

01468

Domotic logic unit



Table of Contents

1. CONFIGURATION	5
1.1 Procedure	5
1.2 Starting the Editor	8
2. GENERAL OVERVIEW	11
2.1 Layout	11
2.2 Main menu	11
2.3 Status bar	14
2.4 Group address bar used	15
2.5 Toolbar	15
2.6 Details pane	16
2.7 Workspace	16
2.8 Messages area	17
3. LOGIC PROGRAMS	18
3.1 Introduction	18
3.2 Creating a new program	18
3.3 Removing or disabling a program	19
3.4 Remote management	19
3.5 Adding blocks to a program	19
3.6 Selecting one or more blocks	21
3.7 Removing one or more blocks	22
3.8 Input and output nodes	22
3.9 Connecting blocks	24
3.10 Types of nodes	26
3.11 Running order	26
3.12 Passing values between programs	28
3.13 Data types	29
3.14 Saving	29
3.15 Simulation	29
4. BY-ME	30
4.1 Introduction	30
4.2 By-me blocks	30
4.3 Lighting	34
4.4 Roller shutters	36
4.5 Climate	38
4.6 Scenarios	40
4.7 Audio	40
4.8 Energy management	40
4.9 Anti intrusion system	44
4.10 Sensors	46
4.11 KNX integration	49
5. BY-ALARM	52
5.1 Introduction	52
6. LOGIC FUNCTIONS	53
6.1 Introduction	53
6.2 Logic blocks	53
6.3 Combinational logic gates	54
6.4 Scenarios and sequences	55
6.5 Gates	57
6.6 Comparisons	63
6.7 Operations	63
6.8 Counters	65
6.9 Timers and planning	67
6.10 Variables	73
7. SIMULATION	74
7.1 Introduction	74
7.2 Types of simulation	74
7.3 Graphical simulation environment	74
7.4 Manual data input	75
7.5 Simulating sending a signal from a trigger node	75
7.6 Stopping the simulation	75
8. COMPILING	76
9. DRAWING TOOLS	79
9.1 Introduction	79
9.2 Labels	79
9.3 Rectangular areas	80
10. DEVICE MANAGEMENT	81
10.1 Introduction	81
10.2 Exporting programs and timelines	82
10.3 Updating firmware	83
10.4 Fault indication	84

Table of Contents

11. REMOTE MANAGEMENT	87
11.1 Introduction	87
11.2 State of program running	87
11.3 Calendar schedules.....	88
12. ANNEXES	89
12.1 Glossary	89
13. APPLICATION EXAMPLES	90
13.1 Activating a scenario via the anti intrusion system	90
13.2 Activating a scenario following an intrusion alarm	90
13.3 Sequential and timed irrigation with start/lock control from button	91
13.4 By-me rocker switch used for 2 separate ON/OFF functions.	93
13.5 Opening/closing shutters to preset positions	94
13.6 Switching on outdoor lights from the twilight sensor and button control.....	96
13.7 Switching on single lights at set times	97
13.8 Consumption management for activating the heat pump	98
13.9 Dehumidifier management via several humidity probes	99
13.10 Multiple activations from a single control	100
13.11 Switching off loads with delayed activation when the available energy is not sufficient to power them	102
13.12 Sequential irrigation, dependent on the clock and rain sensor, forced start-up and with duration parameters that can be changed via the user interfaces	103
13.13 Disabling of loads with delayed enabling (planned manually on the load in time bands with probable PHV production) when the available energy is not sufficient to power them	106
13.14 Parameterization by user of up and down times for actuators (e.g., staircase light function).....	107

Configuration

1. Configuration

1.1 Procedure

In the By-me system the logic unit **can be configured only alone within an Automation group**.

The device must be configured in a By-me line with the following prerequisites:

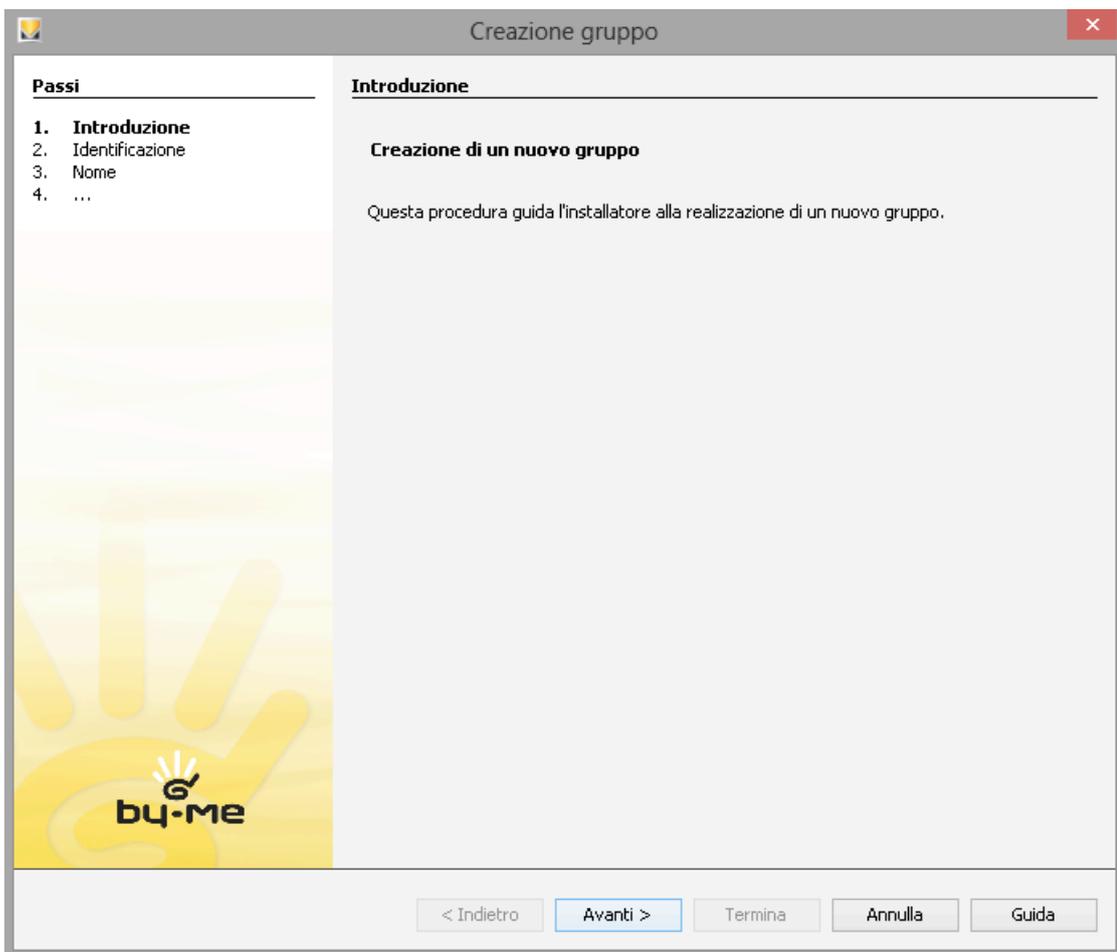
- By-me control unit art. 21509 ver. FW 4.00 or later;
- EasyTool Professional ver. 2.4 or later;
- area > 0 (no SAI line).

