there will be no operation on startup.

Drop - when this box is checked, the variable will transmit its value only when its not equal to the variables previous value. In other words – only new values are transmitted.

Log Level – set the log level or disable logging of the selected Variable.

Log Message – set the text messes to be logged on a Variable event. Anywhere you place a \$ sign, the \$ will be replaced by the Variable’s value.

Use it to:

- Edit Variables properties displayed on the columns.
- Create **Links** by dragging **Variables** and Dropping them on: *Link for* (Input/Output) window and on: **Controller Pages**.
- Using right mouse menu:
 - Add and remove **Variables**.
 - Sort the table

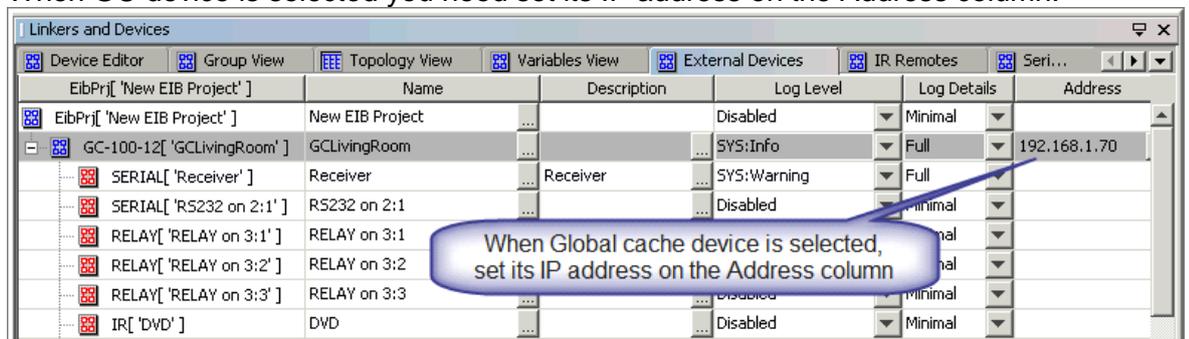
4.4.5. External devices Tab:

This view displays, and enables you to define and edit, the external devices used in the project. External devices are ports used by the maestro to send and receive communication.

Use the right mouse button menu to add external device. Supported devices are:

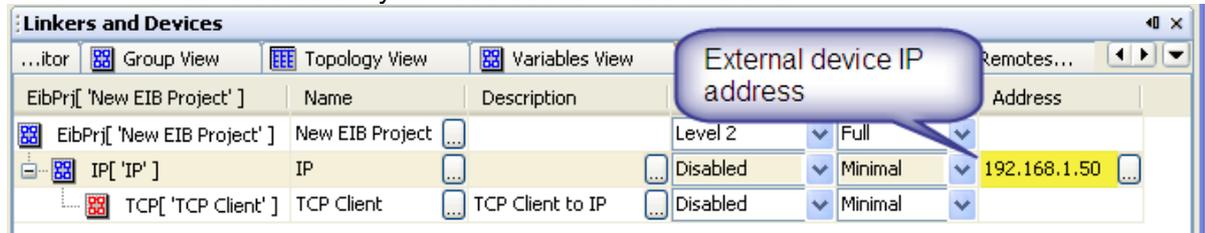
- Global Cache devices GC-100-6 and GC-100-6.
- TCP/IP network devices.
- Maestro's local serial communication ports.
- Maestro's remote serial communication ports.

When GC device is selected you need set its IP address on the Address column.



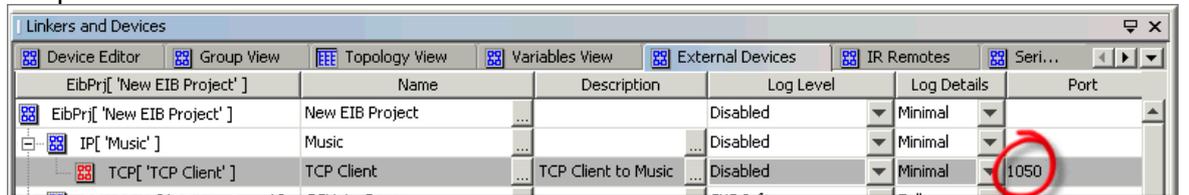
External devices Tab GC-100 settings

When IP device is selected: you need set its IP address and



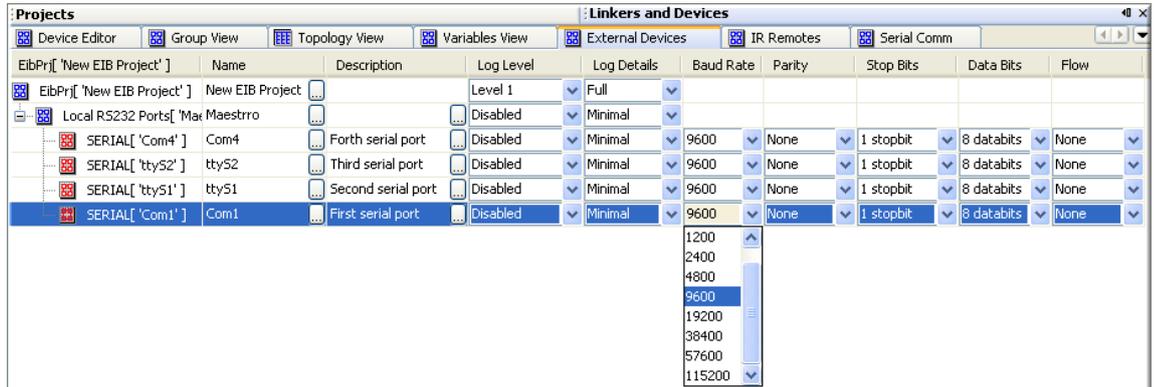
External devices Tab IP address settings

the port on the *Port* column.



External devices Tab IP port setting

when local serial com port is used: you need set its communication characteristics such as *baud rate* and *parity*.



Local port settings

when remote serial com port is used: you need set the IP address of the remote Maestro (IP port is 3999) and communication characteristics such as *baud rate* and *parity*.

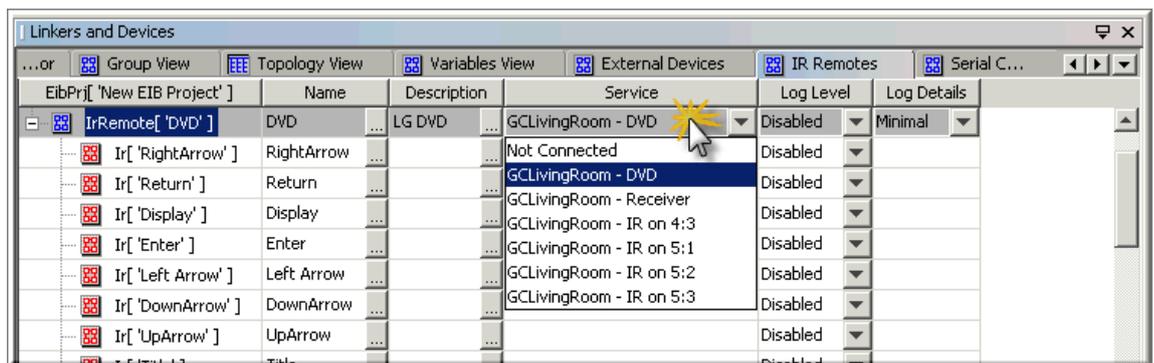
☺ Remote serial com is available only if it is not used by the project running, locally, on the remote Maestro.

☺ Remote serial com is operational when using Maestro Designer *RUN* mode

4.4.6. IR Remotes tab:

This view displays and enables you to define and edit properties of the IR codes and channels used in the project.

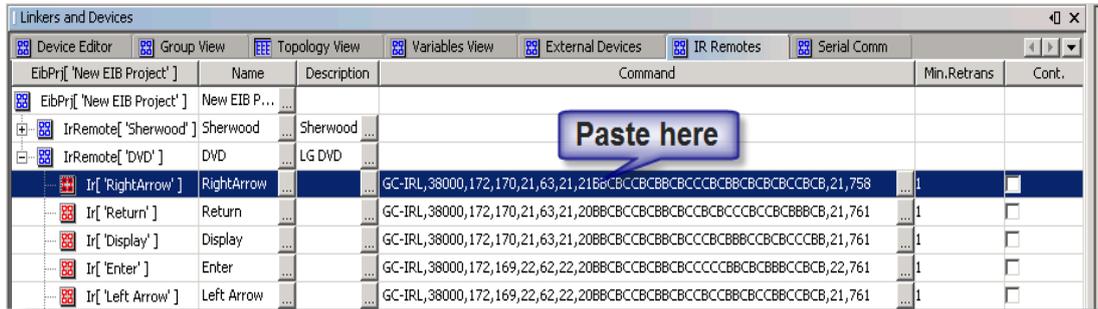
Use right mouse button menu to add an IR remote, and then use the drop-down list from the “Service” column to select the IR port that will be used to transmit the codes. The drop-down list includes all the physical IR ports you define in the *External Devices* tab.



Select the IR port

After you add the physical port you can use the right mouse button menu to add IR codes and then edit their properties.

The IR codes properties are in the following columns:



Paste the IR code to the *Command* column

Min. returns column:

Use it to define the minimum times that this IR code will be transmitted.

Cont. column:

when this parameter is checked -

The device will start transmitting the code when this IR object value turns to True and will continuously repeat the transmission until the IR object value turns to False (this useful for functions such as “Volume Up/Down”).

When this parameter is unchecked -

The device will transmitting the code “Min. Returns” times once when this IR object value turns to True.

IR command **Objects** can be drag and dropped in to the links window, Macros and Scheduler Loads Table and in to the controller pages. IR command **Object** value is true during transmission of IR code throe the port and False when it is not transmitting.

Log Level column:

Set the log level or disable logging of the selected IR commands.

Log Message column:

Set the text messes to be logged on a IR commands event. Anywhere you place a \$ sign, the \$ will be replaced by the IR commands values.

4.4.7. Serial Comm tab.

This tab enables you to define, edit, and display properties and software code of the communication protocols used in the project.

Communication protocols mechanism enables you to implement two different functions:

- 1) Communication protocols: in order to control **external** devices
- 2) **Internal**, complex algorithms and functions.

To use the serial comm. for **internal** functions you have to:

- Create, dummy, TCP, *External Device*, having the following address:

IP address: 127.0.0.1

Port: 7000.

- Use the *service* column to select the dummy as your *External Device*.

- Use the *Serial Comm.* mechanism according to the instructions in the following chapter "[Communication protocol](#)" described below.

Now you can use *transmitters* to input data for your algorithm and return the result of your algorithm by using *receivers*.

An advantage of using this method is that protocols can be exported and imported. This enables the import of the complete function to other projects.

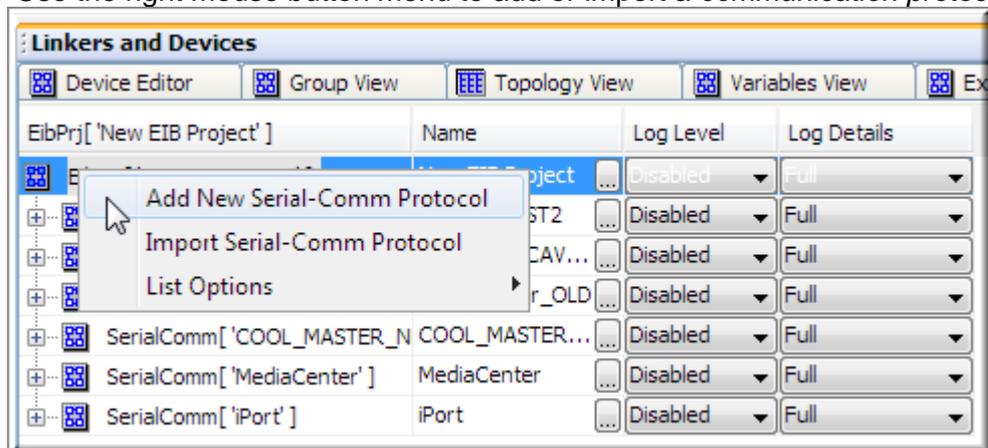
Communication protocols make use of Expression Language.

To learn about expression Language refer to the corresponding chapter [Expression Language](#).

Sample protocols can be downloaded from CDI's web site.

4.4.7.1. Communication protocol:

Use the right mouse button menu to add or import a *communication protocol*.



add or import a *communication protocol*.

The protocol properties are in the following columns:

EibPrj['New EIB Project']	Name	Description	Service	Buffer Size	Buffer Timeout	Init Expr	Connect Expr	Idle Expr	Log Level	Log Details
EibPrj['New EIB Project']	New EIB Project								Disabled	Full
SerialComm['Russound_ST2']	Russound_ST2	== IMPO...	remote - COM1	512 bytes	Disabled	// =...	static bo...	return...	Disabled	Full

serial Comm. Tab

Name column:

Free Text field containing the name of the *communication protocol*. Use letters

only in this field and never use blank spaces.

Description column:

Free text field describing the *communication protocol*. Use it to present information related to the protocol such as: version #, date, documentation it's based on, and communication characteristics such as physical connections and baud rate, instruction for using the protocol, and instructions for setting the initialization parameters.

Service column:

Use the drop down list from the *Service* column to select the *External Device* communication port/IP port that will be used to transmit and receive the communication. The drop down list includes all the physical communication ports you define in the *External Devices* tab.

Buffer size column:

Use the drop down menu to select the incoming data buffer size. When the incoming data does not match any of the filters and its length exceeds the buffer size, Maestro will keep only the fresh data and discard old data beyond buffer size.

Buffer time out:

Use the drop down menu to select the incoming data time out. When the incoming data does not match any of the filters and the period it's on the buffer exceeds the buffer time out, Maestro will discard this data and keep only fresh data.

Init Expression column:

The column's field contains Expression Language code. This code will be executed a single time when the controller starts. You can use it to declare Static Variables and initialize them.

Connect Expr column:

The column's field contains Expression Language code. This code will be executed single time when the controller initializes communication with the external device.

In *Connect Expr*, you can use the *return* command for sending initialization strings to the external device.

Idle Expression column:

The column's field contains Expression Language code.

When the Maestro's processor is not busy, this code will be executed approximately once every second,. You can use it to implement functions such as: delays and cyclic polling of external devices (by calling *transmitters* that generate status requests).

Fast Idle Expression column:

The column's field contains Expression Language code.

When the Maestro's processor is not busy, this code will be executed at the rate set on the Fast Idle Delay. You can use it to implement functions such as: delays cyclic operations and measuring time.

Idle Always:

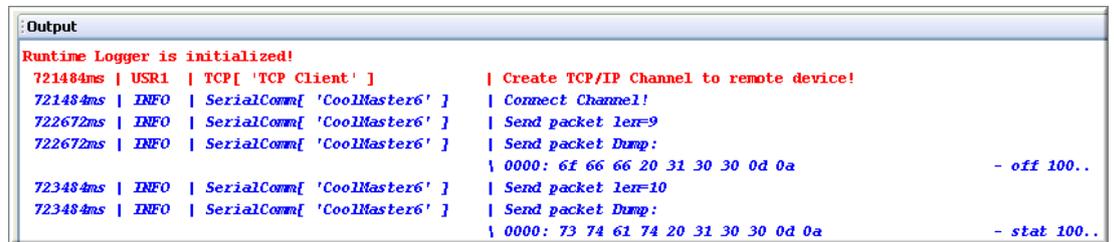
when checked the idle process will be executed even if the protocol is not linked to external device.

Log Level column:

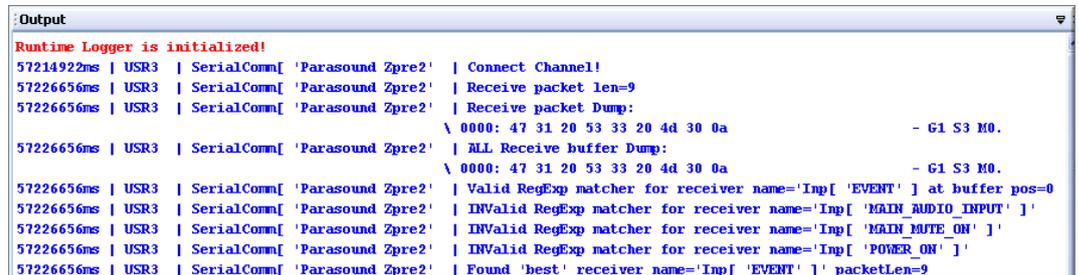
Use it to set the log level or disable logging of the selected *communication protocol*.

The logger is a very useful tool for debugging protocols. It stores all the received values, transmitted values, code and execution errors, and messages of the *communication protocol*.

When using *run* mode, the PC running Maestro designer will simulate the Maestro **Device** and enable you to communicate with GC and IP external devices (with no need to use a Maestro **Device**). In *run* mode, the logged information is displayed in the Output Window in real time.



on run mod: outgoing communication Log is presented on the output window



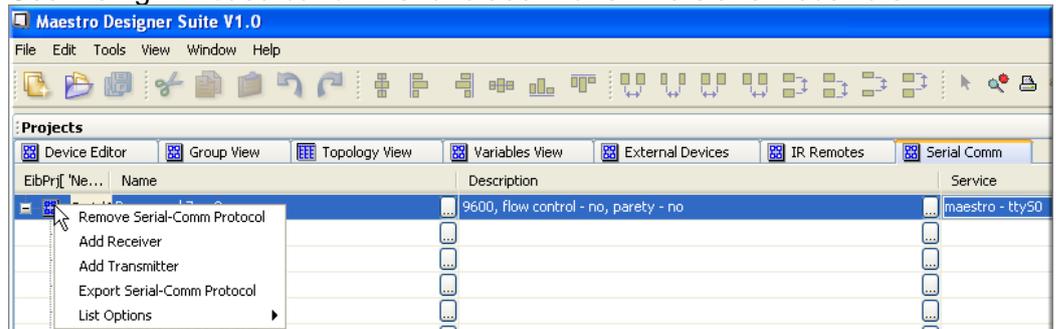
In run mode: received string and filter results Log is presented in the output window

Log Message column (for MS3/MTS3 or older models only):

Use it to set text messages to be logged on a protocol event. Anywhere you place a \$ sign, the \$ will be replaced by the protocol's event value.

Adding Transmitters and Receiver:

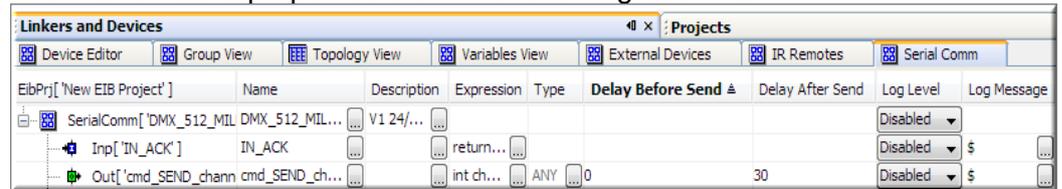
Use the right mouse button menu to add Transmitters and Receivers



add *Transmitters* and *Receivers*

4.4.7.2. *Transmitters:*

Transmitters are used to perform Expression Language code and to trigger and pass communication commands to external devices. *Transmitters* are **Linkers** so you can drag and drop and link their **input** to **buttons** and **function blocks** of the controller and to *Schedules & Macros Loads Table*. The *Transmitter's* properties are in the following columns:



EibPrj['New EIB Project']	Name	Description	Expression	Type	Delay Before Send	Delay After Send	Log Level	Log Message
SerialComm['DMX_512_MIL DMX_512_MIL...]		V1 24/...					Disabled	
Inp['IN_ACK']	IN_ACK		return...				Disabled	\$
Out['cmd_SEND_chann cmd_SEND_ch...']			int ch...	ANY	0	30	Disabled	\$

transmitter properties

Name column:

Free Text field containing the name of the *transmitter*.

To create consistency, it is recommended to start all *transmitter's* names with *CMD_*. Use letters only in this field and never use blank spaces.

Description column:

Free text field describing the *transmitter's* use and functionality.

Expression column:

Contains the Expression Language code to be executed when the *transmitter* is triggered.

When a *transmitter* is triggered – it will execute its expression language code once.

Values passed to the *transmitter* **input** will be assigned to a variable named *arg*.

It is possible to pass more than one value to the *transmitter*, in this case they will be indexed starting with 0 so the first value will be *arg[0]*, the second *arg[1]* and so on.

For example:

```
CMD_SELECT_zone_device = {zone,device};
```

This code line will trigger the transmitter *CMD_SELECT_zone_device* and will send two values to it: *arg[0]* will have the value of *zone* and *arg[1]* will have the value of *device*.

When using the value passed to the *transmitter*, always, assign it's value to a Variable first, for example:

```
int volume = arg;
```

then use only the Variable *volume* in your expression.

The command `return` is used to trigger a transmission to the external device. The string value of the expression placed after the `return` command will be sent to the external device.

For example:

```
return "TD" + "\r";
```

will transmit the string `TD`, to the external device, followed by Carriage Return. When using the command `return;` without any value following it – nothing will be transmitted to the external device.

Within the code you write, you can call and use the value of other communication *transmitters* and *receivers* (used by the same protocol) by using their Name.

For example:

A *transmitter* named `CMD_POWER_on_off` can call the *transmitter* `CMD_POWER_ON` when it is triggered by the value `true` and, call the *transmitter* `CMD_POWER_OFF` when it is triggered by the value `false`. Here is the code for `CMD_POWER_on_off`:

```
if( arg ){                // if incoming value is "true"
    CMD_POWER_ON = 1;    // call the transmitter CMD_POWER_ON
}
else{                    // if incoming value is not "true"
    CMD_POWER_OFF = 1;  // call the transmitter CMD_POWER_OFF
}
return;
```

So in this example, the *transmitter* `CMD_POWER_on_off` is not transmitting directly to the external device but it is triggering other *transmitters* to do it.

Type column:

Use this drop down menu to set Data Type for the `arg` variable.

When data is sent to the *transmitter*, the Maestro will, automatically, cast its type to the Data Type of the `arg` variable.

When you select the value *Any* – `arg` will receive the same Data Type as the incoming data.

The use of *Any* as *Type* is not recommended since it can produce inconsistency in the expression.

For example:

- if your code is:

```
int powerState = arg; // (0=OFF,1=ON,2=STANDBY)
```

- and you select *Any* as data type,

- and you pass the Boolean value `true` to the transmitter:

then the interpreter will generate an error since `arg` is Boolean and `powerState` is an integer.

Nevertheless if you select *integer* as the `arg` Data Type then the incoming Boolean data `true` will be automatically transformed into the integer value "1".

(and Boolean data *false* will be automatically transformed into the integer value “0”).

There is no need to assign Data Type If the transmitter’s expression does not use the *arg* variable.

Delay before send column:

Use it to set a time gap (in milliseconds) between sending from this transmitter to previous transmission of the same protocol.

Maestro will send the data from this transmitter only when a time period equal to the delay (or more) passed after previous transmission from the same protocol.

Delay before send column:

Use it to set a time gap (in milliseconds) between sending from this transmitter to next transmission of the same protocol.

Maestro will send the next transmission of the same protocol only after a time period equal to the delay (or more) passed after transmitting from current transmitter.

Log Level column:

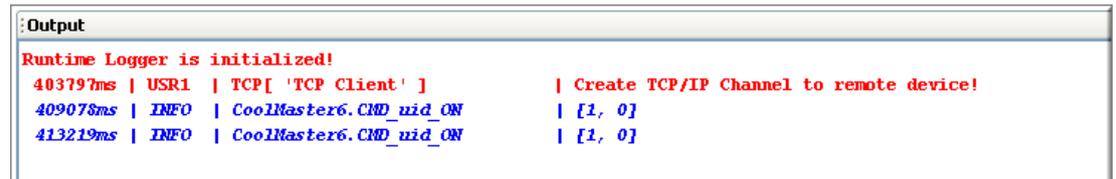
Use it to Set the log level or disable logging of the selected *transmitter*.

When enabled, the Logger will store all the values that the transmitter will return. When using *run* mode, the logged information will be displayed on the *Output Window*.

Log Message column:

Use it to set the text messages to be logged on a *transmitter’s* events.

Anywhere you place a \$ sign, the \$ will be replaced by the *transmitter’s* value.



```

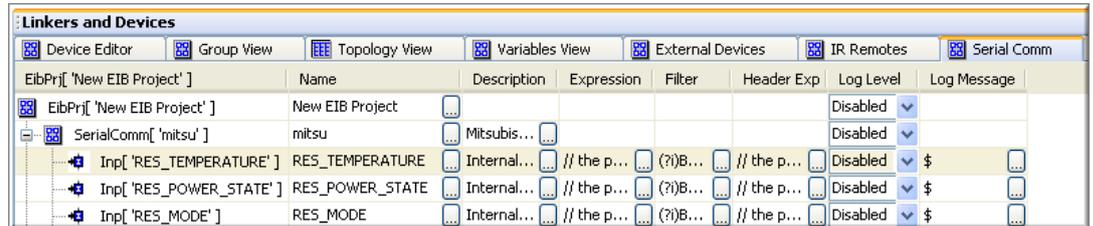
:Output
Runtime Logger is initialized!
403797ms | USR1 | TCP[ 'TCP Client' ] | Create TCP/IP Channel to remote device!
409078ms | INFO | CoolMaster6.CMD_uid_ON | [1, 0]
413219ms | INFO | CoolMaster6.CMD_uid_ON | [1, 0]
  
```

on *run* mode, transmitter’s log is presented on the output window

4.4.7.3. Receivers

Receivers are used to receive communications coming from External Devices. *Receivers* filter the communications, perform Maestro Expression Language code, and return, to their **output point**, values that can be further used by the Maestro controller. *Receivers* are **Linkers** so you can drag and drop and **link** their **output** to **buttons**, **function blocks** of the function blocks controller, and **Schedules & Macros Load Table**.

The receiver’s properties are in the following columns:



Linkers and Devices						
EibPrj['New EIB Project']						
	Name	Description	Expression	Filter	Header Exp	Log Level
EibPrj['New EIB Project']	New EIB Project					Disabled
SerialComm['mitsu']	mitsu	Mitsubis...				Disabled
Inp['RES_TEMPERATURE']	RES_TEMPERATURE	Internal...	// the p...	(?)B...	// the p...	Disabled
Inp['RES_POWER_STATE']	RES_POWER_STATE	Internal...	// the p...	(?)B...	// the p...	Disabled
Inp['RES_MODE']	RES_MODE	Internal...	// the p...	(?)B...	// the p...	Disabled

receiver's properties

Name column:

Free Text field containing the name of *the receiver*.

It is recommended to start all *receivers* name with `IN_`. Use letters only in this field and never use blank spaces.

Description column:

Free text field describing the *receiver's* use and functionality.

Filter column:

The *filter* contains a Regular Expression (also referred to as regex or regexp). It provides a concise and flexible means for matching strings, such as particular characters, or patterns of characters. The regular expression is written in a formal language that can be interpreted by the Maestro to examine incoming strings and identify parts that match a provided communication protocol specifications.

To learn about regular expressions you can use documentation from Oracle Java

<http://download.oracle.com/javase/tutorial/essential/regex/intro.html> and other web sites, for example, Regex simulator:

<http://www.surajshrestha.com.np/projects/apps/RegExpSimulator/RegExpSimulator.php>

The incoming communication from a Maestro external port is a list of characters, sent by external device. The purpose of the filter is to:

- find, within this list, a pattern, that matches a known format, containing information relevant to the *receiver*,
- if necessary, extract the relevant information,
- if necessary, break the extracted information into easy to handle strings,
- if necessary, send the extracted information for further processing and use on the *receivers* expression,
- trigger the execution of the *receivers'* expression language code.

in case of a match between the filter and incoming string:

the filter will extract the characters that match the characteristics that are defined inside the brackets and send them for further processing by the receiver expression code. The extracted characters are passed to the expression as an array of strings using the variable `argv`:

`argv[0]` will contain all of the matched string, `argv[1]` will contain the part that matches the first brackets, `argv[2]` will contain the part that matches the second brackets and so on...

Here is an example for the use of regex in Maestro:

The Parasound Zpre2 PreAmp reports its state by transmitting a string having the following format:

Ga Sb Mc<CR> where

- *a* is power indicator (0=OFF,1=ON)
- *b* is the input number (1=IN1, 2=IN2,3=IN3, 4=IN4)
- *c* is Mute indicator (0=OFF,1=ON)
- <CR> (hexadecimal h0D) is an end of string mark.

Lets assume the PreAmp sends the string: *G1 S3 M0*<CR>

when this string is filtered by the following Maestro's regular expression:

```
(?s)G(.) S(.) M(.)\x0d
```

- 1) it will be a match!
- 2) the regular expression will extract the 3 characters positioned where the brackets are: *(?s)G(.) S(.) M(.)\x0d*.
- 3) the regular expression will assign the extracted values into the *argv* variable in the following way:

argv[0] will hold the whole matched string: *Ga Sb Mc*<CR>

argv[1] will hold the value from the first bracket: "1"

argv[2] will hold the value from the second brackets: "3"

argv[3] will hold the value from the third brackets: "0"

- 4) the filter will pass *argv* to the receiver's expression and will trigger its execution.

As long as no match is found, the Maestro will continue to save the incoming characters in a FIFO buffer, the length of the buffer is 4K bytes.

Any time new character/s arrive from the external device, through the Maestro port, the Maestro trays to find a match to each of the protocol's filters. If more than one match is found - a "best" match will be located and only it will be processed. After a best match is found, all the previous incoming characters will be cleared from the buffer and only new arriving characters will be processed.

When the filter field is left empty and when it has the value *null* – the filter will be skipped, and the expression of the *receiver* will never be executed.

Here is another example:

An audio receiver reports is volume level (0 to 100) using the following format:

<*W,x,VOLUME,y*>*nl*

where:

- *x* is zone "A" or "B"
- *y* is volume level "0" to "100" (here are some examples "0","8","25","100")
- *nl* it the New Line character

the filter for this string is:

```
<W, (.) ,VOLUME, ([0-9]{1,3}?)>\n
```

The first brackets: *(.)* will extract one character ("A" or "B")

The second brackets Will extract 1 to 3 digits: [0-9] means characters "0" to "9" (only digits), {1,3}? Means between one appearance to three

appearances.

So when, for example, the following string is received by the Maestro:

```
<W,A,VOLUME,52>\n
argv[1] will be "A"
argv[2] will be "52"
```

Expression column:

Contains Expression Language code. The code will be executed, one time, when an incoming communication from the External Device will (best) match the *receiver's* filter.

The filter passes an array of strings to the *receiver*. The array is assigned to a variable named `argv` (`argv[0]`, `argv[1]`...

When using the `argv` value: always assign the value to a variable **first**, for example:

```
int volume = (int)argv[1];
```

then use only the variable `volume` in you expression.

The command `return` is used to set a value to the *receiver* and to send it from the *receiver's output point*.

When using the command `return`, without any value and when not using the `return` command at all – the *receiver's* value will not change and nothing will be transmitted from its **output point**.

Within the code, you can call and use values of other *communication protocol transmitters* and *receivers* used by the same protocol by using their Name.

For example:

A *receiver* named `IN_state` receives following hexadecimal status string from audio preamp:

3C 50 01 49 02 4D 01 3C (equals to:

{ "<" , "P" , 0x01 , "I" , 0x02 , "M" , 0x00 , ">" } and meaning power is on input # is 2 and mute is off)

extracts the relevant data using the filter : `<P(.)I(.)M(.)>`

and updates the values of 3 *receivers*: `IN_POWER_ON`, `IN_MAIN_AUDIO_INPUT` and `IN_MAIN_MUTE_ON`:

```
IN_POWER_ON = argv[1];
IN_MAIN_AUDIO_INPUT = argv[2];
IN_MAIN_MUTE_ON = argv[3];
return;
```

So in this example, the *transmitter* `IN_state` receives a status string from the external device, breaks it in to 3 different strings and sends the strings to 3 different *receivers*. Each one of them will be assigned a value of one indication from the audio preamp. As a result the 3 *receivers* will transmit the assigned values out of their **output point**.

Here is another example

```
// filter is: <W,(.),VOLUME,([0-9]{1,3}?)>\n
// sample incoming string is: <W,A,VOLUME,50>\n
```

```

// meaning ZONE A volume is 50%
// argv[0] is the complete string
// argv[1] is Zone "A" or "B"
// argv[2] is volume level "0" to "100"

char zone = argv[1][0];
string stringVolume = argv[2];

int volume = 0 ;
for(int i = 1; i <= volumeLength; i++){ // for every
character
    volume = volume*10 + (stringVolume[i-1] - (int)'0');
}
if( volume > 100){
    return "Invalid volume level";
}
if(zone == 'A'){
    IN_ZONE_A_volume = volume;
    IN_ZONE_TEXT_volume = "ZONE A VOLUME IS: " + volume +
"%";
    return;
}
else{
    if(zone == 'B'){
        IN_ZONE_B_volume = volume;
        IN_ZONE_TEXT_volume = "ZONE A VOLUME IS: " + volume +
"%";
        return;
    }
}
return "Invalid zone";

```

On this example, the filter extracts two strings:

argv[1] contains one character “A” or “B”

argv[2] contains one to three characters representing volume level 0-100(%)

As a result of the receiver’s expression execution, two receivers are being assigned a value.:

IN_ZONE_A_volume Or *IN_ZONE_B_volume* are getting the volume level, as an integer and *IN_ZONE_TEXT_volume* is getting a text string value, this string can be displayed using **Background Button**:



It is possible to send the text string to KNX devices over the KNX bus using 14byte Data Type (if the string is more than 14 bytes the extra characters will be dropped and only the first 14 bytes will be sent to the bus)

Log Level column:

Use it to set the log level or disable logging of the selected *receiver*.

When enabled, the Logger will store all the values that the *receiver* will return. When using *run* mode, the logged information will be displayed on the *Output Window*.

Log Message column:

Use it to set the text messages to be logged when the *receiver* has fresh values. Anywhere you place a \$ sign, the \$ will be replaced by the *receiver's* values.

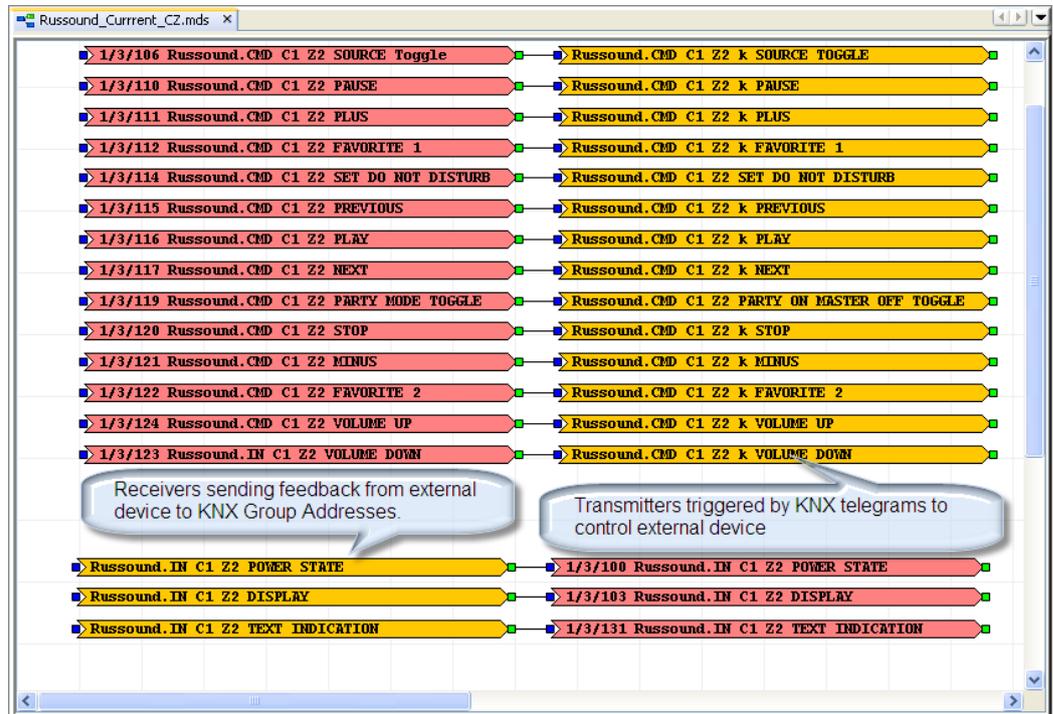
Header Expression column:

Header expression Contains Expression Language code and is executed after a match was found by the *filter* and before the execution of the *receiver's* expression.

The purpose of the Header Expression is to test the validity of the matched string that has been passed by the filter. If the value 1 is returned from the Header Expression, the *receiver's* expression will be executed, if the value -1 is returned – the expression will not be executed. When this field is left empty, or has the value “null”, this procedure will be skipped and the data will be sent from the *filter* directly to the expression.

A common example for the use for Header Expression is checking checksums. Here is a sample code for checking checksum of Russound's multiroom system:

```
int checksum = 0;
int len = argv[0].length;
for (int i = 0; i < len - 2 ; i++) {
    checksum += argv[0][i];
}
checksum = checksum + len - 2 ;
checksum = checksum & 0x7F ;
if ( (int)argv[0][len - 2] == checksum) {
    // valid!
    // now return
    return 1;
}
else{
    // invalid! this is not our packet!
    // return -1 to indicate error!
    return -1;
}
```



using Maestro as gateway between KNX to external device

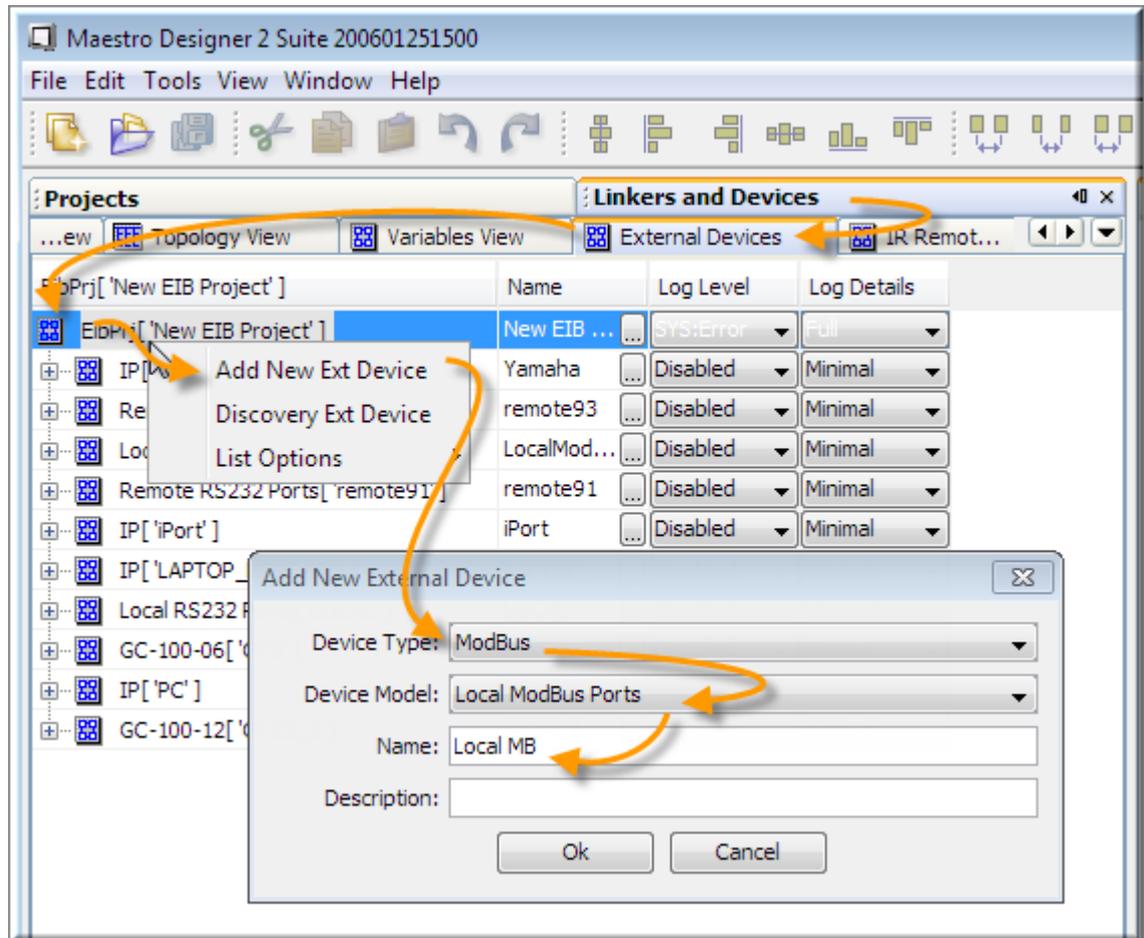
4.4.8. Modbus tab:

Modbus protocol is embedded into Maestro Server's firmware, enabling:

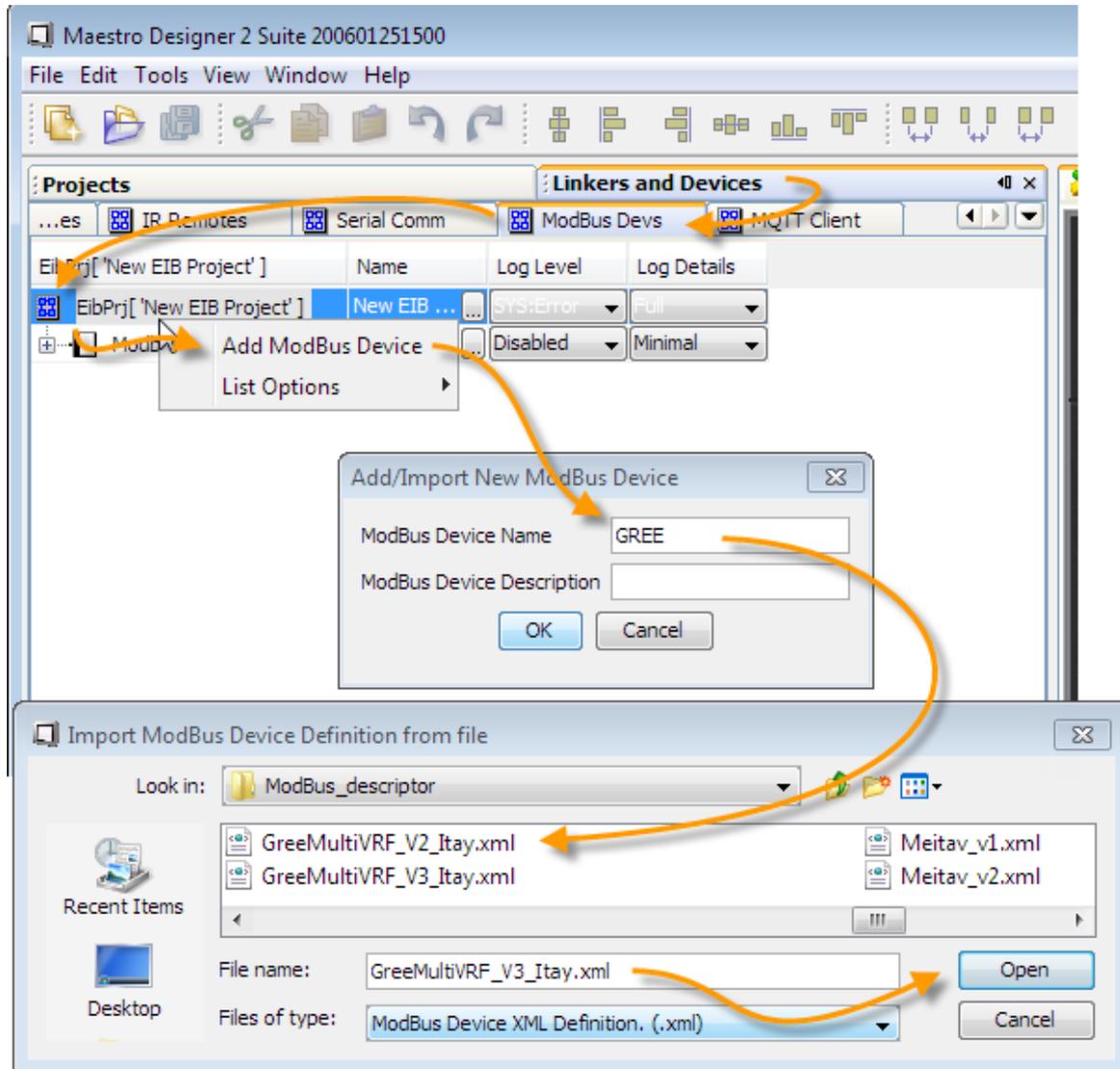
- Seamless control over ModBus client devices.
- Acting as gateway between ModBus to KNX and to any other system connected to Maestro.
- Use of Maestro as ModBus -over-IP to ModBus -over-RS485 gateway, thus, an external ModBus -Master over IP can control Slave- ModBus devices connected to Maestro over RS485.

Here are the steps for Controlling Modbus devices:

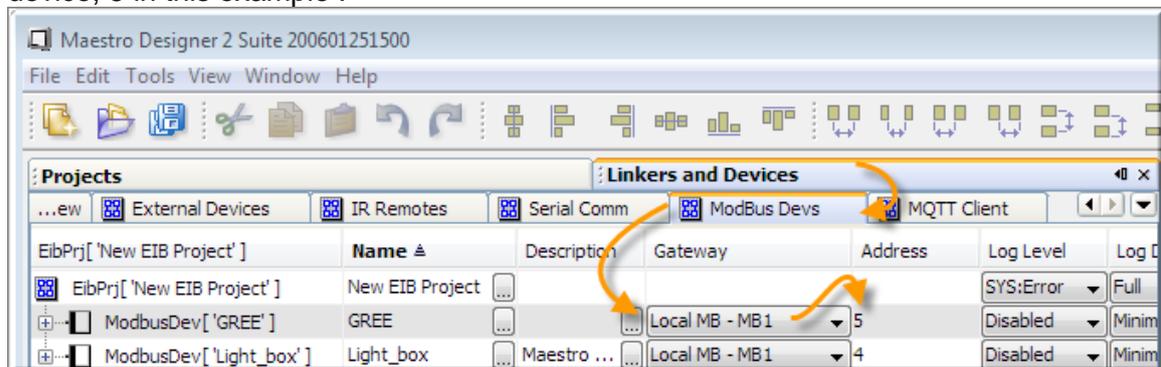
- 4.4.8.1. For controlling a ModBus device – first, an XML descriptor file must be created. The XML file holds information that Maestro Designer needs in order to create interface for communication with the device. The XML file includes the addresses of the registers and coils used by the ModBus product but can have, in addition, formulas, conversions, parameters, filters for out-of-range values, script language code and more. The XML file makes the use of the device on Maestro Designer extremely simple. For more details, about the XML file please contact us at info@cdinnovation.com
- 4.4.8.2. To define the physical media used for the communication:
 - for ModBus over RS485 - Add one local ModBus External-Device
 - for every ModBus over TCP IP address add one External-Device (please notice: ModBus IP port is fixed on the Maestro to port 1502!):



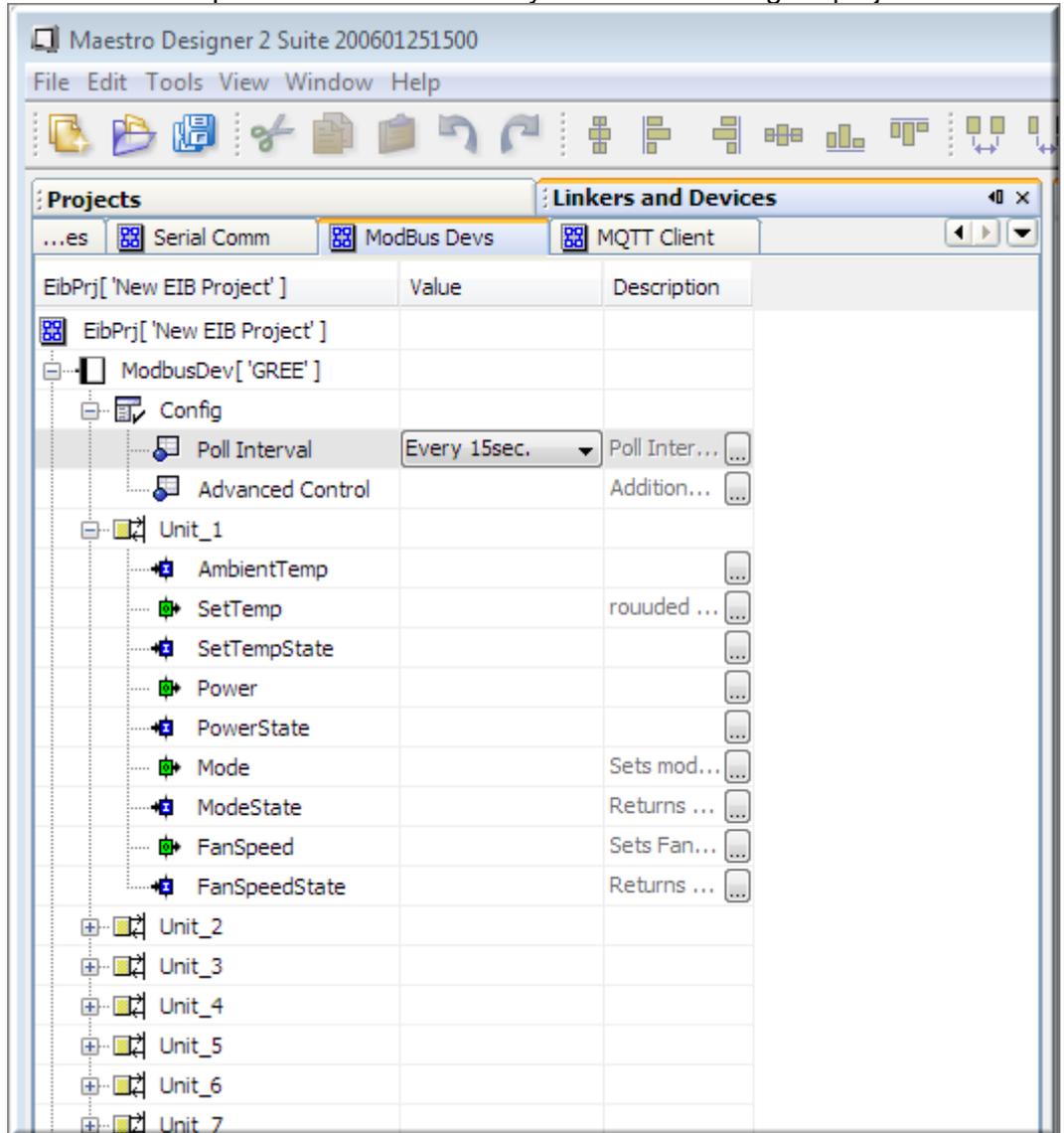
4.4.8.3. For every ModBus device - add new ModBus-Device and point on it's XML descriptor file:



4.4.8.4. Link the device to the physical media (defined on the previous steps) on this example - to the local RS485 line, and set the ModBus address of the device, 5 in this example :



4.4.8.5. Now you can set the parameters for the device and Drag and Drop it's communication points to create links on your Maestro Designer project:

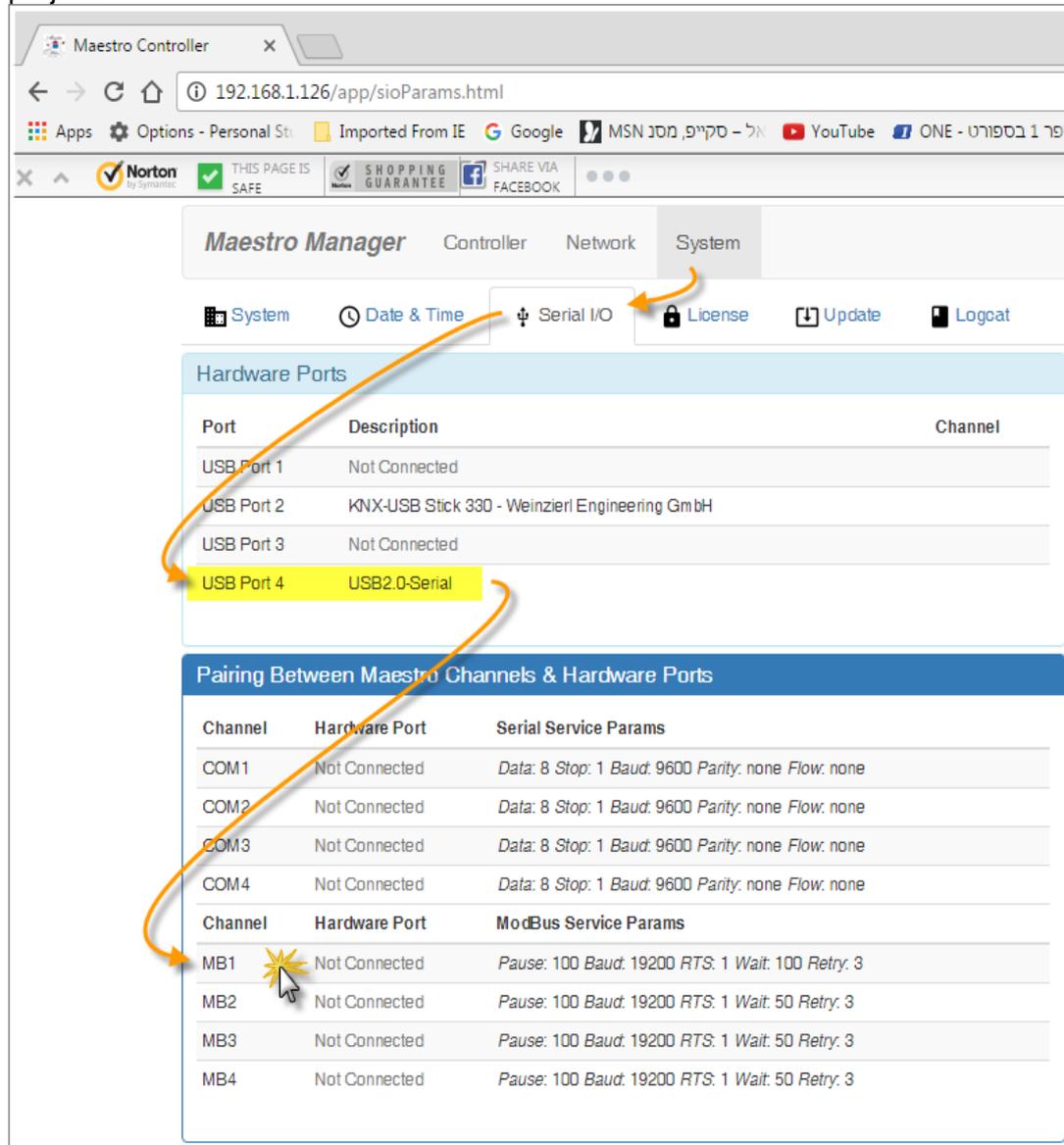


the green points are used to send (write) values to the ModBus device and the blue for data received from the ModBus device (read/feedback), red points are read/write points.

Like every linker on Maestro Designer – you can link it directly to graphic buttons, to function blocks, log it's readings and present them on graphs and even link it directly to KNX telegrams.

4.4.8.6. In case of RS485 communication:

- 4.4.8.6.1. Connect the USB-RS485 adapter to one of Maestro's USB ports
- 4.4.8.6.2. Using a browser:
- 4.4.8.6.3. open Maestro Manager
- 4.4.8.6.4. on the "Hardware Ports" list identify the USB port number where the USB – RS485 is connected
- 4.4.8.6.5. now use Pairing to bind the hardware USB port where RS485 is connected to the software channel used on your Maestro Designer project:



Use the popup dialog to set the correct USB port as well as other Communication parameters:

Edit ModBus: Channel1

Port

Pause
Pause between ModBus requests in milliseconds.

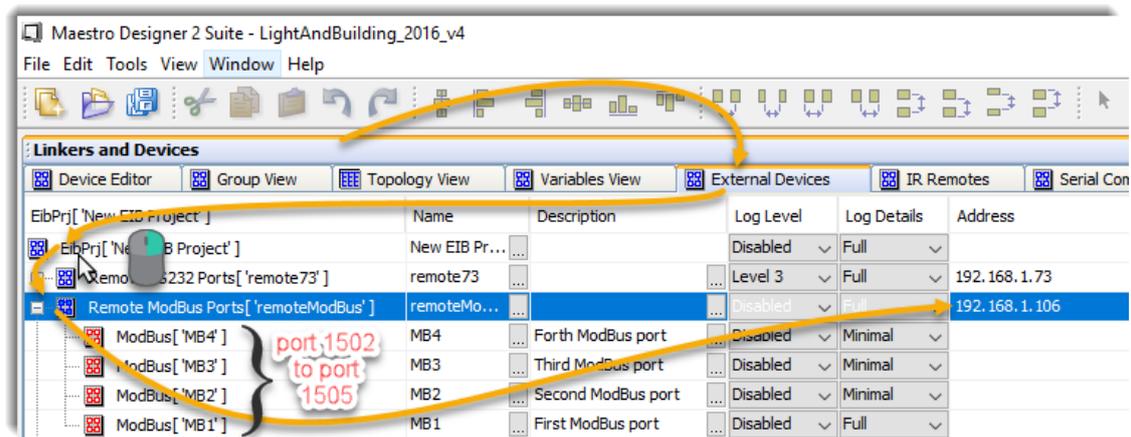
Baud
ModBus RS-485 port baud rate.

RTS
Enable RTS RS-485 data direction control (half duplex transceiver).

Wait
Response wait time in milliseconds.

Retry
Maximum number of request retries

4.4.8.7. in case of ModBus over IP:
add External-Device of the Type Modbus and Model Remote-ModBus-ports
And set the IP address of the slave device here (maestro uses fixed ports
range 1502-5):



Name	Description	Log Level	Log Details	Address
EibPrj['New EIB Project']		Disabled	Full	
EibPrj['New ModBus Project']	New EIB Pr...	Disabled	Full	
Remote ModBus Ports['remote73']	remote73	Level 3	Full	192.168.1.73
Remote ModBus Ports['remoteModBus']	remoteMo...	Disabled	Full	192.168.1.106
ModBus['MB4']	MB4	Disabled	Minimal	
ModBus['MB3']	MB3	Disabled	Minimal	
ModBus['MB2']	MB2	Disabled	Minimal	
ModBus['MB1']	MB1	Disabled	Full	

4.4.9. MQTT tab:

MQTT is light weight protocol ideal for Internet of Things (IoT) contexts.

It can support enormous amount of end points at real-time response. MQTT is based on:

- “Broker” that stores messages on hierarchical topics structure,
- “Agents” that publish the messages to the Broker and/or Subscribe to get real time updates of the messages from the Broker.

MQTT is used for single miniature sensors up to gigantic systems as Facebook and Amazon for their web services.

Now MQTT devices can gain from the wide variety of maestro’s: control functions, Logger, Graphs presentation... and from its advanced gateway capabilities to other system such as Modbus and DMX.

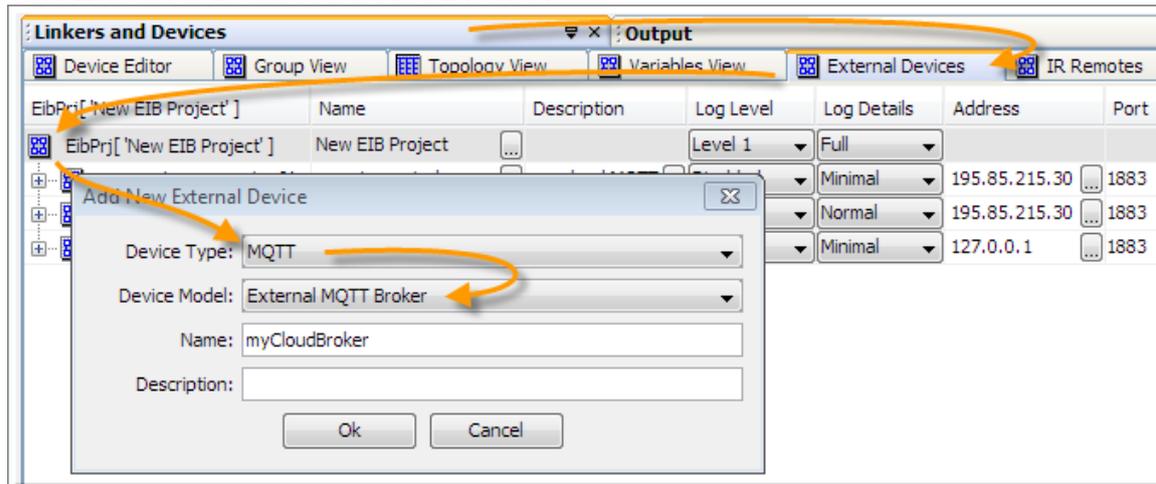
In addition, Maestro can serve as 2 way Gateway between MQTT and KNX, The benefits out of this feature are for example:

- The possibility to connect any number of remote KNX sites over IP combining them into one. Connecting KNX to SCADA and big data software for further processing and integration with external systems.
- Integrating MQTT IoT devices with KNX.

Maestro can act as Broker, Agent or both at the same time.

MQTT settings on Maestro Designer:

4.4.9.1. On External Devices add a Broker:



and set it's parameters:

Name and description – free text used for documentation and clarification.

Log level and log Details – settings used for sending Broker events to Maestro Logger.

Address and Port - IP address of the Broker, if internal broker is

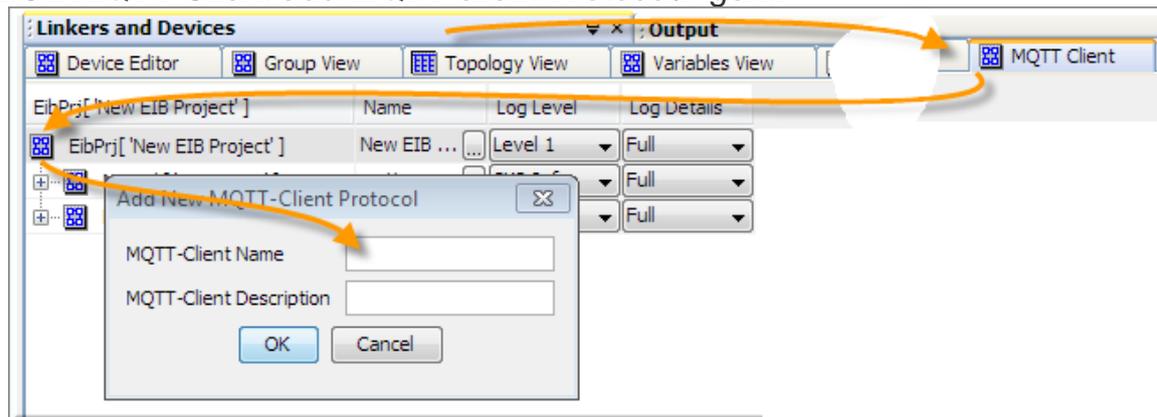
used – IP should be set to 127.0.0.1 and port to 1883

User and Password – Broker’s credentials (optional).

Client ID – optional.

Topic Prefix – the root prefix for all topics used on this connection to the Broker (optional).

4.4.9.2. On MQTT Client add MQTT client Protocol/Agent:



and set it’s parameters:

Name and Description – free text used for documentation and clarification.

Service – dropdown menu that let you select the Broker for Agents, from the list of Brokers defined on External Devices (as described on previous step).

Topic Prefix – the middle part of all topics used by this client protocol (optional).

Init Expression - Can contains Expression Language code. This code will be executed a single time when the controller starts.

You can use it to declare Static Variables and initialize them.

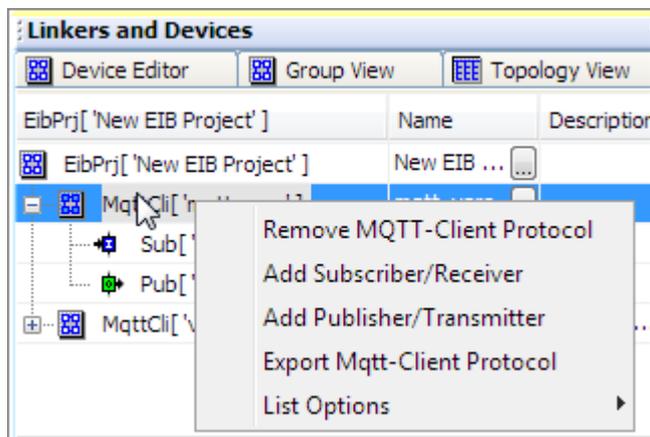
Connect Expr - Can contain Expression Language code. This code will be executed single time when the controller initializes communication with the Broker.

Idle Expression - Can contain Expression Language code.

When the Maestro’s processor is not busy, this code will be executed approximately once every second, You can use it to implement functions such as: delays and cyclic writing of data to the Broker.

Log level and log Details – settings used for sending Agent’s events to Maestro debug Logger.

4.4.9.3. Now you can start adding Subscribers (receiving data from the Broker) and Publishers (sending data to the Broker).



Publishers has the following fields:

Name and Description - free text used for documentation and clarification.

Topic - The last part of the Topic path, The complete Topic path therefore is:

[Broker Prefix][Client Prefix][Publisher Topic], and has to specify the exact path of a unique Topic.

Please notice - and have to including "/", Maestro Designer will not add or remove any characters.

Expression – can Contain Expression Language code. Any time when Publisher is assigned a value – it will execute its expression language code once.

The value will be assigned to a variable named *arg*.

to publish data use the "*return = data;*" command. If the Expression field is left empty – the value assigned to the publisher will be sent directly to the Broker.

Type – the format used for the data sent to the Broker.

Retain – When checked: a published message will be sent immediately to all current subscribers and in addition will be saved, on the Broker, and send immediately to new subscribers.

When unchecked: only current subscribers will get the new messages and then the message will be discarded from the Broker. New subscribers will get no message on subscription.

QoS – Quality of Service as defined by the MQTT standard.

Log level and log Details – settings used for sending Agent's events to Maestro Logger.

Subscribers has the following fields:

Name and Description - free text used for documentation and clarification.

Topic Filter - The last part of the topic path. The complete Topic

path therefore is:

[Broker Prefix][Client Prefix][Subscriber Topic Filter],
please notice - you have to specify the exact path including “/”,
Maestro Designer will not add or remove any characters. A filter
make use of MQTT’s Wildcard + # and \$. To get messages from
more than one publisher.

Expression – Can contain Expression Language code. Any time
when message arrives - the code will be executed once.

The value of the message will be assigned to a variable named
argv[] according to filter’s field settings.

If no expression is set - The value of the message will be
assigned to the Subscriber.

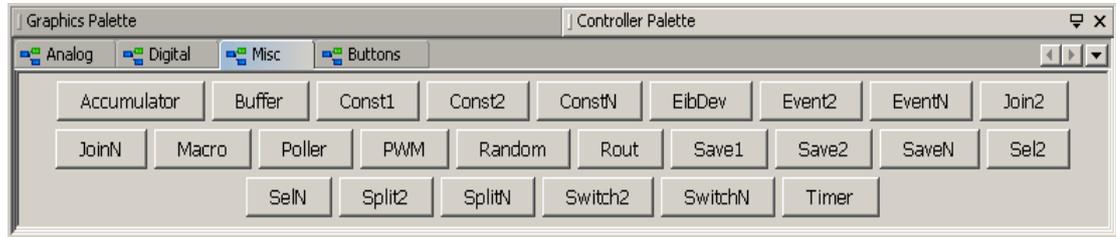
Filter – can contain a regular expression. The regular expression
examine incoming messages, identify parts that match a defined
string patterns and pass selected parts to the Expression via
argv[] variable. If this field is left empty – the complete message
will be past to the Subscriber.

Drop - when unchecked - all incoming messages will be passed to
the subscriber.

When checked - only incoming messages different form last
messes will be passed to the subscriber (“on change” only).

Log level and log Details – settings used for sending Agent’s
events to Maestro Logger.

4.5. Controller Palette window:



Controller Palette, Misc tab

This window contains the **Blocks** for the virtual controller block diagrams. Every Tab holds **Blocks** of common categories.



Controller Palette, Buttons tab

The *Buttons* tab contains **Button Blocks**. These **Blocks** represent **Buttons** that are used in the **Graphic Pages**. Every **Button Block** can be used, on the controller, only once. When a **Button Block** is used on a **Controller Page** it will not be available on the *Buttons* Tab. **Blocks** on *Buttons* tab are sorted by their alphabet name. Hover above a **Function-Block** to open a label with its short description. On chapter 10 "[Function-Blocks](#)" you can find the list and full description of all **Function-Blocks**.

4.6. Graphics Palette window:

This window contains predesigned **Buttons** ready to use on the **Graphic Pages**.



Graphics Palette, Analog tab

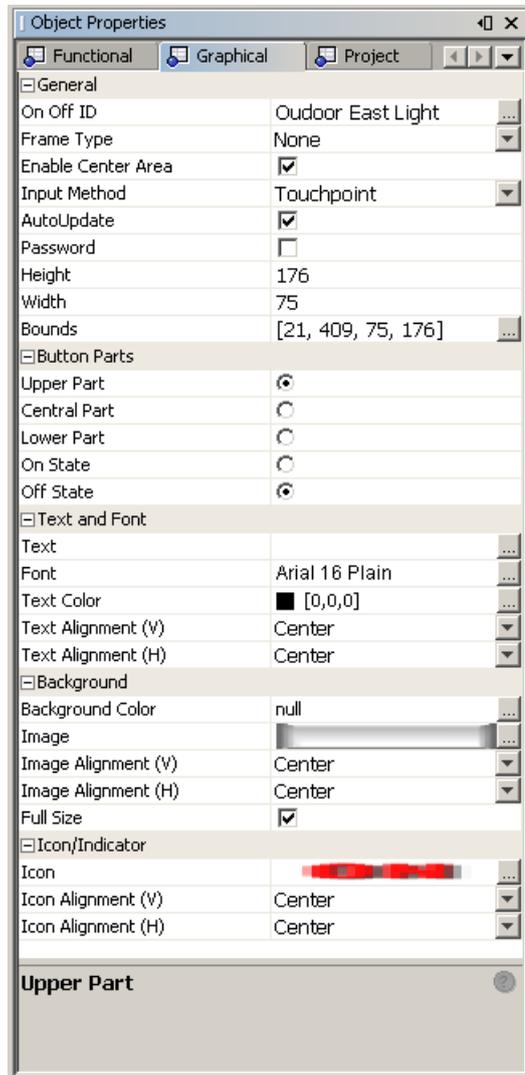
Hover above a **Button** to open a label with its short description.

To place a **Button** on the **Graphic Page** – select the **Button** image from the *Graphics Palette* and then click on the **Graphic Page**.

On Chapter 9 "[Buttons](#)" you can find a list and full description of all **Buttons**.

4.7. *Object properties* window:

The *Object properties* window will open automatically any time you double click on an **Object**. You can open it also by using *Window* menu => *Object properties*.



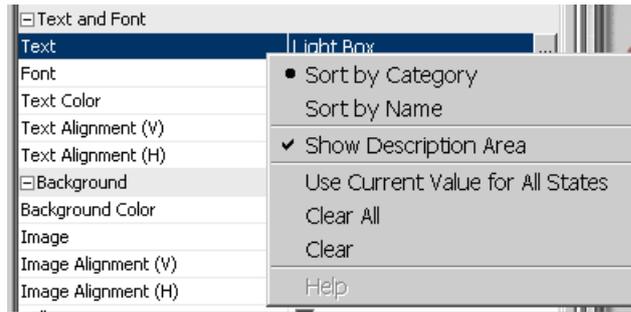
Object properties window / Graphical tab

The *Object Properties* window contains a few tabs:

4.7.1. *Graphical* tab:

Once you select **Graphic Object/s**, you can use the *Graphical* tab to present and edit its/their graphic properties.

4.7.1.1. Right click menu is available for this tab:



Graphical tab right mouse button menu

4.7.1.1.1. *Sort by Category*:

Choose this option to sort the properties by their category, This sort places small letters prior to capital letters.

4.7.1.1.2. *Sort by name*:

Choose this option to sort the properties by Alphabetic order. This sort places small letters prior to capital letters.

4.7.1.1.3. *Show description area*:

Click on it to show/hide the description area. The description area is located at the lower part of the *Graphical* tab:



Description area

The description area presents useful information regarding the currently selected property.

4.7.1.1.4. *Use Current value for All States*:

Use this command to assign the current value of the property to all **Button** states.

4.7.1.1.5. *Clear All:*

Use this command to clear the property's value from all **Button** states.

4.7.1.1.6. *Clear Value:*

Use this command to clear the property's value.

4.7.1.2. *Graphical properties tab areas:*

The graphic Tab has two areas:

☺ you can find the detailed description for all graphic properties in Chapter 9 "[Buttons](#)".

4.7.1.2.1. *Properties area:*

This is the upper section of the tab. It includes a list of properties arranged in a table. The left side of the table includes the name of the property, the right side holds the input field where you can view and edit the value assigned to the property.

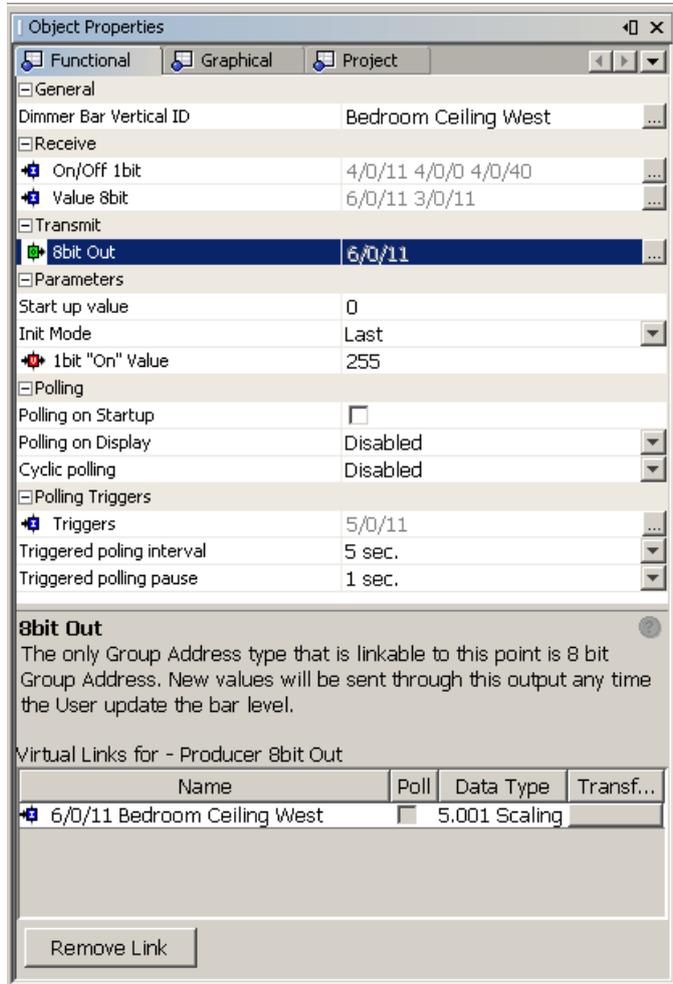
4.7.1.2.2. *Description area:*

Is the lower section of the tab, it presents useful information regarding the currently selected property.

☺ you can show/hide this part by using the right click menu

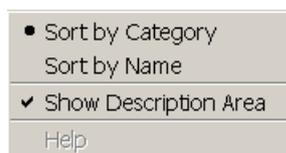
4.7.2. Functional tab:

Once you select a **Graphic Object(s)** or controller **Block**, you can use the *Functional* tab to present and edit the **Object's** functional properties.



Functional tab

4.7.2.1. Right click menu is available for this tab:



Functional tab right mouse button menu

4.7.2.1.1. Sort by category:

Choose this option to sort the properties by their category. This sort places small letters prior to capital letters.

4.7.2.1.2. *Sort by name:*

Choose this option to sort the properties by their name. This sort places small letters prior to capital letters.

4.7.2.1.3. *Show description area:*

Click on this option to show/hide the description area.

4.7.2.2. *Functional* tab areas: the functional tab has three areas:

4.7.2.2.1. *Properties area:*

This is placed at the upper area of the tab. It includes a list of properties arranged in a table. The left side of the table includes the name of the property and the right side is the input field where you can view and edit the value of the property.

4.7.2.2.2. *Description area:*

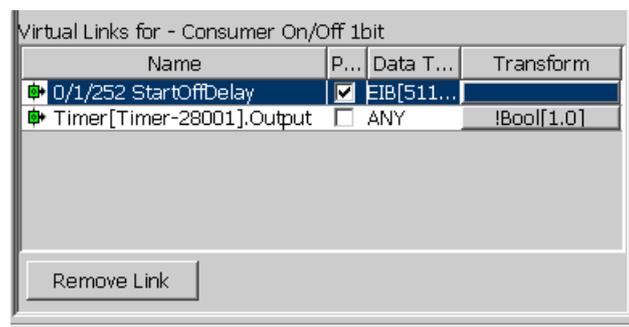
This is located in the lower part of the Tab. It presents useful information regarding the currently selected property.



Description area

☺ you can show/hide this part by using right click menu.

4.7.2.2.3. *Virtual Links for window:*



Links window

The Links window shows you a table including the list of the **Links** made for the selected **Point**. Every line in this table describes one **Link**. The description is combined from 5 sections (arranged in columns from left to right):

4.7.2.2.3.1. Icon:

Shows if the **Link** is an **Input** to the **Object**  or an **Output** .

4.7.2.2.3.2. Name:

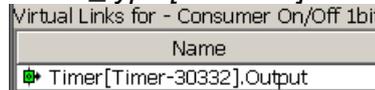
The name of the linked **Point**.

- When the linked **Point** is a KNX Group Address – the KNX Address and its name will be shown:



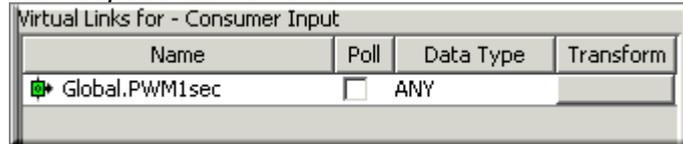
- When the linked **Point** is a **Point** of a **Block**, the description of the **Point** will be presented in the following format:

Block_type.[Block ID].Point :



- When the linked **Point** is a **Point** of a **Variable**, the description of the **Point** will be presented in the following format:

Namespace.Name:

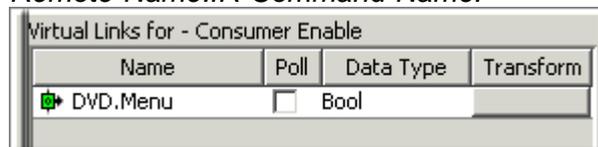


- When the linked **Point** is a **Point** of a Serial Communication, the description of the **Point** will be presented in the following format:

Serial-Protocol-Name.Transmitter/Receiver-Name:

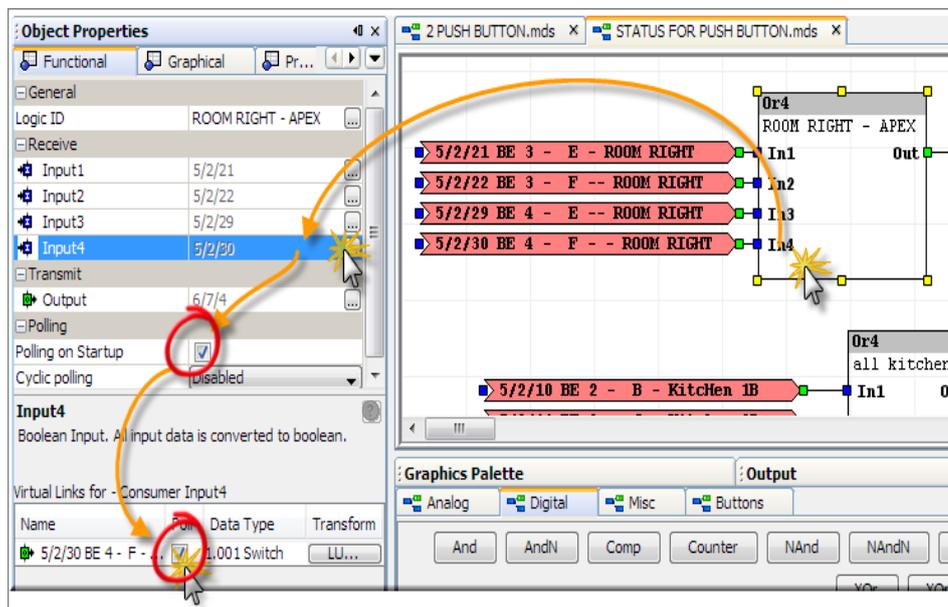


- When the linked **Point** is a **Point** of an IR communication, the description of the **Point** will be presented in the following format: *IR-Remote-Name.IR-Command-Name:*



4.7.2.2.4. Polling:

The Device can send polling telegrams to Group Addresses. A check box enables you to define whether the selected **point** will be used for polling and shows whether polling is currently enabled. Only one of all **links** made to an input can be used for polling.



☺ Polling properties are set from *Object Properties* window => *Functional* tab.

4.7.2.2.5. Data Type:
Presents the Data type of the **linked** point.

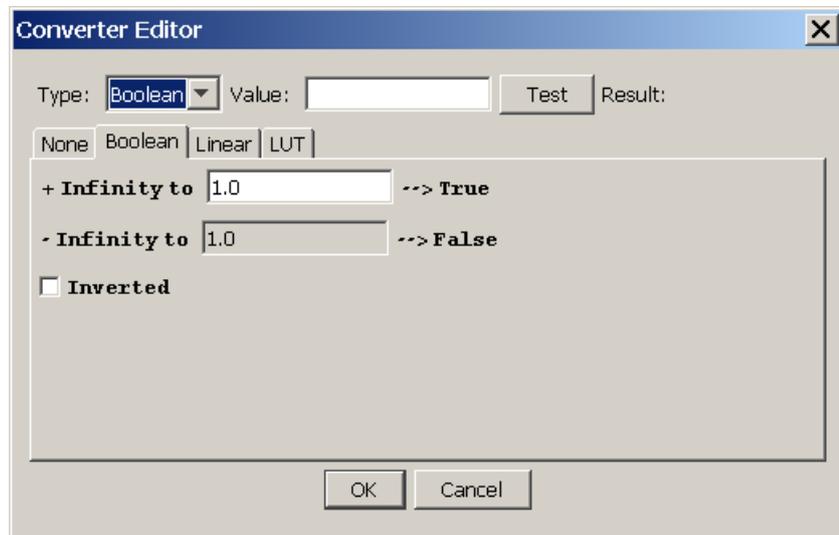
4.7.2.2.6. Transform:

Enables you to view and set a transformation between input and output values of the **link**. Click on the transform button to open the *Converter Editor*. On the upper part of the *Converter Editor* window is an simulator. The simulator allows you to test the result of the transformation for different input values.

4 different setting for transformations are available:

4.7.2.2.6.1. None:
No transform

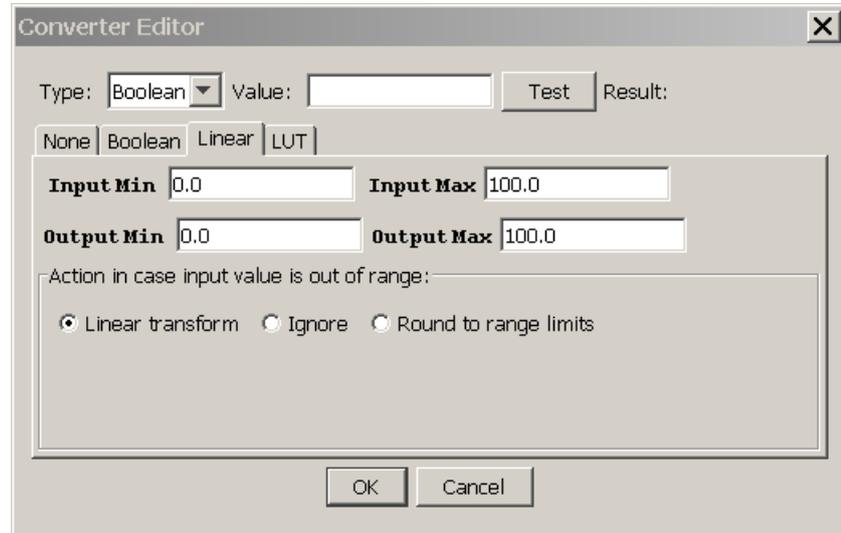
4.7.2.2.6.2. Boolean:



Boolean transformation dialog window

Using this transformation you can define a threshold value. If an input value is equal or greater then this value, the output will receive the Boolean value "True", if the input value is less then threshold the output will receive the Boolean value "False" . Checking the Inverted checkbox will invert the output value.

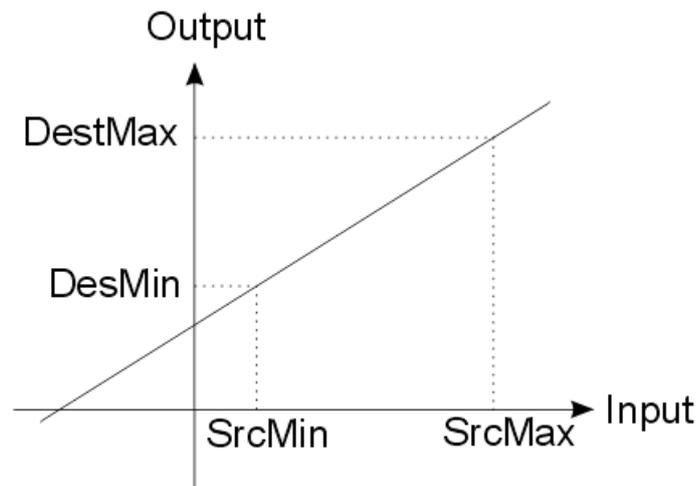
4.7.2.2.6.3. Linear.



Linear transformation dialog window

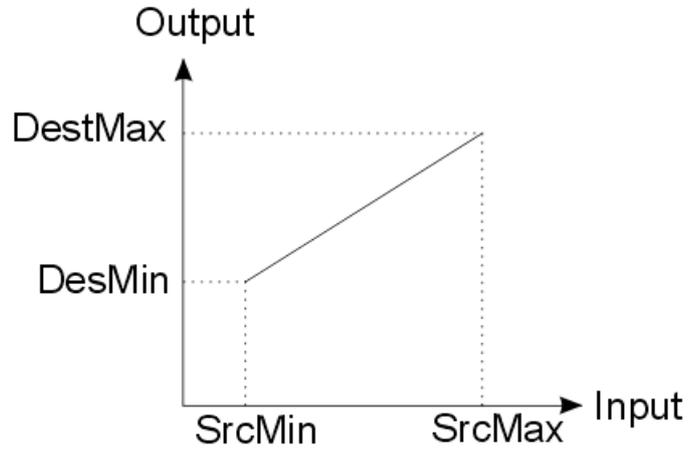
Using this transformation, you define a linear transformation between the input to the output of the **Link**. This transformation allows you to choose one of three options of handling input values that are less than SrcMin or Greater than SrcMax:

- 1) *Linear transform* – use liner transformation for all ranges:



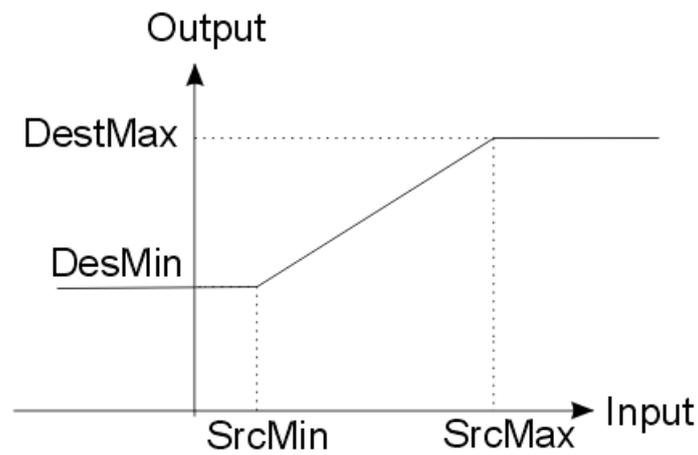
Linear transformation – "Linear transform"

2) *Ignore* – when the input value is out of range, the Maestro will not take any action, so the output value will remain unchanged:



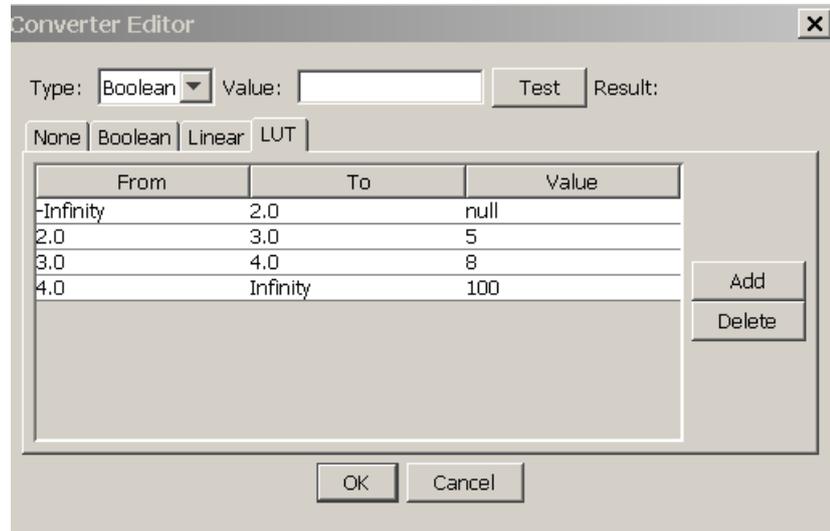
Linear transformation – "Ignore"

3) Round to range limits – when the input value is equal or less than SrcMin, the output will receive the value of DestMin. When the input value is SrcMax or greater, the output will receive the value of DestMax:



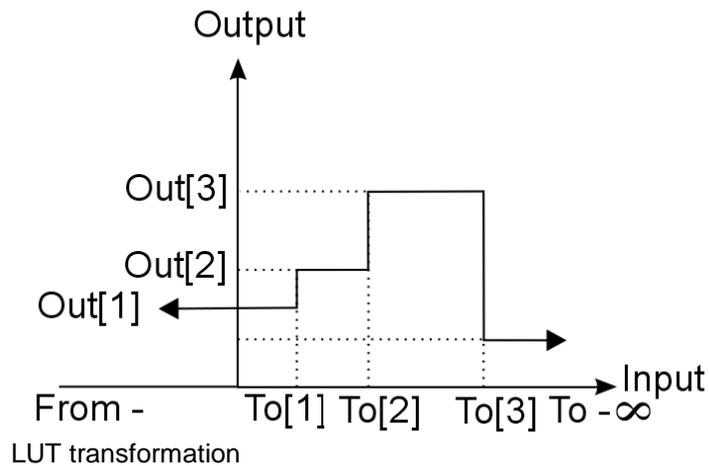
Linear transformation – "Round to range limits"

4.7.2.2.6.4. *LUT:*
Look Up Table



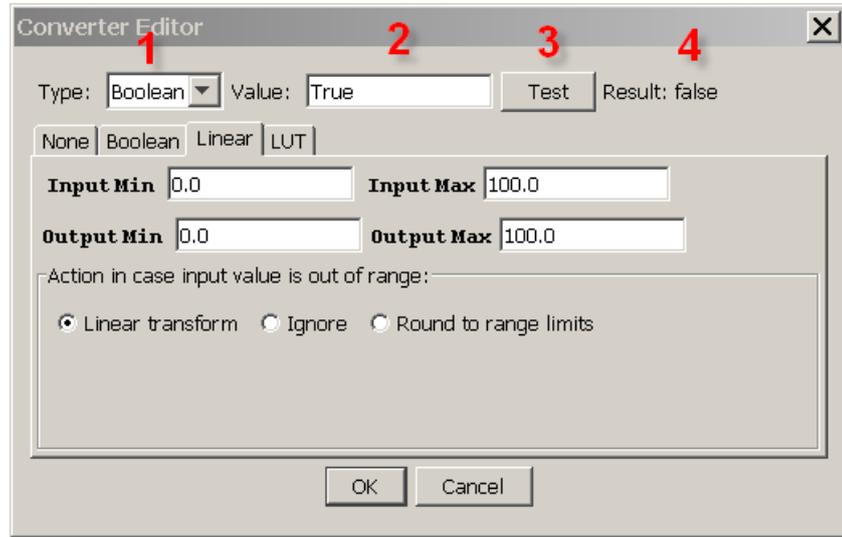
LUT transformation dialog window

This transformation enables you to divide the input range into segments, and to define an output value for every segment. When you leave the value field empty or type “null” on a range of incoming data – the LUT transform will ignore incoming data in the specified rang.



Simulator:

After setting the parameters for a transformation you can test if it is working according your expectations using the simulator:



Test your transformation using the simulator

Flow this steps to test the transformation

- 1) Select from the dropdown menu the type of incoming data
- 2) Type the value of incoming data (it must match the incoming data type)
- 3) Click the test button
- 4) Find out the result of the transform

4.7.2.3. Property types of *Functional* tab:

There are four types of properties:

4.7.2.3.1. Function Inputs:

These are **Inputs**. They have the **Input** icon:  and you can Link them to **Linkers** and to outputs of the controller's **Blocks**.

4.7.2.3.2. Function Outputs:

These are **Outputs**. They have the **Output** icon: . and you can Link them to **Linkers** and to outputs of the controller's **Blocks**.

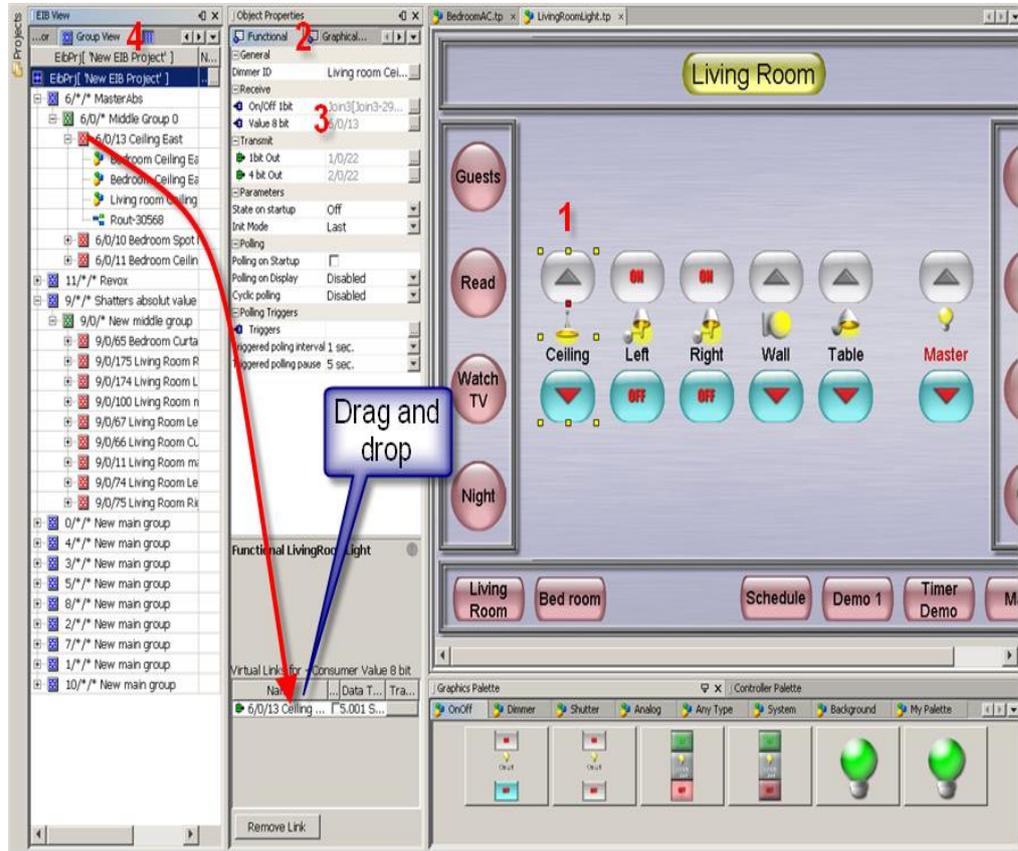
4.7.2.3.3. Unlikable Parameters:

The values of these points are used as parameters by the **Object** algorithm. Their value is constant and they are not linkable.

4.7.2.3.4. Linkable Parameter:

The values of these **points** are used as parameters by the **Object** algorithm. They have the **Linkable Parameter** icon: . These points act as **Input** as well as **Output**. You can **Link** them to any other controller point and **Linker**.

4.7.2.4. Creating a Link to a Linker:



Creating a Link to a Group Address

- 1) Select an **Object** from the *Graphic Editor* window
- 2) Open *Object Properties* window > *Functional* tab
- 3) Select the **Point** you wish to **Link**.
- 4) Open the **Linkers and Devices** window > *Group view* (or any other linker view) tab
- 5) Find the Group Address you want to Link and drag and drop it on the links window placed at the lower part of the *Functional* tab.

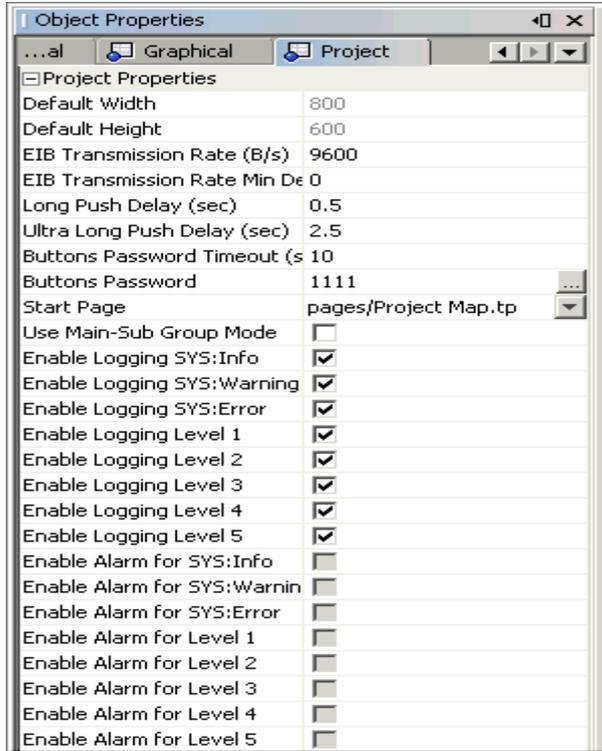
- ☺ There are some restrictions for **linking**:
 - **Linked** Group Addresses must have an EIB Data Type.
 - When **Linking** a Group Address to a **Specialized button** - the **Linked** Group Address and the **Button input/Output** must have the same KNX Data Type.

4.7.2.5. Removing a Link:

- Select the **Link** from the links window.
- Click on the *Remove Link* button on the bottom of the *Functional Properties* tab.

4.7.3. Project properties tab:

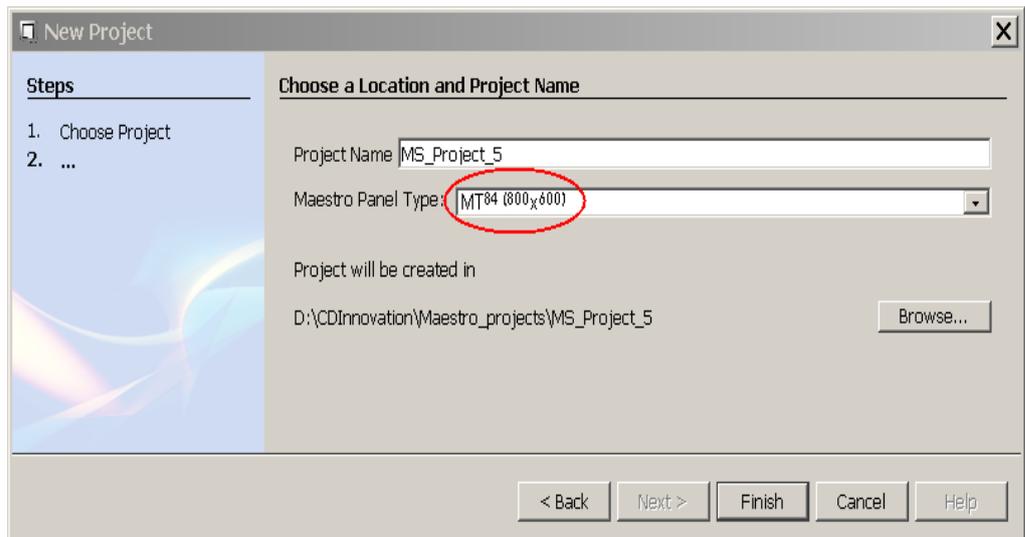
This tab allows you to set the global properties of the project:



Project properties tab

4.7.3.1. Default Width & Default Height:

This is the width and height of the project's graphic display in pixels. This value is set by the wizard that starts the project according to the Maestro hardware used for the project, and cannot be changed later.



New project wizard defines the display Width & Height

4.7.3.2. *EIB transmission rate (B/b):*

The Maestro implements the "Token Bucket" algorithm. This technique allows The Device to transmit using/during bursts of EIB telegrams exercising maximum bus rate, but over the long term the transition rate is limited to the value of this parameter.

In general: When the EIB bus is not loaded The Device will use maximum bus rate. When the Device recognizes that the bus is too loaded for some time, it will gradually reduce its telegrams transition rate over the bus. When bus traffic is reduced - the device will increase its maximum telegrams transmission rate.

Do not change the default value of 9600 unless you experience problems caused by heavy telegrams traffic over the bus.

4.7.3.3. *EIB Transmission rate Min Delay (ms):*

This property defines a minimal delay between EIB telegrams sent by **The Device**.

4.7.3.4. *Long Push Delay (sec):*

The Device will consider the push of a **Button** as a *Long Push*, if it is pressed continuously for a period of time equal or longer then the time set for this parameter. *Long Push* is used by **Buttons** such as Dimmer **Button** and Shutter **Button**.

4.7.3.5. *Ultra long push Delay (sec):*

The Device will consider the push of a **button** as an *Ultra Long* push if it is pushed continuously for a period of time equal to or longer then the time set by this parameter. *Ultra Long* Push is used by the Macro **Button** to bring up the Macro Editor screen.

4.7.3.6. *Buttons Password time out:*

The time duration from entering a valid password (or from using a password enabled button after enabling it by password) to the invalidation of the password entry.

4.7.3.7. *Buttons Password:*

This property let you set the password for the password enabled buttons.

4.7.3.8. *Start Page:*

This property defines the first page that will be presented on **The Device** (and on *Web Control*) when it re-starts after power down or reboot.

4.7.3.9. *Use main-Sub Group Mode:*

This parameter defines if 2 or 3 level group address will be used.

4.7.3.10. *Enable logging [log level]:*

Set the checkbox to enable logging events of the selected log level.

When this checkbox is unchecked – no events of the log level will be logged.

4.7.3.11. *Enable alarm – this function is not supported yet.*



4.8. Output window

The output window is used by Maestro Designer to present outcome of software's actions such as: the result of project tests and logged records (on *run* mode) .

5. Toolbar:

The toolbar includes Shortcut buttons to most used functions:



Hover the mouse over a toolbar button to see the name of the button. Here is a description of the buttons functionality:

- 5.1.  New project:
Is a short cut to the menu command - *File > New Project*
- 5.2.  Open project:
Is a short cut to the menu command - *File > New Project*
- 5.3.  Save All:
Is a short cut to the menu command - *File > Save All*
- 5.4.  Cut:
Is a short cut to the menu command - *Edit > Cut*
- 5.5.  Copy:
Is a short cut to the menu command - *Edit > Copy*
- 5.6.  Paste All:
Is a short cut to the menu command - *Edit > Paste*
- 5.7.  Undo:
Is a short cut to the menu command - *Edit > Undo*
- 5.8.  Redo:
Is a short cut to the menu command - *Edit > Redo*
- 5.9.  Align vertical Left:
Align the selected graphic **Objects** to the left. Relative to the last selected **Object**.
- 5.10.  Align vertical Center:
Align the selected graphic **Objects** center vertically. Relative to the last selected **Object**.

- 5.11.  Align vertical right:
Align the selected graphic **Objects** to the right. Relative to the last selected **Object**.
- 5.12.  Align horizontal Top:
Align the selected graphic **Objects** to the Top. Relative to the last selected **Object**.
- 5.13.  Align horizontal Center:
Align the selected graphic **Objects** center horizontally. Relative to the last selected **Object**.
- 5.14.  Align horizontal bottom:
Align the selected graphic **Objects** to the bottom. Relative to the last selected **Object**.
- 5.15.  Distribute horizontal Center:
Distribute the centers, of the selected graphic **Objects**, horizontally using equal intervals.
Relative to the last selected **Object**.
- 5.16.  Distribute horizontal spaces:
Distribute the selected graphic **Objects**, horizontally using equal spaces between them.
Relative to the last selected **Object**.
- 5.17.  Distribute horizontal Left:
Distribute the left edges, of the selected graphic **Objects**, horizontally using equal intervals.
Relative to the last selected **Object**.
- 5.18.  Distribute horizontal Right:
Distribute the right edges, of the selected graphic **Objects**, horizontally using equal intervals. Relative to the last selected **Object**.
- 5.19.  Distribute vertical Center:
Distribute the centers, of the selected graphic **Objects**, vertically using equal intervals.
Relative to the last selected **Object**.
- 5.20.  Distribute vertical spaces:
Distribute the selected graphic **Objects** vertically using equal spaces between them.
Relative to the last selected **Object**.
- 5.21.  Distribute vertical bottom:
Distribute the bottom of the selected graphic **Objects**, vertically using equal intervals.
Relative to the last selected **Object**.

- 5.22.  Distribute vertical top:
Distribute the top of the selected graphic **Objects**, vertically using equal intervals.
Relative to the last selected **Object**.
- 5.23.  Arrow:
Arrow mode allows you to select, move, and resize **Objects**.
- 5.24.  Check project:
Tests the Maestro Designer project for errors.
- 5.25.  Print:
Is a short cut to the menu command - *File > Print*
- 5.25.1.  Run:
Click this button to start the simulation software. This software will create a pop up window that will simulate the **Device's** graphic dynamics and functionality.
- 5.25.2.  Zip command:
Zip command will prepare a downloadable Zip file of the project. The file will be stored in the "dist" directory of the project and its name will be [project name].Zip. The zip file includes all the directories and files of the project, excluding unused resources, for example image placed on the Images directory of the project but not used in any graphic page – will not be included in the Zip file.

6. Automatic data conversion:

In Maestro, 2 families of Data types are used:

- 1) KNX Data types: as defined by the KNX organization.
- 2) Maestro Data Type: to learn about Maestro Data Types refer to the [Expression Language chapter](#) of the manual.

Automatic **Data Type** conversion:

Any time that data is received by a Maestro **input point** the Maestro automatically cast the received Data Type to the Data Type of the **input point**.

When doing so, the Maestro tries to preserve the value received.

For example: *integer* Data Type having value 64 arriving **input point** of the type KNX 5.001 percent, will be converted to 5.001 percent Data Type having the value 64 (and not 25%).

Here are some examples for data value conversions when incoming Data does not have the same Data Type of the **input point**:

Incoming Data Type	Input point data type	Incoming value	Input point value
KNX 5.001 percentage	int	128	128
KNX 5.001 percentage	Boolean	128	true
KNX 5.001 percentage	Boolean	0	false
KNX 5.001 percentage	Double	128	128.000
KNX 1.001 on/off	Boolean	1	true
KNX 1.001 on/off	Boolean	0	false
KNX 1.001 on/off	int	1	1
KNX 1.001 on/off	int	0	0
Int	KNX 1.001 on/off	255	1
Int	KNX 1.001 on/off	0	0
Int	KNX 1.001 on/off	50	1
Int	KNX 1.001 on/off	-50	1
KNX 9.001 temperature	int	23.56	23
KNX 1.001 on/off	KNX 5.001 percentage	1	1
KNX 1.001 on/off	KNX 5.001 percentage	0	0
Double	KNX 5.001 percentage	55.45	55
Boolean	KNX 5.001 percentage	true	1
Boolean	KNX 5.001 percentage	false	0
Double	Int	55.45	55
KNX 9.001 temperature	KNX 5.001 percentage	24.5	24
Text	KNX 1.001 on/off	"ON"	1
KNX 9.001 temperature	Text	24.5	24""
KNX 1.001 on/off	Text	1	"1"

Examples for data value conversions

For some of the **linkers**, Maestro enables the definition of *Any* as Data Type (normally it's the default value). In this case, the Maestro will cast the Data Type and value of the incoming data to the **input point**. Normally this is the best choice because, this way Maestro can keep the original Data Type and value without any change.

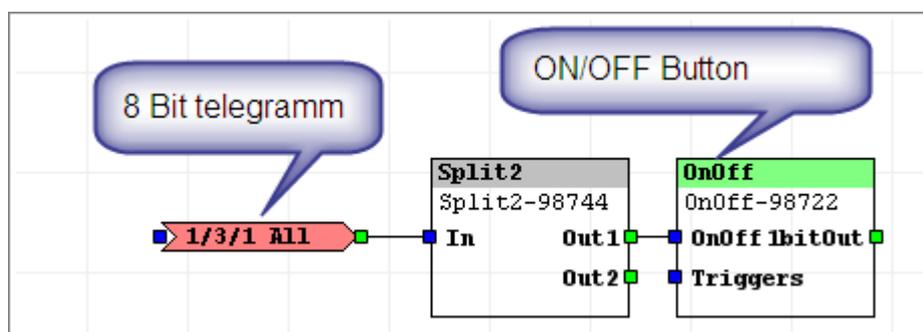
Group Addresses have only KNX Data Types, but other Maestro **linkers** such as **Variables** and Serial Communication *transmitters* can have any of the Maestro's Data Types.

The automatic Data Type conversion enables you to mix different KNX Data Types and other Maestro Data Types conveniently and seamlessly without worrying about their Types.

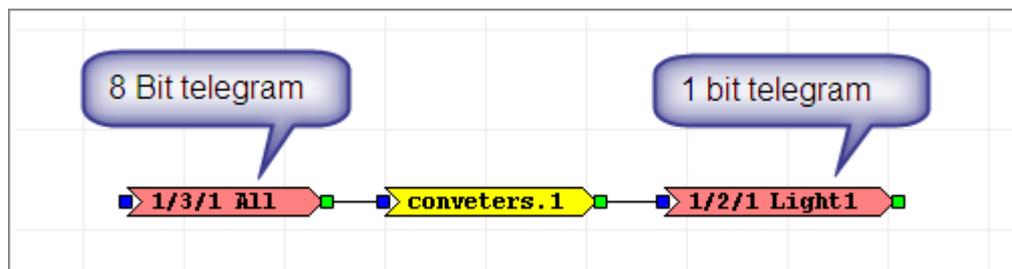
In cases where the automatic conversion of values is not satisfactory you can always use the [transform function](#).

An exception is *On/Off*, *Dimmer* and *Shutter* **buttons**:

The **input** and **output points** of *On/Off*, *Dimmer* and *Shutter* **button** does not perform automatic Data Type conversions. If you want to use data conversion on the **input** and **output point** of these buttons you have to do it indirectly by adding other **objects** for example a **variable** or a **split function block**. Before or after the **button**.



Forcing Data Type conversion to an on/off button using Split function block



Forcing Data Type conversion between Group Addresses using a Variable

7. Temporary data, persistent data:

The Maestro has persistent data storage.

The data of the persistent storage is stored on the Maestro's flash disk and therefore is not erased when the Maestro is turned off or even when its power is cut.

Persistent storage includes the following data:

- *Scheduler* information
- *Macro* information
- Last state of **buttons**
- Values of the *save* function block
- passwords

Remark:

It can take up to 10 minutes between the time of an event taking place until it is saved on the flash.

When downloading a project from the Maestro Designer to the **Maestro device**, the Maestro designer pops up a window that asks whether to delete persistent data or not to delete persistent data.



Persistent data dialog

if "*Delete persistent data*" is selected: persistent values on the device are erased and initialized by the values from the Maestro Designer project.

If "*Do not delete persistent data*" is selected: persistent values on the device are kept and the values from Maestro designer are not passed to the **Maestro Device** (unless it is the first time you download the project to the **Maestro Device**, in this case the Device has no valid persistent data for the project and therefore it will use the values from the downloaded Maestro Designer project).

Exercise caution when downloading: if macros and schedules were edited directly from the **Maestro Device** (by the end user for example) using "Delete persistent data" will erase those settings !!! and reset them to the settings of the Maestro Designer project.

All other information such as: Group Address status, variable values, function block input and output values are temporary, they are stored on the RAM only and are not saved when the Maestro is off; if their initial values are important – you have to initialize them using **Constant Function Block** or a variable that is set to transmit on startup.

8. **Device Set up page:**
for MS3 and MTS3 or older products

General Information Product Name: <input type="text" value="Maestro Controller"/> Firmware Version: <input type="text" value="TP1.8.918 EIB1.10.927 BUS1.6.242"/> Identification Code: <input type="text" value="UMDz-XuKf-CfcG-aATL-IFxn-tq9H-hbrW-w05R"/> Serial Number: <input type="text" value="teble_tests_1111_0000"/> License Code/Status: <input type="text" value="cX4d-zTa8-Eyj4-aXrE-YvIC-SEqG-wve4/OK"/> Licensed To: <input type="text" value="user"/> <input type="button" value="Recalibrate"/> <input type="button" value="Restart"/> <input type="button" value="Power Off"/>	Clock & Regional Settings Greenwich Mean Time: <input type="text" value="06:58:44"/> Greenwich Mean Date: <input type="text" value="יום רביעי, 19 דצמבר 2012"/> Locale and Timezone: <input type="text" value="LOC+1: GMT+2:00 with DLST rules."/> Local Time and Date: <input type="text" value="19-12-2012 09:58:44"/> GPS Coordinates: <input type="text" value="Latitude=32.05 Longitude=34.46"/> Sunrise and Sunset: <input type="text" value="Sunrise at 07:39; Sunset at 17:39"/> <input type="button" value="Position"/> <input type="button" value="Locale"/> <input type="button" value="Time/Date"/>
Project & UI Settings Project Name: <input type="text" value="New EIB Project"/> Project Version: <input type="text" value="537 from 18-12-2012 12:43"/> KNX/EIB Physical Addr: <input type="text" value="15.15.100"/> KNXnet/IP Server: <input type="text" value="Enabled (Ctrl=UDP:3671;Data=UDP:50000)"/> Password & Validity: <input type="text" value="Password is Disabled!"/> Button Push Delays: <input type="text" value="Long=0.5sec. Ultra=2.5sec."/> Sound Settings: <input type="text" value="Master=0% Capture=35% MicBoost=0"/> Brightness & Contrast: <input type="text" value="Bright=96% Contrast=96%"/> Screen Saver: <input type="text" value="Timeout=1800sec. Cycle=300sec."/> <input type="button" value="S- Saver"/> <input type="button" value="Brightness"/> <input type="button" value="Sounds"/> <input type="button" value="Delays"/> <input type="button" value="Password"/> <input type="button" value="KNX/EIB"/>	Network Settings Network Mode: <input type="text" value="IP"/> IP Address: <input type="text" value="192.168.1.71"/> Network Mask: <input type="text" value="255.255.255.0"/> DNS Server IP: <input type="text" value="194.90.1.5"/> Default Gateway IP: <input type="text" value="192.168.1.1"/> Host Name: <input type="text" value="tpctrl"/> Remote User/Pass: <input type="text" value="user/*****"/> <input type="button" value="User/Pass"/> <input type="button" value="Static IP"/> <input type="button" value="Mode"/> Main Actions <input type="button" value="Macro Edit"/> <input type="button" value="Scheduler"/> <input type="button" value="Run/Back"/>

Device setup page

This page is created automatically in every project. It enables the **User** to set up the **Device's** global properties.

☺ It is recommended that every project have at least one Page Flip button that leads to the Setup page.

There is a back door to the set up page. To access the set up page – press, on four different corners of any screen within less than 5 seconds.

The Setup page is uniform for all Maestro **Devices** but some parameters are not functional on some of the devices. For example, Sound Settings is not functional on MSx devices as they do not have speakers.

The Setup page is reachable from Web Control. On web control, the relevant, setup page functions are operative except System Macro Editor.

Set up page is divided to areas:

8.1. General information:

Has the following fields:

- Product Name: for CDI use only.
- Firmware version: this field will automatically update any time you download a new firmware to the **Device**.
- Identification code: for CDI use only.
- serial number: is used for warranty and hardware issues.
- License code: for CDI use only.
- Licensed To: for CDI use only.

The General Information section has the following buttons:

- Recalibrate: use to recalibrate the touch screen. You must restart the **Device** for the recalibration procedure to take effect.
 - Restart: use to restart the **Device**.
 - Power Off: use to power off the **Device**.
- Always turn off/restart the device by using these buttons. Other methods can cause loss of data.

8.2. Project & UI Settings:

Has the following fields:

- Project Name: displays the, installed *Maestro Designer* project name.
- Project Version: displays the installed *Maestro Designer* project version.
- KNX/EIB Physical Addr.: displays the **Device**'s KNX Physical Address.
- KNXnet/IP server: displays KNXnet/IP mode (enabled/disabled) and IP ports.
- Password and Validity: displays the **user** password for **Graphic Buttons** and the validity interval.
- Push Button Delays: displays the Long push and the Ultra Long Push time.
- Sound Settings: displays the volume setting of the **Device**'s speakers.
- Brightness and Contrast: displays the Brightness and Contrast settings of the **Device**'s LCD.
- Screen Saver: displays the timeout settings for the **Device**'s sleep mode. The Sleep mode changes from sleep mode to a deeper sleep mode after an interval equal to "Cycle" time (in seconds). In the first mode the screen turns black. In the second mode the LCD's back light is turned off. The next two modes are related to reducing the power consumption by the graphic processing unit and other internal electronics.
 - ☉ Using the Screen Saver function extends the LCD lifetime and reduces power consumption.
 - ☉ In sleep mode the touch screen is active. When the screen is pushed - the device will turn on the display. It will also react to the push if it was on a functional graphic button!.

The Project & UI section has the following buttons:

- S. Saver: opens a dialog window that allows you to set the Screen Saver time out.
- Brightness: opens a dialog window that allows you to set the LCD's brightness (MTSxxx only).
- Sounds: opens a dialog window that allows you to set the volume level of the **Device**'s

speakers (MTSxxx only).

- Delays: opens a dialog window that allows you to set the Long push and the Ultra Long Push time.
- Password: opens a dialog window that allows you to set the **user** password for **Graphic Buttons** and the validity interval.
- KNX/EIB: opens a dialog window that allows you to set the **Device's** KNX, Physical Address and to Enable/Disable KNXnet/IP. When KNXnet/IP is enabled you can use the Maestro to tunnel KNXnet/IP communication from a remote ETS software to the local KNX bus, the Maestro support all ETS communication except Bus Monitor (group monitor is supported).

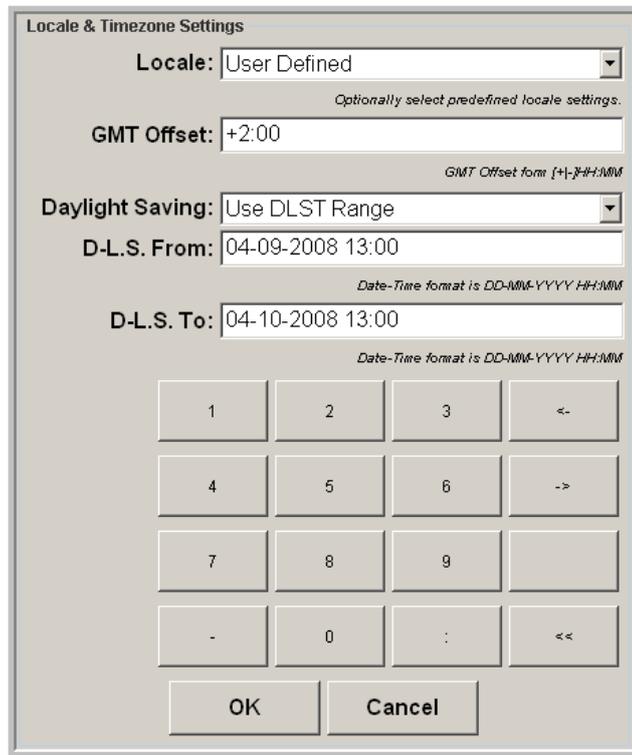
8.3. Clock & Regional Settings:

Has the following fields:

- Greenwich Mean Time: displays the internal clock time. This time must be to the same as Greenwich time in order to, correctly, calculate your local sunset, sunrise, Time and Date.
- Greenwich Mean Date: displays the internal clock date
- Locale and Timezone: Displays your clock settings i. the offset from GMT ii. Daylight saving status.
- Local Time and Date: Displays your local time and date. These are used on the scheduler and Time and Date Buttons.
- GPS Coordinates: Displays your local position. This is used for calculating the sunrise and sun set time.
- Sunrise and Sunset: Displays your local sunrise and sun set time.

Clock & Regional Settings has the following buttons:

- Position: opens a dialog window that allows you to set the Longitude and Latitude coordinates of the **Device's** location. Used for calculating the sunrise and sun set time.
- Local: opens the following dialog window and allows you to edit:



Locale & Timezone Setting dialog

Locale: Maestro utilizes Automatic clock update according to Summer Savings Time: The Maestro has a predefined list of dates and times for Summer Savings Time. If your area uses Summer Savings Time, use the dropdown menu to select your zone from the list. The Maestro will automatically change its internal clock to the correct time. If your zone is not on the list you will have to set the Summer Savings Time manually. If your area does not use Summer Savings Time, you must choose “user defined” under locale, and “disabled” under “Daylight Saving”.

GMT Offset: When *Locale* is set to *User Defined*, use this field to set the offset time from the device’s position time to Greenwich Time.

Daylight saving: When *Locale* is set to *User Defined* you have to choose daylight saving mode from the dropdown menu. The options are:

- *Disabled:* Daylight saving is not implemented (or the user changes the clock to daylight saving modes manually and they are not currently on daylight saving time)
- *Now it is daylight savings:* choose this option to set the clock manually for daylight savings time (+1h)
- *Use DLST Range:* choose this option if you want to manually set the time and date for the Start and End of daylight savings.
- *DLS from & DLS to:* use these fields to manually set the time and date for the Start and End of daylight savings.
- *Time/Date:* opens a dialog window that allows you to set the Time and the Date of the System Clock.

8.4. Network Settings:

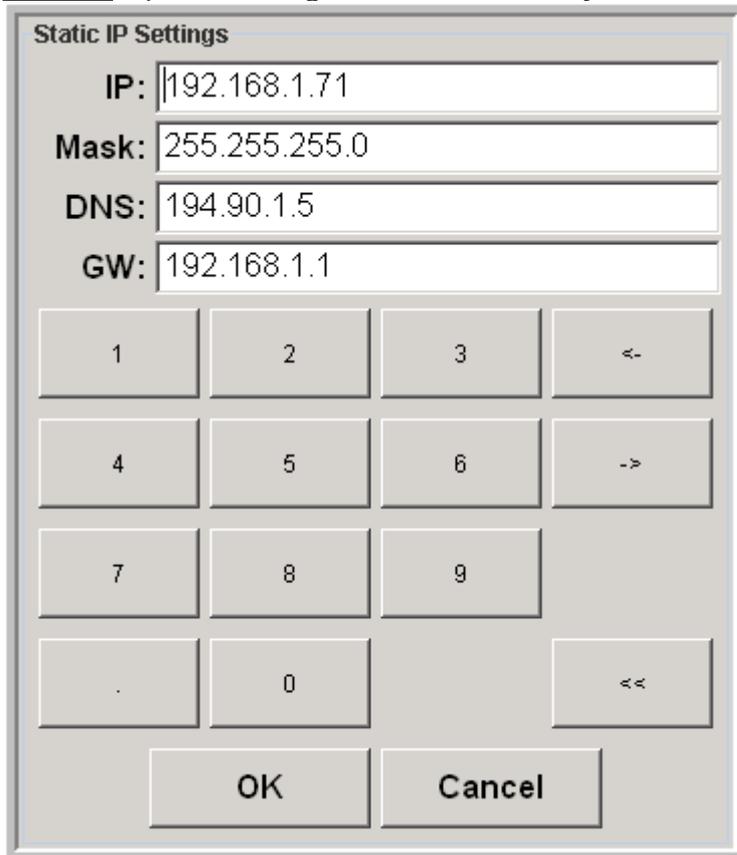
Has the following fields:

- *Network Mode*: Displays network mode.
- *IP Address*: displays the **Device**'s IP address.
- *Network Mask*: Displays network mask.
- *DNS server IP*: Displays network DNS server ID.
- *Default Gateway IP*: Displays network Default gateway ID
- *Host Name*: Displays the **Device**'s host name
- *Remote User/Pass*: Displays User name and password for Web control and App control.

Has the following buttons:

User/Pass – use it to set/disable user name and password for web control

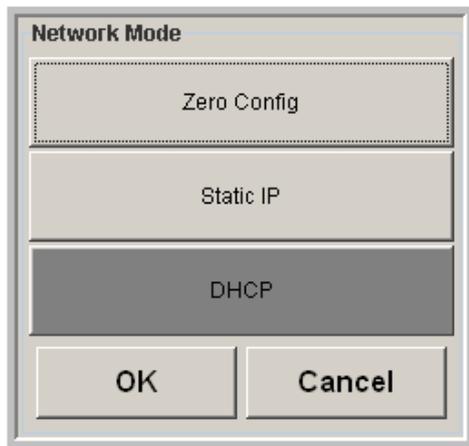
Static IP: opens a dialog window that allows you to set Static IP parameters:



The image shows a dialog box titled "Static IP Settings". It contains four input fields: "IP:" with the value "192.168.1.71", "Mask:" with "255.255.255.0", "DNS:" with "194.90.1.5", and "GW:" with "192.168.1.1". Below these fields is a numeric keypad with buttons for digits 1-9, 0, and a decimal point, along with left and right arrow buttons and a double-left arrow button. At the bottom are "OK" and "Cancel" buttons.

Static IP settings

Mode: opens dialog window that allows you Select IP network mode:



Network mode

Select one of the options:

- *Zero config* – select it when you want a direct, cross cable, physical connection between the **Device** and your PC. When you select this option the device will assign itself an IP address automatically. Use this IP address on your *Maestro designer* and/or browser to communicate with the device.
- *Static IP* – select this option when you want the device to obtain static IP address.
- *DHCP* – Use this option when you want the device to automatically request an IP address from the DHCP server.

8.5. *Main Actions*:

Main Actions has the following buttons:

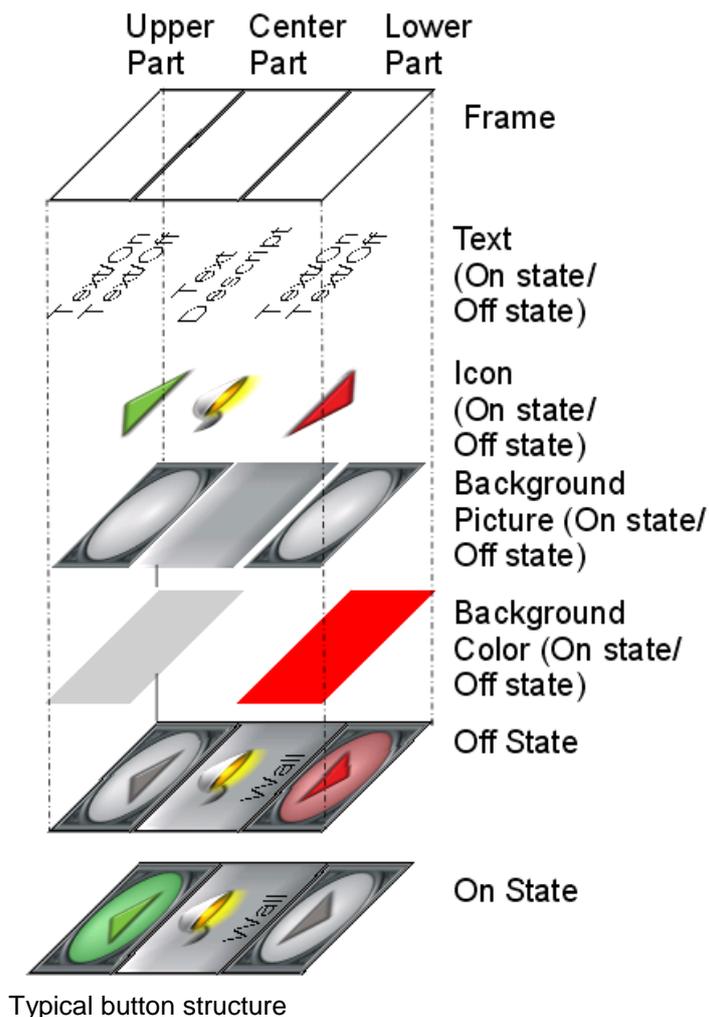
- Macro Edit: opens the System Macro Editor. System Macro Editor allows you to edit all macros of the project including macro of **buttons** and macros of **function blocks**. Unlike editing a macro via its graphic **button**, System Macro Editor works off line. This means that no events are generated during the process of editing a macro setting. The System macro editor is accessible only from the *setup* page on the device. It is not accessible from *web control*.
- Scheduler – opens the scheduler editor
- Run/Back – returns to the previous graphic page.

9. **Buttons:**

Buttons are the main GUI element, They allow the **User** to "communicate" with the **Device**. There are many predefined **Buttons** you can choose from. The **Buttons** are placed in the *Buttons Palette* window. **Buttons** are arranged in tabs. Every tab holds **Buttons** sharing the same functional category. Within a tab, the **Buttons** are separated from each other by their function, look and size. To see the **Button** type and description, hover the mouse over a **Button**. When you do so, a label displaying the **Button** name will pop up. The **Button** name consists of three codes: (i) Theme (ii) **Button** type (iii) **Button** design (when applicable)... Once you select a **Button** and place it on a **Graphic Page**. You can edit its look and functionality by setting its properties.

Buttons are made of graphic layers drawn above each other. Here is the order, bottom to top, for painting/designing a **Button**:

- Back ground color.
- Back ground image.
- Icon/Indicator OR Bar.
- Text and Ticks.
- Border line or frame.



Generally, **Buttons** follow these rules:

- When an **Event** comes through an **Input**:
The **Button's** graphic state will be updated (supplying feedback to the **User**).
The **Output**, of the **Button**, will remain unchanged.
- When the **User** pushes the **Button**:
The **Button's Output** value will be updated and an **event** will be generated through the **Output**.
The **Button's** graphic state will be updated. (in some **Buttons** this can be disabled by disabling the *AutoUpdate* property so that the button state will change only as a result of events arriving at its inputs).
As long as **Button** is pushed and only then it's frame will change its look.

Below, you will find general descriptions for all properties/parameters used by any of the **Buttons**. When you select a few buttons, the Maestro Editor will allow you to edit their common properties. This way you don't have to edit the buttons one by one.

9.1. Graphic properties:

Here is a general description of all available graphic properties. A specific graphic object (e.g. On/Off Button) uses and displays only the relevant properties in its graphic tab.

General properties:

The properties at the top of the properties table are global and are applicable for all **Button** parts.

9.1.1. *Button ID:*

This is the **Button's** unique ID, no other **Button** in the project can have the same ID. ID is used to identify and reference **Buttons** in the different parts of the software. Any time a new **Button** is added, the software will automatically assign it a unique ID number. You may edit the ID, and assign your own ID to the **Button**. If you enter an ID that already exists – the system will automatically assign it an additional serial number: "spot light", "spot light[1]", "spot light[2]....

9.1.2. *Frame Type:*

Use this property to change the frame style. A Frame is complex graphic element on the border of a **Button**. A Frame is used:

- To show the **Button** perimeter.
- As a part of the design of the **Button**.
- To provide the **User** feedback on his push/release action. Whenever a **Button** is pushed/released, the frame will change its appearance as an indication that the action is recognized by the **Device**. An exception to this is the *Line Border* – it is static and does not change its look when the button is pushed/released.

9.1.3. *Frame Color:*

When you decide to use a *Line Border* as a *Frame Type*, you can use *Frame Color* to define the Color of the border line. To select a color; open the color palette or type the name of the color directly into the dialog box. The color names you can type are: White, Light Gray, Gray, Dark Gray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, and Blue.

9.1.4. *Frame Thickness:*

When you decide to use a *Line Border* as *Frame Type*, you can use *Frame Thickness* to define the border line thickness in pixels.

9.1.5. *Central area:*

When you check this checkbox the **Button** will include a central area.



On/Off Button with central area enabled On/Off button with central area disabled

9.1.6. *Input method:*

The Input method for a **button** can be one or more of the following:

- *Disable:*

No action will be taken when the **User** pushes the button. The **Button** can be used only as part of the screen design, and for presenting feedback.

- *Touch point:*

The action that will be performed when the **User** touches this **Button** is defined by the location and characteristics of the **User** touch. See the specific **Button** functional description to find out how it reacts to the different kinds of pushes – long, short, up, down, drag...

- *Keypad:*

Any time the **User** pushes the central part of the **Button** - a keypad will pop up. The keypad will allow the **User** to set a specific value to the **Button**. The new value will become activated when the **User** pushes the OK button on the pop up keypad. Keypad will alert and prevent the **User** from entering out of range values



Pop up keypad

- *Inc/Dec:*

Push the upper part of a **Button** to increase value by "Step size", push lower part of **Button** to decrease value by "Step size".

- *Bar:*

Press or drag over the barograph area the new value will be according to the position of the last touching point.

9.1.7. *Keypad title text:*

Use this property to type the text you want for the title of the keypad popup.



Keypad title

9.1.8. *Input mode:*

Use this property to select the input mode for bar graph.

- When you choose *Jump To New Value*, the bar graph **Button** will set its value to the new value instantly.
- When you choose *step to new value* the bar graph **Button** will set its value toward the new value gradually in steps. Step size and interval are defined by the next two properties.

9.1.9. *Step Size (%):*

Use this property to define the size (%) of single step toward the new value.

9.1.10. *Steps Per Sec'* – use this property to define how many times per second the value of the barograph will step toward its new value.

9.1.11. *AutoUpdate:*

When you enable this function, the **Device** will update the **Button** state

- Automatically any time the **User** touches it
- As a result of new data coming into its **Input**.

If you disable this function, the **Button** will update its state only as a result of new data coming into its **Input**.

9.1.12. *Password:*

When you enable this function, the **User** will be asked for a password in order to perform the operation of the button. Once the **User** enters the correct password, use of the button will be valid for a period of time set on the Setup Page of the device. During this time the **User** can use the **Button**, (or any other password protected button), again with no need to reenter the password. The password is a global property for all password protected buttons in the project. The Password can be modified by the **User** in the device set up page.

9.1.13. *Height:*

This property displays and allows you to edit the **Button**'s vertical dimension in pixels.

9.1.14. *Width:*

This property displays and allows you to edit the **Button**'s horizontal dimension in pixels.

9.1.15. *Button Boundaries:*

This property displays and allows you to edit the **Button** boundaries. Using it, you can resize and reposition the **Button**.

☺ you can use  and  to copy and paste the value of these properties.

Buttons parts properties:

9.1.16. *Upper part/Lower part/Central part:*

Use this property to select the **Button** part you want to edit.

9.1.17. *On state/off state:*

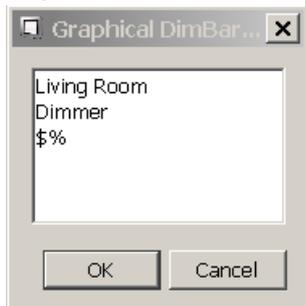
Use this property to select the **Button** state you want to edit and display.

Text and font:

9.1.18. *Text:*

Use this property to edit the **Button** Text. Any where you place the \$ sign the **Device** will replace it with the dynamic value of the **Button**.

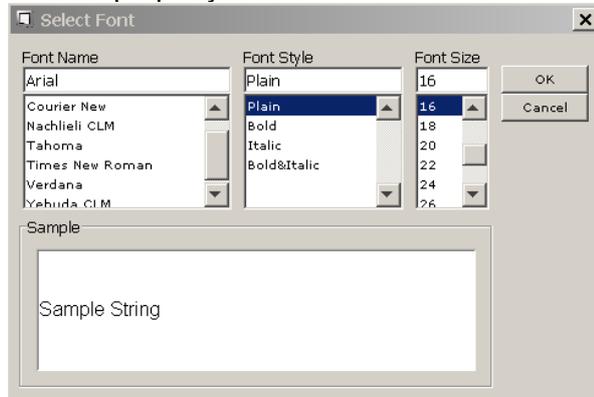
☺ - click on the  **Button** or select the *Text* property and press Ctrl+Space. This will open a custom editor that will allow you to separate the text into more than one line.



Button text custom editor

9.1.19. *Font:*

Use this property to define Text font and size.



Button font editor

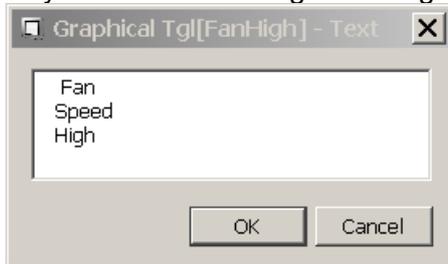
9.1.20. *Color:*

Use this property to edit the Text color.

9.1.21. *Alignment (V):*

Use this property to set the vertical alignment of the Text in relation to the **Container's** left and right borders.

☺ you can use leading or trailing spaces to fine tune the position of the text.

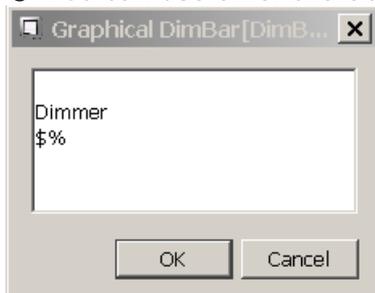


Use leading or trailing spaces

9.1.22. *Alignment (H):*

Use this property to set the horizontal alignment of the Text in relation to the Container's upper and lower borders. The software uses an internal algorithm that keeps the text a few pixels away from the upper and lower border of the button.

☺ You can use enter character (carriage return) to fine tune the position of the text.



Use carriage return

Background:

9.1.23. *Background color:*

Use this property to edit the **Container**'s background color.

9.1.24. *Image:*

Use this property to browse your PC for the **container**'s background image.

9.1.25. *Image Alignment (V):*

Use this property to set the vertical alignment of the background image in relation to the **container**'s left and right borders.

9.1.26. *Image Alignment (H):*

Use this property to set the horizontal alignment of the background image in relation to the **container**'s upper and lower border.

9.1.27. *Full Size:*

When you check this box, the software will expand/shrink the image dimensions so it will exactly fit the **Container**'s dimensions.

Icon/indicator:

9.1.28. *Icon:*

Use this property to browse your PC for the **container**'s Icon.

9.1.29. *Icon Alignment (V):*

Use this property to set the vertical alignment of the Icon in relation to the **container**'s left and right borders.

9.1.30. *Icon Alignment (H):*

Use this property to set the horizontal alignment of the Icon in relation to the **container**'s upper and lower border.

9.1.31. *Current value:*

The value that the analog **Button** (Bar Graph or Analog alpha numeric) is currently displaying. Use it to preview the **Button** look with different values.

9.1.32. *Max value:*

The Maximum value that the analog **Button** (Bar Graph or Analog alpha numeric) can have/display.

9.1.33. *Min value:*

The Minimum value that the analog **Button** (Bar Graph or Analog alpha numeric) can have/display.

9.1.34. *Show ticks:*

Use this property to show/hide bar graph Ticks.

9.1.35. *Number of Ticks:*

Use this property to set the number of bar graph Ticks.

9.1.36. *Ticks Color:*

Color of the Ticks. To select a color, open the color palette or type the name of the color directly into the dialog box. The colors names you can type are: White, Light Gray, Gray, Dark Gray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.

9.2. Functional parameters:

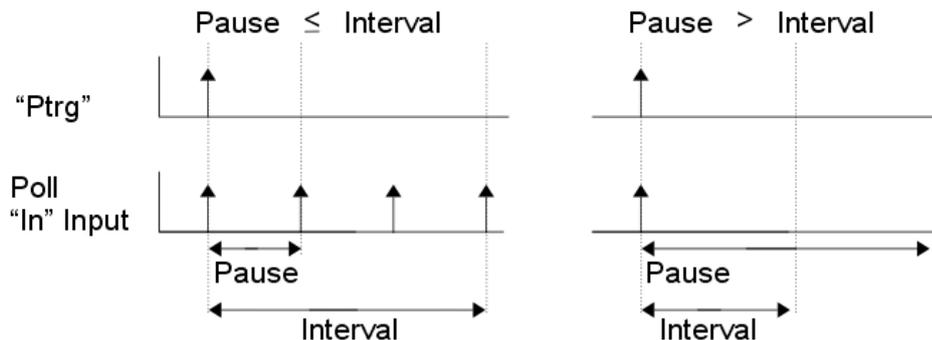
9.2.1. *Startup Value & Initial Mode:*

These properties define the value of a button immediately after **The Device** startup:

- When *Init Mode* property is set to *Init Value*, on startup, The **button** will receive the *Startup Value* property value
- When *Init Mode* property is set to *Last*, on startup, The **button** will receive the last value it had when the power went down (or reboot). If Last value is not available, the button will receive the value of *init*. This can happen in the following cases: i) After the first download that includes this button ii) After a download that uses "Delete Persistent Data".

9.2.2. *Triggered polling interval & Triggered Polling Pause:*

Triggered polling is a procedure composed of series bus polling requests. The total time for the procedure is defined by the *Triggered polling interval* property and the frequency of the polling **Events** is defined by the *Triggered polling pause* property. When the *Triggered polling pause* value is greater than *Triggered polling interval* Value, the Function Block will poll the *In Input* once at the beginning of the interval.



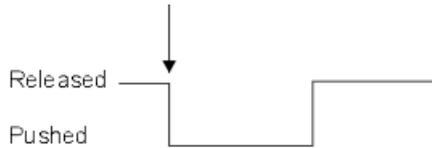
☺ use special care when setting polling and triggered polling as it generates a load on the EIB/KNX bus.

9.3. Terms:

Here are terms used for describing user actions:

On push:

At the moment **User** starts pressing the **Button**.



On push

On release:

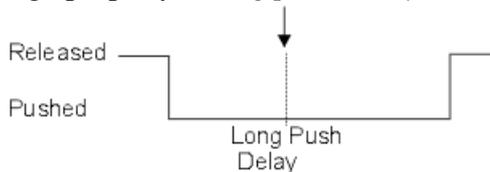
At the moment **User** stops pressing the **Button**.



On release

On Hold:

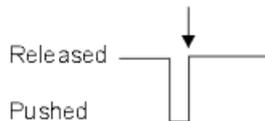
At the moment **User** pushes the **Button** for a period of time equal to the value of Setup Page property - *Long push Delay (sec)*.



On Hold

On click:

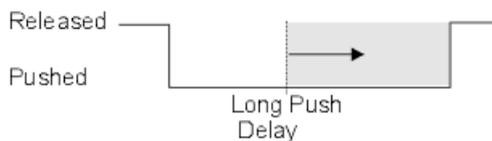
At the moment **User** releases a **Button** in case the period of time that the **Button** was pushed is less than value of Setup Page property *Long push Delay (sec)*.



On click

Long push:

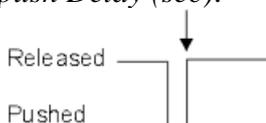
A push of a **Button** for a period of time equal to or more than the value of Setup Page property *Long push Delay (sec)*.



Long push

Short push:

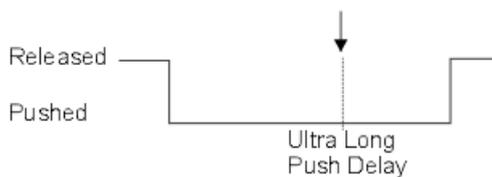
A push of a **Button** for a period of time less than the value of Setup Page property *Long push Delay (sec)*.



Short push

Ultra Long push:

A push of a **Button** for a period of time equal to or more than the value of Setup Page property *Ultra Long push Delay (sec)*.



Ultra Long push

9.4. On/Off Button:

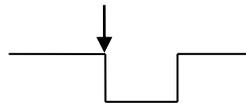
On/Off **Button** is designed especially to control and monitor 1bit data type objects:



9.4.1. Functional description:

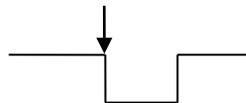


Upper part:



On push - Transmit "1" out of **1bit Out Output**.

Lower part:



On push - Transmit "0" out of **1bit Out Output**.

9.4.2. Functional properties

Icon	Property/Group	Value	Short Description
	ON/OFF Button ID	Text Preview	Unique reference ID of the Button.
	Receive		
→■	On/Off 1bit (OnOff)	List of the links made with this point.	The only Group Address type that is linkable to this input is a 1 bit Group Address. When the value of this input is set to True, the Button state will be set to ON. When this input is set to False the Button state will be set to OFF.

	<i>Transmit</i>		
	<i>1bit Out (1bitOut)</i>	List of the links made with this point.	The only Group Address type that is linkable to this point is a 1 bit Group Address. When the User pushes the upper part of the Button, this output will generate an event with the value "1". When the User pushes the lower part of the Button, this output will generate an event with the value "0".
<i>Parameters</i>			
	<i>State on start up</i>	Drop down menu : On Off	The initial state of the Button immediately after startup. This value is used only when <i>Init mode</i> is set to ' <i>init value</i> ' or when persistent data is not valid (e.g. after first download of the project)
	<i>Init Mode</i>	Drop down menu : Last Init Value	Mode of operation: Last: immediately after start up, the button will have the same state it had before power down. Init Value: immediately after start up, the button will have the value defined by ' <i>State on start up</i> ' property
<i>Polling</i>			
	<i>Polling on startup</i>	Check box	Polls the <i>On/Off 1bit</i> input, once, immediately after the Device startup.
	<i>Polling on display</i>	Drop down menu: Disabled Once 1s 5s 15s 30s 3min 15min 1h 6h 1day.	As long as the object is on display polls the <i>On/Off 1bit</i> input cyclically using the interval specified.
	<i>Cyclic polling</i>	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day.	Polls the <i>On/Off 1bit</i> input cyclically at all times using the interval specified.

	<i>Polling triggers</i>		
→■	<i>Triggers (PTrg)</i>	List of the links made with this point.	Whenever an event is detected on any of the links made for this point – the Device will perform a series of polling commands to the <i>On/Off 1bit</i> input.
	<i>Triggered polling interval</i>	Drop down menu: 1s 5s 15 30s 60s	The total time duration of triggered polling.
	<i>Triggered polling pause</i>	Drop down menu: 1s 5s 15 30s 60s	The time duration between single polling commands during triggered polling interval.

9.4.3. graphic properties:

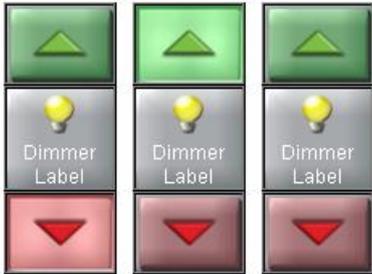
Property/Group	Value	Description
<i>ON/OFF Button ID</i>	Text Preview	Unique reference ID of the Button.
<i>Frame Type</i>	Drop down menu: [list of available frames]	Use this to select the frame style.
<i>Frame color</i>	Color palette And Drop down menu: [list of basic colors]	Select frame color from a Palette. Type: "null" for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, and Blue.
<i>Frame Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Enable Central Area</i>	Check box	Enable/Disable central area.
<i>Input Method</i>	Drop Down menu: None Touch point	None = this Button will not perform any action as a result of User touch. You can use the Button to display status. Touch point = The Button will perform an action when it's pushed by the User. The action is defined by the Button part that was pushed (upper/lower).
<i>AutoUpdate</i>	Check box	Button state will be updated automatically any time the User touches it. This function is not available when the Input Method is "None".
<i>Password</i>	Check box	The User will be asked for the password in order to perform the operation. This function is not available when the Input Method is "None".
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Boundaries</i>	[[X], [Y], [Width],[Height]]	View and edit the position, width and height of the Button.

Button parts properties		
<i>Upper Part</i>	Radio button. Mutually exclusive between: Upper part, Central part and Lower part.	When you select it, – the program will display and allow you to edit the properties of the upper part of the button, (example: text and background). The upper part properties are listed in the remainder of this table.
<i>Central part</i>	Radio button. Mutually exclusive between: Upper part, Central part and Lower part.	When you select it, – the program will display and allow you to edit the properties of the central part of the button, (example: text and background). The central part properties are listed in the remainder of this table.
<i>Lower Part</i>	Radio button. Mutually exclusive between: Upper part, Central part and Lower part.	When you select it, – the program will display and allow you to edit the properties of the lower part of the button, (example: text and background). The lower part properties are listed in the remainder of this table.
<i>On state</i>	Radio button. Mutually exclusive between: On state and Off state.	When you select it, – the program will display and allow you to edit the On state properties of the button, (example: icon and background). The On state properties are listed in the remainder of this table.
<i>Off state</i>	Radio button. Mutually exclusive between: On state and Off state.	When you select it, – the program will display and allow you to edit the Off state properties of the button, (example: icon and background). The Off state properties are listed in the remainder of this table.

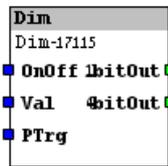
<i>Text and Font</i>		
<i>Text</i>	Text Preview	Use this property to edit the Button Text.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Color</i>	Color palette And Drop down menu: [list of basic colors].	Select Text color from a Palette. Type null for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
<i>Background</i>		
<i>Background color</i>	Color palette And Drop down menu: [list of basic colors].	Select Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions
<i>Icon/Indicator</i>		These properties are for the top and bottom areas of the button only!!!
<i>Icon</i>	Icon Image Preview	Use this to search your PC for Icons.
<i>Icon Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Icon.
<i>Icon Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Icon.

9.5. Dimmer **Button**:

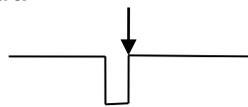
A Dimmer **Button** is designed especially to control and monitor Dimmer actors actions using 1bit, 4bit and 8bit data types. The **Button** has two states, so it can display the On/Off status of the load:



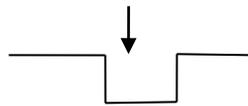
9.5.1. Functional description:



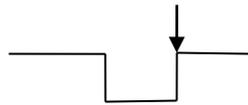
Upper part:



On click - Transmits "0001b" out of **1 bit Out Output**

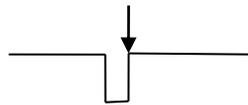


On Hold - Transmits "1001b" out of **4 bit Out Output**

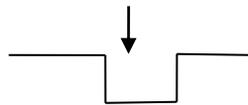


On release of long push – Transmits "0000b" out of **4 bit Out Output**

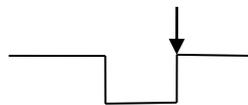
Lower part:



On click - Transmits "0000b" out of **1 bit Out Output**



On hold - Transmits "0001b" out of **4 bit Out Output**



On release of long push – Transmits "0000b" out of **4 bit Out Output**

9.5.2. Functional properties:

	Property/Group	Value	Description
	<i>Dimmer ID</i>	Text Preview	Unique reference ID of the Button.
	<i>Receive</i>		
→■	On/Off 1bit (OnOff)	List of the links made with this point.	The only Group Address type that is linkable to this input is a 1 bit Group Address. When the value of this input is set to True the Button state will be set to ON. When this input is set to False the Button state will be set to OFF.
→■	Value 8 bit (Val)	List of the links made with this point.	The only Group Address type that is linkable to this input is an 8 bit Group Address. When the value of this input is set to False the Button state will be set to OFF. Otherwise, the Button state will be set to ON.
	<i>Transmit</i>		
■→	1 bit Out (1bitOut)	List of the links made with this point.	The only Group Address type that is linkable to this output is a 1bit Group Address. When the User clicks the upper part of the Button, this output generates an event with the value "1". When the User clicks the lower part of the Button, this output generates an event with the value "0".
■→	4 bit Out (4bitOut)	List of the links made with this point.	The only Group Address type that is linkable to this output is a 4 bit Group Address. When the User performs a long push on the upper part of the button, the Button will send a 4 bit control increase to this output. When the user performs a long push on the lower part of the button, the button will send a 4 bit control decrease to this output. When the user releases the button , the button will send a 4 bit control stop to this output.

	Parameters		
	<i>State on start up</i>	Drop down menu : On Off	The initial state of the Button immediately after startup. This value is used only when <i>Init mode</i> is set to ' <i>init value</i> ' or when persistent data is not valid (e.g. after first download of the project)
	<i>Init mode</i>	Drop down menu : Last Init Value	Mode of operation: Last: immediately after start up, the button will have the same state it had before power down. Init Value: immediately after start up, the button will have the value defined by ' <i>State on start up</i> ' property
Polling			
	<i>Polling on start up</i>	Check box	Polls the <i>Value 8bit</i> input, once, immediately after the Device startup.
	<i>Polling on display</i>	Drop down menu: Disabled Once 1s 5s 15s 30s 3min 15min 1h 6h 1day.	As long as the object is on display, polls the <i>Value 8bit</i> input, <i>cyclically</i> , using the interval specified.
	<i>Cyclic polling</i>	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the <i>Value 8bit</i> input cyclically at all times using the interval specified.
Polling triggers			
→■	<i>Triggers (Ptrg)</i>	List of the links made with this point.	Whenever an event is detected on any of the links made for this point – the Device will perform a series of polling commands to the <i>Value 8 bit</i> input.
	<i>Triggered polling interval</i>	Drop down menu: 1s 5s 15s 30s 60s	The total time duration of triggered polling.
	<i>Triggered polling pause</i>	Drop down menu: 1s 5s 15s 30s 60s	The time duration between single polling commands during triggered polling interval.

9.5.3. Graphic properties:

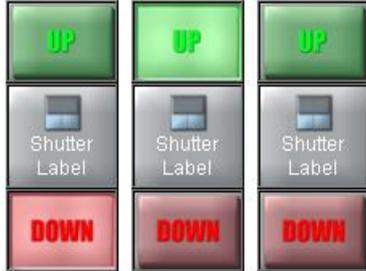
Property/Group	Value	Description
<i>Dimmer ID</i>	Text Preview	Unique reference ID of the Button.
<i>Frame Type</i>	Drop down menu: [list of available frames]	Use this to select the frame style.
<i>Frame color</i>	Color palette And Drop down menu: [list of basic colors].	Select frame color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Frame Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Enable Central Area</i>	Check box	Enable/Disable central area.
<i>Input Method</i>	Drop down menu: None Touch point.	None = this Button will not perform any action as a result of User touch. You can use the Button to display status. Touch point = The Button will perform an action when it's pushed by the User. The action is defined by the Button part that was pushed (upper/lower)
<i>AutoUpdate</i>	Check box	Button state will be updated automatically any time the User touches it. Upper part push – on hold and on click button state will be On. Lower part push - on click button state will be Off (holding lower part has no effect on button state). This function is not available when the Input Method is "None".
<i>Password</i>	Check box	The User will be asked for the password in order to perform the operation. This function is not available when the Input Method is "None".
<i>Height</i>	[Height]	Button height (pixels).
<i>Width</i>	[Width]	Button height (pixels).
<i>Button Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the Button.

Button parts properties		
Upper part	Radio button. Mutually exclusive between: Upper part, Central part and Lower part.	When you select it, – the program will display and allow you to edit the properties of the upper part of the button, (example: text and background). The upper part properties are listed in the remainder of this table.
Central part	Radio button. Mutually exclusive between: Upper part, Central part and Lower part.	When you select it, – the program will display and allow you to edit the properties of the central part of the button, (example: text and background). The central part properties are listed in the remainder of this table.
Lower part	Radio button. Mutually exclusive between: Upper part, Central part and Lower part.	When you select it, – the program will display and allow you to edit the properties of the central part of the button, (example: text and background). The lower part properties are listed in the remainder of this table.
On state	Radio button. Mutually exclusive between: On state and Off state.	When you select it, – the program will display and allow you to edit the On state properties of the button, (example: icon and background). The On state properties are listed in the remainder of this table.
Off state	Radio button. Mutually exclusive between: On state and Off state.	When you select it, – the program will display and allow you to edit the Off state properties of the button, (example: icon and background). The Off state properties are listed in the remainder of this table.

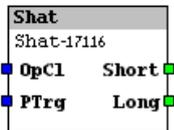
<i>Text and Font</i>		
<i>Text</i>	Text Preview	Use this property to edit the Button Text.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Color</i>	Color palette And Drop down menu: [list of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, and Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
<i>Background</i>		
<i>Background color</i>	Color palette And Drop down menu: [list of basic colors].	Select Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, and Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.
<i>Icon/Indicator</i>		
<i>Icon</i>	Icon Image Preview	Use this to search your PC for Icons.
<i>icon Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Icon.
<i>Icon Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Icon.

9.6. Shutter **Button**:

The shutter **Button** is designed especially to control and monitor shutter actors using 1bit telegrams. The **Button** has two states so, if the shutter actor supports the function, it can display the close/open status as well:

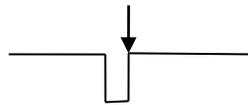


9.6.1. Functional description:

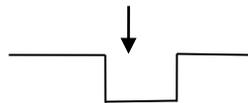


When *Shutter behavior* property is set to "Up/Open=0, Down/Close=1":

Upper part:

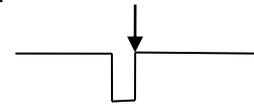


On click - Transmits "0" out of *Short push 1 bit Output*

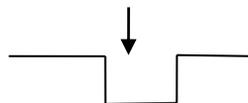


On hold - Transmits "0" out of *Long push 1 bit Output*

Lower part:



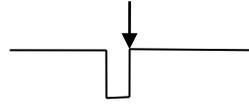
On click - Transmits "1" out of *Short push 1 bit Output*



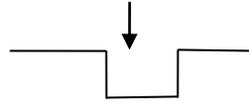
On hold - Transmits "1" out of *Long push 1 bit Output*

When *Shutter behavior* property is set to "Up/Open=1, Down/Close=0":

Upper part:

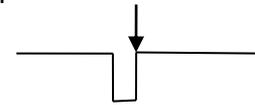


On click - Transmits "1" out of *Short push 1 bit Output*

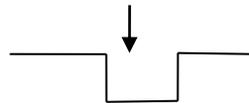


On hold - Transmits "1" out of *Long push 1 bit Output*

Lower part:



On click - Transmits "0" out of *Short push 1 bit Output*



On hold - Transmits "0" out of *Long push 1 bit Output*

9.6.2. Functional properties:

Icon	Property/Group	Value	Description
	<i>Shutter ID</i>	Text Preview	Unique reference ID of the button.
	<i>Receive</i>		
→■	<i>Open/Close 1 bit (OpCl)</i>	List of the links made with this point.	The only Group Address type that is linkable to this input is a 1 bit Group Address. Functionality is according to the Shutter behavior property settings.
	<i>Transmit</i>		
■→	<i>Short push 1 bit (Short)</i>	List of the links made with this point.	The only Group Address type that is linkable to this output is a 1 bit Group Address. Functionality is according to Shutter behavior property settings.
■→	<i>Long push 1 bit (Long)</i>	List of the links made with this point.	The only Group Address type that is linkable to this output is a 1 bit Group Address. Functionality is according to Shutter behavior property settings.

	Parameters		
	<i>State on start up</i>	Drop down menu : Open Close.	The initial state of the Button immediately after startup. This value is used only when <i>Init mode</i> is set to ' <i>init value</i> ' or when persistent data is not valid (e.g. after first download of the project)
	<i>Init mode</i>	Drop down menu : Last Init Value	Mode of operation: Last: immediately after start up, the button will have the same state it had before power down. Init Value: immediately after start up, the button will have the value defined by ' <i>State on start up</i> ' property
	<i>Shutter behavior</i>	Drop down menu: Up/Open=0, Down/Close=1 Up/Open=1, Down/Close=0	Use this property to define buttons value for Up/Open and for Down/Close commands.
<i>Polling</i>			
	<i>Polling on start up</i>	Check box	Polls the <i>Open/Close 1 bit</i> input, once, immediately after the Device startup.
	<i>Polling on display</i>	Drop down menu: Disabled Once 1s 5s 15s 30s 3min 15min 1h 6h 1day.	As long as the object is on display polls the <i>Open/Close 1bit</i> input cyclically using the interval specified.
	<i>Cyclic polling</i>	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day.	Polls the <i>Open/Close 1bit</i> input cyclically at all times using the interval specified.
<i>Polling triggers</i>			
→■	<i>Triggers (PTrg)</i>	List of the links made with this point.	Whenever an event is detected on any of the links made for this point – the Device will perform a series of polling commands to the <i>Open/Close 1bit</i> input.
	<i>Triggered polling interval</i>	Drop down menu: 1s 5s 15s 30s 60s	The total time duration of triggered polling.
	<i>Triggered polling pause</i>	Drop down menu: 1s 5s 15s 30s 60s	The time duration between single polling commands during triggered polling interval.

9.6.3. graphic properties:

Property/Group	Value	Description
<i>Shutter ID</i>	Text Preview	Unique reference ID of the button.
<i>Frame Type</i>	Drop down menu: [list of available frames]	Use this to select the frame style.
<i>Frame color</i>	Color palette And Drop down menu: [list of basic colors].	Select frame color from a Palette. Type 'null' for transparency Or color name: White, Light Gray, Gray, Dark Gray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Frame Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Central Area</i>	Check box	Enable/Disable central area.
<i>Input Method</i>	Drop down menu: None Touch point.	None = this button will not perform any action as a result of User touch. You can use the button to display status. Touch point = The button will perform an action when it's pushed by the User. The action is defined by the button part that was pushed (upper/lower)
<i>AutoUpdate</i>	Check box	Button state will be updated automatically any time the User touches it. Upper part push – on hold and on click button state will be On. Lower part push - on hold button state will be Off. This function is not available when the Input Method is "None".
<i>Password</i>	Check box	The User will be asked for the password in order to perform the operation. This function is not available when the Input Method is "None".
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.

Button parts properties		
Upper part	Radio button. Mutually exclusive between: Upper part, Central part and Lower part.	When you select it, – the program will display and allow you to edit the properties of the upper part of the button, (example: text and background). The upper part properties are listed in the remainder of this table.
Central part	Radio button. Mutually exclusive between: Upper part, Central part and Lower part.	When you select it, – the program will display and allow you to edit the properties of the central part of the button, (example: text and background). The central part properties are listed in the remainder of this table.
Lower part	Radio button. Mutually exclusive between: Upper part, Central part and Lower part.	When you select it, – the program will display and allow you to edit the properties of the lower part of the button, (example: text and background). The lower part properties are listed in the remainder of this table.
On state	Radio button. Mutually exclusive between: On state and Off state.	When you select it, – the program will display and allow you to edit the On state properties of the button, (example: icon and background). The On state properties are listed in the remainder of this table.
Off state	Radio button. Mutually exclusive between: On state and Off state.	When you select it, – the program will display and allow you to edit the Off state properties of the button, (example: icon and background). The Off state properties are listed in the remainder of this table.

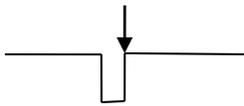
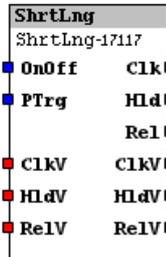
<i>Text and Font</i>		
<i>Text</i>	Text Preview	Use this property to edit the button Text.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Color</i>	Color palette And Drop down menu: [list of basic colors]	Select Text color from a Palette. Type null for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
<i>Background</i>		
<i>Background color</i>	Color palette And Drop down menu: [list of basic colors].	Select Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.
<i>Icon/Indicator</i>		
<i>Icon</i>	Icon Image Preview	Use this to search your PC for Icons.
<i>Icon Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Icon.
<i>Icon Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Icon.

9.7. Short/Long (Click/Hold) **Button**:

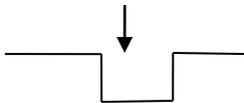
A Short/Long **Button** is a general purpose **Button**. It reacts differently to long pushes and short pushes. It has three different **Outputs**; *Hold* and *Release* for long pushes and *Click* for short pushes. You can use the **Output** to transmit any data type and value. The **Button** has two states so it can display On/Off status as well:



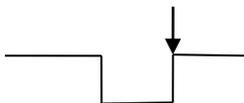
9.7.1. Functional description:



On click - Transmits the value of *Click Value* out of *Click Output*



On hold - Transmits the value of *Hold Value* out of *Hold Output*



On release of long push – Transmits the value of *Release Value* out of *Release Output*

9.7.2. Functional properties:

	Property/Group	Value	Description
	Short/long ID	Text Preview	Unique reference ID of the button.
	Receive		
→■	On/Off (OnOff)	List of the links made for this output.	When the value of this input is set to True the Button state will be set to ON. When this input is set to False the Button state will be set to OFF.
	Transmit		
■→	Click (Clk)	List of the links made for this output.	A short click on the button will produce an event on this output. The value of the event is defined by <i>Click Value</i> parameter.
■→	Hold (Hld)	List of the links made for this output.	An event on this output will happen at the beginning of A long push of the button. The value of the event is defined by <i>Hold Value</i> parameter.
■→	Release (Rel)	List of the links made for this output.	An event on this output will occur at the termination (release) of a long push of the button. The value of the event is defined by the <i>Release Value</i> parameter.
	Parameters		
	State on start up	Drop down menu : On Off	The initial state of the Button immediately after startup. This value is used only when <i>Init mode</i> is set to 'init value' or when persistent data is not valid (e.g. after first download of the project)
	Init mode	Drop down menu : Last Init Value	Mode of operation: Last: immediately after start up, the button will have the same state it had before power down. Init Value: immediately after start up, the button will have the value defined by 'State on start up' property
→■→	Click Value(ClkV)	[Value]	The value of this parameter will be assigned to the <i>Click</i> output.
→■→	Hold Value(HldV)	[Value]	The value of this parameter will be assigned to the <i>Hold</i> output.
→■→	Release Value (RelV)	[Value]	The value of this parameter will be assigned to the <i>Release</i> output.

<i>Polling</i>			
	<i>Polling on start up</i>	Check box	Polls the <i>On/Off</i> Input, once, immediately after the Device startup.
	<i>Polling on display</i>	Drop down menu: Disabled Once 1s 5s 15s 30s 3min 15min 1h 6h 1day.	As long as the object is on display polls the <i>On/Off</i> Input cyclically using the interval specified.
	<i>Cyclic polling</i>	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the <i>On/Off</i> Input cyclically at all times using the interval specified.
<i>Polling triggers</i>			
→■	<i>Triggers (PTrg)</i>	List of the links made with this point.	Whenever an event is detected on any of the links made for this point – the Device will perform a series of polling commands to the <i>On/Off</i> input.
	<i>Triggered polling interval</i>	Drop down menu: 1s 5s 15s 30s 60s	The total time duration of triggered polling.
	<i>Triggered polling pause</i>	Drop down menu: 1s 5s 15s 30s 60s	The time duration between single polling commands during triggered polling interval.

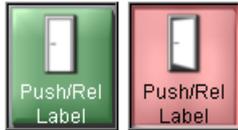
9.7.3. graphic properties:

Property/Group	Value	Description
<i>Short/long Button ID</i>	Text Preview	Unique reference ID of the button.
<i>Frame Type</i>	Drop down menu: [list of available frames]	Use this to select the frame style.
<i>Frame color</i>	Color palette And Drop down menu: [list of basic colors].	Select frame color from a Palette. Type 'null' for transparency Or color name: White, Light Gray, Gray, Dark Gray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Frame Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Input Method</i>	Drop down menu: None Touch point.,	None = this button will not perform any action as a result of User touch. You can use the button to display status. Touch point = The button will perform an action when it's pushed by the User.
<i>Password</i>	Check box	The User will be asked for the password in order to perform the operation. This function is not available when the Input Method is "None".
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.
<i>button parts properties</i>		<i>button parts properties</i>
On state	Radio button. Mutually exclusive between: On state and Off state.	When you select it, – the program will display and allow you to edit the On state properties of the button, (example: icon and background). The On state properties are listed in the remainder of this table.
Off state	Radio button. Mutually exclusive between: On state and Off state.	When you select it, – the program will display and allow you to edit the Off state properties of the button, (example: icon and background). The Off state properties are listed in the remainder of this table.

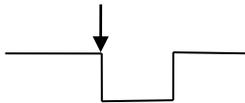
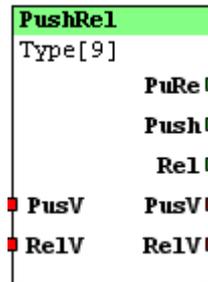
<i>Text and Font</i>		
<i>Text</i>	Text Preview	Use this property to edit the button Text.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Color</i>	Color palette And Drop down menu: [list of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
<i>Background</i>		
<i>Background color</i>	Color palette And Drop down menu: [list of basic colors].	Select Background color from a Palette. Type 'null' for transparency Or color name: White, Light Gray, Gray, Dark Gray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.
<i>Icon/Indicator</i>		
<i>Icon</i>	Icon Image Preview	Use this to search your PC for Icons.
<i>Icon Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Icon.
<i>Icon Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Icon.

9.8. Push Release **Button**:

The Push Release **Button** is a general purpose **Button**. It reacts to the push event, and to the release event. You can use it to transmit any data type and value. This **Button** changes its state to the Push state only during the **User** push, and therefore is not capable of displaying status.

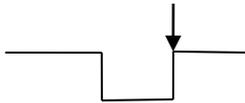


Functional description:



On push - Transmits the value of *Push Value* out of *On push and On Release Output*

On push – Transmits the value of *Push Value* out of *Only On Push Output*



On release – Transmits the value of *Release value* out of *On push and On Release Output*

On release – Transmits the value of *Release value* out of *Only On release Output*

9.8.1. Functional properties:

Icon	Property/Group	Value	Description
	<i>Push/Release ID</i>	Text Preview	Unique reference ID of the button.
	<i>Transmit</i>		
	<i>On push and On Release(PuRe)</i>	List of the links made with this point.	Pressing the button will generate an event on this output with a value equal to the <i>Push Value</i> parameter. The release of the button will generate an event on this output with a value equal to the <i>Release value</i> parameter.
	<i>Only On Push (Push)</i>	List of the links made with this point.	Pressing this button will generate an event on this output with a value equal to the <i>Push Value</i> parameter.
	<i>Only On release (Rel)</i>	List of the links made with this point.	The release of the button will generate an event on this output with a value equal to the <i>Release value</i> parameter.
	<i>Parameters</i>		
	<i>Push Value(PusV)</i>	[Value]	The value of this parameter will be assigned to the push function of the <i>On push and On Release</i> output and to the <i>Only On Push</i> output
	<i>Release value (RelV)</i>	[Value]	The value of this parameter will be assigned to the release function of the <i>On push and On Release</i> output and to the <i>Only On release</i> output

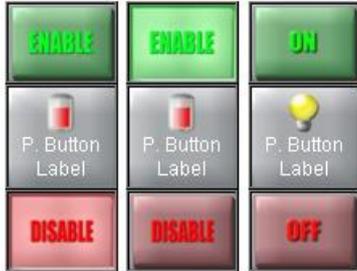
9.8.2. Graphic properties:

Property/Group	Value	Description
<i>Push/Release Button ID</i>	Text Preview	Unique reference ID of the button.
<i>Frame Type</i>	Drop down menu: [list of available frames]	Use this to select the frame style.
<i>Frame color</i>	Color palette And Drop down menu: [list of basic colors].	Select frame color from a Palette. Type 'null' for transparency Or color name: White, Light Gray, Gray, Dark Gray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Frame Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Input Method</i>	Drop down menu: None Touch point.	None = this button will not perform any action as a result of User touch. You can use the button to display status. Touch point = The button will perform an action when it's pushed by the User.
<i>Password</i>	Check box	The User will be asked for the password in order to perform the operation. This function is not available when the Input Method is "None".
<i>Position and size</i>		
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.
<i>button parts properties</i>		<i>button parts properties</i>
<i>Push state</i>	Radio button. Mutually exclusive between: Push state and Release state.	When you select it, – the program will display and allow you to edit the Push state properties of the button, (example: icon and background). The Push state properties are listed in the remainder of this table.
<i>Release state</i>	Radio button. Mutually exclusive between: Push state and Release state.	When you select it, – the program will display and allow you to edit the Release state properties of the button, (example: icon and background). The Release state properties are listed in the remainder of this table.

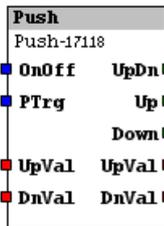
<i>Text and Font</i>		
<i>Text</i>	Text Preview	Use this property to edit the button Text.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Color</i>	Color palette And Drop down menu: [list of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
<i>Background</i>		
<i>Background color</i>	Color palette And Drop down menu: [list of basic colors].	Select Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.
<i>Icon/Indicator</i>		
<i>Icon</i>	Icon Image Preview	Use this to search your PC for Icons.
<i>Icon Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Icon.
<i>Icon Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Icon.

9.9. Push button **Button**:

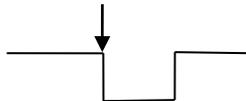
The Push Button is a general purpose **Button**. It reacts **only** to push events. You can use it to transmit any data type and value. It has two states so it can display On/Off status as well.



9.9.1. Functional description:



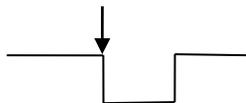
Upper part



On push - Transmit the value of *Push Up value* out of *On push Up & Down Output*.

On push – Transmit the value of *Push Up value* out of *On push Up Output*.

Lower part:



On push - Transmit the value of *Push Down value* out of *On push Up & Down Output*.

On push – Transmit the value of *Push Down value* out of *On push Down Output*.

9.9.2. Functional properties:

Icon	Property/Group	Value	Description
	<i>Push button ID</i>	Text Preview	Unique reference ID of the button.
	<i>Receive</i>		
→■	<i>On/Off (OnOf)</i>	List of the links made with this point.	When the value of this input is set to True, the button state will be set to ON. When this value of this input is set to False, the button state will be set to OFF.
	<i>Transmit</i>		
■→	<i>On push Up & Down (UpDn)</i>	List of the links made with this point.	Pressing the upper part of the button will generate an event on this output with a value equal to the <i>Push Up Value</i> parameter. Pressing the lower part of the button will generate an event on this output with a value equal to the <i>Push Down Value</i> parameter.
■→	<i>On push Up (Up)</i>	List of the links made with this point.	Pressing on the upper part of the button will generate an event on this output with a value equal to the <i>Push Up Value</i> parameter.
■→	<i>On push Down (Down)</i>	List of the links made with this point.	Pressing on the lower part of the button will generate an event on this output with a value equal to the <i>Push Down Value</i> parameter.
	<i>Parameters</i>		
	<i>State on start up</i>	Drop down menu : On Off	The initial state of the Button immediately after startup. This value is used only when <i>Init mode</i> is set to ' <i>init value</i> ' or when persistent data is not valid (e.g. after first download of the project)
	<i>Init mode</i>	Drop down menu : Last Init Value	Mode of operation: Last: immediately after start up, the button will have the same state it had before power down. Init Value: immediately after start up, the button will have the value defined by ' <i>State on start up</i> ' property

→■→	<i>Push Up Value (UpVal)</i>	[Value]	The value of this parameter will be assigned to the Push Up function of the <i>On push UP & Down</i> output and to the <i>Only On push UP</i> output
→■→	<i>Push Down Value (DnVal)</i>	[Value]	The value of this parameter will be assigned to the Push Down function of the <i>On push UP & Down</i> output and to the <i>Only On push Down</i> output
<i>Polling</i>			
	<i>Polling on start up</i>	Check box	Polls the <i>On/Off Input</i> , once, immediately after the Device startup.
	<i>Polling on display</i>	Drop down menu: Disabled Once 1s 5s 15s 30s 3min 15min 1h 6h 1day.	As long as the object is on display polls the <i>On/Off Input</i> cyclically using the interval specified.
	<i>Cyclic polling</i>	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day.	Polls the <i>On/Off Input</i> cyclically at all times using the interval specified.
<i>Polling triggers</i>			
→■	<i>Triggers (PTrg)</i>	List of the links made with this point.	Event on any of the links made for this input will trigger a series of polling commands to the <i>On/Off</i> input.
	<i>Triggered polling interval</i>	Drop down menu: 1s 5s 15s 30s 60s	The total time duration of triggered polling.
	<i>Triggered polling pause</i>	Drop down menu: 1s 5s 15s 30s 60s	The time duration between single polling commands during triggered polling interval.

9.9.3. Graphic properties:

Property/Group	Value	Description
<i>Push button ID</i>	Text Preview	Unique reference ID of the button.
<i>Frame Type</i>	Drop down menu: [list of available frames]	The button unique reference ID
<i>Frame color</i>	Color palette And Dropdown menu: [list of basic colors].	Select frame color from a Palette. Type 'null' for transparency Or color name: White, Light Gray, Gray, Dark Gray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Frame Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Central Area</i>	Check box	Enable/Disable central area.
<i>Input Method</i>	Drop down menu: None Touch point.	None = this button will not perform any action as a result of User touch. You can use the button to display status. Touch point = The button will perform an action when it's pushed by the User.
<i>AutoUpdate</i>	Check box	Button state will be updated automatically any time the User touches it.. This function is not available when the Input Method is "None".
<i>Password</i>	Check box	The User will be asked for the password in order to perform the operation. This function is not available when the Input Method is "None".
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.
button parts properties	<i>button parts properties</i>	
Upper part	Radio button. Mutually exclusive between: Upper part, Central part and Lower part.	When you select it, – the program will display and allow you to edit the properties of the upper part of the button, (example: text and background). The upper part properties are listed in the remainder of this table.
Central part	Radio button. Mutually exclusive between: Upper part, Central part and Lower part.	When you select it, – the program will display and allow you to edit the properties of the central part of the button, (example: text and background). The central part properties are listed in the remainder of this table.

Lower part	Radio button. Mutually exclusive between: Upper part, Central part and Lower part.	When you select it, – the program will display and allow you to edit the properties of the central part of the button, (example: text and background). The lower part properties are listed in the remainder of this table.
On state	Radio button. Mutually exclusive between: On state and Off state.	When you select it, – the program will display and allow you to edit the On state properties of the button, (example: icon and background). The On state properties are listed in the remainder of this table.
Off state	Radio button. Mutually exclusive between: On state and Off state.	When you select it, – the program will display and allow you to edit the Off state properties of the button, (example: icon and background). The Off state properties are listed in the remainder of this table.
<i>Text and Font</i>		
<i>Text</i>	[Text Preview]	Use this property to edit the button Text.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Color</i>	Color palette And Drop down menu: [list of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
<i>Background</i>		
<i>Background color</i>	Color palette And Drop down menu: [List of basic colors].	Select Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.

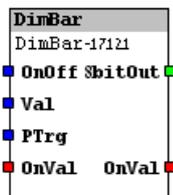
<i>Icon/Indicator</i>		These properties are for the upper and lower areas of the button only!!!
<i>Icon</i>	Icon Image Preview	Use this to search your PC for Icons.
<i>Icon Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Icon.
<i>Icon Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Icon.

9.10. Dimmer vertical bar graph (Dimmer Bar Vertical) **Button:**

Dimmer Bar Vertical is designed especially to transmit absolute values used to control Dimmer actors using 8 bit telegrams. You can also use it to display 0 to 100 (%) values:



9.10.1. Functional description:

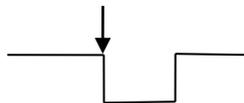


The *Input Method* parameter determines if the user interface for input will be *keypad* or *touch point*.

In cases where the *Input Method* is *touch point*:

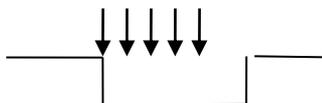
The new value will be determined by the point pressed on the Bar area. It is also possible to push and drag over the bar towards the desired value. When the **User** drags out of the bar area to the top - the value entered will be the maximum value (100%). When the **User** drags beyond the bar area to the bottom - the value entered will be the minimum value (0%).

In cases where the *Input mode* property = *jump*



On push - Immediately Transmits the value out of **8 bit Out Output**.

In cases where the *Input mode* property = *step*



As long as the button is pushed/dragged – the value advances towards the new value gradually, in steps, according to *Step size*

(%) and *Steps per sec* properties. Transmits the values out of **8 bit Out Output**.

In cases where the *Input Method* is *keypad*:



When the keypad Enter button is pressed - Transmits the new value out of **8bit Out Output**.

9.10.2. Functional properties:

	Property/Group	Value	Description
	<i>Dimmer Bar Vertical ID</i>	Text Preview	Unique reference ID of the button.
	<i>Receive</i>		
→■	<i>On/Off 1bit (OnOf)</i>	List of the links made with this point.	The only Group Address type that is linkable to this point is a 1 bit Group Address. When an event with the value False arrives at this input – the bar is set to 0. When an event with the value True arrives at this input - the bar is set to the value of the 1bit "On" Value parameter.
→■	<i>Value 8 bit (8bitIn)</i>	List of the links made with this point.	The only Group Address type that is linkable to this point is an 8 bit Group Address. The bar will automatically scale input values in the range of: 0-255 to display values in the range of: 0-100.

Transmit			
-	<i>8 bit Out (8bitOut)</i>	List of the links made with this point.	The only Group Address type that is linkable to this point is an 8 bit Group Address. New values will be sent through this output any time the User updates the bar level. The bar will automatically scale input values in the range of: 0-100 to transmitted values in the range of: 0-255.
Parameters			
	<i>Start Up Value</i>	Real value: 0 to 255	The initial value of the Button immediately after startup. This value is used only when <i>Init mode</i> is set to ' <i>init value</i> ' or when persistent data is not valid (e.g. after first download of the project). The bar will automatically scale input values in the range of 0-255 to displayed values in the range of: 0-100.
	<i>Init mode</i>	Drop down menu : Last Init Value	Mode of operation: Last: immediately after start up, the button will have the same value it had before power down. Init Value: immediately after start up, the button will have the value defined by ' <i>State on start up</i> ' property
→■→	<i>1bit "On" Value (OnVal)</i>	Real value: 0 to 255	This value will be assigned to the button when the value "1" is received from the <i>On/Off 1bit</i> input. The bar will automatically scale input values in the range of 0-255 to display values in the range of: 0-100.

<i>Polling</i>			
	<i>Polling on start up</i>	Check box	Polls the <i>Value 8bit</i> input, once, immediately after the Device startup.
	<i>Polling on display</i>	Drop down menu: Disabled Once 1s 5s 15s 30s 3min 15min 1h 6h 1day.	As long as the object is on display: polls the <i>Value 8bit</i> input cyclically using the interval specified.
	<i>Cyclic polling</i>	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the <i>Value 8bit</i> input cyclically at all times using the interval specified.
<i>Polling triggers</i>			
→■	<i>Triggers (PTrg)</i>	List of the links made with this point.	An Event on any of the links made for this input will trigger a series of polling commands to the <i>Value 8bit</i> input.
	<i>Triggered polling interval</i>	Drop down menu: 1s 5s 15s 30s 60s	The total time duration of triggered polling.
	<i>Triggered polling pause</i>	Drop down menu: 1s 5s 15s 30s 60s	The time duration between single polling commands during triggered polling interval.

9.10.3. Graphic properties:

Property/Group	Value	Description
<i>Dimmer Bar Vertical ID</i>	Text Preview	Unique reference ID of the button.
<i>Frame Type</i>	Drop down menu: [list of available frames]	Use this to select the frame style.
<i>Frame color</i>	Color palette And Drop down menu: [list of basic colors].	Select Frame color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Frame Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Input Method</i>	Drop down menu: None Bar Keypad.	None = this button will not perform any action as a result of User touch. You can use the button to display status. Bar = User can use the bar to assign new values for the button.. The new value is defined by the location of the push/drag point over the bar. Keypad = when the User touches the button, a keypad will pop up. The keypad enables the User to assign specific value to the button.
<i>Keypad title Text</i>	Text preview	The title is displayed on the upper part of the popup keypad.
<i>Input mode</i>	Drop down menu: Jump to new value step to new value	Jump - Jump immediately to touch point position Step – advance in steps towards touch point value. Not visible when “Display only”, is selected
<i>Step size (%)</i>	[Step]	The size (%) of single step towards the new value. Visible when "step to new value" is selected.
<i>Steps per sec</i>	Drop down menu: 1 2 3 4.	The Number of times per second that the bar graph value will be updated. Visible when "step to new value" is selected.

<i>Password</i>	Check box	The User will be asked for the password in order to perform the operation. This function is not available when the Input Method is "None".
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.
<i>Bar properties</i>		
<i>Text and Font</i>		
<i>Text</i>	Text Preview	Use this property to edit the button Text. \$ Sign will be replaced with the dynamic value of the button.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Color</i>	Color palette And Drop down menu: [list of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
<i>Bar graphic</i>		
<i>Bar Background color</i>	Color palette And Drop down menu: [List of basic colors].	Select Bar Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Bar color</i>	Color palette And Drop down menu: [list of basic colors].	Select Bar color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for images. This image will be placed on top of the bar.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Icon.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Icon.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.

<i>Bar numeric</i>		
Current value %	[Current value]	Use this to preview the appearance of the button while displaying different values. Values can be from 0 to 100 and resolution is about 0.4 (100/255)
Show ticks	Check box	Display Ticks on the left side of the Bar.
Show ticks value	Check box	Display Bar values on the left side of the Bar.
Number of Ticks	[Number of Ticks]	The number of Ticks and/or value label displayed.
<i>Ticks Color</i>	Color palette And Drop down menu: [list of basic colors].	Select Ticks color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
Digits after decimal point	[Number of digits].	This property defines how many digits to the right of the decimal point will be displayed.
Display Resolution	[Step size].	This property defines the resolution of the displayed values. Max resolution is about 0.4 (100/255)

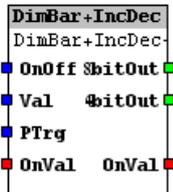
9.11. Dimmer bar graph + Increase/decrease Vertical (Dimmer Bar Inc/Dec Vertical)
Button:

The Dimmer Bar Inc/Dec Vertical is designed especially to control and monitor dimmer actor sending 4 bit and 8 bit telegrams. Its central part has the functionality of Dimmer bar **Button**.

Its top and bottom parts are push **Buttons** made for sending 4 bit control telegrams.



9.11.1. Functional description:



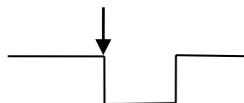
Bar graph:

The *Input Method* parameter determines if the user interface for input will be *keypad* or *touch point*.

In cases where the *Input Method* is *touch point*:

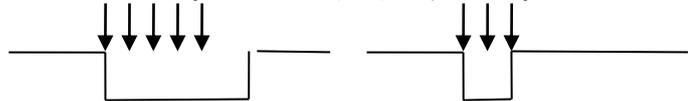
The new value will be determined by the point pressed on the Bar area. It is also possible to push and drag over the bar towards the desired value. When the **User** drags beyond the bar area to the top - the value entered will be the maximum value (100%). When the **User** drags beyond the bar area to the bottom - the value entered will be the minimum value (0%).

In cases where the *Input mode* property = *jump*



On push - immediately transmits the new value out of **8 bit Out Output**.

In cases where the *Input mode* property = *step*



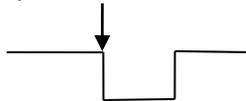
As long as button is pushed/dragged - the value advances towards the new value gradually in steps according to *Step size (%)* and *Steps per sec* properties. Transmits the values out of **8 bit Out Output**.

In cases where the *Input Method* is *keypad*:

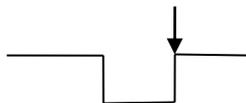


When the keypad Enter button is pressed - Transmits the new value out of **8 bit Out Output**.

Upper part:

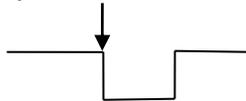


On Push - Transmits "1001b" out of **4 bit Out Output**

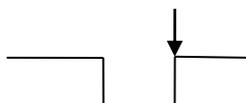


On release – Transmits "0000b" out of **4 bit Out Output**

Lower part:



On Push - Transmits "0001b" out of **4 bit Out Output**





On release – Transmits "0000b" out of *4 bit Out* **Output**

9.11.2. Functional properties:

Icon	Property/Group	Value	Description
	<i>Dimmer Bar Inc/Dec Vertical ID</i>	Text Preview	Unique reference ID of the button.
	<i>Receive</i>		
→■	<i>On/Off 1 bit (1BitIn)</i>	List of the links made with this point.	The only Group Address type that is linkable to this point is a 1bit Group Address. When an event with the value False arrives to this input – the bar is set to 0. When an event with the value True arrives to this input - the bar is set to the value of the <i>1bit "On" Value</i> parameter.
→■	<i>Value 8 bit (8 bitIn)</i>	List of the links made with this point.	The only Group Address type that is linkable to this point is an 8 bit Group Address. The bar will automatically scale input values in the range of: 0-255 to display values in the range of: 0-100.
	<i>Transmit</i>		
■→	<i>4 bit Out (4bitOut)</i>	List of the links made with this point.	The only Group Address type that is linkable to this point is a 4 bit Group Address. The commands: INCREASE, DECREASE and STOP will be sent through this output when the User pushes the upper and lower parts of the button.
■→	<i>8 bit Out (8bitOut)</i>	List of the links made with this point.	The only Group Address type that is linkable to this point is an 8 bit Group Address. New values will be sent through this output any time the User updates the bar level. The bar will automatically scale input values in the range of: 0-100 to transmitted values in the range of: 0-255.
	<i>Parameters</i>		
	<i>Start Up Value</i>	Real value: 0 to 255	The initial value of the Button immediately after startup. This value is used only when <i>Init mode</i> is set to ' <i>init value</i> ' or when persistent data is not valid (e.g. after first download of the

			project). The bar will automatically scale input values range 0-255 to displayed values: 0-100.
	<i>Init mode</i>	Drop down menu : Last Init Value	Mode of operation: Last: immediately after start up, the button will have the same state it had before power down. Init Value: immediately after start up, the button will have the value defined by 'State on start up' property
→■→	<i>1bit "On" Value (OnVal)</i>	[Value]	This value will be assigned to the button when the value "1" is received from the <i>On/Off 1 bit</i> input. The bar will automatically scale input values in the range of 0-255 to displayed values in the range of: 0-100.
	<i>Polling</i>		
	<i>Polling on start up</i>	Check box	Polls the <i>Value (8bit)</i> input, once, immediately after the Device startup.
	<i>Polling on display</i>	Drop down menu: Disabled Once 1s 5s 15s 30s 3min 15min 1h 6h 1day.	As long as the object is on display polls the <i>Value 8 bit</i> input cyclically using the interval specified.
	<i>Cyclic polling</i>	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the <i>Value 8 bit</i> input cyclically at all times using the interval specified.
	<i>Polling triggers</i>		
→■	<i>Triggers(PTrg)</i>	List of the links made with this point.	Event on any of the links made for this input will trigger a series of polling commands to the <i>Value 8bit</i> input.
	<i>Triggered polling interval</i>	Drop down menu: 1s 5s 15s 30s 60s	The total time duration of triggered polling.
	<i>Triggered polling pause</i>	Drop down menu: 1s 5s 15s 30s 60s	The time duration between single polling commands during triggered polling interval.

9.11.3. graphic properties:

Property/Group	Value	Description
<i>Dimmer Bar Inc/Dec Vertical ID</i>	Text Preview	Unique reference ID of the button.
<i>Frame Type</i>	Drop down menu: [list of available frames]	Use this to select the frame style.
<i>Frame color</i>	Color palette And Drop down menu: [list of basic colors].	Select Frame color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Frame Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Input Method</i>	Drop down menu: Up/Down Up/Down + Bar Up/Down + Keypad.	Up/Down = When the Upper/lower part of the button is pressed - Up/Down 4bit control command will be assigned to Dimm (4bit) output. When released – the 4bit control command Stop will be assigned. Bar = User can use the bar to assign new values for the button.. The new value is defined by the location of the push/drag point over the bar. Keypad = when the User touch the bar a keypad will pop up. The keypad enables the User to assign specific value to the button.
<i>Keypad title Text</i>	Text preview	The title is displayed on the upper part of the popup keypad.
<i>Bar Input mode</i>	Drop down menu: Jump to new value step to new value	Jump - Jump immediately to touch point position Step – advance in steps towards touch point position. Not visible when "Display only", is selected
<i>Step size (%)</i>	[Step]	The size (%) of single step towards the new value. Visible when "step to new value" is selected.

<i>Steps per sec</i>	Drop down menu: 1 2 3 4.	The Number of times per second that the bar graph value will be updated. Visible when "step to new value" is selected.
<i>Password</i>	Check box	The User will be asked for the password in order to perform the operation. This function is not available when the Input Method is "None".
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.
<i>button parts properties</i>	<i>button parts properties</i>	
Upper part	Radio button. Mutually exclusive between: Upper part, Central part and Lower part.	When you select it, – the program will display and allow you to edit the properties of the upper part of the button, (example: text and background). The upper part properties are listed in the remainder of this table.
Central part	Radio button. Mutually exclusive between: Upper part, Central part and Lower part.	When you select it, – the program will display and allow you to edit the properties of the central/bar part of the button, (example: text and background). The central/bar part properties are listed in the remainder of this table.
Lower part	Radio button. Mutually exclusive between: Upper part, Central part and Lower part.	When you select it, – the program will display and allow you to edit the properties of the central part of the button, (example: text and background). The lower part properties are listed in the remainder of this table.
Released state	Radio button. Mutually exclusive between: Pushed state and Released state.	When you select it, – the program will display and allow you to edit the Released state properties of the button, (example: icon and background). The Released state properties are listed in the remainder of this table.

Pushed state	Radio button. Mutually exclusive between: Pushed state and Released state.	When you select it, – the program will display and allow you to edit the Pushed state properties of the button, (example: icon and background). The Pushed state properties are listed in the remainder of this table.
Central/Bar properties		These properties are for the bar area of the button only!!!
Text and Font		
<i>Text</i>	Text Preview	Use this property to edit the button Text. When you use the \$ sign on the central part, the \$ Sign will be replaced with the dynamic value of the button.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
Bar graphic		
<i>Bar Background color</i>	Color palette And Drop down menu: [list of basic colors].	Select Bar Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Bar color</i>	Color palette And Drop down menu: [list of basic colors].	Select Bar color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images. This image will be placed on top of the bar.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the

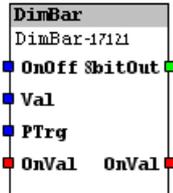
		container dimensions.
Bar numeric		
<i>Current value %</i>	[Current value]	Use this to preview the appearance of the button while displaying different values. Values can be from 0 to 100 and resolution is about 0.4 (100/255)
<i>Show ticks</i>	Check box	Display Ticks on the left side of the Bar.
<i>Show ticks value</i>	Check box	Display Bar values on the left side of the Bar.
<i>Number of Ticks</i>	[Number of Ticks]	The number of Ticks and/or value label displayed
<i>Ticks Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Ticks color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Digits after decimal point</i>	Number of digits.	This property defines how many digits to the right of the decimal point will be displayed.
<i>Display Resolution</i>	[Step size]	This property defines the resolution of the displayed values. Max resolution is about 0.4 (100/255)
Upper and lower button part properties		These properties are for the upper and lower areas of the button only!!!
Text and Font		
<i>Text</i>	Text Preview	Use this property to edit the button Text.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Text Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.

<i>Background</i>		
<i>Background color</i>	Color palette And Drop down menu: [list of basic colors].	Select Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.
<i>Icon/Indicator</i>		
<i>Icon</i>	Icon Image Preview	Use this to search your PC for Icons.
<i>Icon Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Icon.
<i>Icon Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Icon.

- 9.12. Dimmer Horizontal bar graph (Dimmer Bar Horizontal) **Button:**
 Dimmer Bar Horizontal is designed especially to transmit absolute values used to control Dimmer actors using 8 bit telegrams. You can use it also to display values 0 to 100 (%):



9.12.1. Functional description:

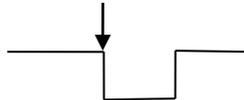


The *Input Method* parameter determines if the user interface for input will be *keypad* or *touch point*.

In cases where the *Input Method* is *touch point*:

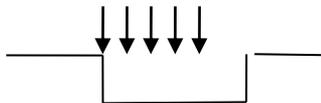
The new value will be determined by the point pressed on the Bar area. It is also possible to push and drag over the bar towards the desired value. When the **User** drags beyond the bar area to the right - the value entered will be the maximum value (100%). When the **User** drags beyond the bar area to the left - the value entered will be the minimum value (0%).

In cases where the *Input mode* property = *jump*



On push - immediately Transmits the value out of **8 bit Out Output**.

In case *Input mode* property = *step*



As long as button is pushed/dragged - the value advances towards the new value gradually in steps according to *Step size (%)* and *Steps per sec* properties. Transmits the values out of **8 bit Out Output**.

In cases where the *Input Method* is *keypad*:



When the keypad Enter button is pressed - Transmits the new value out of **8bit Out Output**.

9.12.2. Functional properties:

	<i>Property/Group</i>	<i>Value</i>	<i>Description</i>
	<i>Dimmer Bar Horizontal ID</i>	Text Preview	Unique reference ID of the button.
	<i>Receive</i>		
→■	<i>On/Off 1bit (OnOf)</i>	List of the links made with this point.	The only Group Address type that is linkable to this point is a 1bit Group Address. When an event with the value False arrives to this input – the bar is set to 0. When an event with the value True arrives to this input - the bar is set to the value of the 1bit "On" Value parameter.
→■	<i>Value 8 bit (8bitIn)</i>	List of the links made with this point.	The only Group Address type that is linkable to this point is an 8 bit Group Address. The bar will automatically scale input values in the range of: 0-255 to display values in the range of: 0-100.

Transmit			
-	<i>8 bit Out (8bitOut)</i>	List of the links made with this point.	The only Group Address type that is linkable to this point is an 8 bit Group Address. New values will be sent through this output any time the User updates the bar level. The bar will automatically scale input values in the range of: 0-100 to transmitted values: in the range of 0-255.
Parameters			
	<i>Start Up Value</i>	Real value: 0 to 255	The initial value of the Button immediately after startup. This value is used only when <i>Init mode</i> is set to ' <i>init value</i> ' or when persistent data is not valid (e.g. after first download of the project). The bar will automatically scale input values range 0-255 to displayed values: 0-100.
	<i>Init mode</i>	Drop down menu : Last Init Value	Mode of operation: Last: immediately after start up, the button will have the same value it has before power down. Init Value: immediately after start up, the button will have the value defined by ' <i>State on start up</i> ' property
→■→	<i>1bit "On" Value (OnVal)</i>	Real value: 0 to 255	This value will be assigned to the button when the value "1" is received from the <i>On/Off 1bit</i> input. The bar will automatically scale input values in the range of 0-255 to displayed values in the range of: 0-100.
Polling			
	<i>Polling on start up</i>	Check box	Polls the <i>Value 8bit</i> input, once, immediately after the Device startup.
	Polling on display	Drop down menu: Disabled Once 1s 5s 15s 30s 3min 15min 1h 6h 1day.	As long as the object is on display polls the <i>Value 8bit</i> input cyclically using the interval specified.

	<i>Cyclic polling</i>	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day.	Polls the <i>Value 8bit</i> input cyclically at all times using the interval specified.
	<i>Polling triggers</i>		
→■	<i>Triggers (PTrg)</i>	List of the links made with this point.	Event on any of the links made for this input will trigger a series of polling commands to the <i>Value 8bit</i> input.
	<i>Triggered polling interval</i>	Drop down menu: 1s 5s 15s 30s 60s	The total time duration of triggered polling.
	<i>Triggered polling pause</i>	Drop down menu: 1s 5s 15s 30s 60s	The time duration between single polling commands during triggered polling interval.

9.12.3. Graphic properties:

Property/Group	Value	Description
<i>Dimmer Bar Horizontal ID</i>	Text Preview	Unique reference ID of the button.
<i>Frame Type</i>	Drop down menu: [list of available frames]	Use this to select the frame style.
<i>Frame color</i>	Color palette And Drop down menu: [List of basic colors].	Select Frame color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Frame Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Input Method</i>	Drop down menu: None Bar Keypad.	None = this button will not perform any action as a result of User touch. You can use the button to display status. Bar = User can use the bar to assign new values for the button. The new value is defined by the location of the push/drag point over the bar. Keypad = when the User touch the button a keypad will pop up. The keypad enables the User to assign specific value to the button.
<i>Keypad title Text</i>	Text preview	The title is displayed on the upper part of the popup keypad.
<i>Input mode</i>	Drop down menu: Jump to new value step to new value	Jump - Jump immediately to touch point position Step – advance in steps towards touch point value. Not visible when "Display only", is selected
<i>Step size (%)</i>	[Step]	The size (%) of single step towards the new value. Visible when "step to new value" is selected.
<i>Steps per sec</i>	Drop down menu: 1 2 3 4.	The Number of times per second that the bar graph value will be updated. Visible when "step to new value" is selected.
<i>Password</i>	Check box	The User will be asked for the password in order to perform the operation. This function is not

		available when the Input Method is "None".
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.
<i>Bar properties</i>		
<i>Text and Font</i>		
<i>Text</i>	Text Preview	Use this property to edit the button Text. \$ Sign will be replaced with the dynamic value of the button.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
<i>Bar graphic</i>		
<i>Bar Background color</i>	Color palette And Drop down menu: [List of basic colors].	Select Bar Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange , Yellow, Green, Magenta, Cyan, Blue.
<i>Bar color</i>	Color palette And Drop down menu: [list of basic colors].	Select Bar color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for images. This image will be placed on top of the bar.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Icon.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Icon.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.

<i>Bar numeric</i>		
Current value %	[Current value]	Use this to preview the appearance of the button while displaying different values. Values can be from 0 to 100 and resolution is about 0.4 (100/255)
Show ticks	Check box	Display Ticks on the left side of the Bar.
Show ticks value	Check box	Display Bar values on the left side of the Bar.
Number of Ticks	[Number of Ticks]	The number of Ticks and/or value label displayed
<i>Ticks Color</i>	Color palette And Drop down menu: [list of basic colors].	Select Ticks color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
Digits after decimal point	[Number of digits].	This property defines how many digits to the right of the decimal point will be displayed.
Display Resolution	[Step size].	This property defines the resolution of the displayed values. Max resolution is about 0.4 (100/255)

9.13. Dimmer bar graph + Increase/decrease Horizontal (Dimmer Bar Inc/Dec Horizontal) **Button**:

Dimmer Bar Inc/Dec Horizontal is designed especially to control and monitor dimmer actor sending 4 bit and 8 bit telegrams. Its central part has the functionality of Dimmer bar **Button**.

its left and right parts are push **Buttons** sending 4 bit control telegrams.



9.13.1. Functional description:



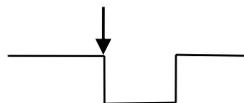
Bar graph:

The *Input Method* parameter determines if the user interface for input will be *keypad* or *touch point*.

In cases where the *Input Method* is *touch point*:

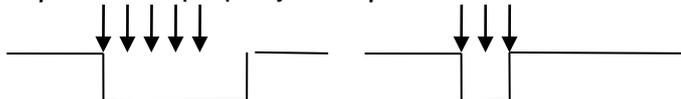
The new value will be determined by the point pressed on the Bar area. It is also possible to push and drag over the bar towards the desired value. When the **User** drags beyond the bar area to the right - the value entered will be the maximum value (100%). When the **User** drags beyond the bar area to the left - the value entered will be the minimum value (0%).

In cases where the *Input mode* property = *jump*



On push - immediately Transmits the new value out of **8 bit Out Output**.

In case *Input mode* property = *step*



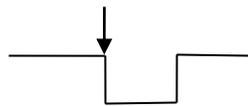
As long as button is pushed/dragged - the value advances towards the new value gradually in steps according to *Step size (%)* and *Steps per sec* properties. Transmits the values out of **8 bit Out Output**.

In cases where the *Input Method* is *keypad*:

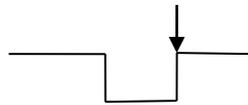


When the keypad Enter button is pressed - Transmits the new value out of **8bit Out Output**.

Right part:

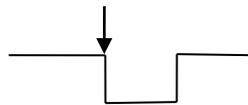


On Push - Transmits "1001b" out of **4 bit Out Output**

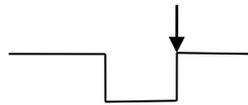


On release – Transmits "0000b" out of **4 bit Out Output**

Left part:



On Push - Transmits "0001b" out of **4 bit Out Output**



On release – Transmits "0000b" out of **4 bit Out Output**

9.13.2. Functional properties:

Icon	Property/ Group	Value	Description
	<i>Dimmer Bar Inc/Dec Horizontal ID</i>	Text Preview	Unique reference ID of the button.
	<i>Receive</i>		
→■	<i>On/Off 1 bit (1BitIn)</i>	List of the links made with this point.	The only Group Address type that is linkable to this point is a 1bit Group Address. When an event with the value False arrives to this input – the bar is set to 0. When an event with the value True arrives to this input - the bar is set to the value of the <i>1bit "On" Value</i> parameter.
→■	<i>Value 8 bit (8 bitIn)</i>	List of the links made with this point.	The only Group Address type that is linkable to this point is an 8 bit Group Address. The bar will automatically scale input values in the range of: 0-255 to display values in the range of: 0-100.
	<i>Transmit</i>		
■→	<i>4 bit Out (4bitOut)</i>	List of the links made with this point.	The only Group Address type that is linkable to this point is a 4 bit Group Address. The commands: INCREASE, DECREASE and STOP will be sent through this output when the User pushes the right and left parts of the button.
■→	<i>8 bit Out (8bitOut)</i>	List of the links made with this point.	The only Group Address type that is linkable to this point is an 8 bit Group Address. New values will be sent through this output any time the User updates the bar level. The bar will automatically scale input values in the range of: 0-100 to transmitted values in the range of: 0-255.
	<i>Parameters</i>		
	<i>Start Up Value</i>	Real value: 0 to 255	The initial value of the Button immediately after startup. This value is used only when <i>Init mode</i> is set to ' <i>init value</i> ' or when persistent data is not valid (e.g. after first download of the

			project). The bar will automatically scale input values range 0-255 to displayed values: range 0-100.
	<i>Init mode</i>	Drop down menu : Last Init Value	Mode of operation: Last: immediately after start up, the button will have the same state it had before power down. Init Value: immediately after start up, the button will have the value defined by 'State on start up' property
→■→	<i>1bit "On" Value (OnVal)</i>	[Value]	This value will be assigned to the button when the value "1" is received from the <i>On/Off 1 bit</i> input. The bar will automatically scale input values in the range of 0-255 to displayed values in the range of: 0-100.
	<i>Polling</i>		
	<i>Polling on start up</i>	Check box	Polls the <i>Value (8bit)</i> input, once, immediately after the Device startup.
	<i>Polling on display</i>	Drop down menu: Disabled Once 1s 5s 15s 30s 3min 15min 1h 6h 1day.	As long as the object is on display polls the <i>Value 8 bit</i> input cyclically using the interval specified.
	<i>Cyclic polling</i>	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1d ay.	Polls the <i>Value 8 bit</i> input cyclically at all times using the interval specified.
	<i>Polling triggers</i>		
→■	<i>Triggers(PTrg)</i>	List of the links made with this point.	Event on any of the links made for this input will trigger a series of polling commands to the <i>Value 8bit</i> input.
	<i>Triggered polling interval</i>	Drop down menu: 1s 5s 15s 30s 60s	The total time duration of triggered polling.
	<i>Triggered polling pause</i>	Drop down menu: 1s 5s 15s 30s 60s	The time duration between single polling commands during triggered polling interval.

9.13.3. graphic properties:

Property/Group	Value	Description
<i>Dimmer Bar Inc/Dec Horizontal ID</i>	Text Preview	Unique reference ID of the button.
<i>Frame Type</i>	Drop down menu: [list of available frames]	Use this to select the frame style.
<i>Frame color</i>	Color palette And Drop down menu: [list of basic colors].	Select Frame color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Frame Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Input Method</i>	Drop down menu: Left/Right Left/Right + Bar Left/Right + Keypad.	Left/Right = When the Right or Left part of the button is pressed – Up or Down 4bit control command will be assigned to Dimm (4bit) output. When released – the 4bit control command Stop will be assigned. Bar = User can use the bar to assign new values for the button.. The new value is defined by the location of the push/drag point over the bar. Keypad = when the User touch the bar a keypad will pop up. The keypad enables the User to assign specific value to the button.
<i>Keypad title Text</i>	Text preview	The title is displayed on the upper part of the popup keypad.
<i>Bar Input mode</i>	Drop down menu: Jump to new value step to new value	Jump - Jump immediately to touch point position Step – advance in steps towards touch point position. Not visible when “Display only”, is selected
<i>Step size (%)</i>	[Step]	The size (%) of single step towards the new value. Visible when "step to new value" is selected.

<i>Steps per sec</i>	Drop down menu: 1 2 3 4.	The Number of times per second that the bar graph value will be updated. Visible when "step to new value" is selected.
<i>Password</i>	Check box	The User will be asked for the password in order to perform the operation. This function is not available when the Input Method is "None".
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.
<i>button parts properties</i>	<i>button parts properties</i>	
Right part	Radio button. Mutually exclusive between: Right part, Central part and left part.	When you select it, – the program will display and allow you to edit the properties of the right part of the button, (example: text and background). The right part properties are listed in the remainder of this table.
Central part	Radio button. Mutually exclusive between: Right part, Central part and Left part.	When you select it, – the program will display and allow you to edit the properties of the central/bar part of the button, (example: text and background). The central/bar part properties are listed in the remainder of this table.
left part	Radio button. Mutually exclusive between: Right part, Central part and Left part.	When you select it, – the program will display and allow you to edit the properties of the left part of the button, (example: text and background). The left part properties are listed in the remainder of this table.
Released state	Radio button. Mutually exclusive between: Pushed state and Released state.	When you select it, – the program will display and allow you to edit the Released state properties of the button, (example: icon and background). The Released state properties are listed in the remainder of this table.

Pushed state	Radio button. Mutually exclusive between: Pushed state and Released state.	When you select it, – the program will display and allow you to edit the Pushed state properties of the button, (example: icon and background). The Pushed state properties are listed in the remainder of this table.
<i>Central/bar properties</i>		These properties are for the bar area of the button only!!!
<i>Text and Font</i>		
<i>Text</i>	Text Preview	Use this property to edit the button Text. When you use the \$ sign on the central part, the \$ Sign will be replaced with the dynamic value of the button.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Color</i>	Color palette And Drop down menu: [list of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
<i>Bar graphic</i>		
<i>Bar Background color</i>	Color palette And Drop down menu: [list of basic colors].	Select Bar Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Bar color</i>	Color palette And Drop down menu: [list of basic colors].	Select Bar color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images. This image will be placed on top of the bar.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the

		container dimensions.
Bar numeric		
<i>Current value %</i>	[Current value]	Use this to preview the appearance of the button while displaying different values. Values can be from 0 to 100 and resolution is about 0.4 (100/255)
<i>Show ticks</i>	Check box	Display Ticks on the left side of the Bar.
<i>Show ticks value</i>	Check box	Display Bar values on the left side of the Bar.
<i>Number of Ticks</i>	[Number of Ticks]	The number of Ticks and/or value label displayed
<i>Ticks Color</i>	Color palette And Drop down menu: [list of basic colors].	Select Ticks color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Digits after decimal point</i>	Number of digits.	This property defines how many digits to the right of the decimal point will be displayed.
<i>Display Resolution</i>	[Step size]	This property defines the resolution of the displayed values. Max resolution is about 0.4 (100/255)
Right and Left button part properties		These properties are for the right and left areas of the button!!!
Text and Font		
<i>Text</i>	Text Preview	Use this property to edit the button Text.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Text Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.

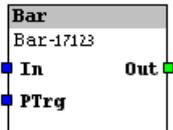
<i>Background</i>		
<i>Background color</i>	Color palette And Drop down menu: [List of basic colors].	Select Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.
<i>Icon/Indicator</i>		
<i>Icon</i>	Icon Image Preview	Use this to search your PC for Icons.
<i>Icon Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Icon.
<i>Icon Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Icon.

9.14. Bar Graph (Bar) **Button**:

A Bar is a general purpose **Button**. It is designed to transmit analog values. You can use it to transmit any numeric data type and value. You can use it also to display analog values:



9.14.1. Functional description:

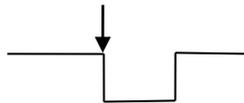


The *Input Method* parameter determines if the user interface for input will be *keypad* or *touch point*.

In cases where the *Input Method* is *touch point*:

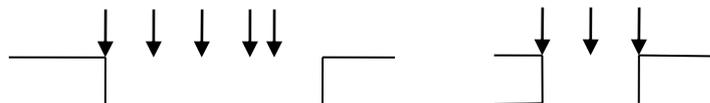
The new value will be determined by the point pressed on the Bar area. It is also possible to push and drag over the bar towards the desired value. When the User drags beyond the bar area to the top - the value entered will be the maximum value. When the User drags beyond the bar area to the bottom - the value entered will be the minimum value.

In cases where the *Input mode* property = *jump*



On push - immediately Transmits the new value out of **Out** **Output**.

In cases where the *Input mode* property = *step*



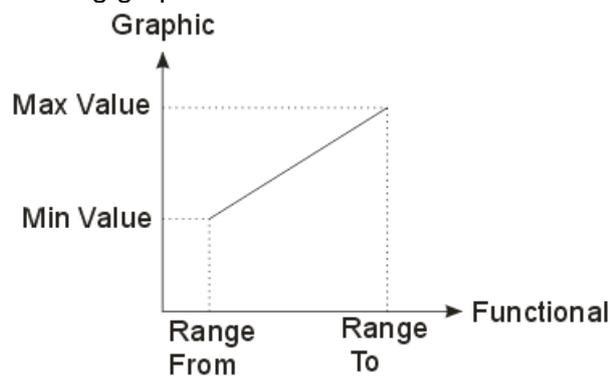
As long as the button is pushed/dragged – the transmitted value advances gradually in steps according to *Step size (%)* and *Steps per sec* properties, until the value reaches the current touch point value. Transmits the values out of **Out Output**.

In cases where the *Input Method* is *Keypad*:



On acceptance of new value - Immediately transmits the new value out of **Out Output**.

The Bar utilizes linear transformations between its graphic properties (display limits) and the functional properties (input and output limits) according to the following graph:



9.14.2. Functional properties:

Icon	Property/Group	Value	Description
	Bar ID	<i>Text Preview</i>	Unique reference ID of the button.
	<i>Receive</i>		
	<i>In</i>	List of the links made with this point.	The bar will display values received from this input. The bar will automatically scale the input values range set by the “Range from” and “Range to” functional properties, to the user interface values range set by “Max value” and “Min value” graphical properties.
	<i>Transmit</i>		
	<i>Out</i>	List of the links made with this point.	New values will be sent through this output any time the user updates the bar level. The bar will automatically scale the user interface values range set by the “Max value” and “Min value” graphical properties, to the output values range set by the “Range from” and “Range to” functional properties.
	<i>Parameters</i>		
	<i>Start Up Value</i>	[Value]	The initial value of the Button immediately after startup. This value is used only when <i>Init mode</i> is set to ' <i>init value</i> ' or when persistent data is not valid (e.g. after first download of the project)
	<i>Init mode</i>	Drop down menu : Last Init Value	Mode of operation: <i>Last</i> : immediately after start up, the button will have the same value it has before power down. <i>Init Value</i> : immediately after start up, the button will have the value defined by ' <i>State on start up</i> ' property
	Range To	[Value]	The upper limit of the bar
	Range From	[Value]	The lower limit of the bar

<i>Polling</i>			
	<i>Polling on start up</i>	Check box	Polls the <i>ln</i> Input, once, immediately after the Device startup.
	<i>Polling on display</i>	Drop down menu: Disabled Once 1s 5s 15 30s 3min 15min 1h 6h 1day.	As long as the object is on display polls the <i>ln</i> input cyclically using the interval specified.
	<i>Cyclic polling</i>	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the <i>ln</i> input cyclically at all times using the interval specified.
<i>Polling triggers</i>			
→■	<i>Triggers</i>	List of the links made with this point.	Event on any of the links made for this input will trigger a series of polling commands to the <i>ln</i> input.
	<i>Triggered polling interval</i>	Drop down menu: 1s 5s 15s 30s 60s	The total time duration of triggered polling.
	<i>Triggered polling pause</i>	Drop down menu: 1s 5s 15s 30s 60s	The time duration between single polling commands during triggered polling interval.

9.14.3. Graphic properties:

Property/Group	Value	Description
<i>Bar ID</i>	<i>Text Preview</i>	Unique reference ID of the button.
<i>Frame Type</i>	Drop down menu: [list of available frames]	Use this to select the frame style.
<i>Frame Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Frame Color</i>	Color palette And Drop down menu: [list of basic colors].	Select Frame color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Input Method</i>	Drop down menu: None Bar Keypad.	None = this button will not perform any action as a result of User touch. You can use the button to display status. Bar = User can use the bar to assign new values for the button.. The new value is defined by the location of the push/drag point over the bar. Keypad = when the User touch the button a keypad will pop up. The keypad enables the User to assign specific value to the button.
<i>Keypad title Text</i>	Text preview	The title is displayed on the upper part of the popup keypad.
<i>Input mode</i>	Drop down menu: Jump to new value step to new value	Jump - Jump immediately to touch point position Step – advance in steps towards touch point position. Not visible when “Display only”, is selected
<i>Step size (%)</i>	[Step]	The size (%) of single step towards the new value. Visible when "step to new value" is selected.
<i>Steps per sec</i>	Drop down menu: 1 2 3 4.	The Number of times per second that the bar graph value will be updated. Visible when "step to new value" is selected.

<i>Password</i>	Check box	The User will be asked for the password in order to perform the operation. This function is not available when the Input Method is "None".
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.
Bar properties		
Text and Font		
<i>Text</i>	Text Preview	Use this property to edit the button Text. \$ Sign will be replaced with the dynamic value of the button.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
Bar graphic		
<i>Bar Background color</i>	Color palette And Drop down menu: [list of basic colors].	Select Bar Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Bar color</i>	Color palette And Drop down menu: [List of basic colors].	Select Bar color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images. This image will be placed on top of the bar.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the

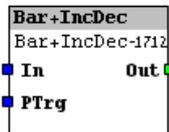
		container dimensions.
<i>Bar numeric</i>		
<i>Current value</i>	[Current value]	Use this to preview the appearance of the button while displaying different values.
<i>Max value</i>	[Max value]	The value of the highest point of the Bar.
<i>Min value</i>	[Min value]	The value of the lowest point of the Bar.
<i>Show ticks</i>	Check box	Display Ticks on the left side of the Bar.
<i>Show ticks value</i>	Check box	Display Bar values on the left side of the Bar.
<i>Number of Ticks</i>	[Number of Ticks]	The number of Ticks and/or values label to display.
<i>Ticks Color</i>	Color palette And Drop down menu: [list of basic colors].	Select Ticks and/or values color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Digits after decimal point</i>	Number of digits.	This property defines how many digits to the right of the decimal point will be displayed.
<i>Display Resolution</i>	Step size.	This property defines the resolution of the displayed values.

9.15. Bar Graph + Increase /Decrease (Bar+Inc/Dec) **Button**:

Bar + Inc/Dec Is a general purpose **Button**. It is designed to transmit analog values. **User** can enter new values by pressing on the bar as well as by pressing the upper and lower parts of the **Button**. You can use it to transmit any numeric data type and value. You can also use it to display analog values:



9.15.1. Functional description:



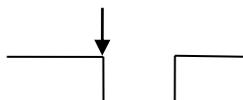
Bar graph:

The *Input Method* parameter determines if the user interface for input will be *keypad* or *touch point*.

In cases where the *Input Method* is *touch point*:

The new value will be determined by the point pressed on the Bar area. It is also possible to push and drag over the bar towards the desired value. When **the User** drags above and beyond the bar area, the value entered will be the maximum value. When the **User** drags below and beyond the bar, the value entered will be the minimum value.

In cases where the *Input mode* property = *jump*



On push/drag - Immediately transmits the new value out of **Value Output**.

In cases where the *Input mode* property = *step*



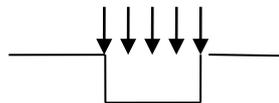
As long as button is pushed/dragged – the transmitted value advances gradually, in steps according to *Step size (%)* and *Steps per sec* properties, until the value reaches the current touch point value. Transmits the values out of **Out Output**.

In case *Input Method* is *Keypad*:



On acceptance of the new value - Immediately transmits, the new value out of **Out Output**.

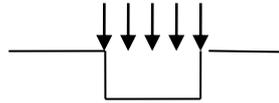
Upper part:



As long as **Button** is pushed

1. Until **Out** = *Max Value*: Increase value by *Step size*, transmit the new value out of **Out Output**.
2. Wait 1sec/ *Steps per sec*.
3. Go to step 1

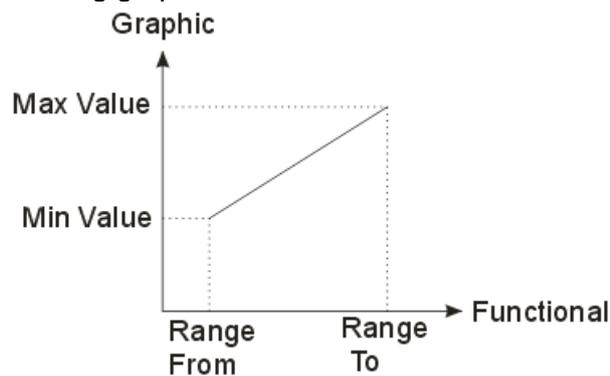
Lower part:



As long as **Button** is pushed

1. Until *Out* = *Min Value*: decrease value by *Step size*, transmit the new value out of *Out Output*.
2. Wait 1sec/ *Steps per sec*.
3. Go to step 1.

The Bar utilizes linear transformations between its graphic properties (display limits) and the functional properties (input and output limits) according to the following graph:



9.15.2. Functional properties:

Icon	Property/Group	Value	Description
	<i>Bar + Inc/Dec</i>	Text Preview	Unique reference ID of the button.
	<i>Receive</i>		
→■	<i>In</i>	List of the links made with this point.	The bar will display values received from this input.
	<i>Transmit</i>		
■→	<i>Out</i>	List of the links made with this point.	New values will be sent through this output any time the User updates the bar level.

Parameters			
	<i>Start Up Value</i>	[Value]	The initial value of the Button immediately after startup. This value is used only when <i>Init mode</i> is set to ' <i>init value</i> ' or when persistent data is not valid (e.g. after first download of the project)
	<i>Init mode</i>	Drop down menu : Last Init Value	Mode of operation: Last: immediately after start up, the button will have the same state it had before power down. Init Value: immediately after start up, the button will have the value defined by ' <i>Start up Value</i> ' property
	Range To	[Value]	The upper limit of the bar
	Range From	[Value]	The lower limit of the bar
Polling			
	<i>Polling on start up</i>	Check box	Polls the <i>In Input</i> , once, immediately after the Device startup.
	Polling on display	Drop down menu: Disabled Once 1s 5s 15s 30s 3min 15min 1h 6h 1day.	As long as the object is on display polls the <i>In Input</i> cyclically using the interval specified.
	<i>Cyclic polling</i>	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the <i>In Input</i> cyclically at all times using the interval specified.
Polling triggers			
→■	<i>Triggers(PTrg)</i>	List of the links made with this point.	Event on any of the links made for this input will trigger a series of polling commands to the <i>In</i> input.
	<i>Triggered polling interval</i>	Drop down menu: 1s 5s 15s 30s 60s	The total time duration of triggered polling.
	<i>Triggered polling pause</i>	Drop down menu: 1s 5s 15s 30s 60s	The time duration between single polling commands during triggered polling interval.

9.15.3. Graphic properties

Property/Group	Value	Description
<i>Bar + Inc/Dec ID</i>	Text Preview	Unique reference ID of the button.
Global Properties		
<i>Frame Type</i>	Drop down menu: [list of available frames]	Use this to select the frame style.
<i>Frame color</i>	Color palette And Drop down menu: [List of basic colors].	Select Frame color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Frame Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Input Method</i>	Drop down menu: Inc/Dec Inc/Dec + bar Inc/Dec + Keypad.	Inc/Dec = when the upper/lower part of the button is pressed the button value will be decreased/increased. bar = User can use the bar to assign new values for the button.. The new value is defined by the location of the push/drag point over the bar. Keypad = when the User touch the bar a keypad will pop up. The keypad enables the User to assign specific value to the button.
<i>Keypad title Text</i>	Text preview	The title is displayed on the upper part of the popup keypad.
<i>Input mode</i>	Drop down menu: Jump to new value step to new value	Jump - Jump immediately to touch point position Step – advance in steps towards touch point position. Not visible when “Display only”, is selected
<i>Step size (%)</i>	[Step size]	The size (%) of single step towards the new value. Visible when "step to new value" is selected.
<i>Steps per sec</i>	Drop down menu: 1 2 3 4.	The Number of times per second that the bar graph value will be updated. Visible when "step to new value" is selected.

<i>Password</i>	Check box	The User will be asked for the password in order to perform the operation. This function is not available when the Input Method is "None".
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.
<i>button parts properties</i>	<i>button parts properties</i>	
Upper part	Radio button. Mutually exclusive between: Upper part, Central part and Lower part.	When you select it, – the program will display and allow you to edit the properties of the upper part of the button, (example: text and background). The upper part properties are listed in the remainder of this table.
Central part	Radio button. Mutually exclusive between: Upper part, Central part and Lower part.	When you select it, – the program will display and allow you to edit the properties of the central/bar part of the button, (example: text and background). The central/bar part properties are listed in the remainder of this table.
Lower part	Radio button. Mutually exclusive between: Upper part, Central part and Lower part.	When you select it, – the program will display and allow you to edit the properties of the central part of the button, (example: text and background). The lower part properties are listed in the remainder of this table.
Released state	Radio button. Mutually exclusive between: Pushed state and Released state.	When you select it, – the program will display and allow you to edit the Released state properties of the button, (example: icon and background). The Released state properties are listed in the remainder of this table.
Pushed state	Radio button. Mutually exclusive between: Pushed state and Released state.	When you select it, – the program will display and allow you to edit the Pushed state properties of the button, (example: icon and background). The Pushed state properties are listed in the remainder of this table.

Central/Bar properties		These properties are for the bar area of the button only!!!
Text and Font		
<i>Text</i>	Text Preview	Use this property to edit the button Text. When you use the \$ sign on the central part, the \$ Sign will be replaced with the dynamic value of the button.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
Bar graphic		
<i>Bar Background color</i>	Color palette And Drop down menu: [list of basic colors].	Select Bar Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Bar color</i>	Color palette And Drop down menu: [List of basic colors].	Select Bar color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images. This image will be placed on top of the bar.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.
Bar numeric		
<i>Current value</i>	[Current value]	Use this to preview the appearance of the button while displaying different values.
<i>Max value</i>	[Max value]	The value of the highest point of the Bar.

<i>Min value</i>	[Min value]	The value of the lowest point of the Bar.
<i>Show ticks</i>	Check box	Display Ticks on the left side of the Bar.
<i>Show ticks value</i>	Check box	Display Bar values on the left side of the Bar.
<i>Number of Ticks</i>	[number of ticks value]	The number of Ticks and/or value label displayed
<i>Ticks Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Ticks color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Digits after decimal point</i>	Number of digits.	This property defines how many digits to the right of the decimal point will be displayed.
<i>Display Resolution</i>	Step size.	This property defines the resolution of the displayed values.
<i>Upper and lower button parts properties</i>		These properties are for the upper and lower areas of the button!!!
<i>Text and Font</i>		
<i>Text</i>	Text Preview	Use this property to edit the button Text.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Color</i>	Color palette And Drop Down menu: [list of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
<i>Background</i>		
<i>Background color</i>	Color palette And Drop down menu: [list of basic colors].	Select Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.

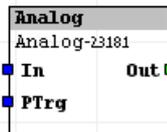
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.
<i>Icon/Indicator</i>		
<i>Icon</i>	<i>Icon Image Preview</i>	<i>Use this to search your PC for Icons.</i>
<i>Icon Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Icon.
<i>Icon Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Icon.

9.16. Analog alphanumeric **Button**:

An Analog alphanumeric **Button** is a general purpose **Button**. It is designed to transmit analog values. You can use it to transmit any numeric data type and value. You can also use it to display analog values:



9.16.1. Functional description:

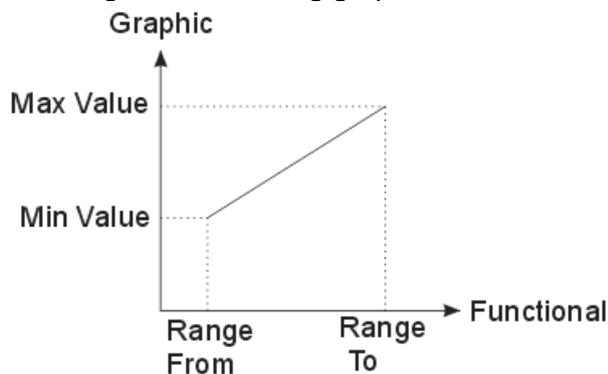


The input method for this button is Keypad:



When the keypad Enter button is pressed – the **Button** immediately transmits the new value out of **Out Output**.

An Analog alphanumeric **Button** utilizes linear transformation between its graphic properties (display limits) and the functional properties (input and output limits) according to the following graph:



9.16.2. Functional properties:

	Property/Group	Value	Description
	<i>Analog alphanumeric ID</i>	Text Preview	Unique reference ID of the button.
	<i>Receive</i>		
	<i>In</i>	List of the links made with this point.	The button will display values received from this input.
	<i>Transmit</i>		
	<i>Out</i>	List of the links made with this point.	New values will be sent through this output any time the User updates the button value.
	<i>Parameters</i>		
	<i>Start Up Value</i>	[Value]	The initial value of the Button immediately after startup. This value is used only when <i>Init mode</i> is set to ' <i>init value</i> ' or when persistent data is not valid (e.g. after first download of the project)
	<i>Init mode</i>	Drop down menu : Last Init Value	Mode of operation: Last: immediately after start up, the button will have the same state it had before power down. Init Value: immediately after start up, the button will have the value defined by ' <i>State on start up</i> ' property
	<i>Range To</i>	[Value]	The upper limit of the button
	<i>Range From</i>	[Value]	The lower limit of the button
	<i>Polling</i>		
	<i>Polling on start up</i>	Check box	Polls the <i>In</i> Input, once, immediately after the Device startup.
	<i>Polling on display</i>	Drop down menu: Disabled Once 1s 5s 15s 30s 3min 15min 1h 6h 1day.	As long as the object is on display polls the <i>In</i> Input cyclically using the interval specified.
	<i>Cyclic polling</i>	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the <i>In</i> Input cyclically at all times using the interval specified.

<i>Polling triggers</i>			
→■	<i>Triggers (PTrg)</i>	List of the links made with this point.	Whenever an event is detected on any of the links made for this point – the Device will perform a series of polling commands to the <i>Value</i> input.
	<i>Triggered polling interval</i>	Drop down menu: 1s 5s 15s 30s 60s	The total time duration of triggered polling.
	<i>Triggered polling pause</i>	Drop down menu: 1s 5s 15s 30s 60s	The time duration between single polling commands during triggered polling interval.

9.16.3. Graphic properties:

Property/Group	Value	Description
<i>Analog alphanumeric ID</i>	<i>Text Preview</i>	Unique reference ID of the button.
<i>Frame Type</i>	Drop down menu: [list of available frames]	Use this to select the frame style.
<i>Frame color</i>	Color palette And Drop down menu: [List of basic colors].	Select Frame color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Frame Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Input Method</i>	Drop down menu: None Keypad	None = this button will not perform any action as a result of User touch. You can use the button to display status. Keypad = when the User touch the button a keypad will pop up. The keypad enables the User to assign specific value to the button.
<i>Keypad title Text</i>	Text preview	The title is displayed on the upper part of the popup keypad.
<i>Password</i>	Check box	The User will be asked for the password in order to perform the operation. This function is not available when the Input Method is "None".
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.
Text and Font		
<i>Text</i>	Text Preview	Use this property to edit the button Text. \$ Sign will be replaced with the dynamic value of the button.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Text Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment</i>	Drop down menu:	Vertical alignment of the Text.

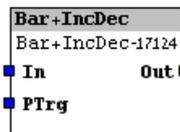
<i>(V)</i>	Top Center Bottom.	
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
<i>Background</i>		
<i>Fill color</i>	Color palette And Drop down menu: [List of basic colors].	Select Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.
<i>Numeric</i>		
<i>Current value</i>	[Current value]	Use this to preview the appearance of the button while displaying different values.
<i>Max value</i>	[Max value]	The maximum value that the button can have/display.
<i>Min value</i>	[Min value]	The minimum value that the button can have/display.
<i>Digits after decimal point</i>	Number of digits.	This property defines how many digits to the right of the decimal point will be displayed.
<i>Display Resolution</i>	Step size.	This property defines the resolution of the displayed values.

9.17. Analog Alpha numeric + Inc/Dec Vertical **Button**:

Analog Alphanumeric + Inc/Dec Vertical is a general purpose **Button**. It is designed to transmit analog values. You can use it to transmit any numeric data type and value. You can also use it to display values:



9.17.1. Functional description:

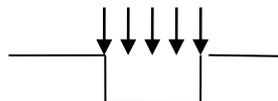


Central part - keypad:



When the keypad Enter button is pressed – The **Button** Transmits the new value out of **Out Output**.

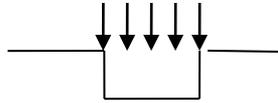
Upper part:



As long as the **Button** is pushed:

1. Until **Out** = Max value: Increase value by *Step size*, transmit the new value out of **Out Output**.
2. Wait 1sec/*Steps per sec*.
3. Go to step 1

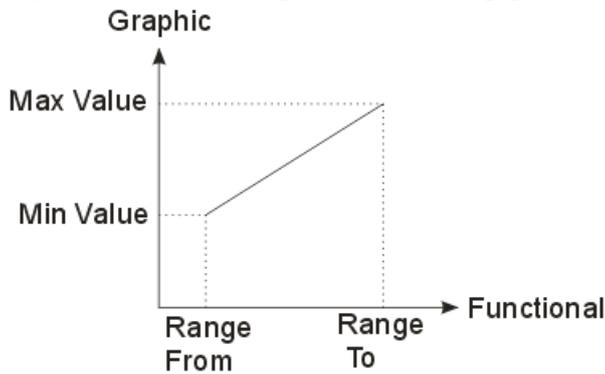
Lower part:



As long as **Button** is pushed:

1. Until **Out** = **Min Value**: decrease value by **Step size**, transmit the new value out of **Out Output**.
2. Wait 1sec/ **Steps per sec**.
3. Go to step 1.

An Analog Alphanumeric + Inc/Dec Vertical **Button** utilizes linear transformation between its graphic properties (display limits) and the functional properties (input and output limits) according to the following graph:



9.17.2. Functional properties:

	Property/Group	Value	Description
	<i>Alphanumeric Inc/Dec Vertical ID</i>	Text Preview	Unique reference ID of the button.
	<i>Receive</i>		
→■	<i>In</i>	List of the links made with this point.	The bar will display values received from this input.
	<i>Transmit</i>		
■→	<i>Out</i>	List of the links made with this point.	New values will be sent through this output any time the User updates the button value.
	<i>Parameters</i>		
	<i>Start up Value</i>	[Value]	The initial value of the Button immediately after startup. This value is used only when <i>Init mode</i> is set to ' <i>init value</i> ' or when persistent data is not valid (e.g. after first download of the project)

	<i>Init mode</i>	Drop down menu : Last Init Value	Mode of operation: Last: immediately after start up, the button will have the same state it had before power down. Init Value: immediately after start up, the button will have the value defined by 'State on start up' property
	Range To	[Value]	The upper limit of the button
	Range From	[Value]	The lower limit of the button
	<i>Polling</i>		
	<i>Polling on start up</i>	Check box	Polls the <i>Value</i> Input, once, immediately after the Device startup.
	Polling on display	Drop down menu: Disabled Once 1s 5s 15 30s 3min 15min 1h 6h 1day.	As long as the object is on display polls the <i>Value</i> Input cyclically using the interval specified.
	<i>Cyclic polling</i>	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day .	Polls the <i>Value</i> Input cyclically at all times using the interval specified.
	<i>Polling triggers</i>		
→■	<i>Triggers(PTrg)</i>	List of the links made with this point.	Whenever an event is detected on any of the links made for this point – the Device will perform a series of polling commands to the <i>Value</i> input.
	<i>Triggered polling interval</i>	Drop down menu: 1s 5s 15s 30s 60s	The total time duration of triggered polling.
	<i>Triggered polling pause</i>	Drop down menu: 1s 5s 15s 30s 60s	The time duration between single polling commands during triggered polling interval.

9.17.3. Graphic properties:

Property/Group	Value	Description
<i>Alpha numeric + Inc/Dec ID</i>	Text Preview	Unique reference ID of the button.
<i>Frame Type</i>	Drop down menu: [list of available frames]	Use this to select the frame style.
<i>Frame color</i>	Color palette And Drop down menu: [List of basic colors].	Select Frame color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Frame Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Input Method</i>	Drop down menu: Inc/Dec Inc/Dec + Keypad	Inc/Dec = when the Upper/Lower part of the button is pressed the button value will be decreased/increased. Keypad = when the User touch the buttons central part - a keypad will pop up. The keypad enables the User to assign specific value to the button.
<i>Keypad title Text</i>	Text preview	The title is displayed on the upper part of the popup keypad.
<i>Step size %</i>	[Step]	The size (%) of single step towards the new value. Visible when "step to new value" is selected.
<i>Steps per sec</i>	Drop down menu: 1 2 3 4.	The Number of times per second that the button value will be updated. Visible when "step to new value" is selected.
<i>Password</i>	Check box	The User will be asked for the password in order to perform the operation. This function is not available when the Input Method is "None".
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.

<i>button parts properties</i>	<i>button parts properties</i>	
Upper part	Radio button. Mutually exclusive between: Upper part, Central part and Lower part.	When you select it, – the program will display and allow you to edit the properties of the upper part of the button, (example: text and background). The upper part properties are listed in the remainder of this table.
Central part	Radio button. Mutually exclusive between: Upper part, Central part and Lower part.	When you select it, – the program will display and allow you to edit the properties of the central/bar part of the button, (example: text and background). The central/bar part properties are listed in the remainder of this table.
Lower part	Radio button. Mutually exclusive between: Upper part, Central part and Lower part.	When you select it, – the program will display and allow you to edit the properties of the central part of the button, (example: text and background). The lower part properties are listed in the remainder of this table.
Released state	Radio button. Mutually exclusive between: Pushed state and Released state.	When you select it, – the program will display and allow you to edit the Released state properties of the button, (example: icon and background). The Released state properties are listed in the remainder of this table.
Pushed state	Radio button. Mutually exclusive between: Pushed state and Released state.	When you select it, – the program will display and allow you to edit the Pushed state properties of the button, (example: icon and background). The Pushed state properties are listed in the remainder of this table.

Button Central part properties		These properties are for the central area of the button only!!!
Text and Font		
<i>Text</i>	Text Preview	When you use the \$ sign on the central part, the \$ Sign will be replaced with the dynamic value of the button.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
Background		
<i>Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.
Numeric		
<i>Current value</i>	[Current value]	Use this to preview the appearance of the button while displaying different values.
<i>Max value</i>	[Max value]	The maximum value that the button can have/display.
<i>Min value</i>	[Min value]	The minimum value that the button can have/display.
<i>Digits after decimal point</i>	Number of digits.	This property defines how many digits to the right of the decimal point will be displayed.
<i>Display Resolution</i>	Step size.	This property defines the resolution of the displayed values.

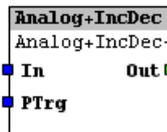
<i>Button upper and lower parts properties</i>		These properties are for the upper and lower areas of the button only!!!
<i>Text and Font</i>		
<i>Text</i>	Text Preview	Use this property to edit the button Text.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
<i>Background</i>		
<i>Background color</i>	Color palette And Drop down menu: [List of basic colors].	Select Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.
<i>Icon/Indicator</i>		
<i>Icon</i>	<i>Icon Image Preview</i>	<i>Use this to search your PC for Icons.</i>
<i>Icon Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Icon.
<i>Icon Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Icon.

9.18. Analog Alpha numeric + Inc/Dec (horizontal) **Button**:

An Analog alphanumeric + Inc/Dec is a general purpose **Button**. It is designed to transmit analog values. You can use it to transmit any numeric data type and value. You can also use it to display values:



9.18.1. Functional description:

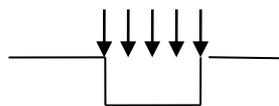


Central part - keypad:



When the keypad OK button is pressed – the **Button** transmits the new value out of **Out Output**.

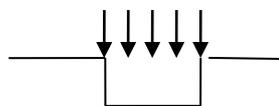
Left part:



As long as **button** is pushed:

1. Until **Out** = Max value: Increase value by *Step size*, transmit the new value out of **Out Output**.
2. Wait 1sec/ *Steps per sec*.
3. Go to step 1

Right part:

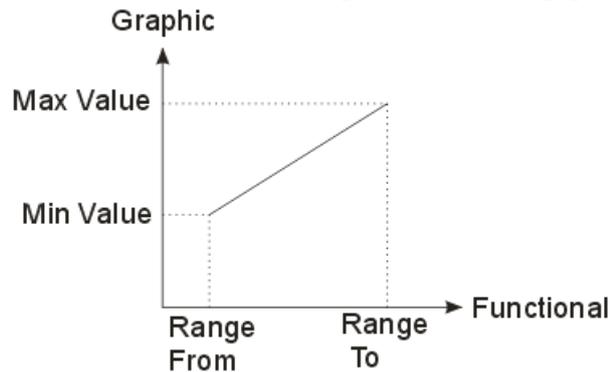


As long as **button** is pushed:

1. Until **Out** = Min Value: decrease value by *Step size*,

- transmit the new value out of **Out Output**.
- 2. Wait 1sec/ *Steps per sec*.
- 3. Go to step 1.

An Analog Alphanumeric + Inc/Dec Vertical **Button** utilizes linear transformation between its graphic properties (display limits) and the functional properties (input and output limits) according to the following graph:



9.18.2. Functional properties:

	Property/Group	Value	Description
	<i>Alphanumeric + Inc/Dec ID</i>	Text Preview	Unique reference ID of the button.
	<i>Receive</i>		
→■	<i>In</i>	List of the links made with this point.	The bar will display values received from this input.
	<i>Transmit</i>		
■→	<i>Out</i>	List of the links made with this point.	New values will be sent through this output any time the User updates the button value.
	<i>Parameters</i>		
	<i>Start up Value</i>	[Value]	The initial value of the Button immediately after startup. This value is used only when <i>Init mode</i> is set to ' <i>init value</i> ' or when persistent data is not valid (e.g. after first download of the project)
	<i>Init mode</i>	Drop down menu : Last Init Value	Mode of operation: Last: immediately after start up, the button will have the same state it had before power down. Init Value: immediately after start up, the button will have the value defined by ' <i>State on start up</i> ' property

	Range To	[Value]	The upper limit of the button
	Range From	[Value]	The lower limit of the button
	<i>Polling</i>		
	<i>Polling on start up</i>	Check box	Polls the <i>Value Input</i> , once, immediately after the Device startup.
	<i>Polling on display</i>	Drop down menu: Disabled Once 1s 5s 15s 30s 3min 15min 1h 6h 1day	As long as the object is on display polls the <i>Value Input</i> cyclically using the interval specified.
	<i>Cyclic polling</i>	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the <i>Value Input</i> cyclically at all times using the interval specified.
	<i>Polling triggers</i>		
→■	<i>Triggers(PTrg)</i>	List of the links made with this point.	Whenever an event is detected on any of the links made for this point – the Device will perform a series of polling commands to the <i>Value</i> input.
	<i>Triggered polling interval</i>	Drop down menu: 1s 5s 15s 30s 60s	The total time duration of triggered polling.
	<i>Triggered polling pause</i>	Drop down menu: 1s 5s 15s 30s 60s	The time duration between single polling commands during triggered polling interval.

9.18.3. Graphic properties:

Property/Group	Value	Description
<i>Alpha numeric + Inc/Dec ID</i>	Text Preview	Unique reference ID of the button.
<i>Frame Type</i>	Drop down menu: [list of available frames]	Use this to select the frame style.
<i>Frame color</i>	Color palette And Drop down menu: [list of basic colors].	Select Frame color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Frame Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Input Method</i>	Drop down menu: Inc/Dec Inc/Dec + Keypad	Inc/Dec = when the lower/upper part of the button is pressed the button value will be decreased/increased. Keypad = when the User touch the buttons central part - a keypad will pop up. The keypad enables the User to assign specific value to the button.
<i>Keypad title Text</i>	Text preview	The title is displayed on the upper part of the popup keypad.
<i>Step size (%)</i>	[Step]	The size (%) of single step towards the new value. Visible when "step to new value" is selected.
<i>Steps per sec</i>	Drop down menu: 1 2 3 4.	The Number of times per second that the button value will be updated. Visible when "step to new value" is selected.
<i>Password</i>	Check box	The User will be asked for the password in order to perform the operation. This function is not available when the Input Method is "None".
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.

<i>button parts properties</i>	<i>button parts properties</i>	
Right part	Radio button. Mutually exclusive between: Right part, Central part and Left part.	When you select it, – the program will display and allow you to edit the properties of the Right part of the button, (example: text and background). The Right part properties are listed in the remainder of this table.
Central part	Radio button. Mutually exclusive between: Upper part, Central part and Lower part.	When you select it, – the program will display and allow you to edit the properties of the central part of the button, (example: text and background). The central part properties are listed in the remainder of this table.
Left part	Radio button. Mutually exclusive between: Upper part, Central part and Lower part.	When you select it, – the program will display and allow you to edit the properties of the Left part of the button, (example: text and background). The Left part properties are listed in the remainder of this table.
Released state	Radio button. Mutually exclusive between: Pushed state and Released state.	When you select it, – the program will display and allow you to edit the Released state properties of the button, (example: icon and background). The Released state properties are listed in the remainder of this table.
Pushed state	Radio button. Mutually exclusive between: Pushed state and Released state.	When you select it, – the program will display and allow you to edit the Pushed state properties of the button, (example: icon and background). The Pushed state properties are listed in the remainder of this table.

Button Central part properties		These properties are for the central area of the button only!!!
Text and Font		
<i>Text</i>	Text Preview	When you use the \$ sign on the central part, the \$ Sign will be replaced with the dynamic value of the button.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
Background		
<i>Fill Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.
Numeric		
<i>Current value</i>	[Current value]	Use this to preview the appearance of the button while displaying different values.
<i>Max value</i>	[Max value]	The maximum value that the button can have/display.
<i>Min value</i>	[Min value]	The minimum value that the button can have/display.

<i>button upper and lower parts properties</i>		These properties are for the Right and Left areas of the button only!!!
<i>Upper part</i>	Radio button. Mutually exclusive between: Upper Part and Lower Part.	When you select it, – the program will display and allow you to edit the properties of the upper part of the button, (example: text and background). The upper part properties are listed in the remainder of this table.
<i>Lower part</i>	Radio button. Mutually exclusive between: Upper Part and Lower Part.	When you select it, – the program will display and allow you to edit the properties of the central part of the button, (example: text and background). The lower part properties are listed in the remainder of this table.
<i>pushed state</i>	Radio button	Select this to edit the Pushed state of the button. When it is set - the underneath properties will affect the Pushed state of the button.
<i>released state</i>	Radio button	Select this to edit the Released state of the button. When it is set - the underneath properties will affect the Released state of the button.
Text and Font		
<i>Text</i>	Text Preview	Use this property to edit the button Text.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.

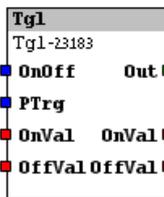
<i>Background</i>		
<i>Background color</i>	Color palette And Drop down menu: [list of basic colors].	Select Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.
<i>Icon/Indicator</i>		
<i>Icon</i>	<i>Icon Image Preview</i>	Use this to search your PC for Icons.
<i>Icon Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Icon.
<i>Icon Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Icon.

9.19. Toggle **Button**:

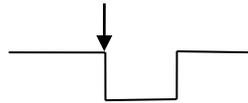
A Toggle **Button** is a general purpose **Button**. It has two states: On and Off, Any time the **User** pushes a Toggle **Button**, it toggles from whichever state it is in to the alternate state. You can use it to transmit any data type and value. It has two states so it can display On/Off status as well.



9.19.1. Functional description:

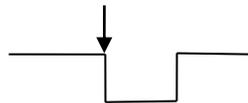


If current State = *On*:



On push - Transmits value *Off Value* out of *Out Output*.

If current State = *Off*:



On push - Transmits value *On Value* out of *Out Output*.

9.19.2. Functional properties:

	<i>Property/Group</i>	<i>Value</i>	<i>Description</i>
	<i>Toggle ID</i>	Text Preview	Unique reference ID of the button.
	<i>Receive</i>		
→■	<i>On/Off (OnOff)</i>	List of the links made with this point.	When the value of this input is set to True, the button state will be set to ON When this input is set to False the button state will be set to OFF.
	<i>Transmit</i>		
■→	<i>Out</i>	List of the links made with this point.	On push –whichever state the button is in, it will transmit the value of the alternate state through this output.

	Parameters		
	<i>State on start up</i>	Drop down menu : On Off	The initial state of the Button immediately after startup. This value is used only when <i>Init mode</i> is set to ' <i>init value</i> ' or when persistent data is not valid (e.g. after first download of the project)
	<i>Init mode</i>	Drop down menu : Last Init Value	Mode of operation: Last: immediately after start up, the button will have the same state it had before power down. Init Value: immediately after start up, the button will have the value defined by ' <i>State on start up</i> ' property
→■→	<i>On value(OnVal)</i>	[Value]	This value will be assigned to <i>Out</i> any time the button switches from the Off state to the On state.
→■→	<i>Off value(OfVal)</i>	[Value]	This value will be assigned to <i>Out</i> any time the button switches from the On state to the Off state.
	<i>Polling</i>		
	<i>Polling on start up</i>	Check box	Polls the <i>On/Off Input</i> , once, immediately after the Device startup.
	<i>Polling on display</i>	Drop down menu: Disabled Once 1s 5s 15s 30s 3min 15min 1h 6h 1day.	As long as the object is on display polls the <i>On/Off Input</i> cyclically using the interval specified.
	<i>Cyclic polling</i>	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the <i>On/Off Input</i> cyclically at all times using the interval specified.
	<i>Polling triggers</i>		
→■	<i>Triggers(PTrg)</i>	List of the links made with this point.	Whenever an event is detected on any of the links made for this point – the Device will perform a series of polling commands to the <i>On/Off input</i> .
	<i>Triggered polling interval</i>	Drop down menu: 1s 5s 15s 30s 60s	The total time duration of triggered polling.
	<i>Triggered polling pause</i>	Drop down menu: 1s 5s 15s 30s 60s	The time duration between single polling commands during triggered polling interval.

9.19.3. Graphic properties:

Property/Group	Value	Description
<i>Toggle ID</i>	<i>Text Preview</i>	Unique reference ID of the button.
<i>Frame Type</i>	Drop down menu: [list of available frames]	Use this to select the frame style.
<i>Frame color</i>	Color palette And Drop down menu: [list of basic colors].	Select Frame color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Frame Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Input Method</i>	Drop down menu: None Touch point.	None = this button will not perform any action as a result of User touch. You can use the button to display status. Touch point = The button will perform an action when it's pushed by the User.
<i>AutoUpdate</i>	Check box	Button state will be updated automatically any time the User touches it. This function is not available when the Input Method is "None".
<i>Password</i>	Check box	The User will be asked for the password in order to perform the operation. This function is not available when the Input Method is "None".
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.
<i>button parts properties</i>		<i>button parts properties</i>
<i>On state</i>	Radio button. Mutually exclusive between: On state and Off state.	When you select it, – the program will display and allow you to edit the On state properties of the button, (example: icon and background). The On state properties are listed in the remainder of this table.
<i>Off state</i>	Radio button. Mutually exclusive between: On state and Off state.	When you select it, – the program will display and allow you to edit the Off state properties of the button, (example: icon and background). The Off state properties are listed in the remainder of this table.

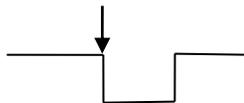
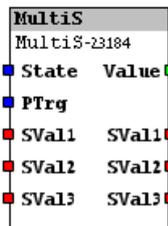
<i>Text and Font</i>		
<i>Text</i>	Text Preview	Use this property to edit the button Text.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
<i>Background</i>		
<i>Background color</i>	Color palette And Drop down menu: [List of basic colors].	Select Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.
<i>Icon/Indicator</i>		
<i>Icon</i>	Icon Image Preview	Use this to search your PC for Icons.
<i>Icon Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Icon.
<i>Icon Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Icon.

9.20. Multi state **Button**:



A Multi state **Button** is a general purpose **Button**. It has one **Output** and up to 10 states. For each state you can define a different output value. There is no limitation to the data type or value. Every state has its own graphic properties so you can use it to display the current state status as well. The **User** switch from one state to another by using the key pad.

9.20.1. Functional description:



When *n* state is selected - Transmits the value of *State Value n* out of *Value Output*.

9.20.2. Functional properties:

	<i>Property/Group</i>	<i>Value</i>	<i>Description</i>
	<i>Multi state ID</i>	Text Preview	Unique reference ID of the button.
	<i>Receive</i>		
→■	<i>State</i>	List of the links made with this point.	When an event arrives to this input, the button state will be set to the state that matches the input value.
	<i>Transmit</i>		
■→	<i>Value</i>	List of the links made with this point.	When state <i>n</i> is selected by the User – this output will transmit the value of the <i>State Value n</i> parameter.
	<i>Parameters</i>		
	<i>State on start up</i>	Drop down menu : [list of all button states]	The initial state of the Button immediately after startup. This value is used only when <i>Init mode</i> is set to ' <i>init value</i> ' or when persistent data is not valid (e.g. after first download of the project)

	<i>Init mode</i>	Drop down menu : Last Init Value	Mode of operation: Last: immediately after start up, the button will have the same state it had before power down. Init Value: immediately after start up, the button will have the state defined by 'State on start up' property
→■→	<i>State Value 1 (SVal1)</i>	[Value]	This value will be assigned to <i>Value</i> any time the User switches the button to state 1.
→■→	<i>State Value... (SVal...)</i>	[Value]	
→■→	<i>State n Value (SValn)</i>	[Value]	This value will be assigned to <i>Value</i> any time the User switches the button to state n.
	<i>Polling</i>		
	<i>Polling on start up</i>	Check box	Polls the <i>State Input</i> , once, immediately after the Device startup.
	<i>Polling on display</i>	Drop down menu: Disabled Once 1s 5s 15s 30s 3min 15min 1h 6h 1day.	As long as the object is on display polls the <i>State Input</i> cyclically using the interval specified.
	<i>Cyclic polling</i>	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the <i>State Input</i> cyclically at all times using the interval specified.
	<i>Polling triggers</i>		
→■	<i>Triggers (PTrg)</i>	List of the links made with this point.	Whenever an event is detected on any of the links made for this point – the Device will perform a series of polling commands to the <i>State</i> input.
	<i>Triggered polling interval</i>	Drop down menu: 1s 5s 15s 30s 60s	The total time duration of triggered polling.
	<i>Triggered polling pause</i>	Drop down menu: 1s 5s 15s 30s 60s	The time duration between single polling commands during triggered polling interval.

9.20.3. graphic properties:

Property/Group	Value	Description
<i>Multi state ID</i>	Text Preview	Unique reference ID of the button.
<i>Frame Type</i>	Drop down menu: [list of available frames]	Use this to select the frame style.
<i>Frame color</i>	Color palette And Drop down menu: [list of basic colors].	Select Frame color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Frame Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Input Method</i>	Drop down menu: None Keypad	None = this button will not perform any action as a result of User touch. You can use the button to display status. Keypad = when the User touch the buttons - a keypad will pop up. The keypad enables the User to assign specific state to the button.
<i>Keypad Title</i>	Text preview	The title displayed on upper part of the popup keypad.
<i>Keypad state 1 Text.</i>		This Text will be displays over the button when it has the current state.
"	"	This Text will be displays over the button when it has the current state.
"	"	"
<i>Keypad state n Text.</i>	"	This Text will be displays over the button when it has the current state.
<i>Password</i>	Check box	The User will be asked for the password in order to perform the operation. This function is not available when the Input Method is "None"
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.
Button state properties		button parts properties
<i>State 1</i>	Radio button	The program will display and allow you to edit the properties, (example: icon and background) of the state you select here. The

		state's properties are listed in the remainder of this table.
....		
<i>State n</i>	Radio button	The program will display and allow you to edit the properties, (example: icon and background) of the state you select here. The state's properties are listed in the remainder of this table.
<i>Text and Font</i>		
<i>Text</i>	Text Preview	Use this property to edit the button Text.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Text Color</i>	Color palette And Drop down menu: [list of basic colors]	Select Text color from a Palette. Type null for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
<i>Background</i>		
<i>Background color</i>	Color palette And Drop down menu: [List of basic colors].	Select Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.
<i>Icon/Indicator</i>		
<i>Icon</i>	Image Preview	Image selector with Thumbnail view
<i>Icon Alignment (V)</i>	Top, Center, Bottom	Drop down box
<i>Icon Alignment (H)</i>	Left, Center, Right,	Drop down box

9.21. Time **Button**:
This **Button** shows the time:



Functional description:

The Time button receives the time from the **Device's** internal Real time clock. The real time clock is setup from the Setup page or by using the Time function block.

The Setup page enables the setting of A) Local time B) Time zone C) Summer saving properties. The time button will display the Local Time by using the following formula: Local time = UTC + Time zone offset + Summer Saving offset.

The Time button displays the time using Analog or digital mode.

9.21.1. Functional properties:

	Property/Group	Value	Description
	<i>Time Button ID</i>	Time ID	Unique reference ID of the button.

9.21.2. graphic properties:

Property/Group	Value	Description
<i>Time Button ID</i>	Text Preview	Unique reference ID of the button.
<i>Frame Type</i>	Drop down menu: [list of available frames]	Use this to select the frame style.
<i>Frame color</i>	Color palette And Drop down menu: [list of basic colors].	Select frame color from a Palette. Type 'null' for transparency Or color name: White, Light Gray, Gray, Dark Gray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Frame Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.

<i>Text and Font</i>		
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Font Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
<i>Background</i>		
<i>Background color</i>	Color palette And Drop down menu: [List of basic colors].	Select Background color from a Palette. Type 'null' for transparency Or color name: White, Light Gray, Gray, Dark Gray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.
<i>Icon/Indicator</i>		
<i>Icon</i>	Icon Image Preview	Use this to search your PC for Icons.
<i>Icon Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Icon.
<i>Icon Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Icon.
<i>format</i>		
<i>Analog clock</i>	Check box	Display the time using analog (Or alphanumeric) format
<i>Time format</i>	Drop down menu: hh:mm hh:mm:ss	Time format – available when Analog clock is not checked
<i>Hour format</i>	Drop down menu: AM/PM 24h	Hour format – available when Analog clock is not checked

9.22. Date **Button**:
 This **Button** shows the date:



Functional description:

The Date button receives the date from the **Device's** internal Real time clock.

The real time clock date is setup from the Setup page or by using the Date function block.

The Setup page enables the setting of A) Local Date B) Time zone C) Summer saving properties. The Date button will display the Local time by using the following formula:
 Local date = Date of [UTC + Time zone offset + Summer saving offset.]

Functional properties:

	<i>Property/Group</i>	<i>Value</i>	<i>Description</i>
	<i>Date Button ID</i>	<i>Date ID</i>	Unique reference ID of the button.

9.22.1. graphic properties:

<i>Property/Group</i>	<i>Value</i>	<i>Description</i>
<i>Date Button ID</i>	<i>Text Preview</i>	Unique reference ID of the button.
<i>Frame Type</i>	Drop down menu: [list of available frames]	Use this to select the frame style.
<i>Frame color</i>	Color palette And Drop down menu: [List of basic colors].	Select frame color from a Palette. Type 'null' for transparency Or color name: White, Light Gray, Gray, Dark Gray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Frame Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.
<i>Text and Font</i>		
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Color</i>	Color palette And Drop down menu: [list of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.

<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
Background		
<i>Background color</i>	Color palette And Drop down menu: [list of basic colors].	Select Background color from a Palette. Type 'null' for transparency Or color name: White, Light Gray, Gray, Dark Gray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.
Icon/Indicator		
<i>Icon</i>	Icon Image Preview	Use this to search your PC for Icons.
<i>Icon Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Icon.
<i>Icon Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Icon.
format		
<i>Date format</i>	Drop down menu: dd:mm:yy mm:dd:yy	Date format

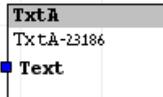
9.23. Text **Button**:



Text

The Text **button** is a background **Button**, therefore it is placed in the *background* layer behind the **Buttons** layer. You can use it to create text titles and to mix them with dynamic texts.

9.23.1. Functional description



The Text **Button** does not react to **User** touch, but it has a *Text Input Point*. You can use this to receive and display values coming from other **Points**.

9.23.2. Functional properties

	Property/Group	Value	Description
	<i>Text ID</i>	Text Preview	Unique reference ID of the button.
	<i>Receive</i>		
→	<i>Text</i>	List of the links made with this point.	When you place the \$ sign on the <i>Text</i> graphic properties of this button - the \$ will be replaced by the values received to this input.

9.23.3. Functional properties

Property/Group	Value	Comment
<i>Text ID</i>	Text Preview	Unique reference ID of the button.
<i>Position and size</i>		
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.
<i>Text and Font</i>		
<i>Text</i>	Text Preview	Use this property to edit the button Text. \$ Sign will be replaced with the dynamic value of the button.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Text Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.

<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
<i>Image</i>		
<i>Image</i>	Image Preview	Use this to search your PC for Images.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.

9.24. Rounded Rectangle **Button**:



The Rounded Rectangle **Button** is a Background **Button**, therefore it is placed in the *Background* layer behind the **Buttons** layer. It can also display text titles and mix them with dynamic texts.

9.24.1. Functional description

The Rounded rectangle **Button** does not react to **User** touch. It has a *Text* property and a linkable *Text Input Point*. You can use it to receive and display values coming from other **Points**.

9.24.2. Functional properties:

	<i>Property/Group</i>	<i>Value</i>	<i>Description</i>
	<i>Rounded rectangle ID</i>	Text Preview	Unique reference ID of the button.
	<i>Receive</i>		
→■	<i>Text</i>	Text Preview	When you place the \$ sign on the <i>Text</i> graphic properties of this button - the \$ will be replaced by the values received to this input.

9.24.3. Graphic properties

Property/Group	Value	Comment
<i>Rounded rectangle ID</i>	Text Preview	Unique reference ID of the button.
<i>Border Line Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Frame color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Border Line Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.
<i>Arc divider</i>	Divider - integer value	Defines the bending level of the border line curve at the corners of the rectangle.
Text and Font		
<i>Text</i>	Text Preview	Use this property to edit the button Text. \$ Sign will be replaced with the dynamic value of the button.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Text Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
Background		
<i>Fill Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images.

<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.

9.25. Rectangle **Button**:

A Rectangle **Button** is Background **Button**, therefore it is placed in the *Background* layer, behind the **Buttons** layer. It can also display text titles and mix them with dynamic texts.



9.25.1. Functional description

Rectangle **Button** does not react to **User** touch. It has a *Text* property and a linkable *Text Input Point*. You can use it to receive and display values coming from other **Points**.

9.25.2. Functional properties

	Property/Group	Value	Description
	<i>Rectangle ID</i>	Text Preview	Unique reference ID of the button.
	<i>Receive</i>		
→■	<i>Text</i>	Text Preview	When you place the \$ sign on the <i>Text</i> graphic properties of this button - the \$ will be replaced by the values received to this input.

9.25.3. Graphic properties

Property/Group	Value	Description
<i>Rectangle ID</i>	Text Preview	Unique reference ID of the button.
<i>Border Line Color</i>	Color palette And Drop down menu: [list of basic colors].	Select Frame color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Border Line Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.
Text and Font		
<i>Text</i>	Text Preview	Use this property to edit the button Text. \$ Sign will be replaced with the dynamic value of the button.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
Background		
<i>Fill Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.

9.26. Oval **Button**:

An Oval **Button** is Background **Button**, therefore it is placed in the *Background* layer, behind the **Buttons** layer. It can also display text titles and mix them with dynamic texts.



9.26.1. Functional description

An Oval **Button** does not react to **User** touch. It has a *Text* property and a linkable **Text Input Point**. You can use it to receive and display values coming from other **Points**.

9.26.2. Functional properties

	<i>Property/Group</i>	<i>Value</i>	<i>Description</i>
	<i>Oval ID</i>	Text Preview	Unique reference ID of the button.
	<i>Receive</i>		
→■	<i>Text</i>	Text Preview	When you place the \$ sign on the <i>Text</i> graphic properties of this button - the \$ will be replaced by the values received to this input.

9.26.3. Graphic properties:

Property/Group	Value	Description
<i>Oval ID</i>	Text Preview	Unique reference ID of the button.
<i>Border Line Color</i>	Color palette And Drop down menu: [list of basic colors].	Select Frame color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Border Line Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.
Text and Font		
<i>Text</i>	Text Preview	Use this property to edit the button Text. \$ Sign will be replaced with the dynamic value of the button.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Color</i>	Color palette And Drop down menu: [list of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
Background		
<i>Fill Color</i>	Color palette And Drop down menu: [list of basic colors].	Select Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.

9.27. Line **Button**:

A Line **Button** is Background **Button**, therefore it is placed in the *Background* layer, behind the **Buttons** layer. You can use it to create horizontal lines in the back ground. It can also display text titles and mix them with dynamic texts.

TEXT

9.27.1. Functional description

A Line **Button** does not react to **User** touch. It has a *Text* property and a linkable *Text Input Point*. You can use it to receive and display values coming from other **Points**.

9.27.2. Functional properties

	Property/Group	Value	Description
	<i>Line ID</i>	<i>Text Preview</i>	Unique reference ID of the button.
	<i>Receive</i>		
→■	<i>Text</i>	Text Preview	When you place the \$ sign on the <i>Text</i> graphic properties of this button - the \$ will be replaced by the values received to this input.

9.27.3. Graphic properties:

Property/Group	Value	Description
<i>Line ID</i>	Text Preview	Unique reference ID of the button.
<i>Border Line Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Frame color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Border Line Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.
Text and Font		
<i>Text</i>	Text Preview	Use this property to edit the button Text. \$ Sign will be replaced with the dynamic value of the button.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Text Color</i>	Color palette And Drop down menu: [List of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
Background		
<i>Fill Color</i>	Color palette And Drop down menu: [list of basic colors].	Select Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.

9.28. Macro **Button**:

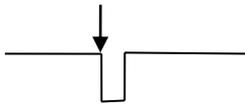


A Macro **Button** is a system **Button**.

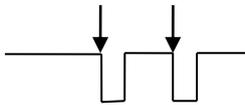
A macro is a list of **Events** and delays that the **Device** will perform in a sequence. The Macro **Button** functionality is to:

- Activate a macro
- Display an indication during the time of the macro execution.
- Enable stopping the execution of the macro.
- open the macro editor.

9.28.1. Functional description



On push – start the execution of the macro.



On the second push, if it is done before the completion of the macro execution, it stops the macro sequence immediately.

The *State* output sends the value True when the macro starts to run and the value False when the macro is ended.



On hold for time equal to *extra long push* – Immediately transmits the final values of the macro and opens the macro editor. Find a detailed description of macro editor in chapter 4.3.3.2 "[Macros tab](#)." Any time the user changes the values of the loads in the macro editor - the device will send these values immediately to the load. It is also possible to edit the macro without immediately sending the values to the loads. This is done by entering into the macro editor through the Set up page. .

9.28.2. Functional properties:

	Property/Group	Value	Description
	<i>Macro ID</i>	Text Preview	Unique reference ID of the button.
	<i>Receive</i>		
→■	<i>Start</i>	List of the links made with this point.	The macro will start running any time a True event comes to this input.
→■	<i>Stop</i>	List of the links made with this point.	The macro execution will stop immediately any time a True event comes to this input.
→■	<i>Edit</i>	List of the links made with this point.	When a true event comes to this input the macro editor will open, allowing the end user to edit the macro.
	<i>Transmit</i>		
■→	<i>State</i>	List of the links made with this point.	This output sends the value True when the macro starts to run, and the value False when the macro is closes.

9.28.3. Graphic properties:

Property/Group	Value	Description
<i>Macro ID</i>	<i>Text Preview</i>	Unique reference ID of the button.
<i>Frame Type</i>	Drop down menu: [list of available frames]	Use this to select the frame style.
<i>Frame color</i>	Color palette And Drop down menu: [List of basic colors].	Select Frame color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Frame Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Input Method</i>	Drop down menu: None Touch point.	None = this button will not perform any action as a result of User touch. You can use the button to display status. Touch point = The button will perform an action when it's pushed by the User.
<i>Password</i>	Check box	The User will be asked for the password in order to perform the operation. This function is not available when the Input Method is "None".
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.

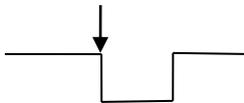
<i>button parts properties</i>		<i>button parts properties</i>
<i>On state</i>	Radio button. Mutually exclusive between: On state and Off state.	When you select it, – the program will display and allow you to edit the On state properties of the button, (example: icon and background). The On state properties are listed in the remainder of this table.
<i>Off state</i>	Radio button. Mutually exclusive between: On state and Off state.	When you select it, – the program will display and allow you to edit the Off state properties of the button, (example: icon and background). The Off state properties are listed in the remainder of this table.
<i>Text and Font</i>		
<i>Text</i>	Text Preview	Use this property to edit the button Text.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Color</i>	Color palette And Drop down menu: [list of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
<i>Background</i>		
<i>Background color</i>	Color palette And Drop down menu: [list of basic colors].	Select Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.
<i>Icon/Indicator</i>		
<i>Icon</i>	Icon Image Preview	Use this to search your PC for Icons.
<i>Icon Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Icon.
<i>Icon Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Icon.

9.29. Page flip **Button**:



When this **Button** is pushed it switches the graphic page on display.

9.29.1. Functional description:



On push – switches the graphic display to the page pointed to by the *flip to page* parameter.

9.29.2. Functional properties:

	<i>Property/Group</i>	<i>Value</i>	<i>Description</i>
	<i>Page flip ID</i>	Text Preview	Unique reference ID of the button.
	<i>Receive</i>		
→■	<i>Text</i>	Text Preview	When you place the \$ sign on the <i>Text</i> graphic properties of this button - the \$ will be replaced by the values received to this input.

9.29.3. Graphic properties:

<i>Property/Group</i>	<i>Value</i>	<i>Description</i>
<i>Page flip ID</i>	<i>Text Preview</i>	Unique reference ID of the button.
<i>Frame Type</i>	Drop down menu: [list of available frames]	Use this to select the frame style.
<i>Frame color</i>	Color palette And Drop down menu: [List of basic colors].	Select Frame color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Frame Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Password</i>	Check box	The User will be asked for the password in order to perform the operation. This function is not available when the Input Method is "None".
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.

<i>Flip To Page</i>	Drop Down menu [List of all project pages]	This property let you select specific page name. The device will switch to this page any time the user push this button.
button parts properties		button parts properties
<i>Current Page</i>	Radio button. Mutually exclusive between: Current Page state and <i>Flip Page</i> state.	When you select it, – the program will display and allow you to edit the Current Page state properties of the button, (example: icon and background). The Current Page state properties are listed in the remainder of this table.
<i>Flip Page</i>	Radio button. Mutually exclusive between: Current page state and <i>Flip Page</i> state.	When you select it, – the program will display and allow you to edit the Flip Page state properties of the button, (example: icon and background). The Flip Page state properties are listed in the remainder of this table.
Text and Font		
<i>Text</i>	Text Preview	Use this property to edit the button Text. \$ Sign will be replaced with the dynamic value of the button.
<i>Font</i>	[[Font] [Size] [style]]	Font/Size/Style selector.
<i>Color</i>	Color palette And Drop down menu: [list of basic colors].	Select Text color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Text Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Text.
<i>Text Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Text.
Background		
<i>Background color</i>	Color palette And Drop down menu: [list of basic colors].	Select Background color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Image</i>	Image Preview	Use this to search your PC for Images.
<i>Image Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Image.
<i>Image Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Image.
<i>Full Size</i>	Check box	Fit the image size exactly to the container dimensions.



<i>Icon/Indicator</i>		
<i>Icon</i>	Icon Image Preview	Use this to search your PC for Icons.
<i>Icon Alignment (V)</i>	Drop down menu: Top Center Bottom.	Vertical alignment of the Icon.
<i>Icon Alignment (H)</i>	Drop down menu: Left Center Right.	Horizontal alignment of the Icon.

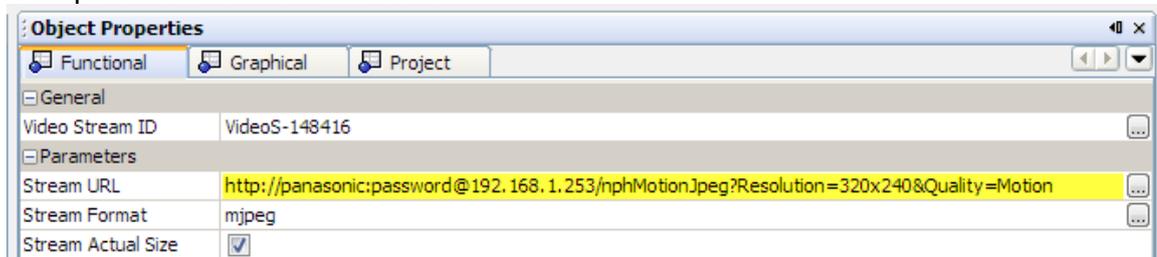
9.30. Video stream button

The video stream button presents video stream, in addition on remote Apps (iPad/iPhone/Android...) it can start a browser page.

1) Functional description

The video stream button presents video stream having MJPG format. Multiple cameras can be displayed on single graphic page and there is no limitation to the number of pages showing video on a project, video can be presented on original resolution or resized to fit the button's dimensions. Video stream can overload the Maestro's CPU follow the instructions below to avoid overloading the CPU.

To present the stream from a streaming device you have to enter its URL on the *Stream URL* parameter of the video button.



The URL should include the IP address of the streaming device followed by a control command. The control commands vary from one manufacturer to the other and even between different models of the same manufacturer. Refer to the streaming device's documentation to find the correct command used by the device
Here are some samples:

Robotix:

`http://<IPofCamer>/cgi-bin/faststream.jpg?preview&size=320x240&fps=8.0&quality=60`

Axis:

`http://<IPAddress>/axis-cgi/mjpg/video.cgi?camera=1&resolution=CIF`

Panasonic:

`http://panasonic:password@<IPofCamera>/nphMotionJpeg?Resolution=320x240&Quality=Motion`

Linksys:

`http://<IPofCamer>/img/video.mjpeg`

Grand stream:

`http://admin:admin@<IPofCamera>/goform/stream?cmd=get&channel=0`

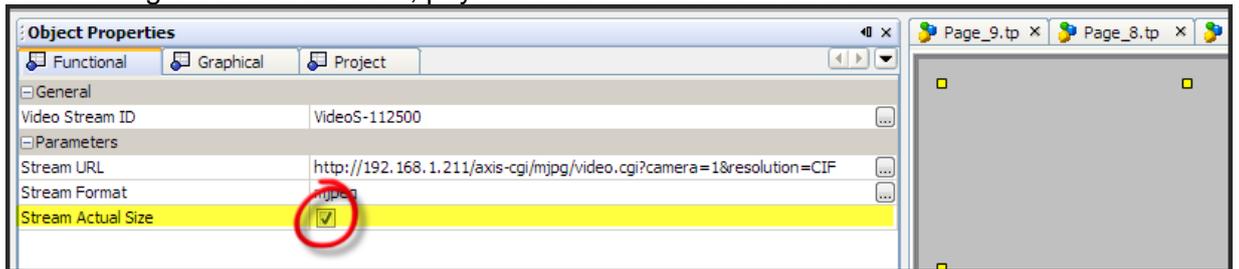
Vivotech:

`http://<IPofCamma:port>/cgi-bin/viewer/video.mjpg`

By default, the Maestro presents the video on its original resolution, so if the camera is streaming image having a resolution of 640x480 you have to use a video button having exactly the same size in order to see the whole image.

In case your application demands using a video button having a different size from the streamed video you can uncheck the Stream Actual Size parameter. Now Maestro will resize the original video resolution streamed by the camera and fit it to exact dimensions of the video button on the Maestros graphic screen, pay attention that resizing the image consumes additional CPU power from the Maestro and therefore not recommended.

When using the resize function, pay extra care to maestros CPU load.



To monitor CPU load:

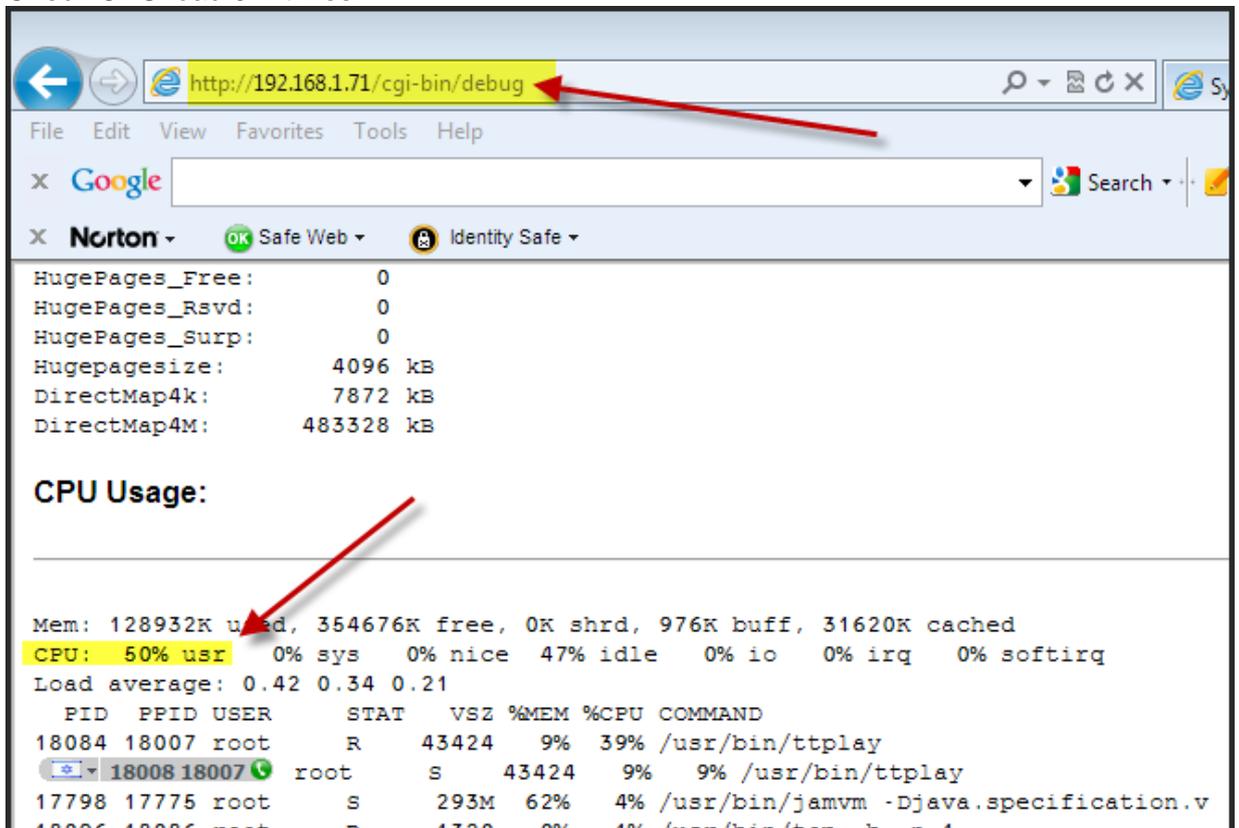
Use a browser to open the maestros debug file,.

Download the project to the Maestro.

Flip to the IP video stream and while video is presented

Use a browser to open the debug file of Maestro (it is located on the `http://<Maestro's IP address>/cgi-bin/debug`) and

Check CPU load 3-4 times:



do not exceed 70% CPU load for long times (continuously).

if CPU is overloaded - you can: reduce frame rate, resolution and use Stream Actual Size.

Normally at 5 frames per second the Maestro MTS3 should be capable of showing one VGA image, 4 x CIF images or 12 x QCIF images on a single page.

9.30.1. Functional properties

	Property/Group	Value	Description
	<i>Video Stream ID</i>	<i>Text Preview</i>	Unique reference ID of the button.
	<i>Parameters</i>		
	Stream URL	URL	IP address and command sent to the streaming device
	Stream Format	Text field options are: - mjpg - html	mjpg - Motion JPEG video stream. html – browser page
	Stream Actual Size	Check box	When checked: image will be displayed on original resolution. When unchecked: Maestro will resize the incoming stream resolution to fit the buttons dimensions. It is recommended to always use Stream Actual Size: this way the CPU dos not need to work on stretching your image and <u>can show higher frame rate/resolution</u> .

9.30.2. Graphic properties:

Property/Group	Value	Description
<i>Video Stream ID</i>	Text Preview	Unique reference ID of the button.
<i>Frame Type</i>	Drop down menu: [list of available frames]	Use this to select the frame style.
<i>Frame color</i>	Color palette And Drop down menu: [List of basic colors].	Select Frame color from a Palette. Type 'null' for transparency Or color name: White, LightGray, Gray, DarkGray, Black, Red, Pink, Orange, Yellow, Green, Magenta, Cyan, Blue.
<i>Frame Thickness</i>	[Thickness]	Use this to set the Frame thickness (pixels). This field accepts natural numbers as input.
<i>Height</i>	[Height]	Button Height (in pixels).
<i>Width</i>	[Width]	Button Width (in pixels).
<i>Bounds</i>	[[X], [Y], [Width], [Height]]	View and edit the position, width and height of the button.

10. Function Blocks Controller:

The maestro **Devices** are running a powerful internal, event driven, virtual controller. The *Maestro Designer* software enables you to design the functionality of the controller by drawing the block diagram of the application you need. Maestro is event driven unsynchronized: generally in case of few events triggered simultaneously there is no way to determine in which order they will be executed.

Types of **Blocks**:

- Button-blocks:

These **blocks** represent the functional part of **Buttons** that were placed on the **Graphic Pages**. A Button-block is related to a specific **Button**, therefore it can be placed only once on the block diagram and it can not be copied. See chapter 9 "[Buttons](#)" for description of all available **Buttons**.

- Function-Blocks:

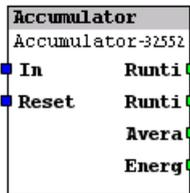
Function-Blocks are operators used in the virtual controller block diagram. They can have **Inputs**, **Outputs**, parameters and an algorithm. The Function-Block uses its **Inputs**, parameters and algorithm to calculate the values for its **Outputs**. These **Blocks** have no direct representation on the **Graphic Pages**. Below is a description of all available Function-Blocks

Features relating to **Events**:

- Any telegram coming from the bus is an **Event**.
 - Any value change of a **Block's Output** is an **Event**.
 - **Buttons** are generating **Events** as a result of **User** touch and the **Button** algorithm.
 - Function-Block algorithms are routing triggers from their **Input** to their **Output**; therefore an **Event** on an **Input** of a **Block** will (with a few exceptions) produce an **Event** on the **Outputs** of the **block**, even if the value of the **Output** was not changed.
 - **Linkable Parameters**: An **Event** on some of the **Linkable parameters** will generate an **Event** on the Function-Block **Outputs** and on other **Linkable Parameters** it will not generate an **Event**, for more details see the description of the specific Function-Block.
 - An **Event** on a **Linkable Parameter Output** will occur **only** when the same Linkable parameter **Input** had an incoming **Event** having new value.
 - An **Event** on the **Input** of a Button-Block can affect the **Button** state, but it can not cause an **Event** on the **Button Output**.
- ☺ *Rout* and *Buffer* Function-Blocks are giving you the ability to control the timing of transmissions and to block unwanted **Events** from going through a **Link**.

Here is a description of the available Function-Blocks:

10.1. Accum (Accumulator) Function-Block:



The accumulator **Function Block** is calculating:

$$\sum Load * \frac{in}{InMax - InMin} * \Delta t$$

And sending the outcome of the calculation to its *Energ* output.

All calculations and the updates of the outputs are executed on an interval that its length (sec) is defined by the *polling* parameter

the *In* value used for the calculation is the last value received on the *In* input.

When a group Address is linked to the *In* input and the link's poll checkbox is checked – the Maestro will send cyclic poll commands to the Group Address (according to the specified interval).

if

In addition the Maestro sends the total time in seconds to the **Function Block's** *Run Time Sec* output – *the* and the total time as a string to is *Run Time String* output (having the format of “XXXXh XXm XXs”).

10.1.1. Functional description:

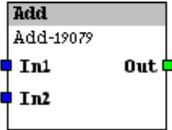
Time diagram: Accumulator Function-Block.

10.1.2. Functional properties:

Icon	Property/Group	Value	Description
	Accumulator <i>Function-Block ID</i>	Text Preview	Unique reference ID of the Function-Block.
	<i>Receive</i>		
→■	<i>In</i>	List of the links made with this point.	
→■	<i>Reset(Rst)</i>	List of the links made with this point.	Reset to 0 all calculated values
	<i>Transmit</i>		
■→	<i>Run Time Sec (Runti)</i>	List of the links made with this	Run time is seconds

		point.	
■→	<i>Run Time String (Runti)</i>	List of the links made with this point.	Run time in Text format: For example "156h 12m 30s"
■→	<i>Average (Avera)</i>	List of the links made with this point.	Has to be fixed
■→	<i>Energy (Ener)</i>	List of the links made with this point.	Accumulated value
<i>Parameters</i>			
	<i>polling</i>		Time interval (sec) for calculations
	<i>Load (KWh)</i>		The maximum load.
	<i>In Min</i>		The input value representing 0 load
	<i>In Max</i>		The input value representing 100% load. When <i>In</i> equals or grater to this value: the load is considered as maximum load

10.2. Add Function-Block:



This Function-Block performs arithmetical sum operations.

10.2.1. Functional description:

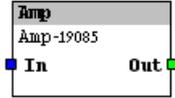
This Function-Block sums the value of its two **Inputs**, and sends the result to the **Output**.

$$\text{Out} = \text{In 1} + \text{in 2}.$$

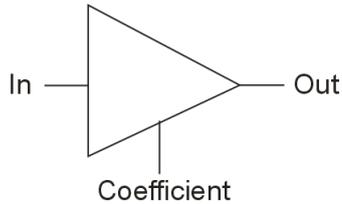
10.2.2. Functional properties:

Icon	Property/Group	Value	Description
	Add Function-Block ID	Text Preview	Unique reference ID of the Function-Block.
	Receive		
→■	In 1	List of the links made with this point.	addend1
→■	In 2	List of the links made with this point.	addend2
	Transmit		
■→	Out	List of the links made with this point.	The sum of the inputs.
	Polling		
	Polling on start up	Check box	Polls the State Input, once, immediately after the Device startup.
	Cyclic polling	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the State Input cyclically at all times using the interval specified.

10.3. Amp Function-Block:



This function is an amplifier.



10.3.1. Functional description:

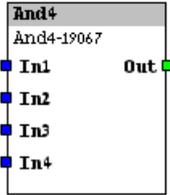
The value of the **Output** *Out* is the product of multiplying the *In* **Input** by the *Coefficient* parameter.

$$Out = In \times Coefficient$$

10.3.2. Functional properties:

	Property/Group	Value	Description
	<i>Amp Function-Block ID</i>	Text Preview	Unique reference ID of the Function-Block.
	<i>Receive</i>		
→■	<i>In</i>	List of the links made with this point.	Input of amplifier.
	<i>Transmit</i>		
■→	<i>Out</i>	List of the links made with this point.	$Out = In \times Coefficient.$
	<i>Parameters</i>		
	<i>Coefficient</i>		Coefficient value of amplifier.
	<i>Polling</i>		
	<i>Polling on start up</i>	Check box	Polls the State Input, once, immediately after the Device startup.
	<i>Cyclic polling</i>	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the State Input cyclically at all times using the interval specified.

10.4. And, AndN Function-Blocks:



And Function-Block performs Logic And operation over its **Inputs**.

Functional description:

When the values of all the **Inputs** of this Function-Block are True the value of its **Output** will be True. In all other cases - the value of the **Output** is False. The number of **Inputs** is assigned by the parameter *number of inputs*.

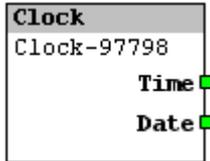
<i>In 1</i>	0	1	0	1	0	1	0	1
<i>In 2</i>	0	0	1	1	0	0	1	1
<i>In 3</i>	0	0	0	0	1	1	1	1
<i>Out</i>	0	0	0	0	0	0	0	1

Truth table for: And Function-Block with 3 **Inputs** (0 stands for False and 1 stands for True).

10.4.1. Functional properties:

Icon	Property/Group	Value	Description
	<i>AND ID</i>	Text Preview	Unique reference ID of the Function-Block.
	<i>Receive</i>		
→■	<i>In 1</i>	List of the links made with this point.	Input
→■	<i>In 2</i>	List of the links made with this point.	Input
→■	<i>In...</i>	List of the links made with this point.	Input
→■	<i>In n</i>	List of the links made with this point.	Input
	<i>Transmit</i>		
■→	<i>Out</i>	List of the links made with this point.	The result of the logic AND function
	<i>Polling</i>		
	<i>Polling on start up</i>	Check box	Polls the State Input, once, immediately after the Device startup.
	<i>Cyclic polling</i>	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the State Input cyclically at all times using the interval specified.

10.5. Clock Function Block:



Clock Function Block transmits the Time and date, as a KNX 3 byte Time and Date Data Type. The interval between transmissions is a parameter that can be set from 10sec up to 7200sec (2 hours)

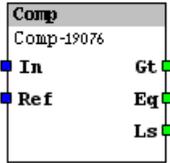
10.5.1. Functional description

The **Function Block** will transmit the *Time* and *Date*, at an interval set on the *Interval* parameter

10.5.2. functional properties:

	<i>Property/Group</i>	<i>Value</i>	<i>Description</i>
	<i>ID</i>	Text Preview	Unique reference ID of the Function-Block.
	<i>Transmit</i>		
	<i>Time</i>	List of the links made with this point.	Sends the Time as 3 byte, KNX, Time Data Type. The period of time between transitions is set by the <i>Interval</i> parameter
	<i>Date</i>	List of the links made with this point.	Sends the Date as 3 byte, KNX, Time Data Type. The period of time between transitions is set by the <i>Interval</i> parameter
	<i>Parameters</i>		
	<i>Interval</i>		The period of time between transitions in seconds. Minimum is 10sec and Maximum is 7200sec
	<i>Polling</i>		
	<i>Polling on start up</i>	Check box	Polls the State Input, once, immediately after the Device startup.
	<i>Cyclic polling</i>	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the State Input cyclically at all times using the interval specified.

10.6. Comp (Comparator) Function-Block:



Comparator Function-Block compares the value of its **Inputs**.

10.6.1. Functional description:

This Function-Block compares between its two **Inputs**, and updates its three **Outputs**.

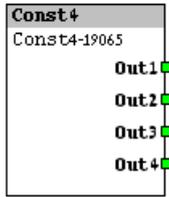
<i>In 1</i>	x	y	x	y
<i>Ref</i>	x	x	y	y
<i>Greater Then</i>	0	1	0	0
<i>Equal</i>	1	0	0	1
<i>Less Then</i>	0	0	1	0

Truth table for Comparator Function-Block, $x < y$ (0 stands for False and 1 stands for True)

10.6.2. Functional properties:

Icon	Property/Group	Value	Description
	Comparator Function-Block ID	Text Preview	Unique reference ID of the Function-Block.
	Receive		
→■	<i>In</i>	List of the links made with this point.	Input
→■	<i>Reference (Ref)</i>	List of the links made with this point.	Reference input
	Transmit		
■→	<i>Greater Then (GT)</i>	List of the links made with this point.	The value of this out will be True if <i>In</i> is greater then <i>Reference</i>
■→	<i>Equal (EQ)</i>	List of the links made with this point.	The value of this out will be True if <i>In</i> is Equal to <i>Reference</i>
■→	<i>Less Then (LT)</i>	List of the links made with this point.	The value of this out will be True if <i>In</i> is less then <i>Reference</i>
	Polling		
	<i>Polling on start up</i>	Check box	Polls the State Input, once, immediately after the Device startup.
	<i>Cyclic polling</i>	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the State Input cyclically at all times using the interval specified.

10.7. Const 1,2,N (Constant) Function-Blocks:



Constant Function-Block holds constant values.

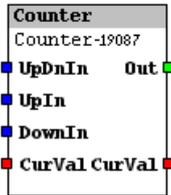
10.7.1. Functional description:

Constant Function-Block holds constant values.
It transmits these values any time the **Device** starts.

10.7.2. Functional properties:

	Property/Group	Value	Description
	<i>Constant Function-Block ID</i>	Text Preview	Unique reference ID of the Function-Block.
	<i>transmit</i>		
	<i>Out1</i>	List of the links made with this point.	Default value for this constant
	<i>Out2</i>	List of the links made with this point.	Default value for this constant
	<i>Out ..</i>	List of the links made with this point.	Default value for this constant
	<i>Out n</i>	List of the links made with this point.	Default value for this constant
	<i>parameters</i>		
	Constant 1 (C1)	[Initial value]	Value of this constant
	Constant 2 (C2)	[Initial value]	Value of this constant
	Constant ..(C..)	[Initial value]	Value of this constant
	Constant n (Cn)	[Initial value]	Value of this constant
	<i>Polling</i>		
	Polling on start up	Check box	Polls the State Input, once, immediately after the Device startup.
	Cyclic polling	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the State Input cyclically at all times using the interval specified.

10.8. Counter Function-Block:

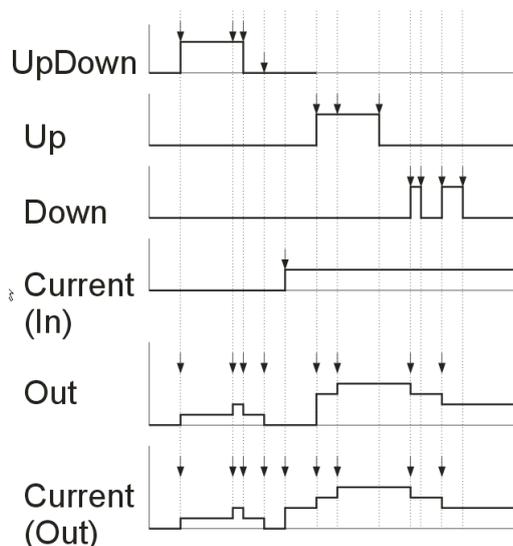


A Counter Function-Block is an up and down counter. It is possible to define the Delta of the counter, and to set an initial value to the count.

10.8.1. Functional description:

The counting starts with the value of the **CurVal Linkable Parameter**. The size of the step for every count is defined by the *Step* variable. Here is description of the behavior of the Function-Block:

- When an **Event** with the value True comes through the UpDown **Input** – the **CurVal** value will be increased by 1 *Step*.
 - When an **Event** with the value False comes through the UpDown **Input** - the **CurVal** value will be decreased by 1 *Step*.
 - When an **Event** with the value True comes through the Up **Input** - the **CurVal** value will be increased by 1 *Step*.
 - When an **Event** with the value False comes through the Up **Input** - the **CurVal** value will remain unchanged.
 - When an **Event** with the value True comes through the Down **Input** - the **CurVal** value will be decreased by 1 *Step*.
 - When an **Event** with the value False comes through the Down **Input** - the **CurVal** value will remain unchanged.
 - When an **Event** with the value X comes through the **CurVal Input** – the value X will be assigned to the **CurVal** value.
- Out** will have the same value as **CurVal** except for the case of **Event** on **CurVal**. In this case the **Out** will remain unchanged, (the change will take effect only on the next counting **Event**).



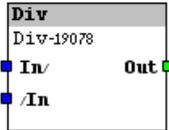
Time diagram Counter Function-Block.

10.8.2. Functional properties:

Icon	Property/Group	Value	Description
	Counter Function-Block ID	Text Preview	Unique reference ID of the Function-Block.
	<i>Receive</i>		
→■	<i>Up Down In (UpDnIn)</i>	List of the links made with this point.	A True event on this input will add the value of <i>Step</i> to the counter. A False event on this input will subtract the value of <i>Step</i> from the counter.
→■	<i>UpIn</i>	List of the links made with this point.	A True event on this input will add the value of <i>Step</i> to the counter. False event on this input will be ignored.
→■	<i>DownIn</i>	List of the links made with this point.	True event on this input will subtract the value of <i>Step</i> from counter. A False event on this input will be ignored..
	<i>Transmit</i>		
■→	<i>Out</i>	List of the links made with this point.	The Value of the counter
	<i>Parameters</i>		
→■→	<i>Current value (CurVal)</i>	List of the links made with this point.	Use this variable to set a new value to the counter.
	<i>Step</i>		The value will be added/subtracted from the count on an Up/Down event.

	<i>Polling</i>		
	Polling on start up	Check box	Polls the State Input, once, immediately after the Device startup.
	Cyclic polling	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the State Input cyclically at all times using the interval specified.

10.9. Div (Divide) Function-Block:



This Function-Block performs arithmetical Division operations.

10.9.1. Functional description:

This Function-Block performs division operations between its two **Inputs** and sends the result to the **Output**.

$$Out = Dividend / Divisor.$$

10.9.2. Functional properties:

Icon	Property/Group	Value	Description
	<i>Divide Function-Block ID</i>	Text Preview	Unique reference ID of the Function-Block.
	<i>Receive</i>		
→■	<i>Dividend (In/)</i>	List of the links made with this point.	dividend
→■	<i>Divisor (/In)</i>	List of the links made with this point.	divisor
	<i>Transmit</i>		
■→	<i>Out</i>	List of the links made with this point.	The quotient of dividing <i>Dividend</i> by <i>Divisor</i> .
	<i>Polling</i>		
	Polling on start up	Check box	Polls the State Input, once, immediately after the Device startup.
	Cyclic polling	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the State Input cyclically at all times using the interval specified.

10.10. Event2, EventN Function-Blocks:



Event Function-Block generates an **Event** on its **Output** whenever it detects an **Event** on its **Inputs**.

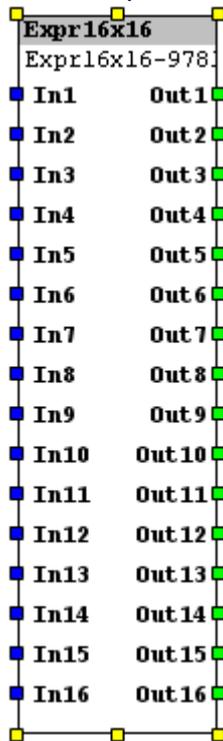
10.10.1. Functional description:

An event will be generated, on the **Output**, any time the **Event** is detected on one of the Function-Blocks **Inputs**. The value of the *Out* is equal to the input number that generates the last event: (example: If there is an event on input 3, the output will transmit the value “3”)

10.10.2. Functional properties:

<i>Icon</i>	<i>Property/Group</i>	<i>Value</i>	<i>Description</i>
	<i>Event Function-Block ID</i>	Text Preview	Unique reference ID of the Function-Block.
	<i>Receive</i>		
→■	<i>In 1</i>	List of the links made with this point.	Input
→■	<i>In 2</i>	List of the links made with this point.	Input
→■	<i>In...</i>	List of the links made with this point.	Input
→■	<i>In n</i>	List of the links made with this point.	Input
	<i>Transmit</i>		
■→	<i>Out</i>	List of the links made with this point.	The value of the out is The order number of the input that generated the event

10.11. Expression function block



The Expression function block enables you to write/create/generate your own functions using expression language.

It has 1 to 16 inputs and 1 to 16 outputs.

To learn about expression language, refer to the corresponding chapter: [“Maestro script language”](#).

10.11.1. Functional description

when the controller starts, it creates a variable for every input and output of the expression **Function Block**. The names of these variables are in# and out# (for example in1, in2.....out1, out2..) these are static variables so you can use them and they are shared in all of the expressions of the expression **Function Block**.

When the controller starts it executes, one time, the expression language code placed on *Expr start* parameter

When an event arrives to one of the expression **Function Block** inputs, it will trigger the execution of the expression language code placed on its corresponding *Expr In#* parameter. For example: an event on Group Address that is linked to In2 will trigger the execution of the expression of in2.

The value that is received by input of the expression function block is stored in a local variable named “in”. The value returned from *Expr In#* will be set to corresponding in# variable.

If no value is returned from *Expr in#* - no further action will be executed.

If a value is returned from *Expr In#* , then the corresponding In# variable will receive the returned value and the controller will execute, one time, the expression language code of the *Expr process* parameter.

The default command “=in;” of the *Expr in#* is equivalent to the operation: “return in”. In other words, the *in#* variable will receive the value received by this input and the *Expr process* will be triggered.

Please note that the statement *in1=in;* also assigns the input value to the *in1* variable but it will not trigger the execution of the *Expr process*. If you wish to trigger the execution of the *Expr process*, you must use the return command. In addition, when the processor is not busy it will execute the expression placed on the “*Expr Idle*” parameter, approximately once per second.

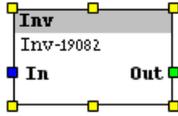
10.11.2. Functional properties:

Icon	Property/Group	Value	Description
	<i>ID</i>	Text Preview	Unique ID reference of the Function-Block.
	<i>Receive</i>		
→■	<i>In1</i>	List of the links made with this point.	In1 Input
→■	<i>In2</i>	List of the links made with this point.	In2 input
→■	List of the links made with this point.
→■	<i>Inn</i>	List of the links made with this point.
	<i>Transmit</i>		
■→	<i>Out1</i>	List of the links made with this point.	Out1 output, this output will transmit the values of the out1 variable
■→	<i>Out2</i>	List of the links made with this point.	Out2 output, this output will transmit the values of the out2 variable
■→	...	List of the links made with this point.	...
■→	<i>Outn</i>	List of the links made with this point.	...
	<i>Parameters</i>		
	<i>Expr Process</i>		The code placed here will be executed any time a value is returned from one of the function block's <i>inputs</i>
	<i>Expr Start</i>		The code placed here will be executed a single time

			when the controller starts
	<i>Expr Idle</i>		The code placed here will be executed approximately once a second when the CPU is not busy
	<i>Expr In1</i>		The code placed here will be executed a single time, any time fresh data arrives to In1 <i>input</i> . If a value is returned from this input – it will trigger the execution of <i>Expr Process</i> .
	<i>Expr In2</i>		The code placed here will be executed a single time, any time fresh data arrives to In2 <i>input</i> . If a value is returned from this input – it will trigger the execution of <i>Expr Process</i> .

	<i>Expr Inn</i>		The code placed here will be executed a single time, any time fresh data arrives to In2 <i>input</i> . If a value is returned from this input – it will trigger the execution of <i>Expr Process</i> .
	<i>Polling</i>		
	Polling on start up	Check box	Polls the State Input, once, immediately after the Device startup.
	Cyclic polling	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the State Input cyclically at all times using the interval specified.

10.12. Inv (Inverter) Function-Block:



This function inverts the value of the **Input**.

10.12.1. Functional description:

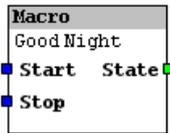
This function calculates the inverted value of the **Input** and sends the result to the **Output**.

$$Out = - In$$

10.12.2. Functional properties:

<i>icon</i>	<i>Property/Group</i>	<i>Value</i>	<i>Description</i>
	<i>Inverter Function-Block ID</i>	Text Preview	Unique reference ID of the Function-Block.
	<i>Receive</i>		
→■	<i>In</i>	List of the links made with this point.	Input values
	<i>Transmit</i>		
■→	<i>Out</i>	List of the links made with this point.	$Out = - In$
	<i>Polling</i>		
	Polling on start up	Check box	Polls the State Input, once, immediately after the Device startup.
	Cyclic polling	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the State Input cyclically at all times using the interval specified.

10.13. Macro Function Block:



A macro is list of **Events** and delays that the **Device** will perform in a sequence. The Macro **Function Block** functionality:

- Activates a macro
- sends an indication during the time of the macro execution.
- Enables the stopping of the execution of the macro.

10.13.1. Functional description

When an **Event** with the value True comes through the **Start Input**, the **Function Block** will start the execution of the macro.

The **State** output sends the value True when the macro starts to run and the value False when the macro is ended.

A True Event on the **Stop Input**, if it comes before the completion of the macro execution, will stop the macro sequence immediately.

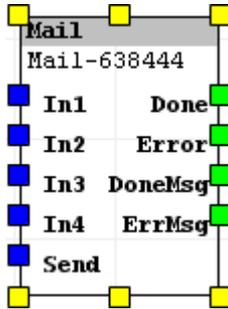
It is possible to edit the macro setting from your PC using the *Maestro Designer* Macro Editor. You can also access the Macro Editor from the Setup page on the **Device**.

Find a detail description of macro editor in chapter 4.3.3 "[Macros and Scheduler](#)"

10.13.2. Functional properties:

	Property/Group	Value	Description
	<i>Macro ID</i>	Text Preview	Unique reference ID of the Function-Block.
	<i>Receive</i>		
→■	Start	List of the links made with this point.	The macro will start running any time a True event arrives to this input.
→■	Stop	List of the links made with this point.	The macro execution will stop immediately any time a True event arrives to this input.
	<i>Transmit</i>		
■→	State	List of the links made with this point.	This output sends the value True when the macro starts to run and the value False when the macro is closed.

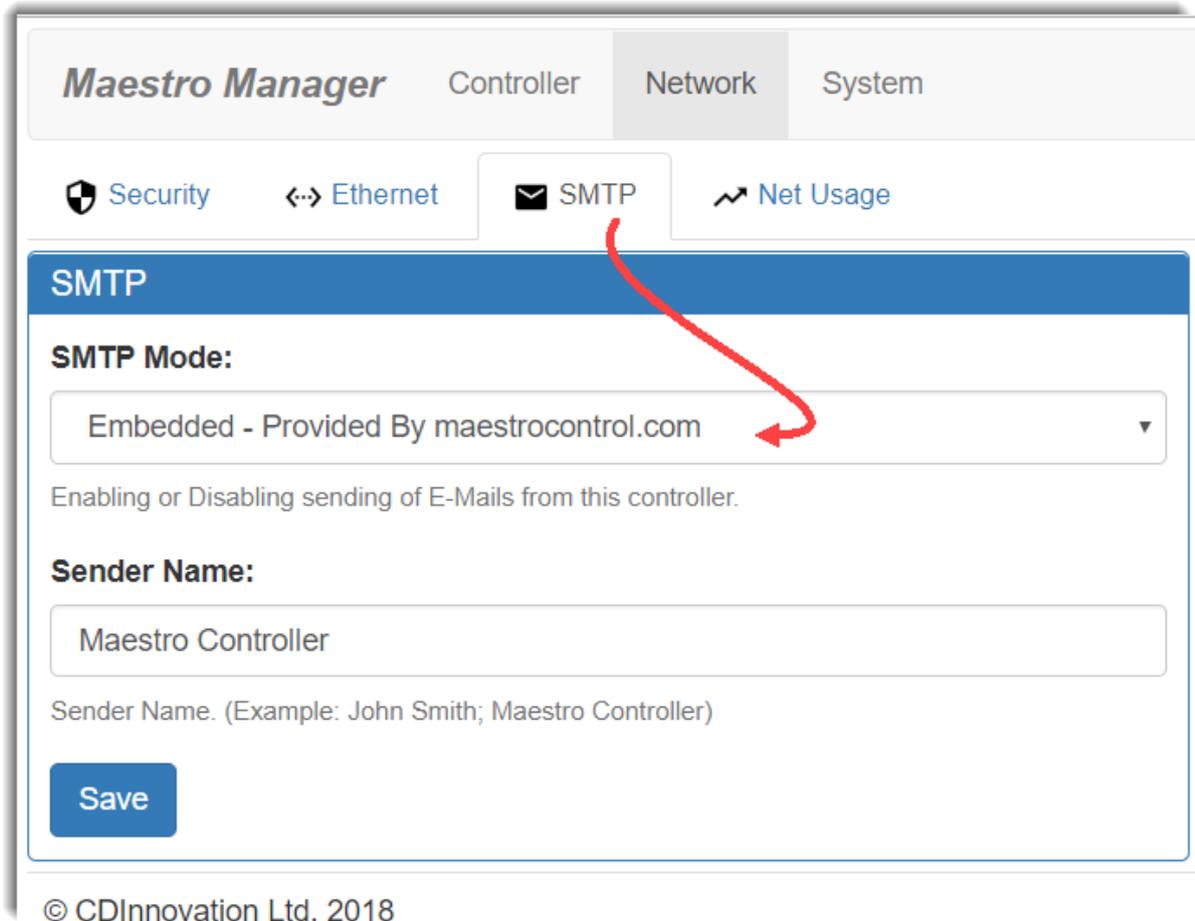
10.14. Mail Faction block



Maestro enables the generation and transmission of emails in real time. Emails can be programmed to notify the end user of weather developments, emergencies, reports, actions taken by Maestro and much more. By Using Maestro's artificial intelligence Email content is dynamic and the Maestro adapts it in real time according to events as they change from minute to minute

To send Email follow, the instructions below:

10.14.1. Using a browser, enable one of the SMTP modes:



Maestro Manager Controller Network System

Security Ethernet SMTP Net Usage

SMTP

SMTP Mode:

Embedded - Provided By maestrocontrol.com

Enabling or Disabling sending of E-Mails from this controller.

Sender Name:

Maestro Controller

Sender Name. (Example: John Smith; Maestro Controller)

Save

© CDInnovation Ltd. 2018

Embedded – for using Mail box created by CD Innovation.

External – to enable interface for setting the parameters of user defined, external Mail box.

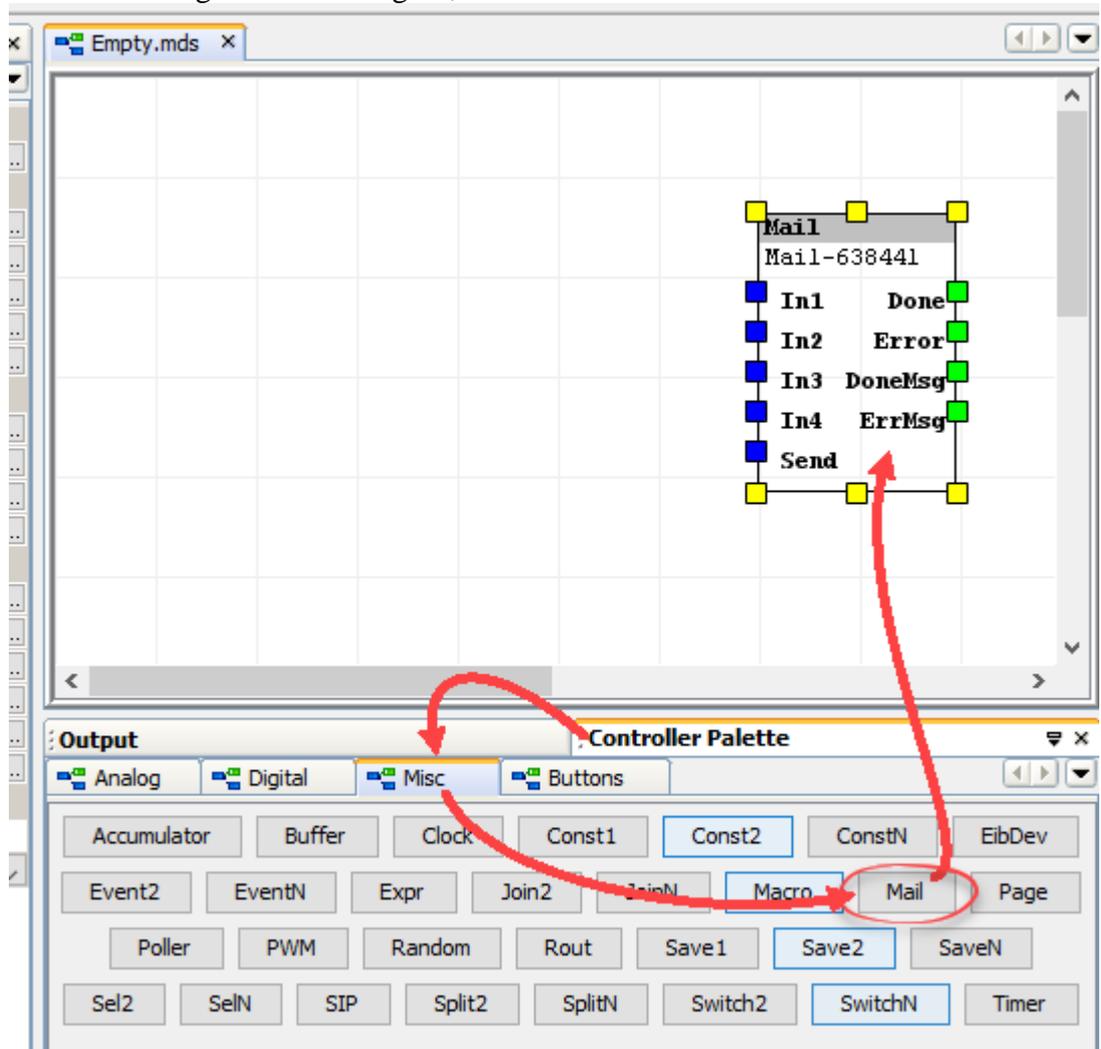
It is recommended to select the option "Embedded – Provided By maestrocontrol.com", when you select this option - Maestro will use a mail box on CD Innovation's cloud server.

it has 2 advantages:

you will not have to set any parameters for the Mail box.

some external Mail box providers are blocking the option of remotely sending Mails by 3rd party App and even if set correctly – will block Maestro from sending the eMails.

10.14.2. Using Maestro Designer, add Mail function block:

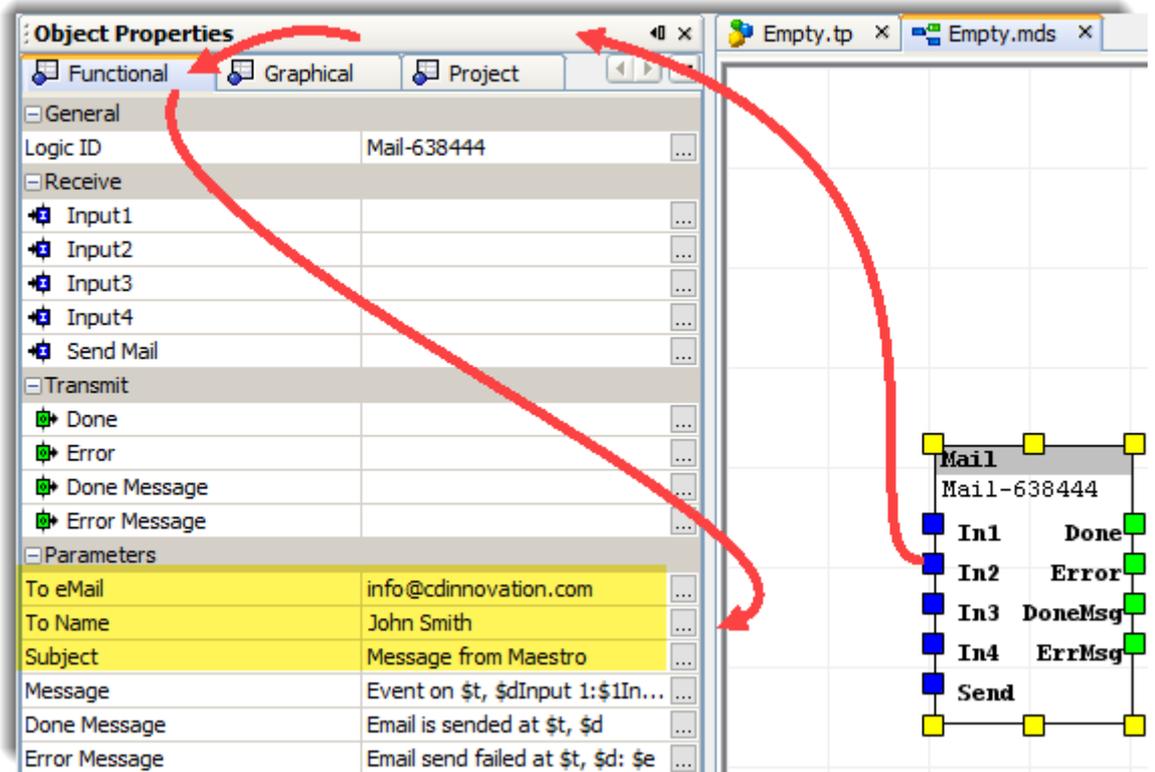


10.14.3. Set the:

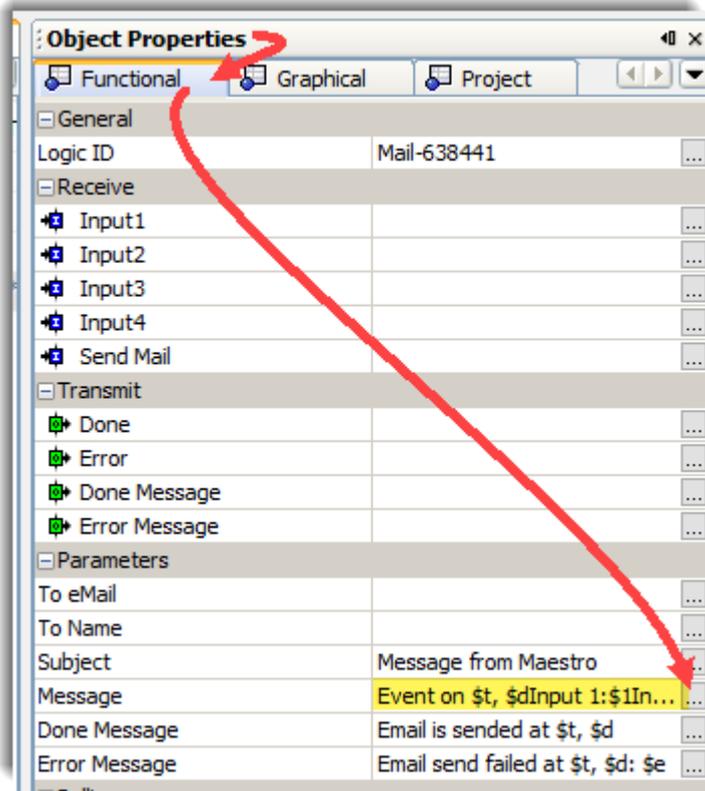
To EMail: recipient Email address (only one is possible for each Mail function block).

To Name - free text

Subject – email subject, free text



10.14.4. Set the Message,
Type your eMail message here:



on this field some character combinations will be replaced by dynamic content when the mail sent, thus allowing to create dynamic content to the message (not just fixed text).
 The character combination \$t - will be replaced with the current time.
 The characters combination \$d – will be replaced with the current date.
 The characters combinations \$1,\$2,\$3 and \$4 - it will be replaced with last value received on the corresponding input (In1,In2,in3 and In4).
 Prior to sending the eMail you can use Group Addresses, logic, multistate button and even script language (maestros artificial intelligence) to compose complex emails: describing dynamic situation, values of parameters, Maestro actions, reports, lists..... then send this dynamic content on eMail by linking it to the eMail Function Block inputs. here is example for a message:

"Hi it's your Maestro server,

Today \$d at \$t I have detected that sun brightness was \$1Lux and therefor I have set the artificial light at the patio to \$2%

have a nice day,

Maestro"

\$d \$t \$1 and \$2 on the actual email will be replaces with the current values at the time the email is sent.

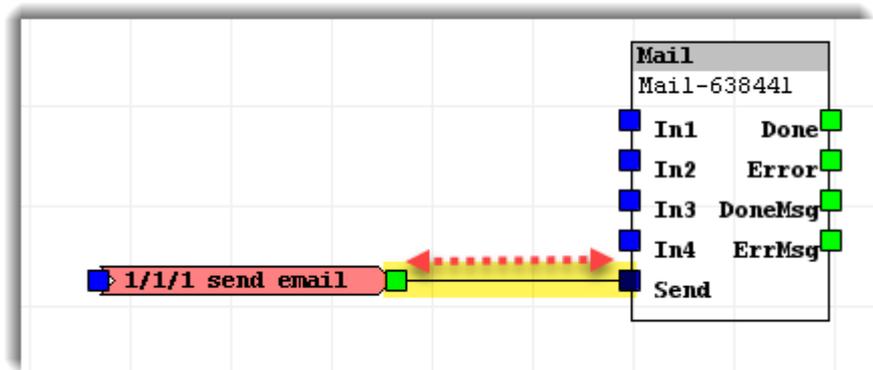
\$d – curren date

\$t – current time

\$1 – input 1 value comminb from weather stetion to input 1

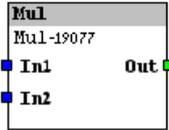
\$2 – input 2 value coming from dimmer absolute value

10.14.5. Link group address/logic/other to the Send input:



Approximately 2 seconds after a true event sent to this input – the Maestro Function Block will send the Email.

10.15. Mul (Multiply) Function-Block:



This Function-Block performs arithmetical multiplication operations.

10.15.1. Functional description:

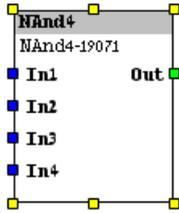
This Function-Block performs multiplication operations between its two **Inputs** and sends the result to the **Output**.

$$Out = In\ 1 \times in\ 2.$$

10.15.2. Functional properties:

Icon	Property/Group	Value	Description
	<i>Multiply Function-Block ID</i>	Text Preview	Unique reference ID of the Function-Block.
	<i>Receive</i>		
→■	<i>In 1</i>	List of the links made with this point.	Coefficient 1
	<i>In 2</i>	List of the links made with this point.	Coefficient 2
	<i>Transmit</i>		
■→	<i>Out</i>	List of the links made with this point.	The product of multiplying the inputs.
	<i>Polling</i>		
	Polling on start up	Check box	Polls the State Input, once, immediately after the Device startup.
	Cyclic polling	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the State Input cyclically at all times using the interval specified.

10.16. NAnd, NAndN Function-Blocks:



NAnd Function-Block performs Logic NAnd operation over its **Inputs**.

10.16.1. Functional description:

When all the values of the **Inputs** of this Function-Block are True: the value of its **Output** will be False. In all other cases - the value of the **Output** is True.

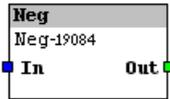
<i>In 1</i>	0	1	0	1
<i>In 2</i>	0	0	1	1
<i>Out</i>	1	1	1	0

Truth table for: NAnd Function-Block (0 stands for False and 1 stands for True)

10.16.2. Functional properties:

<i>Icon</i>	<i>Property/Group</i>	<i>Value</i>	<i>Description</i>
	<i>NAND ID</i>	Text Preview	Unique reference ID of the Function-Block.
	<i>Receive</i>		
→■	<i>In 1</i>	List of the links made with this point.	Input
→■	<i>In 2</i>	List of the links made with this point.	Input
	<i>Transmit</i>		
■→	<i>Out</i>	List of the links made with this point.	The result of the logic NAND function
	<i>Polling</i>		
	Polling on start up	Check box	Polls the State Input, once, immediately after the Device startup.
	Cyclic polling	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the State Input cyclically at all times using the interval specified.

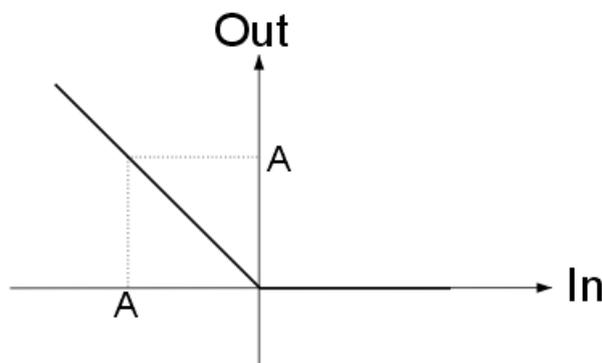
10.17. Neg (Negative) Function-Block:



This function routes the absolute value of negative values and turns positive values to 0.

1.1.1. Functional description:

If the **Input** value is equal to, or more than 0, the **Output** will be set to zero.
 If the **Input** value is less than 0, its absolute value will be routed to the **Output**.

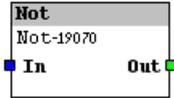


In to Out mapping for Negative Function-Block.

1.1.2. Functional properties:

Icon	Property/Group	Value	Description
	Negative Function-Block ID	Text Preview	Unique reference ID of the Function-Block.
	Receive		
→■	In	List of the links made with this point.	Input values
	Transmit		
■→	Out	List of the links made with this point.	If the <i>In</i> Input value is equal to, or more than 0, the <i>Out</i> Output will be set to zero. If the <i>In</i> Input value is less than 0, its absolute value will be routed to the <i>Out</i> Output.
	Polling		
	Polling on start up	Check box	Polls the State Input, once, immediately after the Device startup.
	Cyclic polling	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1 day	Polls the State Input cyclically at all times using the interval specified.

10.18. Not Function-Block:



Not Function-Block performs Logic NOT operation over its **Inputs**.

10.18.1. Functional description:

The Not function converts True to False and False to True.

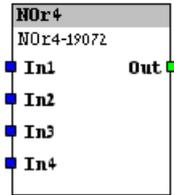
<i>In</i>	0	1
<i>Out</i>	1	0

Truth table for: Not Function-Block (0 stands for False and 1 stands for True).

10.18.2. Functional properties:

<i>Icon</i>	<i>Property/Group</i>	<i>Value</i>	<i>Description</i>
	<i>Not Function-Block ID</i>	Text Preview	Unique reference ID of the Function-Block.
	<i>Receive</i>		
→■	<i>In</i>	List of the links made with this point.	Input
	<i>Transmit</i>		
■→	<i>Out</i>	List of the links made with this point.	The result of the logic Not function
	<i>Polling</i>		
	Polling on start up	Check box	Polls the State Input, once, immediately after the Device startup.
	Cyclic polling	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the State Input cyclically at all times using the interval specified.

10.19. NOR, NORN Function-Blocks:



And Function-Block performs Logic NOR operation over all its **Inputs**.

Functional description:

When all the values of the **Inputs** of this Function-Block are False, the value of its **Output** will be True. In all other cases - the value of the **Output** is False.

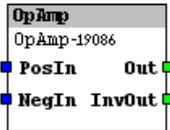
<i>In 1</i>	0	1	0	1
<i>In 2</i>	0	0	1	1
<i>Out</i>	1	0	0	0

Truth table for NOR Function-Block (0 stands for False and 1 stands for True).

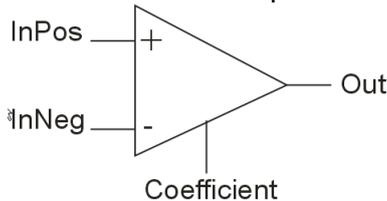
10.19.1. Functional properties:

<i>Icon</i>	<i>Property/Group</i>	<i>Value</i>	<i>Description</i>
	<i>NOR ID</i>	Text Preview	Unique reference ID of the Function-Block.
	<i>Receive</i>		
→■	<i>In 1</i>	List of the links made with this point.	Input
→■	<i>In 2</i>	List of the links made with this point.	Input
	<i>Transmit</i>		
■→	<i>Out</i>	List of the links made with this point.	The result of the logic Nor function
	<i>Polling</i>		
	Polling on start up	Check box	Polls the State Input, once, immediately after the Device startup.
	Cyclic polling	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the State Input cyclically at all times using the interval specified.

10.20. OpAmp Function-Block:



This function is an operational amplifier.



10.20.1. Functional description:

The value of the **Output** *Out* of this Function-Block is: the difference between its **Inputs** (*PosIn NegIn*) multiplied by the *Coefficient* parameter.

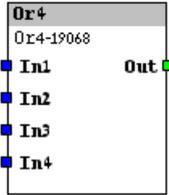
$$Out = (Positive\ In - Negative\ In) \times Coefficient$$

$$InvOut = -Out$$

10.20.2. Functional properties:

	Property/Group	Value	Description
	<i>OpAmp Function-Block ID</i>	Text Preview	Unique reference ID of the Function-Block.
	<i>Receive</i>		
→■	<i>Positive In (Pos)</i>	List of the links made with this point.	Positive input of operational amplifier.
→■	<i>Negative In (Neg)</i>	List of the links made with this point.	Negative input of operational amplifier.
	<i>Transmit</i>		
■→	<i>Out</i>	List of the links made with this point.	$Out = (Positive\ In - Negative\ In) \times Coefficient$
■→	<i>Inv Out (Inv)</i>	List of the links made with this point.	$Inv\ Out = -Out$
	<i>Parameters</i>		
	<i>Coefficient</i>		Coefficient value of operational amplifier.
	<i>Polling</i>		
	Polling on start up	Check box	Polls the State Input, once, immediately after the Device startup.
	Cyclic polling	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the State Input cyclically at all times using the interval specified.

10.21. Or, OrN Function-Blocks:



Or Function-Block performs Logic Or operation over its **Inputs**.

10.21.1. Functional description:

When the values of all **Inputs** of this Function-Block are False the value of its **Output** will be False. In all other cases - the value of the **Output** is True. The number of **Inputs** for this function is assigned by the parameter *Number of inputs*.

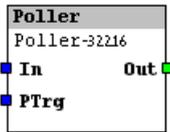
<i>In 1</i>	0	1	0	1	0	1	0	1
<i>In 2</i>	0	0	1	1	0	0	1	1
<i>In 3</i>	0	0	0	0	1	1	1	1
<i>Out</i>	0	1						

Truth table for: Or Function-Block with 3 **Inputs** (0 stands for False and 1 stands for True).

10.21.2. Functional properties:

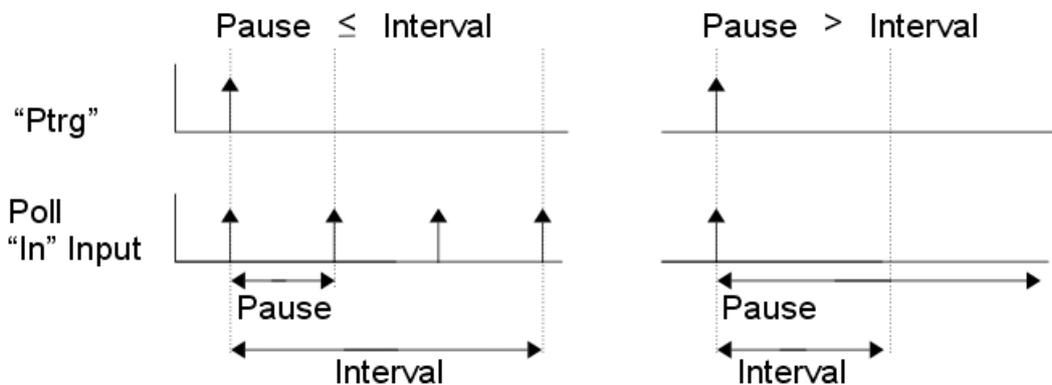
<i>Icon</i>	<i>Property/Group</i>	<i>Value</i>	<i>Description</i>
	OR Function-Block ID	Text Preview	Unique reference ID of the Function-Block.
	<i>Receive</i>		
→■	<i>In 1</i>	List of the links made with this point.	Input
→■	<i>In 2</i>	List of the links made with this point.	Input
→■	<i>In...</i>	List of the links made with this point.	Input
→■	<i>In n</i>	List of the links made with this point.	Input
	<i>Transmit</i>		
■→	<i>Out</i>	List of the links made with this point.	The result of the logic OR function
	<i>Polling</i>		
	Polling on start up	Check box	Polls the State Input, once, immediately after the Device startup.
	Cyclic polling	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the State Input cyclically at all times using the interval specified.

10.22. Poller Function-Block:



This **Function Block** enables the polling of data.

10.22.1. Functional description:



This **Function Block** can be parameterized to send a series of polling requests, to **Links** made to its **In Input**, whenever an **Event** is detected on the **PTrg Input**. On the links window, a check box displays and enables you to define which **Link** will be used for polling. Only one of all the **links** made to the **In Input** can be used for polling.

Use the *Triggered polling interval* & *Triggered Polling Pause* parameters to define the polling procedure: The total time for the procedure is defined by the *Triggered polling interval* property and the frequency of the polling **Events** is defined by the *Triggered polling pause* property. When the *Triggered polling pause* value is greater than the *Triggered polling interval* Value; the Function Block will poll the **In Input** only once at the beginning of the interval.

Any **Event** arriving at **In Input** is immediately routed to the **Out Output**.

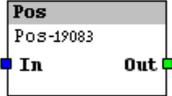
Only Group Addresses can be polled, controller points do not respond to polling requests.

☺ use extral care when setting polling and triggered polling as it generates loads on the EIB/KNX bus.

10.22.2. Functional properties:

Icon	Property/Group	Value	Description
	<i>Poller Function-Block ID</i>	Text Preview	Unique reference ID of the Function-Block.
	<i>Receive</i>		
→■	<i>In</i>	List of the links made with this point.	The Events on this Input are always routed to <i>Out</i> Output.
	<i>Transmit</i>		
■→	<i>Out</i>	List of the links made with this point.	The Events on the <i>In</i> Input are always routed to this Output.
	<i>Polling triggers</i>		
→■	<i>Triggers (PTrg)</i>	List of the links made with this point.	Whenever an event is detected on any of the links made for this point – the Device will perform a series of polling commands to the <i>In</i> input.
	<i>Triggered polling interval</i>	Drop down menu: 1s 5s 15s 30s 60s	The total time duration of triggered polling.
	<i>Triggered polling pause</i>	Drop down menu: 1s 5s 15s 30s 60s	The time duration between single polling commands during triggered polling interval.

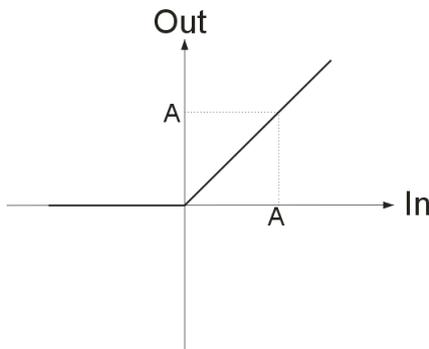
10.23. Pos (Positive) Function-Block:



This function routes positive values and turns negative values to 0.

10.23.1. Functional description:

If the **Input** value is equal to, or more 0, it will be routed to the **Output**.
 If the **Input** value is less than 0, the **Output** will be set to zero.

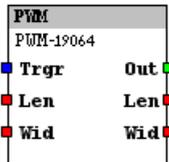


In to Out mapping for Positive Function-Block.

10.23.2. Functional properties:

Icon	Property/Group	Value	Description
	Positive Function-Block ID	Text Preview	Unique reference ID of the Function-Block.
	Receive		
→■	In	List of the links made with this point.	Input values
	Transmit		
■→	Out	List of the links made with this point.	If the In Input value is equal to, or more than 0, it will be routed to the Out Output . If the In Input value is less than 0, the Out Output will be set to zero.
	Polling		
	Polling on start up	Check box	Polls the State Input, once, immediately after the Device startup.
	Cyclic polling	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the State Input cyclically at all times using the interval specified.

10.24. PWM Function-Block:



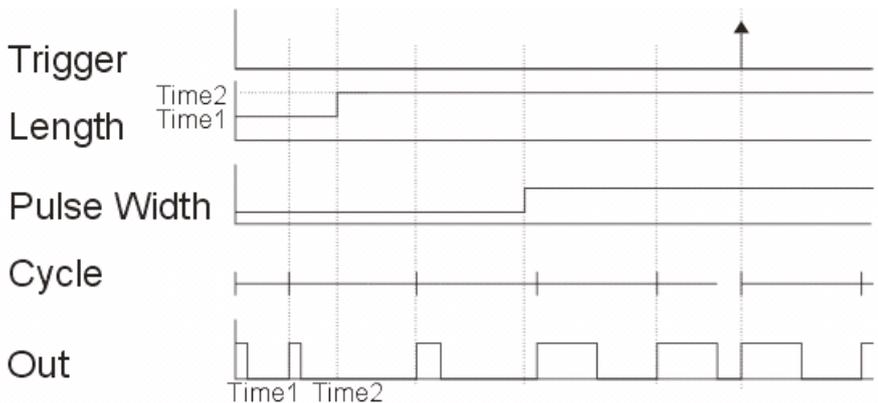
A PWM Function-Block generates a square wave form with variable length and cycle.

10.24.1. Functional description:

This Function-Block generates a regularly repeating square wave form. The length of a cycle is defined by the value of the **Length Input** (seconds), the relative width of the True part is defined by **Pulse width % Input**.

New values for **Length** and **Pulse width (%)** parameters will take effect immediately when received. If **Pulse width** is set to 0, the Oscillator will generate momentary pulses.

After startup, the oscillator immediately starts with the value true on its **Output**. The maximum resolution is 0.1sec, the minimum wave length is 0.5 Sec and the minimum length of the True or False part of the wave is 250ms.

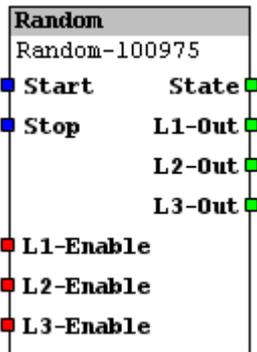


Time diagram for PWM Function-Block.

10.24.2. Functional properties:

Icon	Property/Group	Value	Description
	<i>PWM ID</i>	Text Preview	Unique reference ID of the Function-Block.
	<i>Transmit</i>		
	<i>Out</i>	List of the links made with this point.	The PWM square wave output.
	<i>Receive</i>		
	<i>Trigger (Trgr)</i>	List of the links made with this point.	A True event on this input will start a new wave instantly.
	<i>Parameters</i>		
	<i>Length (Leng (sec))</i>	(sec)	Wave length (seconds).
	<i>Pulse width (Wid (%))</i>	%	Relative width (%) of the True part of the wave.

10.25. Random Function-Block:



Random **Function Block** sends one of 2 predefined values randomly every few seconds at random interval.

10.25.1. Functional description:

For every output, 2 values are predefined using the *Lx OnValue* and the *Lx OffValue*. The Random **Function Block**, randomly, selects one of them and transmits it thru the *Lx-Out* output after, random, period of time of few seconds. The *Start* and the *Stop* inputs are enabling and disabling the activation of the entire function block

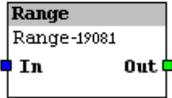
The *Lx-Enable* enables or disables the activation of specific output.

10.25.2. Functional properties:

Icon	Property/Group	Value	Description
	Counter Function-Block ID	Text Preview	Unique reference ID of the Function-Block.
	<i>Receive</i>		
→■	<i>Start</i>	List of the links made with this point.	Enable the random function block
→■	<i>Stop</i>	List of the links made with this point.	Disable the random function block
	<i>Transmit</i>		
■→	<i>State</i>	List of the links made with this point.	State of random function block. 1=enabled, 0=disabled
■→	<i>L1-Out</i>	List of the links made with this point.	One of the two predefined values for this output are chosen randomly and being sent every few seconds at random interval
■→	...	List of the links made with this point.	One of the two predefined values for this output are chosen randomly and being

			sent every few seconds at random interval
■→	<i>Ln-Out</i>	List of the links made with this point.	One of the two predefined values for this output are chosen randomly and being sent every few seconds at random interval
	<i>Parameters</i>		
→■→	<i>L1-Enable</i>	List of the links made with this point.	Enable/disable random transmitting for this load
	<i>NumLoads</i>	integer	Number of loads
	<i>L1 Name</i>		Free text field – name of load
	<i>L1 OnValue</i>	[Value]	First value
	<i>L1 OffValue</i>	[Value]	Second value

10.26. Range Function-Block:



This function routes the **Input** to the **Output**, if the **Input** values are within a predefined range.

10.26.1. Functional description:

this function uses two parameters to define a range:

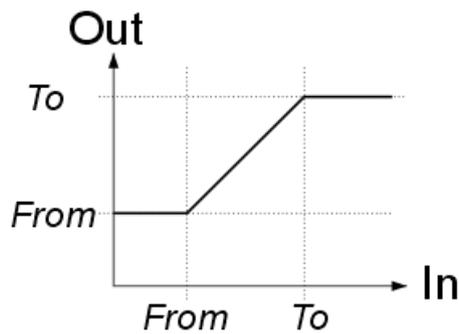
From - is the lower limit of the range.

To - is the upper limit of the range.

If the value of the **Input** is within the range limits, it will be routed to the **Output**.

If the **Input** value is more than *To* - the **Output** value will be *To*.

If the **Input** value is less than *From* - the **Output** value will be equal to *From*.

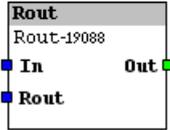


In to Out mapping for Range Function-Block.

10.26.2. Functional properties:

<i>icon</i>	<i>Property/Group</i>	<i>Value</i>	<i>Description</i>
	<i>Range Function-Block ID</i>	Text Preview	Unique reference ID of the Function-Block.
	<i>Receive</i>		
→■	<i>In</i>	List of the links made with this point.	Input values
	<i>Transmit</i>		
■→	<i>Out</i>	List of the links made with this point.	<p>If the value of the <i>In Input</i> is within the range limits, it will be routed to the <i>Out Output</i>.</p> <p>If the <i>Input</i> value is more than the <i>To parameter</i> - the Output value will be equal to the <i>To parameter</i>.</p> <p>If the <i>In Input</i> value is less than the <i>From parameter</i>- the Output value will be equal to the parameter <i>From</i>.</p>
	<i>Parameters</i>		
	<i>From</i>		Lower limit of range
	<i>To</i>		Upper limit of range
	<i>Polling</i>		
	Polling on start up	Check box	Polls the State Input, once, immediately after the Device startup.
	Cyclic polling	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the State Input cyclically at all times using the interval specified.

10.27. Rout Function-Block:

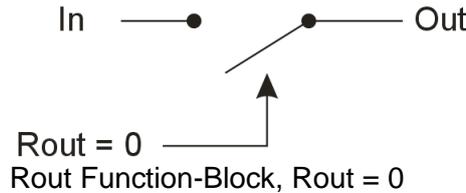


This function routes or blocks **Events** coming from the **Input** to the **Output**.

10.27.1. Functional description:

When the *Rout* value is True, the **Input** will be routed to the **Output**; every **Event** coming to the **Input** will be sent to the *Out*. When the "Send On Enable" parameter is set; a True event on the *Rout* **Input** will trigger **Event** on the *Out* **Output** with the last value of the *In* **Input**.

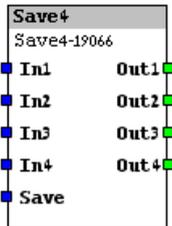
When the *Rout* value is equal to False, the **Input** will be cut from the **Output**. *Out* will remain unchanged and **Events** coming to the **Input** will be ignored.



10.27.2. Functional properties:

Icon	Property/Group	Value	Description
	<i>Rout Function-Block ID</i>	Text Preview	Unique reference ID of the Function-Block.
	<i>Receive</i>		
→■	<i>Input (In)</i>	List of the links made with this point.	When <i>Rout</i> is True - the events on this input will be routed to <i>Out</i> .
→■	<i>Rout</i>	List of the links made with this point.	When this input is true - the input events will be routed to <i>Out</i> .
	<i>Transmit</i>		
■→	<i>Out</i>	List of the links made with this point.	When <i>Rout</i> is True - the <i>input</i> events will be routed to this output.
	<i>Parameters</i>		
	Send On Enable	Check box	
	<i>Polling</i>		
	Polling on start up	Check box	Polls the State Input, once, immediately after the Device startup.
	Cyclic polling	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1 day	Polls the State Input cyclically at all times using the interval specified.

10.28. Save 1,2,N Function-Blocks:



A Save Function-Block saves values in the **Device** flash memory.

10.28.1. Functional description:

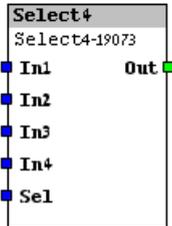
The Save Function-Block can store and recall values. The values are stored in a flash memory. If there is a power failure, they are retained. All outputs transmit their value automatically immediately after startup of the controller.
 2-3 seconds after True **Event** on the **Save Input**: the **Input** values will be copied to the Flash memory and transmitted through the outputs (in practice, the Flash update can occur up to 10 minutes after the Save Event. If during this time there is a power failure, – the new values will not be saved).

10.28.2. Functional properties:

Icon	Property/Group	Value	Description
	Save Function-Block ID	Text Preview	Unique reference ID of the Function-Block.
	Receive		
→■	In 1	List of the links made with this point.	The last value sent to this input will be saved to the flash when the Save input is triggered.
→■	In 2	List of the links made with this point.	The last value sent to this input will be saved to the flash when the Save input is triggered.
→■	In ..	List of the links made with this point.	The last value sent to this input will be saved to the flash when the Save input is triggered.
→■	In n	List of the links made with this point.	The last value sent to this input will be saved to the flash when the Save input is triggered.
→■	Save	List of the links made with this point.	2-3 seconds after True event on this input. The Maestro will perform a

			Save-to-flash operation for the input values and send them from the outputs.
	<i>Transmit</i>		
■→	<i>Out 1</i>	List of the links made with this point.	Saved value 1.
■→	<i>Out 2</i>	List of the links made with this point.	Saved value 2.
■→	<i>Out ..</i>	List of the links made with this point.	Saved value..
■→	<i>Out n</i>	List of the links made with this point.	Saved value n.
	<i>Parameters</i>		
	<i>Save value1</i>	[Initial value]	Initial value
	<i>Save value2</i>	[Initial value]	Initial value
	<i>Save value..</i>	[Initial value]	Initial value
	<i>Save value n</i>	[Initial value]	Initial value
	<i>Polling</i>		
	Polling on start up	Check box	Polls the State Input, once, immediately after the Device startup.
	Cyclic polling	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the State Input cyclically at all times using the interval specified.

10.29. Sel2, SelN (Select) Function-Blocks:



This Function-Block routes one of the **Inputs** to the **Output**.

10.29.1. Functional description:

Inputs are assigned by matching order number:

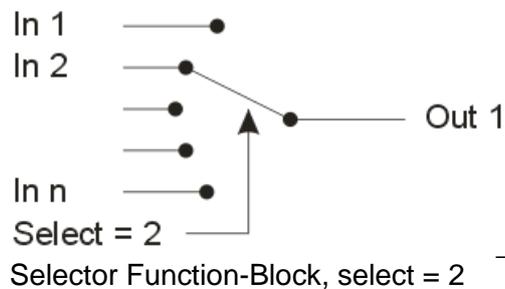
In 1 order number is a 1.

In 2 order number is 2,

...

In n order number is n.

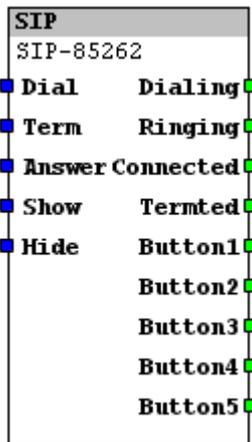
To route the desirable **Input** to the **Output**: you must assign the **Inputs** a “select” number to the **Select Input**. If the value you assigned to the **Select Input** is out of range, the **Input** selection will not be changed. When the "Send On Enable" parameter is set: Event on the **Select Input** will trigger **Event** on the **Out** output having the current value of the selected *input*.



10.29.2. Functional properties:

Icon	Property/Group	Value	Description
	<i>Selector ID</i>	Text Preview	Unique reference ID of the Function-Block.
	<i>Receive</i>		
→■	<i>In 1</i>	List of the links made with this point.	when the value of the Select Input is equal to this input order number. The value of this input will be routed to the Out output,
→■	<i>In ..</i>	List of the links made with this point.	when the value of the Select Input is equal to this input order number. The value of this input will be routed to the Out output,
→■	<i>In n</i>	List of the links made with this point.	when the value of the Select Input is equal to this input order number. The value of this input will be routed to the Out output,
→■	<i>Select (sel)</i>	List of the links made with this point.	The value of this input defines which input will be routed to the Out output
	<i>Transmit</i>		
■→	<i>Out</i>	List of the links made with this point.	The selected input will be routed to this output.
	<i>Parameters</i>		
	<i>Start up selection</i>	Selection 1-n	The initial value of Select immediately after startup.
	<i>Send on Select</i>	Check box	When this function is enabled an event will be generated on any event on the Select input . When this function is disabled event will be generated only when there is event on the selected input.
	<i>Polling</i>		
	Polling on start up	Check box	Polls the State Input, once, immediately after the Device startup.
	Cyclic polling	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the State Input cyclically at all times using the interval specified.

10.30. SIP function block



Maestro supports IP SIP (Session Initiation Protocol) protocol.

SIP is signaling protocol, widely used for controlling multimedia communication sessions such as voice and video calls over Internet Protocol (IP).

By using SIP, Maestro can initiate, modify, and terminate two-party, two way audio and/or video streaming.

The voice and video stream communications are performed using audio and video codecs. The Maestro supports the following codec standards:

- G711 for audio
- H.263 and +H.263 for video.

Therefore Maestro can enable audio (only), video (only) and audio+video calls with another Maestro device or with third party devices supporting the same standards.

(Such as Mobotix x24 product line, AmRoad intercoms, alphatechtechnologies intercoms Linphone software and IP telephones and videophones).

SIP enables features present in the public switched telephone network (PSTN). SIP by itself does not define these features; rather, its focus is call-setup and signaling.

However, it was designed to enable the construction of functionalities of network elements designated as proxy servers and user agents. These are features that permit familiar telephone-like operations such as: dialing a number, causing a phone to ring, hearing ringback tones or a busy signal. Therefore Maestro can perform additional functions using third party devices; for example when using IP SIP PBX normally it will be possible to transfer ongoing calls from Maestro to other devices, record messages, access voicemail box etc..

Maestro can dial to other devices in two ways

- 1) Dial directly (peer to peer) to other devices using their IP address
- 2) Register to IP SIP PBX and dial to other devices through the IP SIP PBX using the other device's registrar ID (for example extension No 201).

In order to activate SIP on the Maestro you have to place one SIP function block on one of the controller sheets of your project.

10.30.1. Functional description

Dial – send to this input the address you want to dial. The address has to be sent as Text data type. When an event arrives to this input, the Maestro will dial the address, using the value passed to the Dial input.

Examples of data sent to this input are:

192.168.1.70 (when using peer to peer communication)

204@192.168.1.2 (dialing to extension 204 of PBX at 192.168.1.2)

the functionality of other linkable inputs of this function block is described in the table below

SIP dialog:

SIP configuration page:

The sip configuration page is located at <http://maestroIP/sip> (for example: <http://192.168.1.70/sip>). It enables you to set SIP communication parameters

SIP Configuration		
Firewall Policy:	No Firewall	
NAT Address:	212.143.214.24 <i>(firewall/router)</i>	<i>Enter Real-IP of your NAT box</i>
STUN Server:	stunserver.org	<i>Enter STUN Server IP or host name</i>
SIP Port:	5060	<i>Default port is 5060</i>
Audio RTP Port:	7078	<i>Default port is 7078</i>
Video RTP Port:	9078	<i>Default port is 9078</i>
SIP Registrar/Proxy mode:	Register	
SIP Proxy/Registrar:	sip:192.168.1.2	<i>Enter IP or host name. Example: sip:a.b.c.d</i>
Proxy/Registrar Identity:	sip:271@192.168.1.71	<i>Example: sip:my_name@my_host_or_ip</i>
Echo Cancellation:	N/A	
DTMF Mode:	Inband (Audio)	
SIP Authentication		
User Name:	271	<i>Leave empty for disable</i>
User ID:	271	<i>Usually the same as username or blank</i>
Password:	••••••	
Security Realm:	asterisk	<i>(auth domain) can be left empty if not known</i>
<input type="button" value="Apply Changes"/> <input type="button" value="Reset"/>		

10.30.1.1. Firewall policy:

enable this parameter when your call are going out of your local LAN throe firewall.

10.30.1.2. NAT Address:

Real IP Address of the firewall

10.30.1.3. STUN Server:

STUN server address

10.30.1.4. SIP Port:

IP port for SIP protocol. Default is: 5060.

10.30.1.5. Audio RTP Port:

IP port for audio streaming. Default is: 7078.

10.30.1.6. Video RTP Port:

IP port for video streaming. Default is: 9078.

10.30.1.7. SIP Registrar/Proxy mode:

- select Enable to activate Registrar/Proxy mode (the Maestro will register to the PBX). Peer to peer is calling is enabled even on registered mode.

- select Disable to use peer to peer mode only.

SIP Registrar/Proxy: Enter the IP or name of host.

For example: sip:192.168.1.71 where 192.168.1.71 is the IP address of the Maestro.

10.30.1.8. SIP Proxy/Registrar:

Enter here your PBX IP Address, for example: sip:192.168.1.2.

10.30.1.9. Proxy/Registrar Identity:

Enter here your Registrar identity, for example:

sip:271@192.168.1.71 where 271 is the Maestro's extension name and 192.168.1.71 is the Maestro's IP address.

10.30.1.10. Echo Cancellation:

this field is not functional. Acoustic echo cancellation on MTS2-104W-S and MTS3-104W-S is implemented by a hardware audio card assembled on S models (for example MTS3-104-S). This card also implements other techniques to improve audio quality such as noise cancellation and automatic gain control. Non S models have no option for acoustic echo canceller

10.30.1.11. DTMF Mode:

Inband (Audio) – when you select this option, DTMF signals will be sent from the Maestro as voice-frequency signals (audio tones).

SIP Info Message – when you select this option, DTMF signals will be sent from the Maestro using SIP Info Message (digital communication).

10.30.1.12. SIP Authentication

The following fields are used to enter information used by the Maestro during authentication process when logging in to a PBX.

10.30.1.12.1. User Name:

User name as set on the PBX

10.30.1.12.2. User ID:

User ID as set on the PBX

10.30.1.12.3. Password:

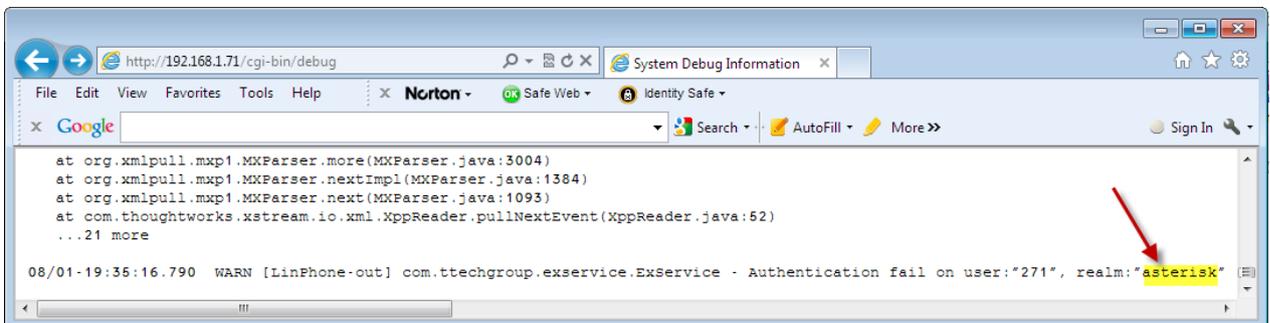
Password as set on the PBX

10.30.1.12.4. Security Realm:

Security Realm of PBX.

To find out the Security Realm of the PBX:

- 1) Connect the PBX to the LAN and set all of the parameters of the Maestros extension.
- 2) Download to the Maestro a project having SIP function block on one of its controller sheets.
- 3) Set all of the SIP page parameters (according to the description above) and push the *Apply Changes* button.
- 4) Using a browser, open the Maestros debug page (located on `http://<IP of Maestro>/cgi-bin/debug`). scroll down to the bottom of the page, the phrase inside the double quotes is the PBX's Realm.



Copy the realm from the Maestros debug file and paste it to the Security realm field on the SIP setting page of the Maestro.

10.30.2. Functional properties:

<i>Icon</i>	<i>Property/Group</i>	<i>Value</i>	<i>Description</i>
	<i>ID</i>	Text Preview	Unique ID reference of the Function-Block.
	<i>Receive</i>		
→■	<i>Dial</i>	List of the links made with this point.	When an event arrives to this input, the Maestro will dial, using the value passed to the <i>Dial</i> input, (in Text format), as the address.
→■	<i>Term</i>	List of the links made with this point.	True event on this input will terminate current call, or current dial, or current ring.
→■	<i>Answer</i>	List of the links made with this point.	True event on this input answers the incoming call
→■	<i>Show</i>	List of the links made with this point.	True event on this input shows the SIP phone communication dialog
→■	<i>Hide</i>	List of the links made with this point.	True event on this input hides the SIP phone communication dialog
	<i>Transmit</i>		
■→	<i>Dialing Number (Dialing)</i>	List of the links made with this point.	This output transmits the destination number in Text format.
■→	<i>Ringing Number (Ringing)</i>	List of the links made with this point.	This output transmits the originator number in Text format (caller ID).
■→	<i>Connected</i>	List of the links made with this point.	This output produces a True event any time a call is established
■→	<i>Terminated</i>	List of the links made with this point.	This output produces a True event any time a call is terminated
■→	<i>Button1</i>	List of the links made with this point.	When this button (The1 st button from the left of the SIP dialog window) is enabled, this output produces a true event when the button is pressed. To enable this button; place any text on "Button1 Name"

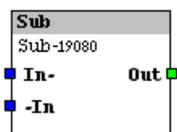
			parameter.
	<i>Button2</i>	List of the links made with this point.	When this button (The 2 nd button from the left of the SIP dialog window) is enabled, this output produces a true event when the button is pressed. To enable this button place any text on “Button2 Name” parameter.
	<i>Button3</i>	List of the links made with this point.	When this button (The 3 rd button from the left of the SIP dialog window) is enabled, this output produces a true event when the button is pressed. To enable this button place any text on “Button3 Name” parameter.
	<i>Button4</i>	List of the links made with this point.	When this button (The 4 th button from the left of the SIP dialog window) is enabled, this output produces a true event when the button is pressed. To enable this button place any text on “Button4 Name” parameter.
	<i>Button5</i>	List of the links made with this point.	When this button (The 5 th button from the left of the SIP dialog window) is enabled, this output produces a true event when the button is pressed. To enable this button place any text on “Button5 Name” parameter.
	<i>Parameters</i>		
	<i>Button1 Name</i>		Place here the Text that will be displayed on the 1 st button (from the left) of the SIP dialog. When this field is left empty the button will be disabled and will not have any functionality.
	<i>Buttn1 Dial</i>		Place here the address you

			want this button to dial. Dialing will be initiated only if the button is pushed when there is no active call
	<i>Button1 DTMF</i>		Place here the DTMF number/s you want this button to dial. Dialing will be initiated only if the button is pushed when there is active call
	<i>Button2 Name</i>		Place here the Text that will be displayed on the 2 nd button (from the left) of the SIP dialog. When this field is left empty the button will be disabled and will not have any functionality.
	<i>Button2 Dial</i>		Place here the address you want this button to dial. Dialing will be initiated only if the button is pushed when there is no active call
	<i>Button2 DTMF</i>		Place here the DTMF number/s you want this button to dial. Dialing will be initiated only if the button is pushed when there is active call
	<i>Button3 Name</i>		Place here the Text that will be displayed on the 3 rd button (from the left) of the SIP dialog. When this field is left empty the button will be disabled and will not have any functionality.
	<i>Button3 Dial</i>		Place here the address you want this button to dial. Dialing will be initiated only if the button is pushed when there is no active call
	<i>Button3 DTMF</i>		Place here the DTMF number/s you want this button to dial. Dialing will be initiated only if the button is pushed when there is active call

	<i>Button4 Name</i>		Place here the Text that will be displayed on the 4th button (from the left) of the SIP dialog. When this field is left empty the button will be disabled and will not have any functionality.
	<i>Buttn4 Dial</i>		Place here the address you want this button to dial. Dialing will be initiated only if the button is pushed when there is no active call
	<i>Button4 DTMF</i>		Place here the DTMF number/s you want this button to dial. Dialing will be initiated only if the button is pushed when there is active call
	<i>Button5 Name</i>		Place here the Text that will be displayed on the 5th button (from the left) of the SIP dialog. When this field is left empty the button will be disabled and will not have any functionality.
	<i>Buttn5 Dial</i>		Place here the address you want this button to dial. Dialing will be initiated only if the button is pushed when there is no active call
	<i>Button5 DTMF</i>		Place here the DTMF number/s you want this button to dial. Dialing will be performed only if the button is pushed when there is active call
	<i>Auto answer</i>	Check box	When this function is enabled the Maestro will automatically answer all incoming calls
	<i>Show Dlg on dial</i>	Check box	When this function is enabled the Maestro will automatically pop up the SIP dialog any time dialing is initiated
	<i>Show Dlg on ring</i>	Check box	When this function is

			enabled the Maestro will automatically pop up the SIP dialog any time an incoming call is ringing.
	<i>Show Dlg on connect</i>	Check box	When this function is enabled the Maestro will automatically pop up the SIP dialog any time a call is established.
	<i>Hide Dlg on terminate</i>	Check box	When this function is enabled the Maestro will close the SIP dialog automatically any time a call, ring, or dial is terminated.

10.31. Sub (Subtract) Function-Block:



This Function-Block performs arithmetical Subtraction operations.

10.31.1. Functional description:

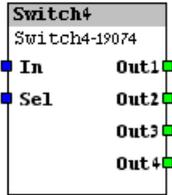
This Function-Block performs subtraction between its two **Inputs**; and sends the result to the **Output**.

$$Out = Minuend\ 1 - Subtrahend\ 2.$$

10.31.2. Functional properties:

	Property/Group	Value	Description
	Subtract Function-Block ID	Text Preview	Unique reference ID of the Function-Block.
	<i>Receive</i>		
→■	<i>minuend (In-)</i>	List of the links made with this point.	The <i>minuend</i>
→■	<i>Subtrahend (-In)</i>	List of the links made with this point.	The <i>subtrahend</i>
	<i>Transmit</i>		
■→	<i>Out</i>	List of the links made with this point.	The difference <i>In 1 - In 2</i> .
	<i>Polling</i>		
	Polling on start up	Check box	Polls the State Input, once, immediately after the Device startup.
	Cyclic polling	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the State Input cyclically at all times using the interval specified.

10.32. Switch2, SwitchN Function-Blocks:



This function routes the **Input** to the selected **Output**.

10.32.1. Functional description:

All **Outputs** are assigned by matching order number:

Out 1 order number is a 1.

Out 2 order number is 2,

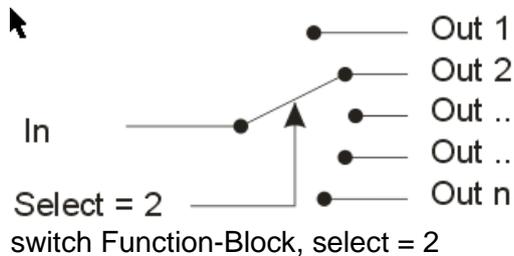
...

Out n order number is n.

To route the *input* to the desirable **Output**, you must assign the **Outputs** order number to the *Select Input*. If the value you assign to the *Select Input* is out of range, the switching will not change.

When the "Send On Enable" parameter is set: Event on the *Select Input* will trigger **Event** on the selected output having the current value of the *Input*

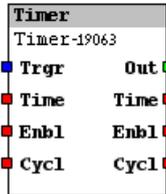
When the "Send On Enable" parameter is not set: Event on the *Select Input* will not trigger **Event** on the selected output.



10.32.2. Functional properties:

Icon	Property/Group	Value	Description
	Switch ID	Text Preview	Unique reference ID of the Function-Block.
	<i>Receive</i>		
→■	<i>Input (In)</i>	List of the links made with this point.	The value of this input will be routed to the selected output
→■	<i>Select (sel)</i>	List of the links made with this point.	This input selects the active output
	<i>Transmit</i>		
■→	<i>Out 1</i>	List of the links made with this point.	The Input will be routed to this Output when the <i>Select</i> input is equal to 1
■→	<i>Out ..</i>	List of the links made with this point.	
■→	<i>Out n</i>	List of the links made with this point.	The Input will be routed to this Output when the <i>Select</i> input is equal to n
	<i>Parameters</i>		
	<i>Start up selection</i>		The initial value of <i>Select</i> immediately after startup.
	<i>Send on Select</i>	Check box	When this function is enabled an event will be generated on any event on the <i>Select</i> input. When this function is disabled event will be generated only when there is event on the selected input.
	<i>Polling</i>		
	Polling on start up	Check box	Polls the State Input, once, immediately after the Device startup.
	Cyclic polling	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the State Input cyclically at all times using the interval specified.

10.33. Timer Function-Block:



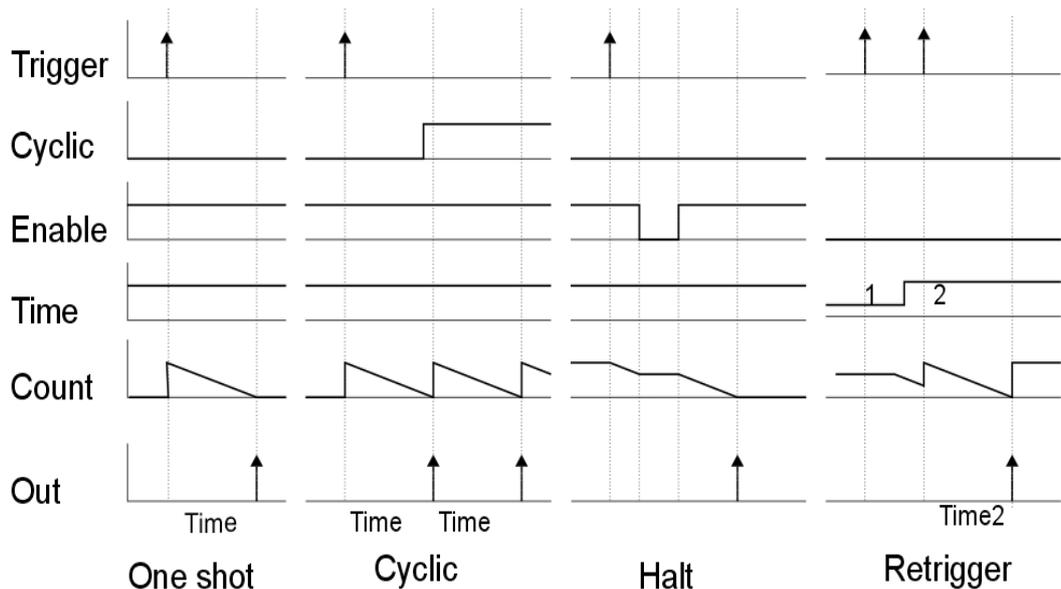
A Timer Function-Block is a count down timer. Once triggered, the timer starts measuring a predetermined time interval. When the time interval is complete, it will generate a True **Event** at its **Output**.

10.33.1. Functional description:

A True **Event** on the **Trigger Input** assigns the current value of **Time Input** to the timer; and starts the timer's countdown. When the countdown is completed, the timer will generate a True **Event** on the **Out Output**.

If the Cyclic **Input** value is True, the Timer will retrigger itself automatically anytime the countdown is completed.

As long as the **Enable Input** is False, the countdown will be halted. Countdown will continue from the point it was halted, as soon as the **Enable Input** is ' True '.

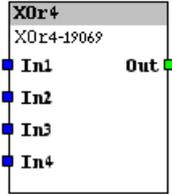


Time diagram Timer Function-Block.

10.33.2. Functional properties:

Icon	Property/Group	Value	Description
	<i>Timer ID</i>	Text Preview	Unique reference ID of the Function-Block.
	<i>Receive</i>		
→■	<i>Trigger (Trgr)</i>	List of the links made with this point.	A True event on this input starts the timer's count down.
	<i>Transmit</i>		
■→	<i>Out</i>	List of the links made with this point.	The Timer's Output.
	<i>Parameters</i>		
→■→	<i>Time</i>	Time (sec)	The starting value (in seconds) of the timer count down Input.
→■→	<i>Enable (Enbl)</i>	Check box	When this parameter value is True,, the timer countdown is enabled, and when this parameter value is False, the timer countdown is halted.
→■→	<i>Cyclic (Cycl)</i>	Check box	If the value of this input is True at the completion of the countdown - the Timer will retrigger itself automatically.

10.34. XOr, XOrN Function-Blocks:



XOr Function-Block performs Logic XOr operation over its **Inputs**.

1.1.3. Functional description:

The XOr function is similar to a binary addition without any carry operations. The number of **Inputs** for this Function-Block is assigned by the parameter *Number of inputs*.

<i>In 1</i>	0	1	0	1	0	1	0	1
<i>In 2</i>	0	0	1	1	0	0	1	1
<i>In 3</i>	0	0	0	0	1	1	1	1
<i>Out</i>	0	1	1	0	1	0	0	1

Truth table for XOR Function-Block with 3 **Inputs** (0 stands for False and 1 stands for True)..

1.1.4. Functional properties:

Icon	Property/Group	Value	Description
	XOR ID	Text Preview	Unique reference ID of the Function-Block.
	<i>Receive</i>		
→■	<i>In 1</i>	List of the links made with this point.	Input
→■	<i>In 2</i>	List of the links made with this point.	Input
→■	<i>In...</i>	List of the links made with this point.	Input
→■	<i>In n</i>	List of the links made with this point.	Input
	<i>Transmit</i>		
■→	<i>Out</i>	List of the links made with this point.	The result of the logic XOR function.
	<i>Polling</i>		
	Polling on start up	Check box	Polls the State Input, once, immediately after the Device startup.
	Cyclic polling	Drop down menu: Disabled 1s 5s 30s 3min 15min 1h 6h 1day	Polls the State Input cyclically at all times using the interval specified.

11. *Web Control*:

Web Control enables you to view and use the Maestro graphic interface from a remote location using IP network and a JAVA enabled browser.

All of the Maestro's functions are available from *Web Control* except:

- The use of the Macro Editor from the Set Up Page.
- Viewing IP camera video
- SIP video phone.
- Back door to set up page by tapping on 4 different corners

To create the link between your browser and the Maestro you have to type the Maestro IP address on your browser.

In case you have a firewall you need to redirect some ports in your "Router":

- 1) Your router has a REAL IP address "RealIP"
- 2) Your Maestro Device has a LOCAL IP address "LocalIP"

Use the administrative web interface of your router to redirect the ports 80 and 1999:

First option is:

Redirect TCP RealIP:80 <-> LocalIP:80

Redirect TCP RealIP:1999 <-> LocalIP:1999

In this case, from outside of the LAN, the address of the Maestro will be: `http://"RealIP"`

Second option (not all routers support this)

In a situation where you don't want to use your RealIPport 80 for the Maestro, you can use any other port No. for the Maestro that is > 1024

For example if you want to use port 8080 you have to redirect:

TCP RealIP:8080 <-> LocalIP:80

TCP RealIP:1999 <-> LocalIP:1999

And from outside of the LAN, the address of the Maestro will be: `http://"RealIP":8080/`

Vista and windows7 users

In order to work around a bug caused by Java and Vista/Windows 7 incommutability you have to perform the following procedure:

step 1: Right click on an empty part of the desktop

step 2: Select *Personalize*

step 3: Vista :Click *Window Color and Appearance*

step 4: Vista: If currently using Aero, then click *Open classic appearance properties* link at bottom, OTHERWISE skip to step 5

step 5: In the Color scheme/themes list, click *Windows Classic* (windows 7 users can select *Windows 7 basic* as well) and then click OK

step 6: Restart your broken Java app -- the exception should now not appear.

Web control is not recommended for long continues sessions: if connection between Maestro to browser breaks, the browser will still show the last page but will not send commands to the Maestro, in this case you have to refresh the browser

12. Logger

The Maestro implements the data logger function. When Logger is used on the **Device**, it will save the logged records on the Maestro's flash disk. It is possible to view log files from a browser on a PC and to download them to your PC.

The log files are in csv format so you can process them using PC software such as Microsoft Excel.

On *run* mode the log records are presented on the *output window*. Using this method is a very useful method to debug complex functions and communication protocols.

The log records are stored on files. Every file can store up to 63K events.

When a file reaches its capacity, a new file will be opened and start logging the incoming events.

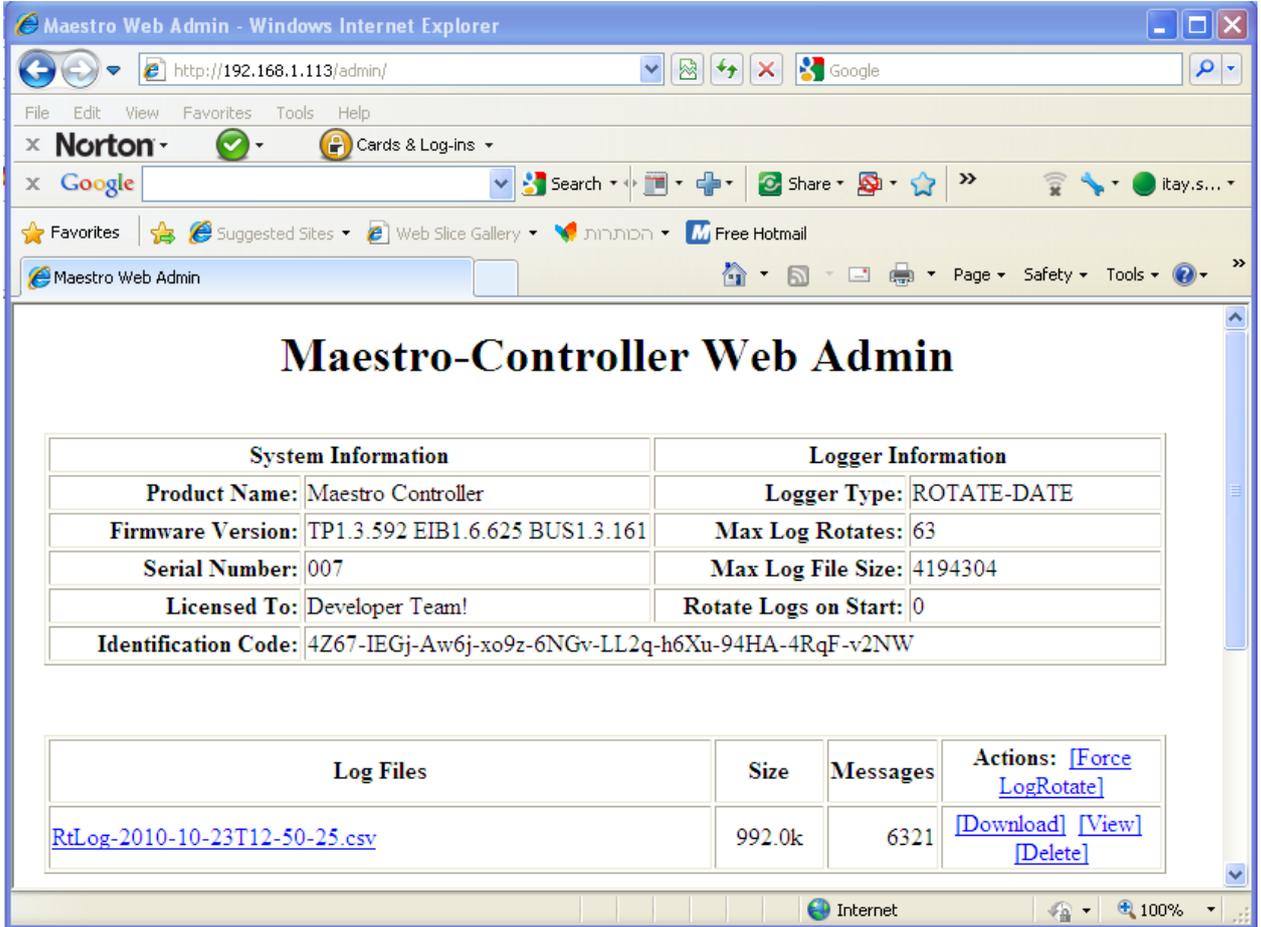
When the log files reach the Maestro capacity for log files, it will free memory space by erasing the oldest log file (FIFO). Log file are named "RtLog" followed by the date and time of file creation.

MS2 and MTS2 or later devices reserve at least 250 Megabytes of flash disk space for logs. The minimum size of a single log is 64 bytes therefore a maestro can log at least 4,000,000 log events.

A **linker's** log has a free text field you can edit by yourself. If your additional text is long, it is possible that 64 bytes (characters) will not be sufficient for the logged data. In this case the Maestro will enlarge the reserved size for this log record by multiplications of 64 bytes until its size will be sufficient to store the data. This will result in a reduction of the total number capacity of logged events. For example, if all of your records consume 128 bytes, the logger will be able to log about 2,000,000 events. In this case the flash can store the logs of 4 events per minute during an entire year before it will have to erase the first log file in order to free memory space.

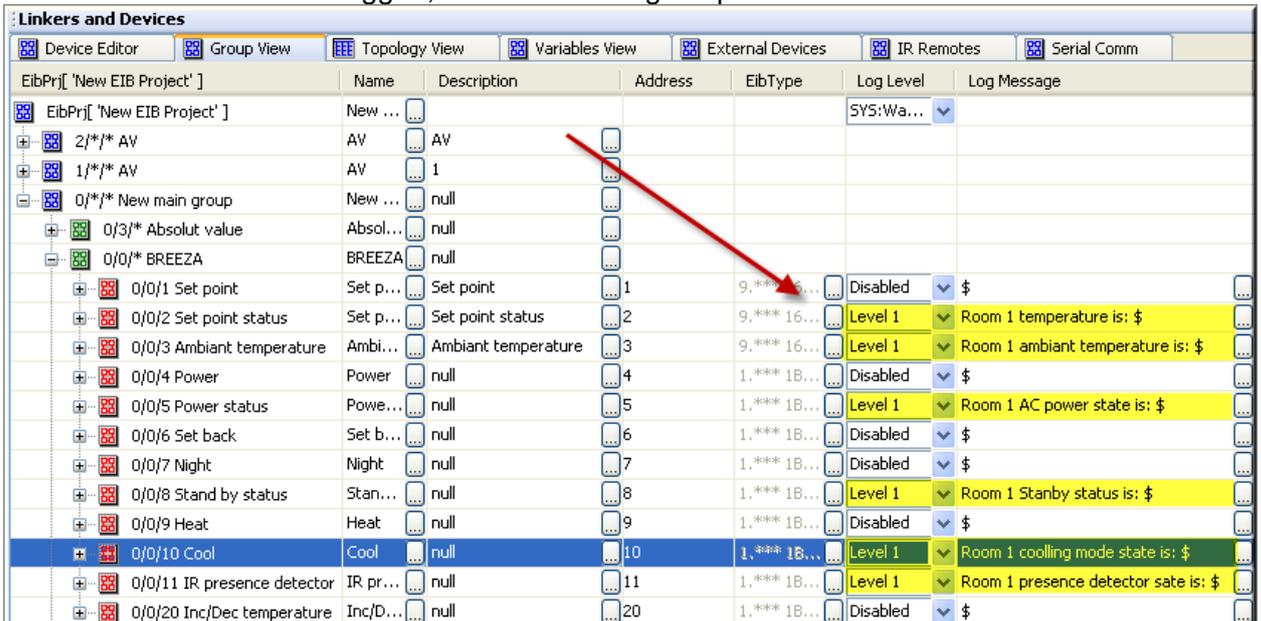
It is not recommended to log more than a few events per minute.

Logged information is accessible through the admin page of the maestro's web server. The IP address you have to type is *maestro/IPaddress/admin* (for example 192.168.1.71/admin) here is the view of the admin page:

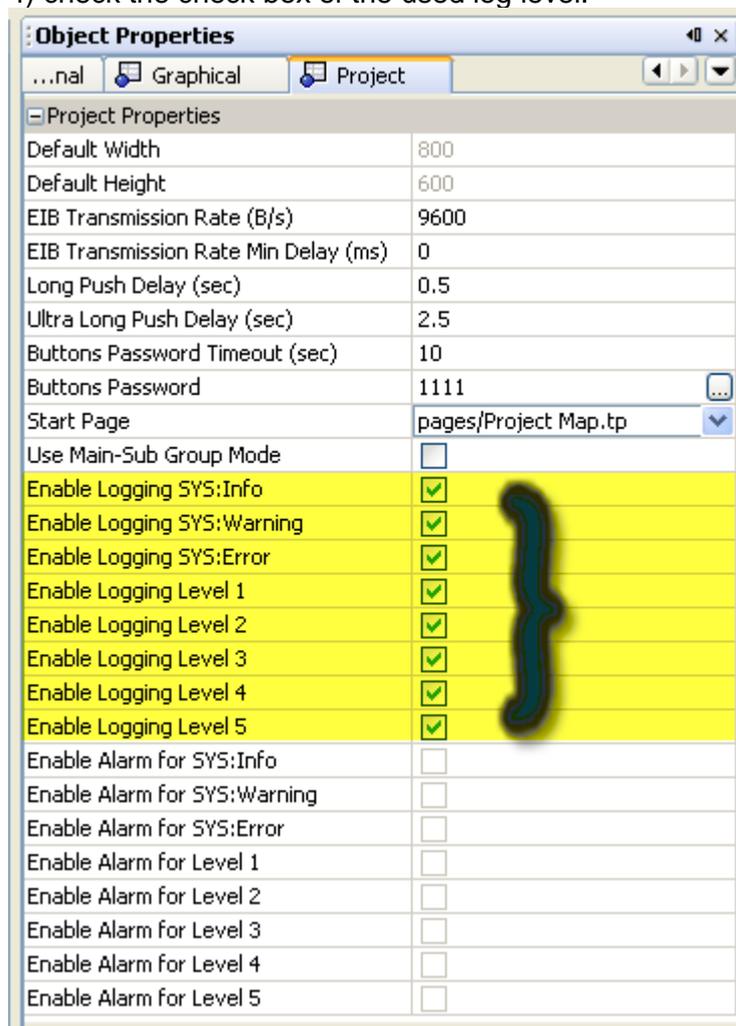


to start a new log file click on *Force LogRotate*
 to view the log on your browser click on *View* or the file name
 to delete a log file click on *Delete*

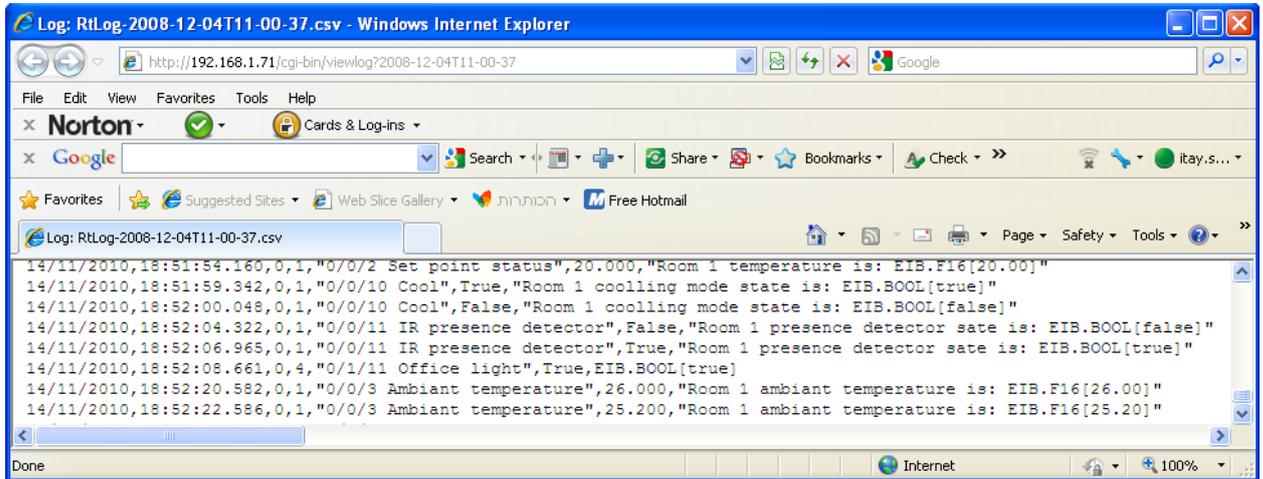
All linkers events can be logged, to enable the log of specific linker:



- 1) select it
- 2) set its log level
- 3) if necessary, add free text at the *Log Message* field. When you use the \$ sign on this field the \$ will be replaced by the linker value.
- 4) check the check box of the used log level:

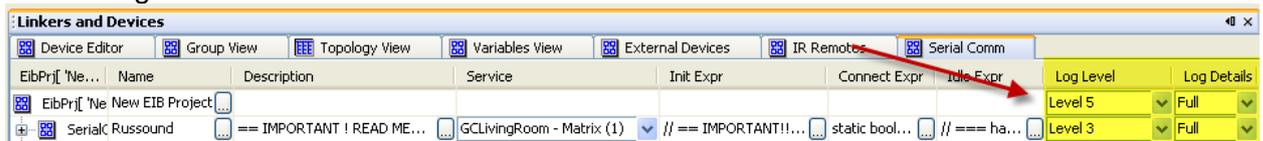


here is a view of a log file as shown by a browser



a log file as shown by a browser

It is also possible to log Serial Protocols communication, to enable the protocol Log Level and set log details to *full*

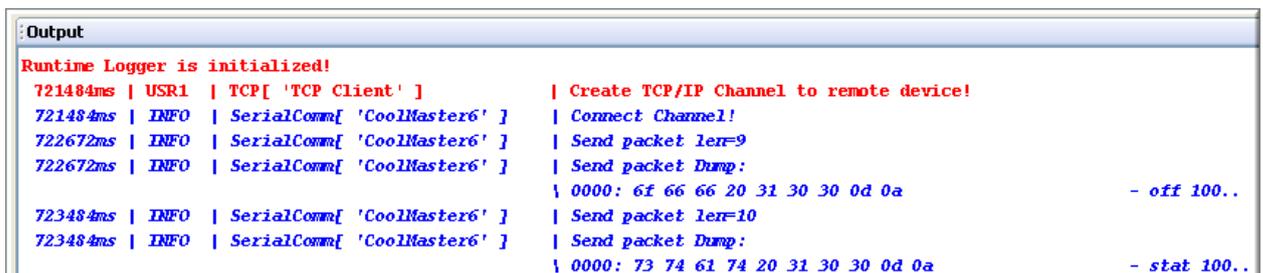


enable the protocol Log

this is useful method for debugging communication protocols using *run* mode and viewing the log records on the *output* window on your Maestro Designer.

IMPORTANT:

When finished with the debugging process, disable all unwanted logs before downloading the project to the Maestro device –. Logging more than few records per minute during long periods of time is not recommended.



on *run* mod: outgoing communication Log is presented on the *output* window

```

Output
Runtime Logger is initialized!
57214922ms | USR3 | SerialComm[ 'Parasound Zpre2' | Connect Channel!
57226656ms | USR3 | SerialComm[ 'Parasound Zpre2' | Receive packet len=9
57226656ms | USR3 | SerialComm[ 'Parasound Zpre2' | Receive packet Dump:
\ 0000: 47 31 20 53 33 20 4d 30 0a - G1 S3 M0.
57226656ms | USR3 | SerialComm[ 'Parasound Zpre2' | ALL Receive buffer Dump:
\ 0000: 47 31 20 53 33 20 4d 30 0a - G1 S3 M0.
57226656ms | USR3 | SerialComm[ 'Parasound Zpre2' | Valid RegExp matcher for receiver name='Inp[ 'EVENT' ] at buffer pos=0
57226656ms | USR3 | SerialComm[ 'Parasound Zpre2' | INVALID RegExp matcher for receiver name='Inp[ 'MAIN_AUDIO_INPUT' ]'
57226656ms | USR3 | SerialComm[ 'Parasound Zpre2' | INVALID RegExp matcher for receiver name='Inp[ 'MAIN_MUTE_ON' ]'
57226656ms | USR3 | SerialComm[ 'Parasound Zpre2' | INVALID RegExp matcher for receiver name='Inp[ 'POWER_ON' ]'
57226656ms | USR3 | SerialComm[ 'Parasound Zpre2' | Found 'best' receiver name='Inp[ 'EVENT' ]' packetLen=9

```

on *run* mode: received string and filter results Log of is presented on the output window

Remark:

It can take up to 10 minutes from the occurrences of an event until it is written on the flash. When the device shuts down by the Power Off button on the set up page – it will write all buffered records to the flash before shutting down.

When the device is cut off from the power, all buffered records will be lost so, up to the last 10 minutes of log will be lost.

13. Maestro script language

The Maestro enables you to write software using scripted language. Many of the Maestro's script language characteristics are similar to Java script programming language. The expression language is used in Serial Communication and in Expression **Function Blocks**.

The expression language is programming language executed by an interpreter, this means that its commands are compiled and executed one by one in real time. If an error is encountered during the execution of one of the commands, the interpreter will stop the execution of the current code and skip all following commands of the current code (only).

The Maestro stops executing the commands of an expression after executing the `return` command or when it reaches the last line of code. If an expression is placed on a line of code after the `return` command, the expression will return the value of this expression.

The text, in a line of code, next to two sequential slashes `//` is used for comments and not executed.

13.1. Variables

The scripted language is statically-typed, which means that all variables must first be declared before they can be used. This involves stating the variable's type and name. for example:

```
int temperature = 22;
```

Maestro utilizes two types of variables

13.1.1. `static` variables:

is any variable declared with the `static` modifier, for example:

```
static int sZone = 1;  
// static integer named sZone initialized with the value 1
```

This tells the compiler that there is exactly one copy of this variable in existence. And it exists at all times.

On Serial Communication:

A `static` variable is valid within the specific protocol where it was declared.

The `static` variable is valid in all expressions of a protocol (transmitters, receivers, ideal expression...).

It is also valid throughout all the expressions of the project including other Communication Protocols and expression **Function Blocks**. When a `static` variable is used outside of the Communication Protocol where it was declared it should be called by its full name: `<protocolName>.<name>`.

In expression **Function Block**:

It is not possible to declare static variables in expression **Function Blocks**.

But all inputs and outputs (`in1, in2, ...out1, out2...`) of the expression **Function**

Block are actually `static` variables and you can use spare inputs and outputs as `static` variables. And use them throughout all the code of the expression **Function Block** (*process-expression*, *init-expression*, *ideal-expression* and *in-expression*). You can also declare static variables on a serial protocol and use them in your expression function block.

13.1.2. Local Variables

A local variable is valid only within the specific expression code segment where it was declared.

Its value is temporary and is lost once the execution of the code segment is finished.

Naming variables:

Variable names are case-sensitive. A variable's name can be any unlimited-length sequence of Unicode letters digits and “_” beginning with a letter. White space is not permitted.

When choosing a name for your variables, use full words instead of cryptic abbreviations. Doing so will make your code easier to read and understand. In many cases it will also make your code self-documenting. Keep in mind that the name you choose must not be a keyword or reserved word.

If the name you choose consists of only one word, spell that word in all lowercase letters. If it consists of more than one word, capitalize the first letter of each subsequent word. The names `zone1Volume` and `roomTemperature` are prime examples of this convention. If your variable stores a constant value, such as

```
static int NUM_ZONES = 6;
```

the convention changes slightly, capitalizing every letter and separating subsequent words with the underscore character. By convention, the underscore character is never used elsewhere.

13.2. Primitive Data Types:

`byte`:

The `byte` Data Type is an 8-bit signed two's complement integer. It has a minimum value of -128 and a maximum value of 127 (inclusive). The `byte` Data Type can be useful for saving memory in large arrays, where the memory savings actually matters. They can also be used in place of `int` where their limits help to clarify your code; the fact that a variable's range is limited can serve as a form of documentation.

`short`:

The `short` data type is a 16-bit signed two's complement integer. It has a minimum value of -32,768 and a maximum value of 32,767 (inclusive). As with `byte` Data Type, the same guidelines apply: you can use a `short` to save memory in large arrays, in situations where the memory savings actually matters.

`int`:

The `int` Data Type is a 32-bit signed two's complement integer. It has a minimum value of -2,147,483,648 and a maximum value of 2,147,483,647 (inclusive). For integral values, this Data Type is generally the default choice unless there is a reason (such as above) to choose something else. This Data Type will most likely be large enough for the numbers your program will use, but if you need a wider range of values, use `long` instead.

long:

The long Data Type is a 64-bit signed two's complement integer. It has a minimum value of -9,223,372,036,854,775,808 and a maximum value of 9,223,372,036,854,775,807 (inclusive). Use this Data Type when you need a range of values wider than those provided by int.

float:

The float Data Type is a single-precision 32-bit IEEE 754 floating point. Its range of values is beyond the scope of this discussion, but is specified in section 4.2.3 of the Java Language Specification. float represents its value by a mantissa and exponent: 1.2345678E30

double:

The double Data Type is a double-precision 64-bit IEEE 754 floating point. Its range of values is beyond the scope of this discussion, but is specified in section 4.2.3 of the Java Language Specification. double represents its value by a mantissa and exponent: 1.2345678901234567E50 For decimal values, this Data Type is generally the default choice.

boolean:

The boolean data type has only two possible values: true and false. Use this Data Type for simple flags that track true/false conditions. This Data Type represents one bit of information, but its "size" isn't something that's precisely defined.

char:

The char Data Type is a single 16-bit Unicode character. It has a minimum value of '\u0000' (or 0) and a maximum value of '\uffff' (or 65,535 inclusive)

13.3. string Data Type:

string is a sequence of characters. Enclosing a character string within double quotes will automatically create a string:

```
string display = "CURRENT SOURCE: DVD"
```

You can get a character at a particular index within a string. The index of the first character is 0, while the index of the last character is length()-1. For example, the following code gets the character "4" at index 9 in a string:

```
string volumeText = "Volume: -40db";  
char tens = volumeText[9];
```

You can also set characters of a string:

```
string volumeText = "Volume: -40db";  
volumeText[9] = "2"; //the outcome will be "Volume: -20db"
```

Or

```
volumeText[9] = "15"; //the outcome will be "Volume: -15db"
```

In this example the assignment of the string "15" will start on character index 9 and since it has 2 characters – will end on index 10.

The + operator can be used for concatenating (joining) two strings together, as shown in the following line of code:

```
string cmd = sRoomZone + ":" + "GET_STATUS:" + "ROOM" + "\r";
```

13.4. Default values

It's not always necessary to assign a value when a variable is declared.

The Maestro never assigns a default value to an uninitialized local variable.

Therefore it is strongly recommended to assign a value when a variable is declared.

If you cannot initialize your local variable where it is declared, make sure to assign it a value before you attempt to use it. Accessing an uninitialized local variable will result in a compile-time error.

13.5. Expressing values:

The integral types `byte`, `short`, `int`, and `long` can be expressed using decimal, binary, octal, or hexadecimal number systems. Decimal is the number system you already use every day; it's based on 10 digits, numbered 0 through 9. The binary number system is base 2, consisting of the digits 0 and 1. The octal number system is base 8, consisting of the digits 0 through 7. The hexadecimal system is base 16, whose digits are the numbers 0 through 9 and the letters A through F. For general-purpose programming, the decimal system is likely to be the only number system you'll ever use. However, if you need octal or hexadecimal, the following example shows the correct syntax. The prefix 0 indicates octal, whereas 0x indicates hexadecimal.

```
int binVal = 0b11010; // The number 26, in binary
int decVal = 26;      // The number 26, in decimal
int octVal = 032;    // The number 26, in octal
int hexVal = 0x1a;   // The number 26, in hexadecimal
```

The floating point types `float` and `double` can also be expressed using `E` or `e` (for scientific notation), `F` or `f` (32-bit float literal) and `D` or `d` (64-bit double literal; this is the default and by convention is omitted).

```
double d1 = 123.4;
double d2 = 1.234e2; //same value as d1, but in scientific notation
float f1 = 123.4f;
```

Literals of the `char` and `String` types may contain any Unicode (UTF-16) characters. If your editor and file system allow it, you can use such characters directly in your code. If not, you can use a "Unicode escape" such as `'\u0108'` (capital C with circumflex), or `"S\u00ED se\u00F1or"` (SÍ Señor in Spanish). Always use 'single quotes' for `char` literals and "double quotes" for `String` literals.

The programming language also supports a few special escape sequences for `char` and `String` literals: `\b` (backspace), `\t` (tab), `\n` (line feed), `\f` (form feed), `\r` (carriage return), `\"` (double quote), `\'` (single quote), and `\\` (backslash).

`null`:

There's also a special `null` literal that can be used as a value for any reference type.

`null` may be assigned to any variable, except variables of primitive types (`byte`, `short`, `int`, `long`, `float`, `double`, `Boolean` AND `char`). There's little you can do with a `null` value beyond testing for its presence. Therefore, `null` is often used in programs as a marker to indicate that some object is unavailable.

13.6. Array:

An array is a container of objects that holds a fixed number of values of a single type. The length of an array is established when the array is created. After creation, its length is fixed.

```
static string[] sSourcesName = {"DVD", "TV", "CD"};
```

Each array element is accessed by its numerical index:

```
string source = sSourcesName[0]
// assigns "DVD" to the variable source
```

13.7. Operators:

Operators are special symbols that perform specific operations on one, two, or three *operands*, and then return a result.

As we explore the *operators* of the Maestro script programming language, it may be helpful for you to know ahead of time which *operators* have the highest precedence. The *operators* in the following table are listed according to precedence order. The closer to the top of the table an *operator* appears, the higher its precedence. *Operators* with higher precedence are evaluated before *operators* with relatively lower precedence. *Operators* on the same line have equal precedence. When *operators* of equal precedence appear in the same expression, a rule must govern which is evaluated first. All binary *operators*, except for the assignment *operators*, are evaluated from left to right. Assignment operators are evaluated right to left.

Operator Precedence	
Operators	Precedence
postfix	<i>expr</i> ++ <i>expr</i> --
unary	++ <i>expr</i> -- <i>expr</i> + <i>expr</i> - <i>expr</i> ~ !
multiplicative	* / %
additive	+ -
shift	<< >> >>>
relational	< > <= >= instanceof
equality	== !=
bitwise AND	&
bitwise exclusive OR	^
bitwise inclusive OR	
logical AND	&&

logical OR	
ternary	? :
assignment	= += -= *= /= %= &= ^= = <<= >>= >>>=

13.7.1. The Simple Assignment *Operator*

One of the most common *operators* that you'll encounter is the simple assignment *operator* "=". It assigns the value on its right to the *operand* on its left:

```
int counter = 0;
```

13.7.2. The Arithmetic *Operators*:

The Maestro script programming language provides *operators* that perform addition, subtraction, multiplication, and division. There's a good chance you'll recognize them by their counterparts in basic mathematics. The only symbol that might look new to you is "%", which divides one *operand* by another and returns the remainder as its result.

- + additive operator (also used for String concatenation)
- subtraction operator
- * multiplication operator
- / division operator
- % remainder operator

The following code lines show the functionality of the arithmetic *operators*.

```
int result = 1 + 2; // result is now 3
result = result - 1; // result is now 2
result = result * 2; // result is now 4
result = result / 2; // result is now 2
result = result + 8; // result is now 10
result = result % 7; // result is now 3
```

You can also combine the arithmetic *operators* with the simple assignment *operator* to create compound assignments. For example, `x+=1;` and `x=x+1;` both increment the value of `x` by 1.

The `+` *operator* can also be used for concatenating (joining) two strings together, as shown in the following lines of code:

```
String firstString = "This is";
String secondString = " a concatenated string.";
String thirdString = firstString+secondString;
```

By the end of the execution of these 3 lines the variable `thirdString` contains "This is a concatenated string."

13.7.3. The Unary Operators:

The unary *operators* require only one *operand*; they perform various operations such as incrementing/decrementing a value by one, negating an expression, or inverting the value of a boolean.

- + Unary plus *operator*; indicates positive value (numbers are positive without this, however)
- Unary minus *operator*; negates an expression
- ++ Increment *operator*; increments a value by 1
- Decrement *operator*; decrements a value by 1
- ! Logical complement *operator*; inverts the value of a Boolean

The following the following lines of code, tests the unary *operators*:

```
int result = +1; // result is now 1
result--; // result is now 0
result++; // result is now 1
result = -result; // result is now -1
boolean success = false; // result is false
success = !success; // result is now true
```

The increment/decrement *operators* can be applied before (prefix) or after (postfix) the *operand*. The code `result++;` and `++result;` will both end in `result` being incremented by one. The only difference is that the prefix version (`++result`) evaluates to the incremented value, whereas the postfix version (`result++`) evaluates to the original value. If you are just performing a simple increment/decrement, it doesn't really matter which version you choose. But if you use this *operator* in part of a larger expression, the one that you choose may make a significant difference.

These two examples, illustrates the prefix/postfix unary increment *operator*:

```
int i = 3;
int j= i; // j = 3
int k= i; // k = 3
j = i++; // j = 3
k = i; // k = 4
```

but

```
int i = 3;
int j= i; // j = 3
int k= i; // k = 3
j = ++i; // j = 4
k = i; // k = 4
```

The equality and relational *operators*:

The equality and relational *operators* determine if one *operand* is greater than, less than, equal to, or not equal to another *operand*. The majority of these *operators* will probably look familiar to you as well. Keep in mind that you must use "==" , not "=", when testing if two primitive values are equal.

```

==      equal to
!=      not equal to
>       greater than
>=     greater than or equal to
<       less than
<=     less than or equal to

```

The following examples demonstrate the comparison *operators*:

```

value1 = 1
value2 = 2
if(value1 == value2) // the condition result is: false
if(value1 != value2) // the condition result is: true
if(value1 > value2)  // the condition result is: false
if(value1 < value2)  // the condition result is: true
if(value1 <= value2) // the condition result is: true

```

13.7.4. The Conditional *Operators*

The `&&` and `||` *operators* perform Conditional-AND and Conditional-OR operations on two boolean expressions. These *operators* exhibit "short-circuiting" behavior, which means that the second operand is evaluated only if needed.

`&&` Conditional-AND

`||` Conditional-OR

The following code, tests these *operators*:

```

int value1 = 1;
int value2 = 2;
if((value1 == 1) && (value2 == 2))
    string text = "value1 is 1 AND value2 is 2";
if((value1 == 1) || (value2 == 1))
    string text = "value1 is 1 OR value2 is 1";

```

13.7.5. The Type Comparison *Operator* `instanceof`

The `instanceof` operator compares a variable to a specified type.

The following example shows how to test if; the first value sent to a transmitter is of the correct type (`int`) and kills the execution if it is of the wrong type:

```

if (!(arg[0] instanceof int))
    return null; // input argument 0 is not integer!

```

When using the `instanceof` *operator*, keep in mind that `null` is not an instance of anything.

13.7.6. Bitwise and bit shift operations:

The Maestro expression programming language also provides *operators* that perform bitwise and bit shift operations on integral types.

The unary bitwise complement *operator* `~` inverts a bit pattern; it can be applied to any of the integral types, making every "0" a "1" and every "1" a "0". For example, a byte contains 8 bits; applying this operator to a value whose bit pattern is "00000000" would change its pattern to "11111111".

The signed left shift *operator* "<<" shifts a bit pattern to the left, and the signed right shift operator ">>" shifts a bit pattern to the right. The bit pattern is given by the left-hand operand, and the number of positions to shift by the right-hand operand. The unsigned right shift operator ">>>" shifts a zero into the leftmost position, while the leftmost position after ">>>" depends on sign extension. Here is an example:

```
int binaryData = 0b00001101;
int flags = binaryData << 4; // flags will get the value
0b11010000
```

The bitwise & operator performs a bitwise AND operation.

The bitwise ^ operator performs a bitwise exclusive OR operation.

The bitwise | operator performs a bitwise inclusive OR operation.

The following program lines, uses the bitwise AND operator to mask data stored on `val` variable:

```
int bitmask = 0x000F;
int val = 0x01A8;
int firstByte = val & bitmask; // firstByte will get the value 8
```

13.8. Expressions statements and blocks.

Operators may be used in building expressions which compute values. Expressions are the core components of statements; statements may be grouped into blocks.

13.8.1. Expressions

An *expression* is a construct made up of variables and *operators*, which are constructed according to the syntax of the language that evaluates to a single value. You've already seen examples of expressions, illustrated in bold below:

```
int cadence = 0;
anArray[0] = 100;
int result = 1 + 2; // result is now 3
if(value1 == value2) string text = "value1 == value2";
```

The data type of the value returned by an expression depends on the elements used in the expression. The expression `cadence = 0` returns an `int` because the assignment operator returns a value of the same data type as its left-hand operand; in this case, `cadence` is an `int`. As you can see from the other expressions, an expression can return other types of values as well, such as `boolean` or `String`.

The Maestro expression programming language allows you to construct compound expressions from various smaller expressions **as long as the data type required by one part of the expression matches the data type of the other**. Here's an example of a compound expression

```
1 * 2 * 3
```

In this particular example, the order in which the expression is evaluated is unimportant because the result of multiplication is independent of order; the outcome is always the same, no matter which order you apply the multiplications. However, this is not true of all expressions. For example, the following expression gives different results, depending on whether you perform the addition or the division operation first:

```
x + y / 100 // ambiguous
```

13.8.2. You can specify exactly how an expression will be evaluated using balanced parenthesis: (and). For example, to make the previous expression unambiguous, you could write the following:

```
(x + y) / 100 // unambiguous, recommended
```

If you don't explicitly indicate the order for the operations to be performed, the order is determined by the precedence assigned to the operators in use within the expression. *Operators* that have a higher precedence get evaluated first. For example, the division *operator* has a higher precedence than does the addition *operator*. Therefore, the following two statements are equivalent:

```
x + y / 100
```

```
x + (y / 100) // unambiguous, recommended
```

When writing compound expressions, be explicit and indicate with parentheses which operators should be evaluated first. This practice makes code easier to read and to maintain.

13.8.3. Statements:

Statements are roughly equivalent to sentences in natural languages. A *statement* forms a complete unit of execution. The following types of expressions can be made into a statement by terminating the expression with a semicolon (;).

- Assignment expressions

Any use of ++ or --

Object creation expressions

Such statements are called *expression statements*. Here are some examples of expression statements.

```
aValue = 8933.234; // assignment statement  
aValue++; // increment statement
```

In addition to *expression statements*, there are two other kinds of statements: *declaration statements* and *control flow statements*. A *declaration statement* declares a variable. You've seen many examples of declaration statements already:

```
double aValue = 8933.234; //declaration statement
```

Finally, *control flow statements* regulate the order that statements get executed. You'll learn about control flow statements in the next section, Control Flow Statements

13.8.4. Blocks:

A *block* is a group of zero or more statements between balanced braces and can be used anywhere a single statement is allowed. The following example illustrates the use of blocks:

```
boolean condition = true;  
if (condition) { // begin block 1  
    string text = "Condition is true.";  
} // end block one  
else { // begin block 2  
    string text = "Condition is false.";  
} // end block 2  
}
```

13.8.5. Control Flow Statements:

The statements inside your source code are generally executed from top to bottom, in the order that they appear. *Control flow statements*, however, break up the flow of execution by employing decision making, looping, and branching, enabling your program to *conditionally* execute particular blocks of code. This section describes the decision-making statements `if-then` and `if-then-else`, the looping statements `for`, `while`, `do-while`, and the branching statement `return` supported by the Maestro expression programming language.

`if-then` statement

The `if-then` statement is the most basic of all the control flow statements. It tells your program to execute a certain section of code *only if* a particular test evaluates to `true`. For example, the PIR will control the light *only if* the Night Mode is enabled:

```
if (nightMode){ // if nightMode value is true
    lightState = pirState;
}
```

If this test evaluates to `false` (meaning that the night mode is not enabled), control jumps to the end of the `if-then` statement.

In addition, the opening and closing braces are optional, provided that the "then" clause contains only one statement:

```
if (nightMode) lightState = pirState;
```

Or

```
if (nightMode)
    lightState = pirState;
```

Deciding when to omit the braces is a matter of personal taste. Omitting them can make the code more brittle. If a second statement is later added to the "then" clause, a common mistake would be forgetting to add the newly required braces. The compiler cannot catch this sort of error; you'll just get the wrong results.

13.8.6. The `if-then-else` Statement

The `if-then-else` statement provides a secondary path of execution when an "if" clause evaluates to `false`. You could use an `if-then-else` statement to take some action if the operating mode is not `nightMode`. In this case, the action is to simply to assign the `switchchState` value to the `lightState` variable.

```
if (nightMode){ // if nightMode value is true
    lightState = pirState;
}
else{
    lightState = switchchState;
}
```

Once a condition is satisfied, the appropriate statements are executed and the remaining conditions are not evaluated.

13.8.7. The while statement

The `while` statement continually executes a block of statements while a particular condition is true. Its syntax can be expressed as:

```
while (expression) {  
    statement(s)  
}
```

The `while` statement evaluates `expression`, which must return a `boolean` value. If the `expression` evaluates to `true`, the `while` statement executes the `statement(s)` in the `while` block. The `while` statement continues testing the `expression` and executing its block until the `expression` evaluates to `false`.

Please exercise caution when using it: you may implement an infinite loop using the `while` statement as follows:

```
while (true){  
    // your code goes here  
}
```

The Maestro expression programming language also provides a `do-while` statement, which can be expressed as follows:

```
do {  
    statement(s)  
} while (expression);
```

The difference between `do-while` and `while` is that `do-while` evaluates its `expression` at the bottom of the loop instead of the top. Therefore, the statements within the `do` block are always executed at least once.

13.8.8. for statement

The `for` statement provides a compact way to iterate over a range of values. Programmers often refer to it as the "for loop" because of the way in which it repeatedly loops until a particular condition is satisfied. The general form of the `for` statement can be expressed as follows:

```
for (initialization; termination; increment) {  
    statement(s)  
}
```

When using this version of the `for` statement, keep in mind that:

- The *initialization* expression initializes the loop; it's executed once, as the loop begins.
- When the *termination* expression evaluates to `false`, the loop terminates.
- The *increment* expression is invoked after each iteration through the loop; it is perfectly acceptable for this expression to increment *or* decrement a value.

The following code, uses the general form of the `for` statement to set an array's values :

```
int[] numbers = {0,0,0,0,0,0,0,0,0,0};  
for(int i=1; i<11; i++){  
    intArrey[i-1] = i;  
}
```

The outcome of this program is: `intArrey equals {1,2,3,4,5,6,7,8,9,10}`
Notice how the code declares a variable within the initialization expression. The scope of this variable extends from its declaration to the end of the block governed by the `for` statement, so it can be used in the termination and increment expressions as well. If the variable that controls a `for` statement is not needed outside of the loop, it's best to declare the variable in the initialization expression. The names `i`, `j`, and `k` are often used to control `for` loops; declaring them within the initialization expression limits their life span and reduces errors.

The three expressions of the `for` loop are optional; an infinite loop can be created as follows:

```
for ( ; ; ) { // infinite loop
           // your code goes here
}
```

13.8.9. return statement

The return statement exits from the current code segment, The return statement has two forms: one that returns a value, and one that doesn't. To return a value, simply put the value (or an expression that calculates the value) after the return keyword.

```
return "Volume is: " + volumeLevelString + "db";
```

14. List of IP ports

Here is a list of IP ports being used by Maestro:

No	Port	USE	MS4/MTS+ (new models)	MS3/MTS3- (old models)	Remarks
1	80	Web control	√	√	html
	8080	Alternative port for Web control	√		html
	1999	Web control		√	
2	22	<ul style="list-style-type: none"> - <i>Maestro Designer</i> upload and download. - Client Apps (iPad/iPhone/Android , windows...) - Remote service (developers). 	√	√	<p>Secured port – it is recommended to leave this port always open!!!</p> <p>Using Secure Shell (SSH) - a cryptographic and password protected network protocol providing a secure path over the Internet.</p>
3	7000	<ul style="list-style-type: none"> - <i>Remote shell</i> for External-systems control. - Simple ASCII-KNX gateway protocol. 	√	√	<p>recommended for use with AMX, Crestron....</p> <p>Not a secured port - Do not open for use out of the firewall!!! Recommended only for internal LAN use.</p>
	3671	KNXnetIP	√	√ works with ETS3 only	<p>Disable firewalls and antiviruses software if it does not work properly. Supports tunneling only, does not support routing!</p>
4	50,000	KNXnetIP		√ works with ETS3 only	<p>Disable firewalls and antiviruses software if it dos not work properly. Supports tunneling only, dos not support routing!</p>
	1502	Modbus	√		
	5055	DMX	√		

	5056 5057 5058				
5	5060 7078 9078	IP SIP VideoPhone	√	√	Default ports can be changed from SIP configuration page: <a href="http://<IPofMaestro>/sip">http://<IPofMaestro>/sip



15. List of http web pages:

Here is a list of http web pages being used by the maestro MS3/MTS3 or older models and their use.

No	IP	USE
1	MaestroIP/admin	Administration and logger
2	MaestroIP/sip	IP SIP VideoPhone settings used with SIP function block
3	MaestroIP/cgi-bin/debug	Developers debug information
4	MaestroIP/eth	LAN2 settings
5	MaestroIP/smtp	Outgoing mail server settings used with Mail function block

16. Remote Shell:

Maestro's *Remote Shell* is a mechanism for interfacing with the Maestro.

Maestro's *Remote Shell* uses Text protocol enabling external system/software to control and monitor the Maestro using Maestro's IP port 7000.

Maestro's *Remote Shell* enables the control and monitoring of the Maestro's system parameters, including elements such as buttons and variables. It also enables the use of the Maestro as a Gateway to KNX bus, sending, receiving, and polling Group Addresses using very simple text protocol.

More than one external system can communicate with the Maestro, using port 7000, at the same time.

16.1. List of commands:

Here is a list of Maestro's *Remote Shell* commands:

To learn and practice Remote Shell commands, you Start embedded shell on port 7000 by using Telnet application on a PC and connect to [MaestroIPAddress:7000](#).

16.1.1. 'help' or '?':

type "help" or "?" to get online help presenting the list of available *Remote Shell* commands.

```
>> help
registry(reg) [action],[name],[id] : Manage current project registry (R/W project storage).
variables(vars) [filter] : List all matching Variables in current running project.
phone(sip) [subcmd],[param] : Manage SIP VoIP Audio-Video phone.
license(lic) [token] : Shows current license information for device.
log(logger) [subcmd],[logger] : Manage Loggers and debug levels.
prj-list(prj-ls) : List all running EIB Projects.
system(java) [subcmd] : Manage JVM.
ui-events(uievt) [action],[id],[context] : Manage event subscriptions for UI Controls.
app-stop : Gracefully Stop Application?
logger(log) [action] : Control runtime project logger.
eib-sub-group-events(sub-group-events) [action],[filter],[context] : Manage event subscriptions for EIB Sub Groups.
eib-sub-groups(sub-groups) [filter] : List all matching EIB Sub Groups in current running project.
exit(quit) : Exit shell.
eib-sub-group(sub-group) addr,[action],[data] : EIB Sub-Group command.
help(?) [command] : Print this help screen.
ui-control(uictrl) id,action,[arg1],[arg2],[arg3] : Simulate UI interaction.
env(env) [subcmd],[variable],[value] : Manage Shell environment variables.
variable-events(var-events) [action],[filter],[context] : Manage event subscriptions for Variables.
variable(var) name,[action],[data] : Variable command.
>>
```

16.1.2. help [command]:

Get help on specific commands describing the specified command format and its parameters.

for example: "help registry"

16.1.3. registry(reg) [action],[name],[id]

Manage current project registry (R/W project storage).

Parameters:

action - What to do: 'get', 'flush', or 'set' (Opt, def='get')

name - Registry variable name. (Opt, def='')

id - Ctrl object ID (use 0 for global registry values). (Opt, def='0')

16.1.4. variables(vars) [filter]:

List all matching Variables in current running project.

Parameters:

filter - Variables filter. Format is <namespace-name>.<variable-name>, use * for ANY! (Opt, def='com.ttechgroup.eib.ProjectShell\$VarFilter@aee23ac8')

16.1.5. phone(sip) [subcmd],[param]:

Manage SIP VoIP Audio-Video phone.

Parameters:

subcmd:

start - Start SIP Phone Service.

stop - Stop SIP Phone Service.

restart - Restart SIP Phone Service.

call - Call remote party.

terminate - Terminate current call.

param - Optional parameter. Used in call subcommand (Opt, def='')

16.1.6. license(lic) [token]:

Shows current license information for device.

Parameters:

token - What to show: 'ident', 'status', 'exp', 'fuses', 'consumer' or 'serial' (Opt, def='status')

16.1.7. log(logger) [subcmd],[logger]:

Manage Loggers and debug levels.

log help:

help for Loggers and debug levels sub commands.

Parameters:

list - Print list with all loggers and their debug levels.

ALL - Set log level to ALL for logger(s).

OFF - Disable logging for logger(s).

DEBUG - Set log level to DEBUG for logger(s).

INFO - Set log level to INFO for logger(s).

WARN - Set log level to WARN for logger(s).

ERROR - Set log level to ERROR for logger(s).

FATAL - Set log level to FATAL for logger(s).

logger - Logger name. (Opt, def='')

16.1.8. pri-list(pri-ls):

List all running EIB Projects.

16.1.9. system(java) [subcmd]:
Manage JVM.

16.1.10. system help:
get the list of System/Java Sub Commands:

Parameters:

- exit - Terminates the currently running JVM.
- halt - Forcibly terminates the currently running JVM.
- gc - Runs the garbage collector.
- fin - Runs the finalization methods of any objects pending finalization.
- mem - Prints Java Virtual Machine memory statistics (in KBytes).
- stack - List All active threads and stack trace in JVM.

16.1.11. ui-events(uievt) [action],[id],[context]:
Manage event subscriptions for UI Controls.

Parameters:

- action - What to do: 'list', 'add', 'del', 'delall' event subscription (Opt, def='list')
- id - UI Object ID to add/del. (Opt, def='0')
- context - Event context. We will pass this data in the corresponding event notification. (Opt, def='')

16.1.12. app-stop:
Gracefully Stop Application!

16.1.13. logger(log) [action]:
Control runtime project logger.

Parameters:

- action - What to do: 'flush', 'rotate', or 'clear' (Opt, def='flush')

16.1.14. eib-sub-group-events(sub-group-events) [action],[filter],[context]:
Manage event subscriptions for EIB/KNX Group Addresses.

Parameters:

- action - What to do: 'list', 'add', 'del' event subscription (Opt, def='list')
- filter - EIB Sub-Group address filter. Format is ##/# or ##/ # is number or * for Any! (Opt, def='Flt[net=0/0/0 mask=0/0/0]')
- context - Event context. We will pass this data in the corresponding event notification. (Opt, def='')

16.1.15. eib-sub-groups(sub-groups) [filter]:
List all matching EIB Sub Groups in current running project.

Parameters:

- filter - EIB Sub-Group address filter. Format is ##/# or ##/ # is number or * for Any! (Opt, def='Flt[net=0/0/0 mask=0/0/0]')

16.1.16. exit(quit):
Exit shell.

- 16.1.17. eib-sub-group(sub-group) addr,[action],[data]:
 EIB Sub-Group command.
 Parameters:
 addr - EIB Sub-Group address. Format is #/#/# or #/#
 action - What to do: 'info', 'send', or 'poll' (Opt, def='info')
 data - Data to be send from 'send' action. (Opt, def=")
- 16.1.18. ui-control(uictrl) id,action,[arg1],[arg2],[arg3]:
 Simulate UI interaction.
 Parameters:
 id - UI Object ID to control.
 action - UI Action:
 {onAnalog,onPress,onRelease,onClick,onLongClick,onClickOn,onClickOff,onHoldOn,onHoldOff,onReleaseOn,onReleaseOff,onPushInc,onPushDec,onReleaseInc,onReleaseDec,onSetState}
 arg1 - Action arg1, depends on type of action. (Opt, def=")
 arg2 - Action arg2, depends on type of action. (Opt, def=")
 arg3 - Action arg3, depends on type of action. (Opt, def=")
- 16.1.19. env(env) [subcmd],[variable],[value]:
 Manage Shell environment variables.
 Parameters:
 subcmd - Subcommand. Use 'help' for list. (Opt, def='help')
 variable - Variable name to set/get. (Opt, def=")
 value - Variable Value to set. (Opt, def=")
- 16.1.20. variable-events(var-events) [action],[filter],[context]:
 Manage event subscriptions for Variables.
 Parameters:
 action - What to do: 'list', 'add', 'del' event subscription (Opt, def='list')
 filter - Variable filter. Format is <namespace-name>.<variable-name>, use * for Any! (Opt, def='com.ttechgroup.eib.ProjectShell\$VarFilter@aae23ac8')
 context - Event context. We will pass this data in the corresponding event notification. (Opt, def=")
- 16.1.21. variable(var) name,[action],[data]:
 Variable command.
 Parameters:
 name - Variable name. Format is <namespace-name>.<variable-name>
 action - What to do: 'info', 'get', or 'set' (Opt, def='info')
 data - Data for the 'set' action. (Opt, def=")
- 16.2. Here is the procedure for connecting to Maestro using ASCII protocol via IP port 7000:
 Make sure the relevant Maestro-Designer-project, having all the group addresses, is loaded to Maestro and that Maestro is Running then follow these instructions:
- 16.2.1. On connection - You have to subscribe for (all) group addresses:
 to get the list of Group addresses as feedback use:
eib-sub-group-events add,*/**
 or quietly with no feedback:

eib-sub-group-events addq,*/*/* (it is OK to send this command every 1 minute if you are not sure your system can track subscription state in all situations)

After subscription – Maestro will start sending updates of telegrams coming from the KNX bus.

The format of the Updates is:

"EVENT[" + context + "]: EIB-SubGroup[" + sg.getGroupAddress().toShortString() + "]= " + data

for example:

EVENT[]: EIB-SubGroup[1/2/3]=EIB.BOOL[true]

means: the value "true" was received from the KNX bus on the Boolean group address "1/2/3"

if Group address 0/3/41 is an 8 bit Group Address and it will get the values 255 and 10 from the KNX bus then Maestro will send back the following Text strings:

EVENT[]: EIB-SubGroup[0/3/41]=EIB.U8[255]

EVENT[]: EIB-SubGroup[0/3/41]=EIB.U8[10]

if Group address 0/0/11 is a 2 Byte Floating Group Address and it will get the values 26.5, 0.055 and -10 from the KNX bus then Maestro will send back the following Text strings:

EVENT[]: EIB-SubGroup[0/3/41]=EIB.F16[26.5]

EVENT[]: EIB-SubGroup[0/3/41]=EIB.F16[0.05]

EVENT[]: EIB-SubGroup[0/3/41]=EIB.F16[-10.00]

Remark: when Maestro initiates a procedure transmitting of telegram to the KNX bus, it will send the event to port 7000 regardless of the transmission success to the KNX bus. So, for example, if the KNX bus is disconnected from the Maestro and the user pushes a button on the Maestro touch screen, in order to send a telegram to the bus: Maestro will report to port 7000 the new value on the Group Address even though it was not transmitted to the bus.

16.2.2. You can query the last known state of group addresses by sending (normally used on connection):

eib-sub-group-events listv,###/#

For example:

eib-sub-group-events listv, 12/1/105

Maestro will return the value in the following format:

12/1/105 | 1. 1Bit Bool| on_off_ch5_state| | <true>**

And if the value is unknown it will return "null"

12/1/105 | 1. 1Bit Bool| on_off_ch5_state| | <null>**

You can use * as wild cards

eib-sub-group-events listv, *//*

16.2.3. To send values to KNX group Address on the KNX bus, use the eib-sub-group command, for example:

To send the value 1 to Group Address 0/1/11, of the type one bit, use one of the following command:

eib-sub-group 0/1/11,send,1 or

eib-sub-group 0/1/11,send,on or

eib-sub-group 0/1/11,send,true

To send the values 128 and 20 to Group Address 0/1/61 of the type 8 bit use the following commands:

eib-sub-group 0/1/61,send,128

eib-sub-group 0/1/61,send,20 or **eib-sub-group 0/1/61,send,020**

16.2.4. You can use **prj-list** command as hart beat for monitoring Maestros status:

no project running:

Active Eib Projects (0 items)

project running:

Active Eib Projects (1 items)

c25dc7a8-fff7-41d9-958d-08e70d7563cb | ObjId[0] | New EIB Project

any time you detect a Maestro-controllers-restarts you can presubscribe and poll for status.

16.2.5. To poll KNX devices for Grope Address state use the **eib-sub-group** command.

For example to poll the Group Address 0/3/161, use the following command:

eib-sub-group 0/3/161,poll

As a result the Maestro will acknowledge:

0/3/161 | poll OK (this is an automatic Maestro acknowledgement and does not mean that the poll telegram was transmitted on the KNX BUS). After a KNX device responds to the polling request, you will get the returned value (only if you previously subscribed this group address using: eib-sub-group-events add...) as a normal EIB sub group event, for example

EVENT[]: EIB-SubGroup[0/3/161]=EIB.F16[26.5]

REMARK:

At the end of every command you must add carriage return (0Dh) followed by line feed (0Ah).

17. LAN web page (MS3/MTS3 or older models only):

The LAN configuration page is located at <http://maestroIP/eth> (for example: <http://192.168.1.70/eth>). It enables you to set LAN parameters for all LAN ports of the Maestro.

Maestro-Controller Web Admin

System Information (1.2.3)	Logger Information
Product Name: Maestro Controller	Logger Type: ROTATE-DATE
Firmware Version: TP1.8.918 EIB1.10.927 BUS1.6.242	Max Log Rotates: 63
Serial Number: 007	Max Log File Size: 4194304
Licensed To: Developer Team!	Rotate Logs on Start: 0
Identification Code: iUMU-4cFo-sDRs-Vsxa-CfSA-Hexb-l59G-sAtd-4uy0-JcY0	

LAN1 Configuration

Network Mode:	Static IP	
IP Address:	192.168.1.74	<i>Enter Static IP Address of LAN1</i>
Subnet Mask:	255.255.255.0	<i>Enter Static Subnet Mask of LAN1</i>
Default Gateway:	192.168.1.1	<i>Enter Static Default Gateway IP Address</i>
DNS:	194.90.1.5	<i>Enter Static DNS IP Address</i>

LAN2 Configuration

Network Mode:	DHCP Client	
IP Address:	192.168.1.75	<i>Enter Static IP Address of LAN2</i>
Subnet Mask:	255.255.255.0	<i>Enter Static Subnet Mask of LAN2</i>
Default Gateway:	192.168.1.1	<i>Enter Static Default Gateway IP Address</i>

Current Network Configuration:

```

eth0      Link encap:Ethernet  HWaddr 00:04:5F:8E:8D:07
          inet addr:192.168.1.74  Bcast:192.168.1.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:855309 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1578 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:75861511 (72.3 MiB)  TX bytes:96905 (94.6 KiB)
          Interrupt:11 Base address:0x6000

eth1      Link encap:Ethernet  HWaddr 00:04:5F:8E:8D:08
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
          Interrupt:10 Base address:0xe000

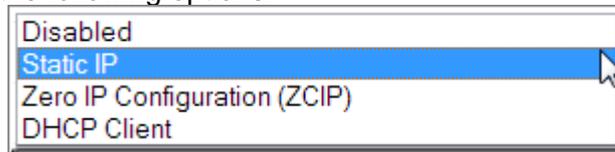
lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
    
```

configuration page (html page on <http://maestroIP/eth>)

LAN

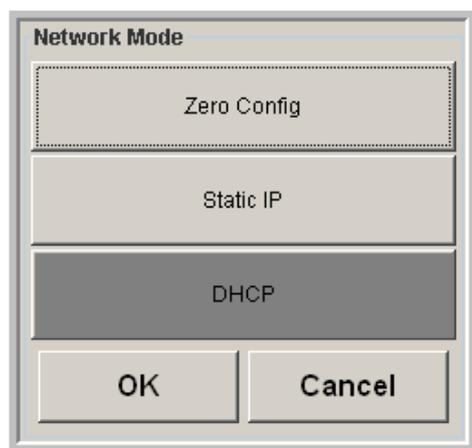
For every LAN port, it has the following fields:

- *Network Mode*: is a drop down menu used to present and set the network mode, it has the following options:



- *IP Address*: used to present and set the network Address on the LAN port
- *Subnet Mask*: used to present and set the network mask.
- *Default Gateway IP*: used to present and set the network Default gateway IP
- *DNS*: used to present and set the network DNS.

Mode: opens dialog window that allows you Select IP network mode:



Network mode

Select one of the options:

- *Zero config* – select it when you want a direct, cross cable, physical connection between the **Device** and your PC. When you select this option the device will assign itself an IP address automatically. Use this IP address on your *Maestro designer* and/or browser to communicate with the device.
- Static IP – select this option when you want the device to obtain static IP address.
- DHCP – Use this option when you want the device to automatically request an IP address from the DHCP server.

18. Maestro KNX – App for iOS:

18.1. General information

Maestro KNX is an App for iOS devices.
Maestro KNX is a client application for the Maestro device.

Maestro KNX enables the same user friendly graphic interface of the Maestro on your iOS device, controlling and monitoring your automation systems via your Maestro device .

Maestro KNX can be downloaded from the Apple App store.

The Maestro can support multiple clients simultaneously, (normally up to 64).

It is possible to have a few iOS devices running the App with different functionalities. For example; children's iPads will have limited control while parents' iPhone will have full control. This can be achieved by using a different starting page for each iOS device and building the Maestro Designer project with different, isolated sections that can't access each other.

The App on the iOS device enables you to setup a few Maestro devices to connect with: (i.e., Maestro in the office, Maestro in Home), and to connect the same maestro using a few IP addresses; Maestro's local IP address, Maestro's external IP address, host name, etc. However, you can only connect to one Maestro at a time.

When using MTS3/MS3 or older devices, There are some differences between the functions directly available from the Maestro device and the functions directly available from the App:

- The App does not support the Macro editor.
- The App does not support the schedule editor.
- The App does not support the setup page.
- The App supports web browser buttons (for details see [video button](#)).
- The iOS devices support more colors than the Maestro so some graphics will look better on the iOS devices.

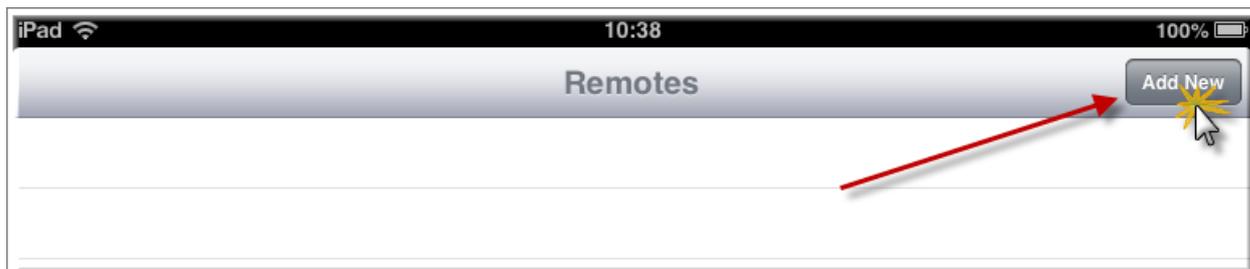
The App connects to the Maestro using port 22, utilizing Secure Shell (SSH) - a **cryptographic and password protected network protocol providing a secure path over the Internet between the iOS device and the Maestro. This optimizes security even when communicating from**

the internet.

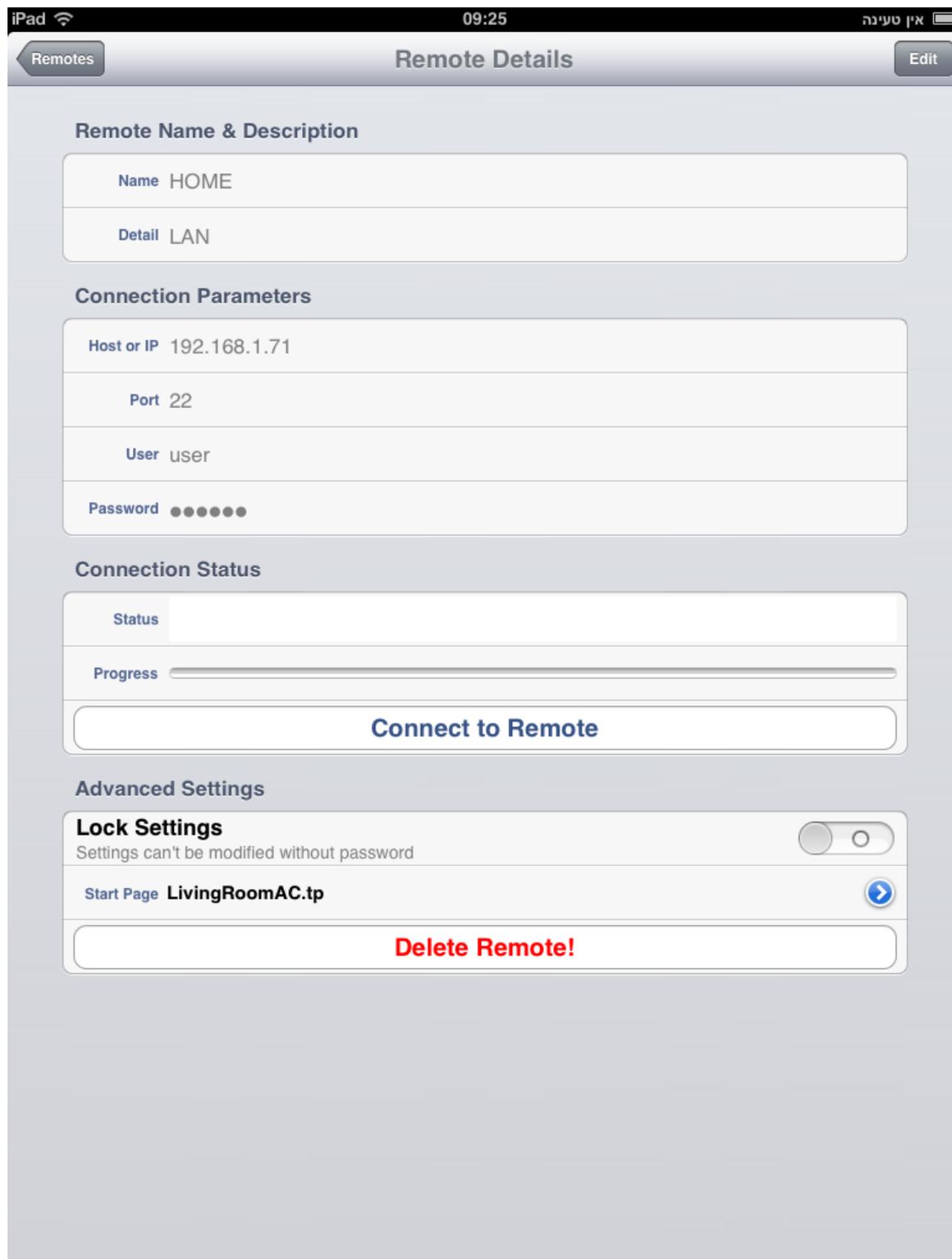
Communication to the Maestro can be executed via Wi-Fi or GPRS.

18.2. Setting up a remote:

To set a connection to a Maestro device you need to add "Remote":



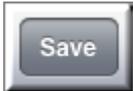
This will open the *Remote Details* configuration page:



To edit the parameters of this page you must enter the Edit mode by pressing the Edit button on the top right corner:



When you have completed choosing your settings you must press the Save button in the same position:



The *Remote Details* page has the following fields:

Remote Name & Description:

- *Name* – free text name of the Maestro device.
- *Detail* – free text details of the Maestro device.

Connection Parameters:

- *Host or IP* – Host or IP address of the Maestro device.
- *Port* – IP port used for communication with the Maestro - must be set to the number "22".
- *User* – user name as set up on Maestro's device (setup page [Remote User/Pass](#))
- *Password* - password as set up on Maestro's device (setup page [Remote User/Pass](#)).

Connection status:

- *Status* – connection status to the Maestro device.
- *Progress* – connection status progress
- *Connect to Remote* – push this button to connect to the Maestro device and start the project.

Advanced settings:

- *Lock Settings* – unlock to edit *Advanced Settings*. You must enter your password to unlock the advanced settings. You must also set the password any time you lock the *Advanced Settings*.
- *Start page:* enables you to define the starting page. If you build your Maestro designer project in isolated segments that can't access each other, (thereby disabling page flips between the segments), then you can connect with different iOS devices having different functionalities. Each iOS device will have its own starting page in a different segment with no possibility of accessing pages in other segments).
- *Delete Remote* – delete the entered remote for the iOS device.

- When the App starts up, it searches for the project name and version currently running on the Maestro. If the Maestro device project and version are similar to the current version on the App, the App will immediately start the project. If the Maestro device version is different from the current version on the App, the App will load the project from the Maestro and only then start it up.
- When the App is already open and recalled it will immediately start the current project without checking for the project name and version.

This means that you need to restart the App after any and all updates of the project on the Maestro device – (In other words, if the App is already open you must close and reopen it first.). This will force the App to look for the project name and version, realize that there is a new project/version, and upload it from the Maestro device.

19. Maestro Graphs and data logger:

Maestro is capable of collecting various values such as measurements of temperature, wind speed, power consumption, solar radiation etc' and presenting them on graphs. The source of such data can be received directly from equipment connected to Maestro using communication means such as: MQTT, MODBUS and KNX or the result of internal calculations performed by Maestro such as the schematic of various loads, calculation of electricity costs, water throughputs, etc'.

Maestro locally collects and stores the required data. It can store tens of millions of samplings enabling to collect history data from up to 1,000 different sources for up to 3 years each.

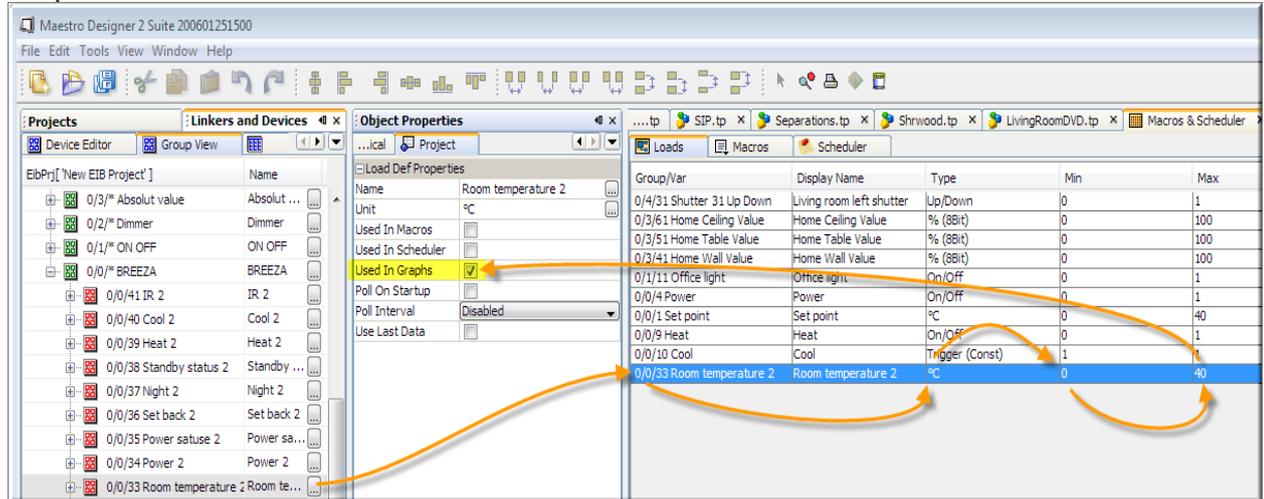
Data can be presented, monitored and analyzed by the end-user for maximum optimization, using various graph types: daily, weekly, monthly and annually. Comparative graphs may be used as well. e.g. electricity consumption this month as compared to the previous month or to the same month last year. The end-user can build his own integrative graphs by selecting different data sources and presenting them in one graph allowing him to compare their values over a common time base.

The end-user can also export all data along any period of time as CSV file type utilizing for instance, Maestro as a KNX data logger, to process and present the logged data by other platforms regardless of what can be presented by Maestro.

Following are the instructions for creating graphs on Maestro:

- First the system integrator has to use **Maestro Designer** for selecting the list of loads to be used for Graphs.
- Then the end-user can use **Maestro server's real-time interface** (App) for presenting graphs, creating custom graphs and exporting CSV files.

- 19.1. To log a data source (such as a KNX group address) follow these steps:

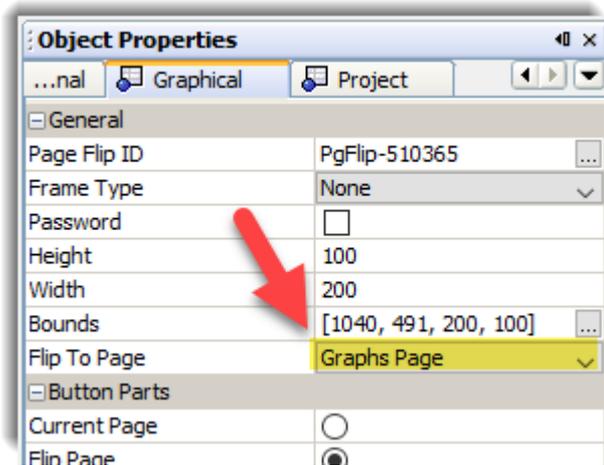


- 19.2. Drag and drop any Linker (KNX group address, Modbus data point, Variable....) to the Loads table (from Macro And Schedule window).
- 19.3. Set the data type, and minimum and maximum values that the graph can present.
- 19.4. Check the Used-In-Graphs box.
- 19.5. Set the additional parameters:
- 19.6. Name – To be shown on the graph legend.
- 19.7. Unit - To be shown on the graph legend.
- 19.8. Poll on start-up and Poll interval – send Polling commands to KNX Group Address cyclically for the data (useful if the sensor can't send the values at the desired interval by itself).
- 19.9. Use Last Data – Maestro's data base stores values on fixed predefined interval. Use Last Data let you select what Maestro should do when time comes to store a new value but, since it's last store, no fresh data arrived to Maestro.

When unchecked – maestro will leave the current storage location empty and when the graph is drawn – at this point time, the graph will be cleared. In other words - Maestro will draw the graph only between real data points.

When checked – maestro will duplicate the last saved value and store the duplicate value on current storage, when the graph is drawn – at this point time the graph will be continued as flat line having the last stored value. In other words - Maestro will use last stored values in order to draw continues graph without gaps.

- 19.10. To enable accesses to graphs from the Graphic interface (APP) - add a Page-Flip Button, select it, and from the Flip-To-Page drop-down menu – select "Graphs Page".



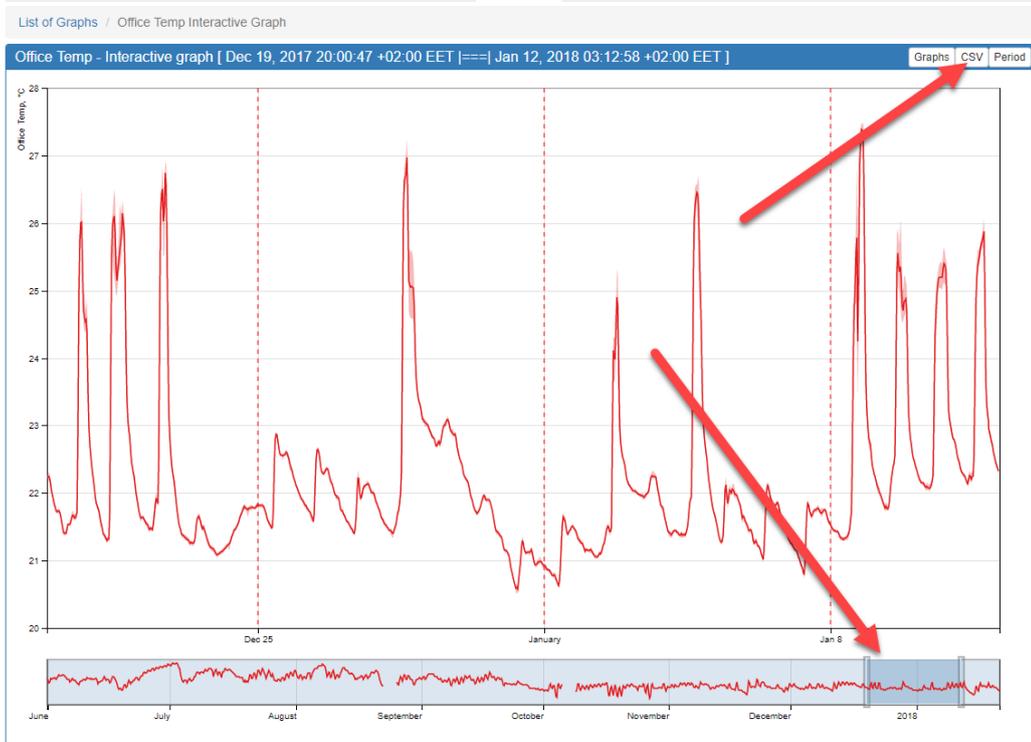
When this Page-Flip button is pressed - Maestro will present a list of all available graphs:

Load Graphs		Search for specific Load	Search
Name	Type	Links	
Ambient	°C	Graphs	Interactive
Office Ext Temp	°C	Graphs	Interactive
Office Press	Value	Graphs	Interactive
Office Temp	°C	Graphs	Interactive
Water	°C	Graphs	Interactive

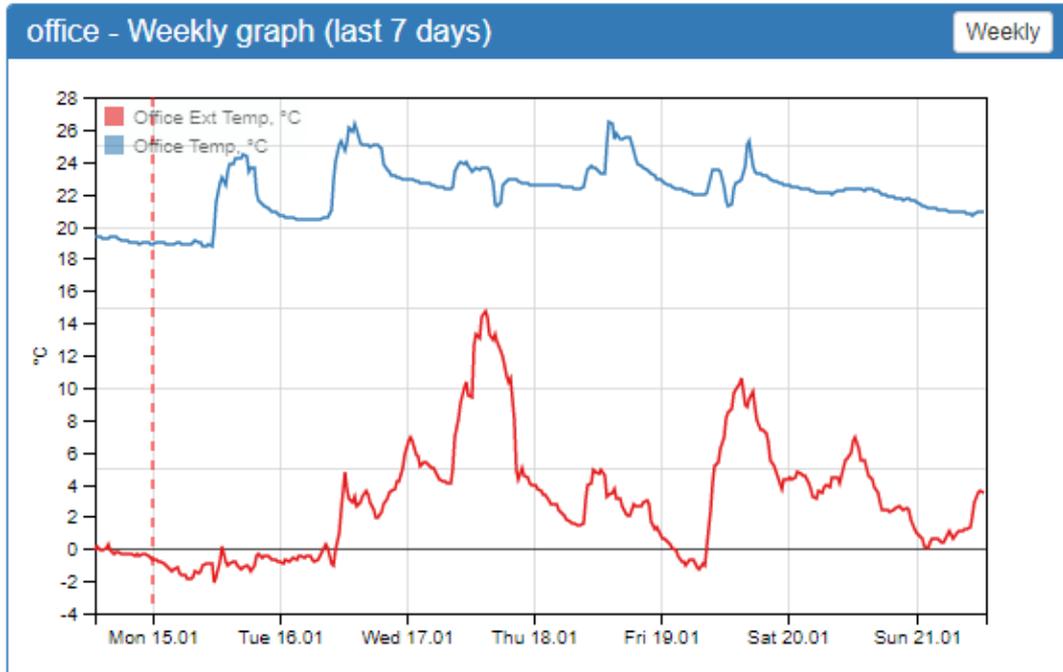
- 19.11. To present basic, predefined, graphs - press the  button, this will open the following views:

- last 120 seconds graph
- Last 24 hours graph
- Last week graph
- Last month graph
- Last year graph

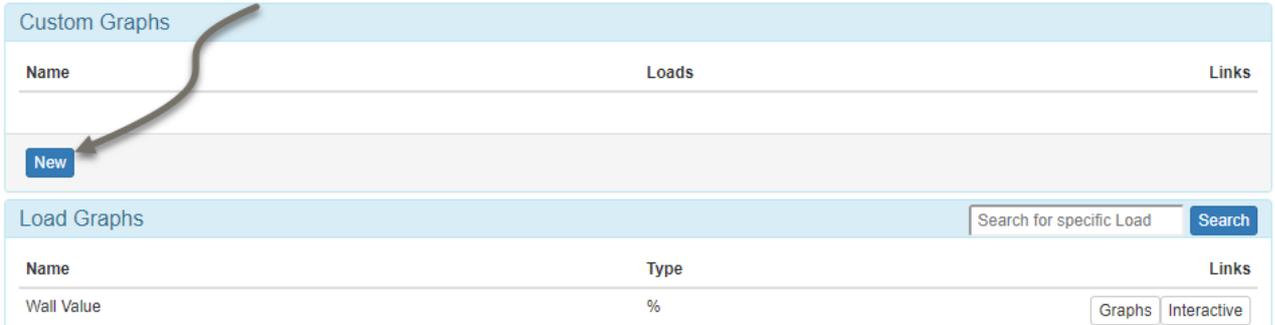
19.12. To present zoomable graph press the **Interactive** button, you can use this view to zoom and to export Data on CSV format:



19.13. To create custom graph, presenting more than one data source on one graph:



Open graph page, then press the "New" button



Custom Graphs

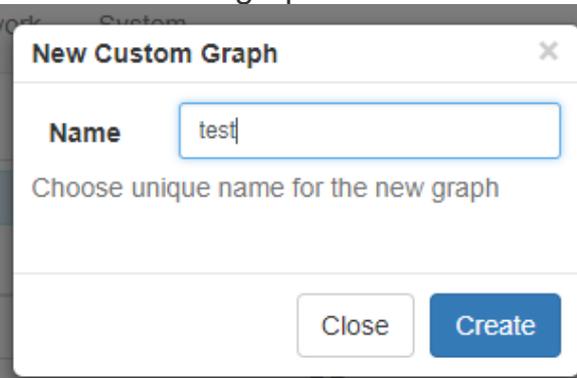
Name	Loads	Links
New		

Load Graphs

Search for specific Load

Name	Type	Links
Wall Value	%	<input type="button" value="Graphs"/> <input type="button" value="Interactive"/>

Name the new graph

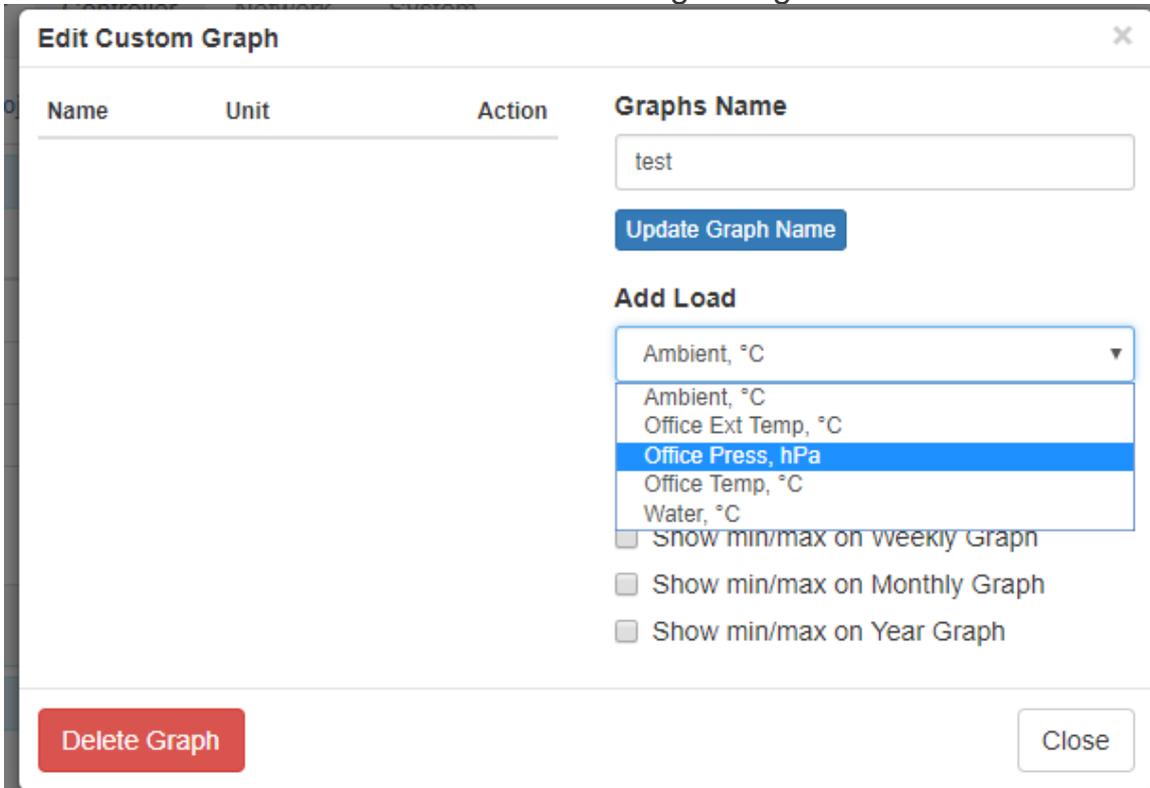


New Custom Graph

Name

Choose unique name for the new graph

Press the "Edit" button and use the following dialog to add loads to the Graph:



Edit Custom Graph

Name	Unit	Action
Graphs Name: test		
Update Graph Name		
Add Load		
<input type="text" value="Ambient, °C"/> <ul style="list-style-type: none"> Ambient, °C Office Ext Temp, °C <li style="background-color: #007bff; color: white;">Office Press, hPa Office Temp, °C Water, °C 		
<input type="checkbox"/> Show min/max on weekly Graph <input type="checkbox"/> Show min/max on Monthly Graph <input type="checkbox"/> Show min/max on Year Graph		
<input type="button" value="Delete Graph"/>		<input type="button" value="Close"/>



General information regarding the data logger:

data is saved on Maestro according to the following time intervals:

- Every 1 minute for the last 48 hours,
- Every 5 minutes for the last 8 days
- Every 15 minutes for the last 1 year and one month
- And every 1 hour for the last 3 years

The actual act of saving to the Flash memory is done once every about 10 minutes, therefor in case of power fail, maestro could lose up to 10 minutes of the latest readings.

when data is transferred to a lower resolution storage (for example from 1-minute interval to 5 minute interval) Maestro stores 3 "points":

- The average value during the interval
- The highest reading during the interval
- The lowest reading during the interval

All 3 points will be exported to the CSV file and can be presented on Maestro Graphs.

20. MQTT support and MQTT to KNX gateway:

MQTT is light weight protocol ideal for Internet of Things (IoT) contexts. It can support enormous amount of end points at real-time response.

MQTT is based on:

- “Broker” that stores messages on hierarchical topics structure, and on
- “Agents” that publish the messages to the Broker and/or Subscribe to get real time updates of the messages from the Broker.

MQTT is used for single miniature sensors up to gigantic systems as Facebook and Amazon for their web services.

Now MQTT devices can gain from the wide variety of maestro’s: control functions, Logger, Graphs presentation... and from its advanced gateway capabilities to other system such as Modbus and DMX.

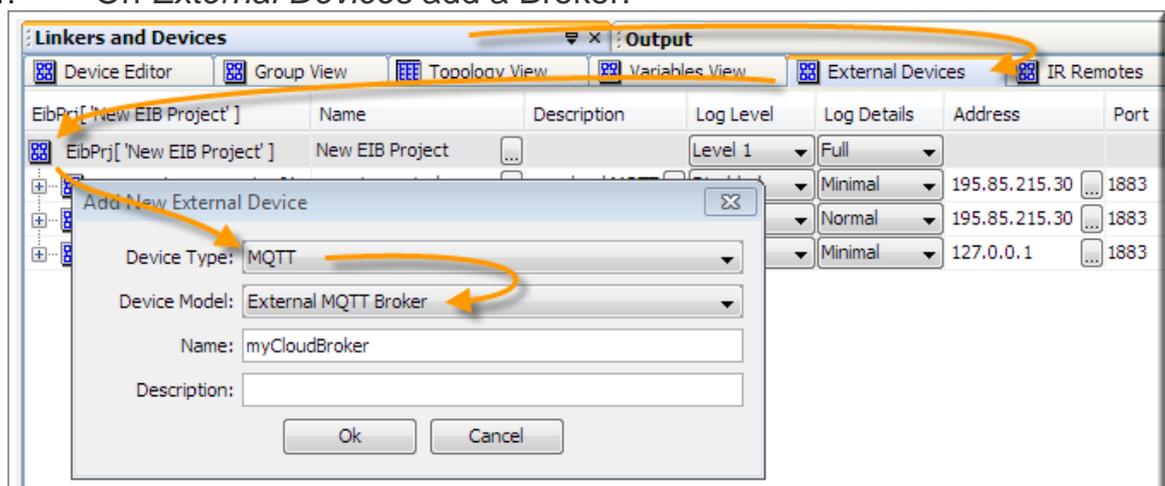
In addition, Maestro can serve as 2 way Gateway between MQTT and KNX, The benefits out of this feature are for example:

- The possibility to connect any number of remote KNX sites over IP combining them into one.
- Connecting KNX to SCADA and big data software for further processing and integration with external systems.
- Integrating MQTT IoT devices with KNX.

Maestro can act as Broker, Agent or both at the same time.

MQTT settings on Maestro Designer:

20.1. On *External Devices* add a Broker:

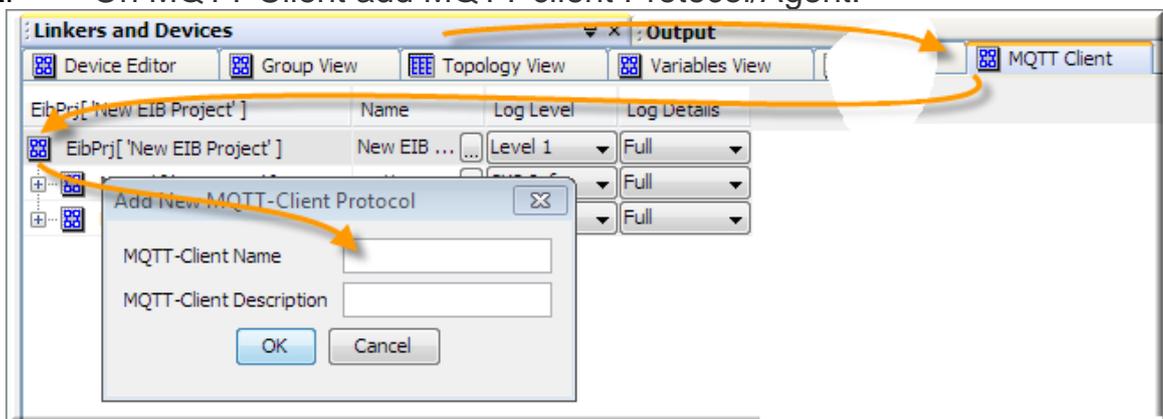


and set it's parameters:

20.1.1. *Name and description* – free text used for documentation and clarification.

- 20.1.2. *Log level and log Details* – settings used for sending Broker events to Maestro Logger.
- 20.1.3. *Address and Port* - IP address of the Broker, if internal broker is used – IP should be set to 127.0.0.1 and port to 1883
- 20.1.4. *User and Password* – Broker’s credentials (optional).
- 20.1.5. *Client ID* – optional.
- 20.1.6. *Topic Prefix* – the root prefix for all topics used on this connection to the Broker (optional).

20.2. On MQTT Client add MQTT client Protocol/Agent:

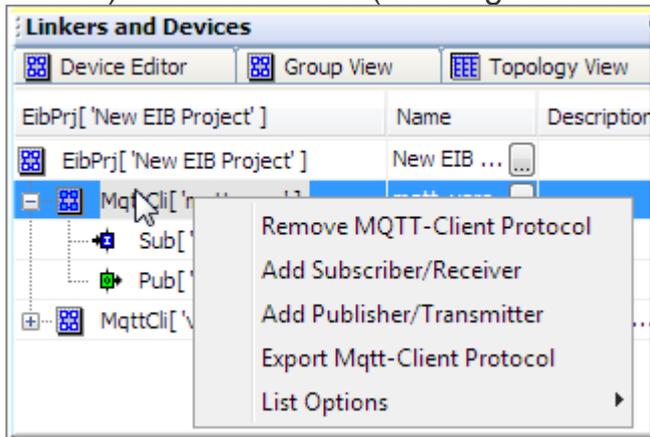


and set it's parameters:

- 20.2.1. *Name and Description* – free text used for documentation and clarification.
- 20.2.2. *Service* – dropdown menu that let you select the Broker for Agents, from the list of Brokers defined on External Devices (as described on previous step).
- 20.2.3. *Topic Prefix* – the middle part of all topics used by this client protocol (optional).
- 20.2.4. *Init Expression:*
Can contains Expression Language code. This code will be executed a single time when the controller starts.
You can use it to declare Static Variables and initialize them.
- 20.2.5. *Connect Expr:*
Can contain Expression Language code. This code will be executed single time when the controller initializes communication with the Broker.
- 20.2.6. *Idle Expression:*
Can contain Expression Language code.
When the Maestro’s processor is not busy, this code will be executed approximately once every second, You can use it to implement functions such as: delays and cyclic writing of data to the Broker.

20.2.7. *Log level and log Details* – settings used for sending Agent’s events to Maestro debug Logger.

20.3. Now you can start adding Subscribers (receiving data from the Broker) and Publishers (sending data to the Broker).



20.3.1. Publishers has the following fields:

20.3.1.1. *Name and Description* - free text used for documentation and clarification.

20.3.1.2. *Topic* - The last part of the Topic path, The complete Topic path therefore is:
[Broker Prefix][Client Prefix][Publisher Topic], and has to specify the exact path of a unique Topic.

Please notice - and have to including “/”, Maestro Designer will not add or remove any characters.

20.3.1.3. *Expression* – can Contain Expression Language code. Any time when Publisher is assigned a value – it will execute its expression language code once.

The value will be assigned to a variable named *arg*.

to publish data use the “*return = data;*” command. If the

Expression field is left empty – the value assigned to the publisher will be sent directly to the Broker.

20.3.1.4. *Type* – the format used for the data sent to the Broker.

20.3.1.5. *Retain* – When checked: a published message will be sent immediately to all current subscribers and in addition will be saved, on the Broker, and send immediately to new subscribers. When unchecked: only current subscribers will get the new messages and then the message will be discarded from the Broker. New subscribers will get no message on subscription.

20.3.1.6. *QoS* – Quality of Service as defined by the MQTT standard.

20.3.1.7. Log level and log Details – settings used for sending Agent's events to Maestro Logger.

20.3.2. Subscribers has the following fields:

20.3.2.1. *Name and Description* - free text used for documentation and clarification.

20.3.2.2. *Topic Filter* - The last part of the topic path. The complete Topic path therefore is:

[Broker Prefix][Client Prefix][Subscriber Topic Filter],

please notice - you have to specify the exact path including "/", Maestro Designer will not add or remove any characters. A filter make use of MQTT's Wildcard + # and \$. To get messages from more than one publisher.

20.3.2.3. *Expression* – Can contain Expression Language code. Any time when message arrives - the code will be executed once. The value of the message will be assigned to a variable named *argv[]* according to filter's field settings.

If no expression is set - The value of the message will be assigned to the Subscriber.

20.3.2.4. *Filter* – can contain a regular expression. The regular expression examine incoming messages, identify parts that match a defined string patterns and pass selected parts to the Expression via *argv[]* variable. If this field is left empty – the complete message will be past to the Subscriber.

20.3.2.5. *Drop*: when unchecked - all incoming messages will be passed to the subscriber.

When checked - only incoming messages different form last messes will be passed to the subscriber ("on change" only).

20.3.2.6. *Log level and log Details* – settings used for sending Agent's events to Maestro Logger.

21. DMX control, DMX to ASCII gateway and KNX to DMX gateway

The DMX 512 protocol is embedded into Maestro's firmware. Maestro runs a software daemon that acts as DMX master. It continuously sends the current value to the DMX channels. The daemon receives and sends simple ASCII commands and thus provides a simple interface and maximum flexibility for controlling DMX devices.

It is possible to use Maestro Designer to send commands to the DMX daemon by using Serial-Protocol and in addition it is possible to send simple ASCII commands to the DMX daemon over LAN using TCP/IP. The daemon processes the incoming commands and accordingly updates the value of the DMX channels.

When using Maestro Designer's Serial-Protocol, the communication to the daemon is achieved via External-Device of type TCP and having the IP Address 127.0.0.1 and:

- port 5055 for DMX line 1
- port 5055 for DMX line 2
- port 5055 for DMX line 3
- port 5055 for DMX line 4

The Daemon supports the following format of commands:

- *fill addr [len] [value]:*
Fill <len> DMX slots from address <addr> with <value>.
- *stop addr [len]:*
Stop Fader and Read <len> DMX slots from address <addr>.
- *write addr val 1[(fader)] val2...:*
Write <valX> DMX slots from address <addr>.
Positive (fader) means delay in seconds.
Example: 127(13.5) - Fade to 127 in 13.5s.
Negative (fader) means steps per second.
Example: 127(-8) - Fade to 127 with speed of 8steps/sec.
- *read addr [len]:*
Read <len> DMX slots from address <addr>.

Here is an example for ASCII command *write*:

write 1 102(-255)

This command writes to DMX channel 1 the value 102 and the channel level will get to 102 at the rate of 255 DMX steps per sec.

The *D* command reports the current state of DMX channel (Feedback).

Maestro will send the *D* command as a response to a read command and after a stop command.

Here is an example for *D* command:

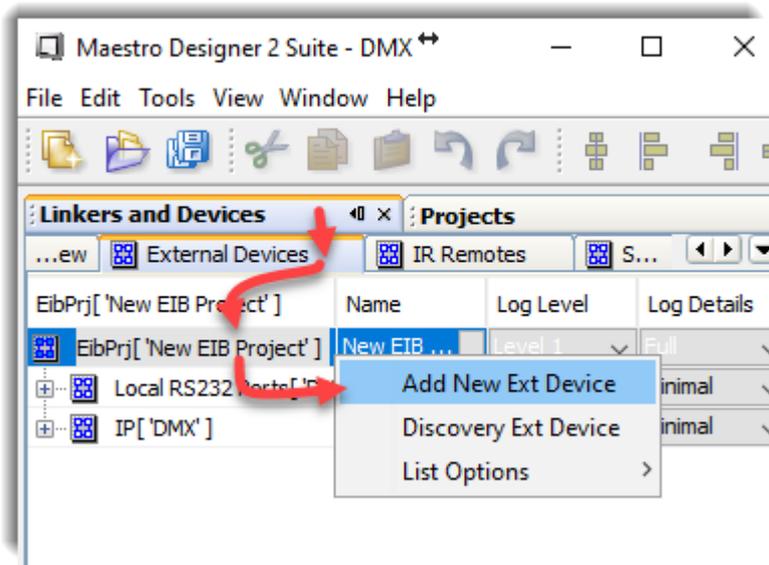
D 1 125

This command coming from the daemon means that the current value of DMX channel 1 is 125.

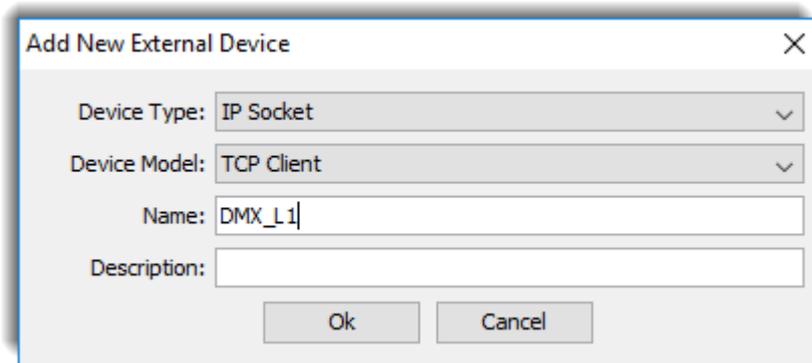
Remark: add carriage return (0A) followed by new line (0D) at the end of any ASCII string sent.

On CD Innovation web site, you can find serial protocol that provides basic DMX control. Here are the instructions how to use it:

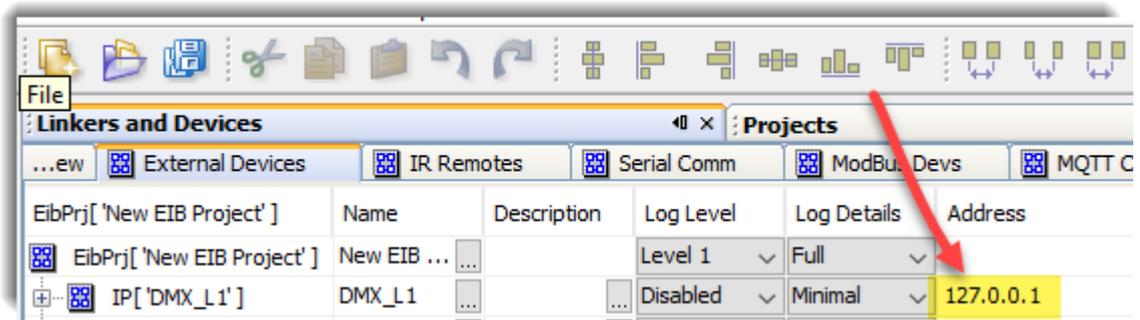
21.1. Add external TCP device:



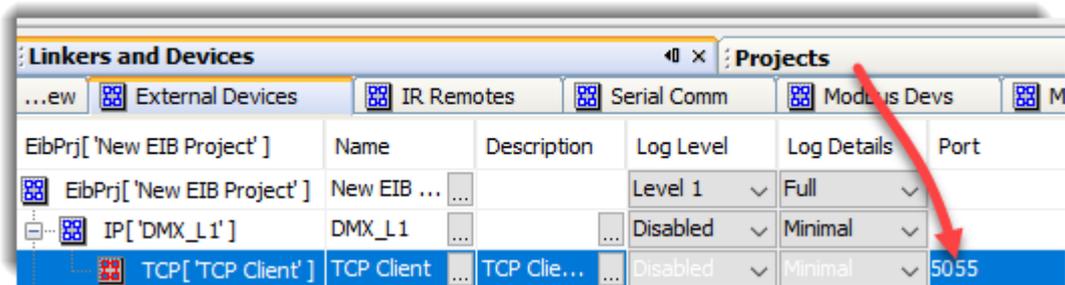
Name it



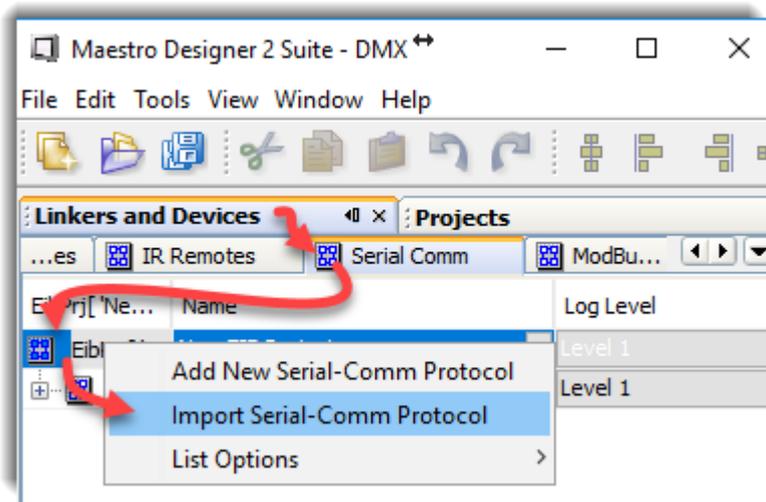
Set the IP address to 127.0.0.1



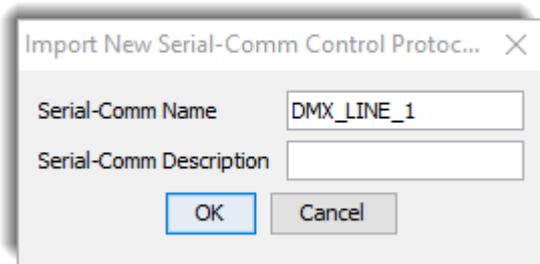
and port to 5055 (when using DMX1, 5056 when using DMX2....):



21.2. Import the DMX protocol:



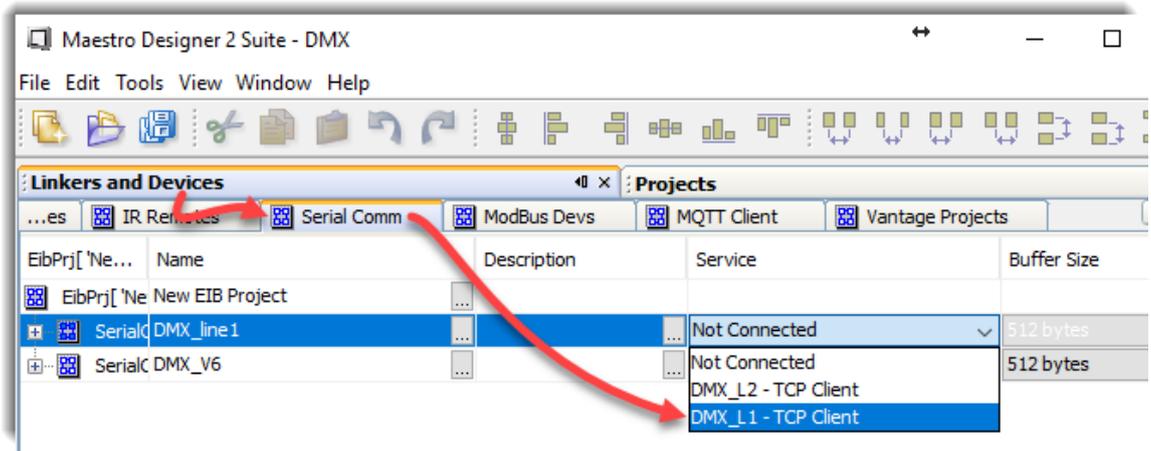
Name the protocol



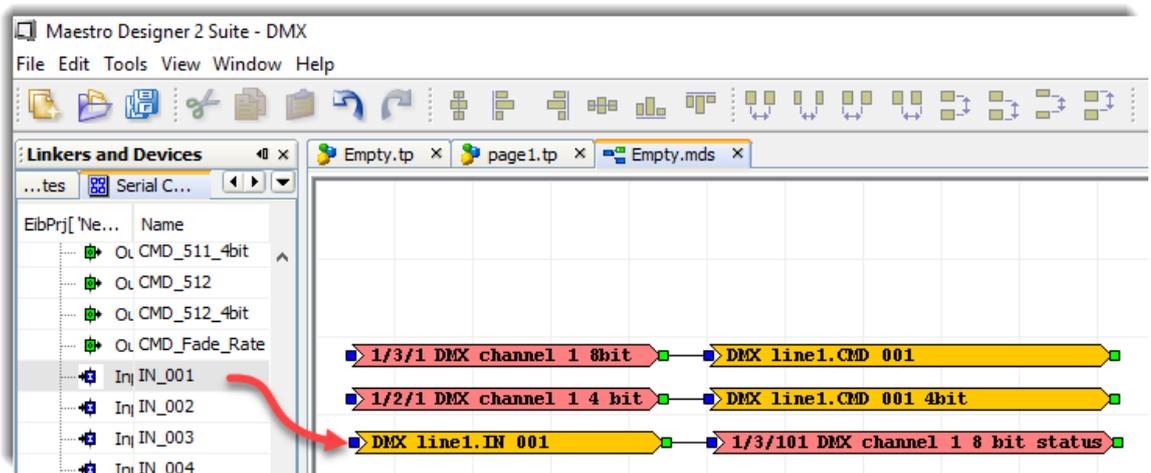
Select the DMX .xsproto file, on this tutorial we will use the file named:

DMX_512_FTDI_V6.xsproto.

21.3. Link the Serial Protocol to the External Device:

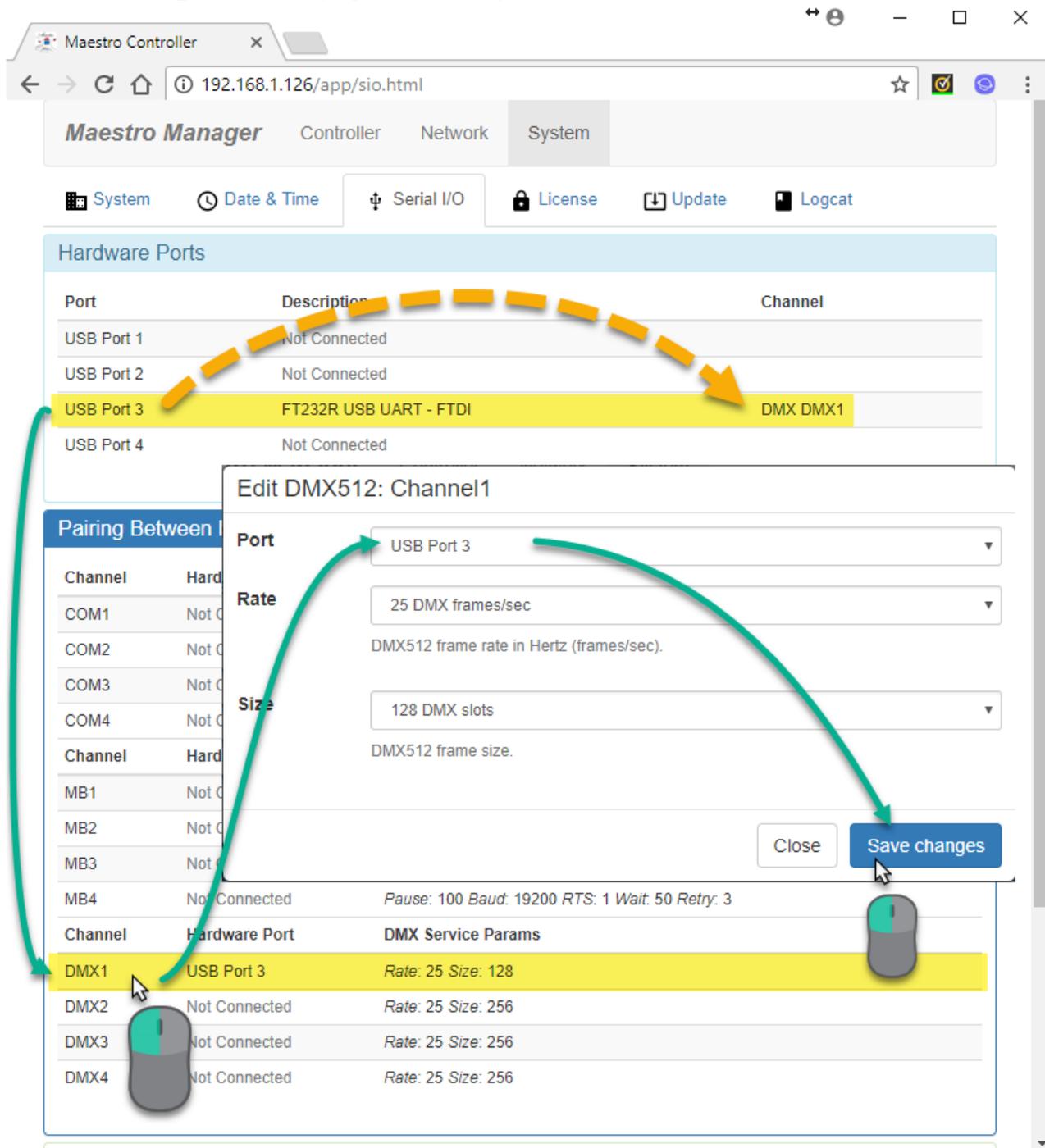


21.4. Now you are ready to use the DMX commands on you Maestro Designer project – just drag and drop them to buttons, load table and function blocks controller. To use it as KNX gateway, open a controller sheet and link Group Addresses directly to DMX commands:



21.5. Download the project to the Maestro.

- 21.6. Connect the USB to KNX dongle to Maestro and, using a browser, Open Maestro-Manager and pair between the logical channel set on Maestro Designer to the physical USB port where the DMX is connected:



The screenshot shows the Maestro Manager web interface in a browser window. The address bar shows the URL `192.168.1.126/app/sio.html`. The interface has tabs for **Maestro Manager**, **Controller**, **Network**, and **System**. Under the **System** tab, there are sub-tabs for **System**, **Date & Time**, **Serial I/O**, **License**, **Update**, and **Logcat**.

The **Hardware Ports** section contains a table with the following data:

Port	Description	Channel
USB Port 1	Not Connected	
USB Port 2	Not Connected	
USB Port 3	FT232R USB UART - FTDI	DMX DMX1
USB Port 4	Not Connected	

An orange dashed arrow points from the **USB Port 3** row to the **DMX DMX1** column. A green arrow points from the **DMX DMX1** column to the **Edit DMX512: Channel1** dialog box.

The **Edit DMX512: Channel1** dialog box has the following fields:

- Port:** USB Port 3
- Rate:** 25 DMX frames/sec
- Size:** 128 DMX slots

Buttons for **Close** and **Save changes** are visible. A mouse cursor is shown clicking the **Save changes** button.

Below the dialog box, there is a table for **DMX Service Params**:

Channel	Hardware Port	DMX Service Params
DMX1	USB Port 3	Rate: 25 Size: 128
DMX2	Not Connected	Rate: 25 Size: 256
DMX3	Not Connected	Rate: 25 Size: 256
DMX4	Not Connected	Rate: 25 Size: 256

A green arrow points from the **DMX1** row in this table back to the **USB Port 3** row in the **Hardware Ports** table. A mouse cursor is shown hovering over the **DMX1** row.

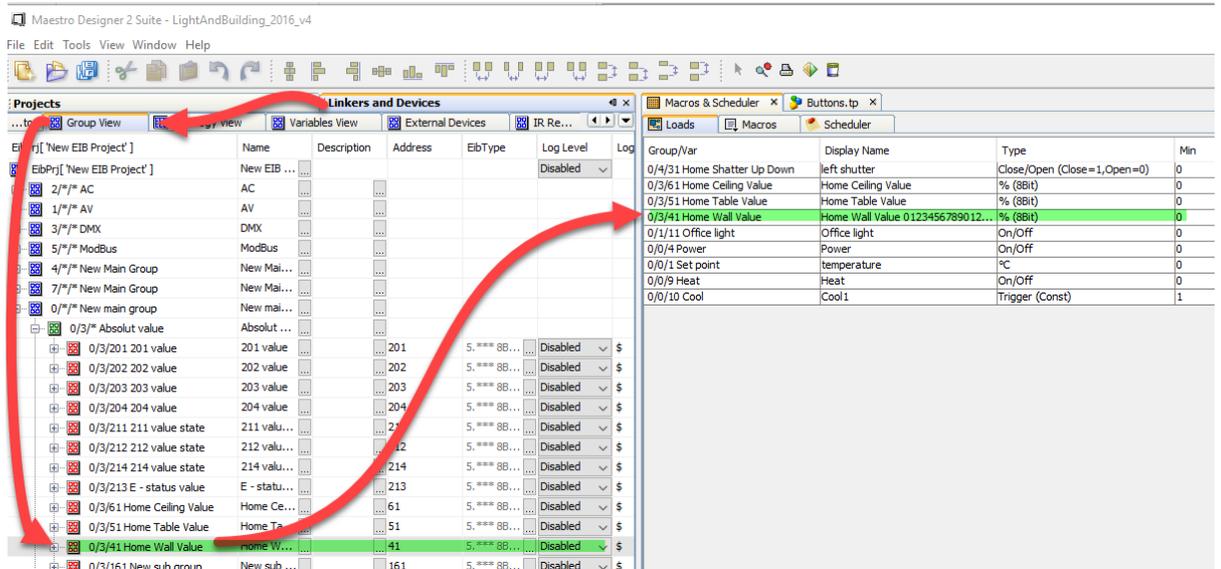
in addition, this interface enables setting how many times per second the Maestro will send the DMX frame and the last DMX channel on the frame (/the length of the frame).

22. Amazon echo (Alexa) integration with Maestro Server

Following are the instruction for the integration of Amazon Echo (Alexa) with Maestro Server.

If the loads are already inserted to the loads table (used for Macro, schedules, graphs....) you can start with step 4.

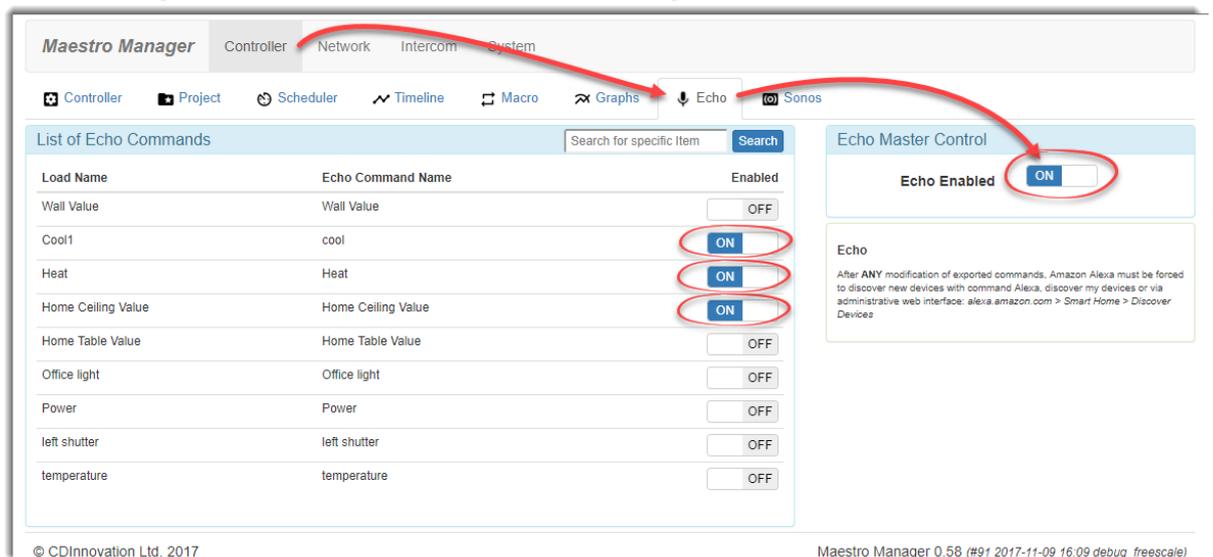
22.1. Using Maestro Designer Drag and drop the commands the loads table:



Set the correct data "Type" and edit the "Display Name" to be used later with Alexa:

22.2. Download the project to the Maestro

22.3. Using a browser, open Maestro Manager – Controller – Echo



enable Echo

- enable all loads you want to use with Echo (Alexa)

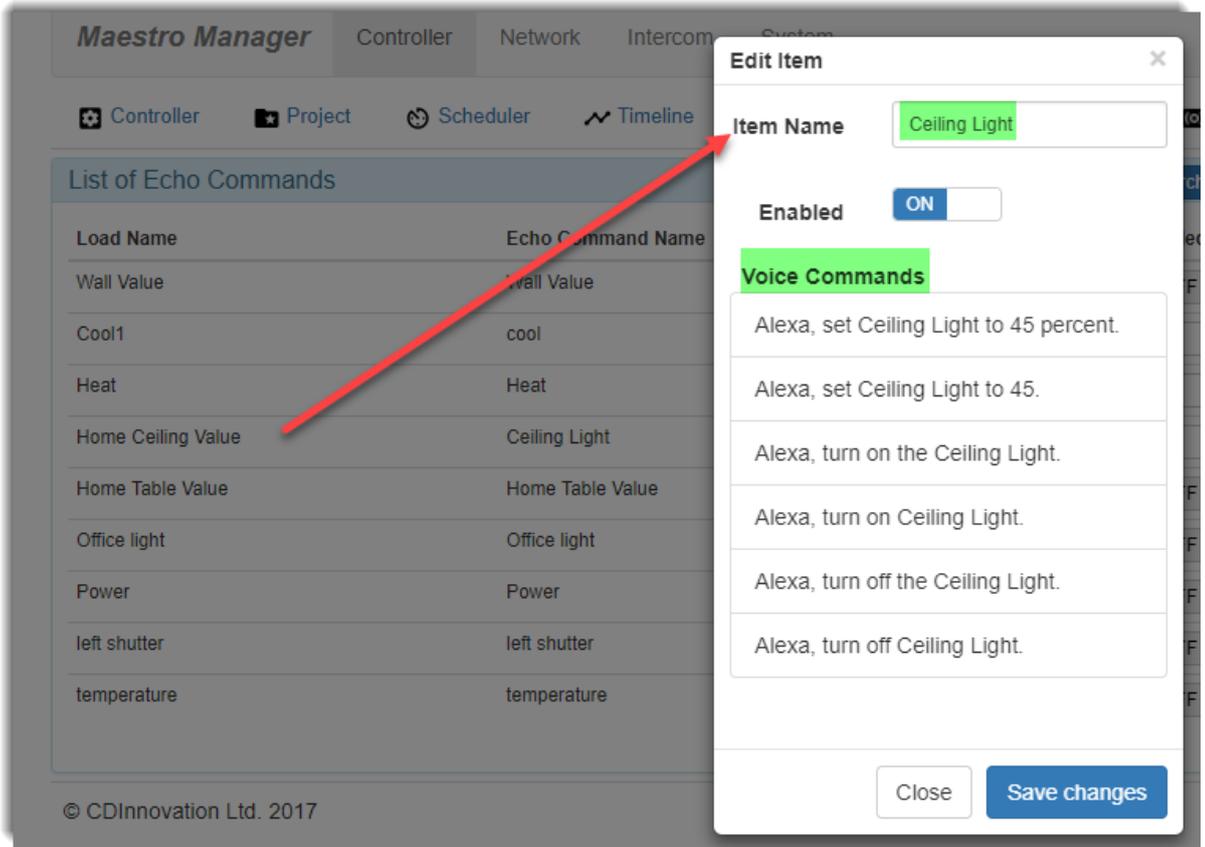
Maestro Designer user manual V1.5

www.cdinnovation.com,

info@cdinnovation.com

please notice: the maximum number of command supported is 29 while some commands such as shutter control are using 2 commands so the total number of loads in this case will be less than 29.

- 22.4. Now you can click on the loads line and:
- edit the text used by Echo (Alexa)
 - see the exact voice commands you must use



you can Add a Page Flip button, to your Maestro Designer graphic pages, pointing to this page thus letting the end user selects the loads he wants to control and the text uses for it.

- 22.5. Use the following voice command with your Echo (Alexa) device: "Alexa discover my devices". After few seconds it should discover Maestro will be ready to use the voice commands. Any time changes are made on Maestro's Echo page you have to repeat this voice command so the changes will take effect.

23. Frequently Asked Questions

Question:

KNXnetIP does not work:

Answer:

Make sure KNXnetIP is enabled on the setup page, then disable all software fire walls (such as windows firewall on your PC), restart ETS set NAT mode, restart Maestro and try again.

Question (MS3/MTS3 and older models only):

With LAN, web control runs fine but remotely, from the internet, web control is slow:

Answer:

This is normal, Web control from the internet can take about 2-3 minutes to start. To reduce this time use the fastest possible internet upload rate available at the location where the Maestro is installed and reduce the size of your images files.

Question:

Analog button is not transmitting/receiving the correct values.

Answer:

Make sure the Functional and the Graphical Min and Max values are set correctly.

Question:

I don't know the device's IP and I don't have a Page Flip Button to setup page and

Answer:

Using your finger or a USB mouse, tap within less than 5 seconds, on 4 different corners of the LCD, this will flip the page to the setup page (this does not work from web control)

Question:

Polling does not work

Answer:

Make sure the Group address you want to poll is checked on the "Virtual Links For" for window of the functional properties.

Question (MS3/MTS3 and older models only):

Approximately 1 minute after the device starts the screen turns black or the project reloads, after an additional 10 minutes the device shuts off.

Answer:

The license of your device is not valid – contact CDI.

Question (MS3/MTS3 and older models only):

After changing the time and date of the device, from the setup page, the device does not work properly

Answer:

Always restart your device after changing the time or date.

Question:

Serial communication with GC-100 does not work from run mode.

Answer:

Make sure that your PC is the only device communicating with the GC-100. The GC-100 can support only one device simultaneously. If another device is still communicating with the GC-100, (for example a Maestro or another PC) – you must turn off this device or disconnect it from the LAN while you are using run mode.

Question:

Using Windows, I have renamed one of my Graphic Pages, now the Maestro does not work properly.

Answer:

The only way to edit a project is from Maestro Designer.

It is strictly prohibited to manipulate any of the Maestro Designer project files directly from Windows!!!. Once you have edited the project directly from Windows – the project is corrupted and there is no way to fix it – you have to start a new project.

Question:

Bus monitor does not work on ETS when I am using the Maestro as my IP gateway

Answer:

You have to use Group Monitor. Maestro does not support Bus Monitor.