The entire logic configuration can instead take place only via EasyTool Professional.

To configure the logic unit, proceed as for any other Automation device (refer to chapters 6 and 7 of this manual).

To work correctly, the logic unit must not be configured in groups containing other devices. The logic unit can only be configured by automation group and without any other device.

Images of the procedure applied to the program are shown below:



✕
Creazione gruppo

Passi

1. Introduzione
2. **Identificazione**
3. Nome
4. ...

Identificazione

Inserire i dati identificativi del nuovo gruppo

Centrale

Applicazione

Indice

Descrizione

Selezionare la centrale dove memorizzare il nuovo gruppo, il campo di applicazione e l'indice. In funzione dell'applicazione scelta cambiano gli indici validi. La scelta degli indici viene fatta su quelli non ancora utilizzati nella centrale selezionata.

✕
Creazione gruppo

Passi

1. Introduzione
2. Identificazione
3. **Nome**
4. ...

Nome

Impostare i seguenti parametri per definire il nome

Gruppo 40 UNITA' LOGICA

Tipo (*)

Numero

Stanza

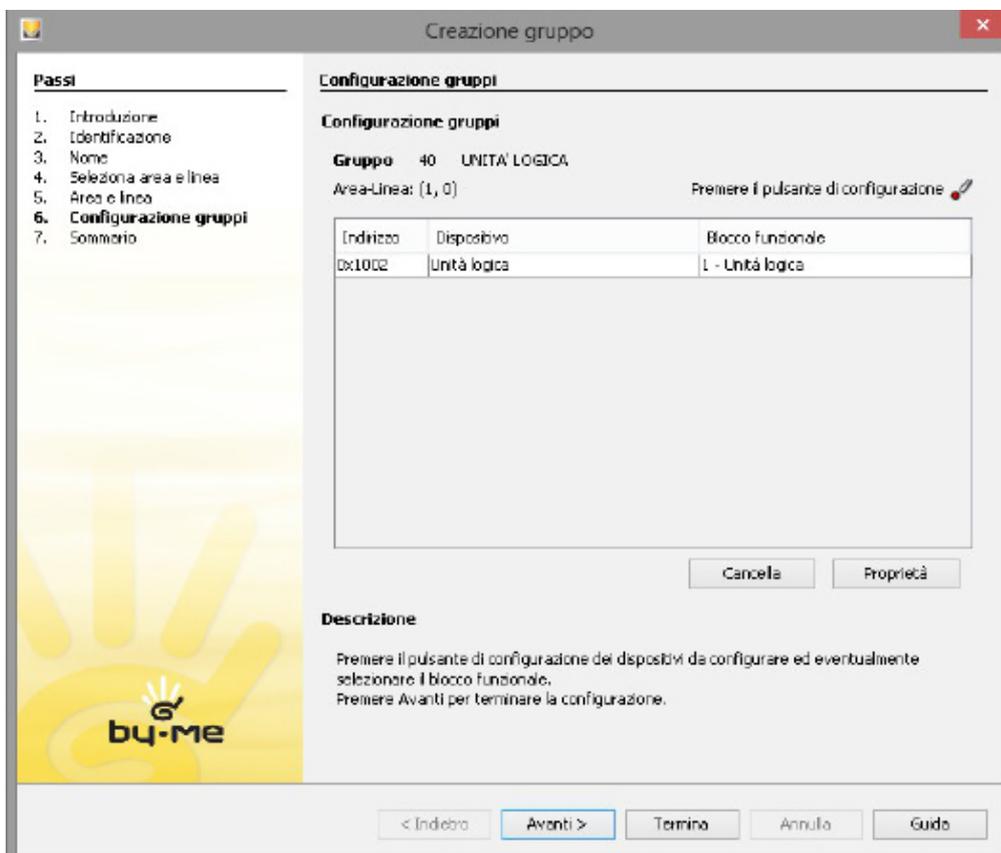
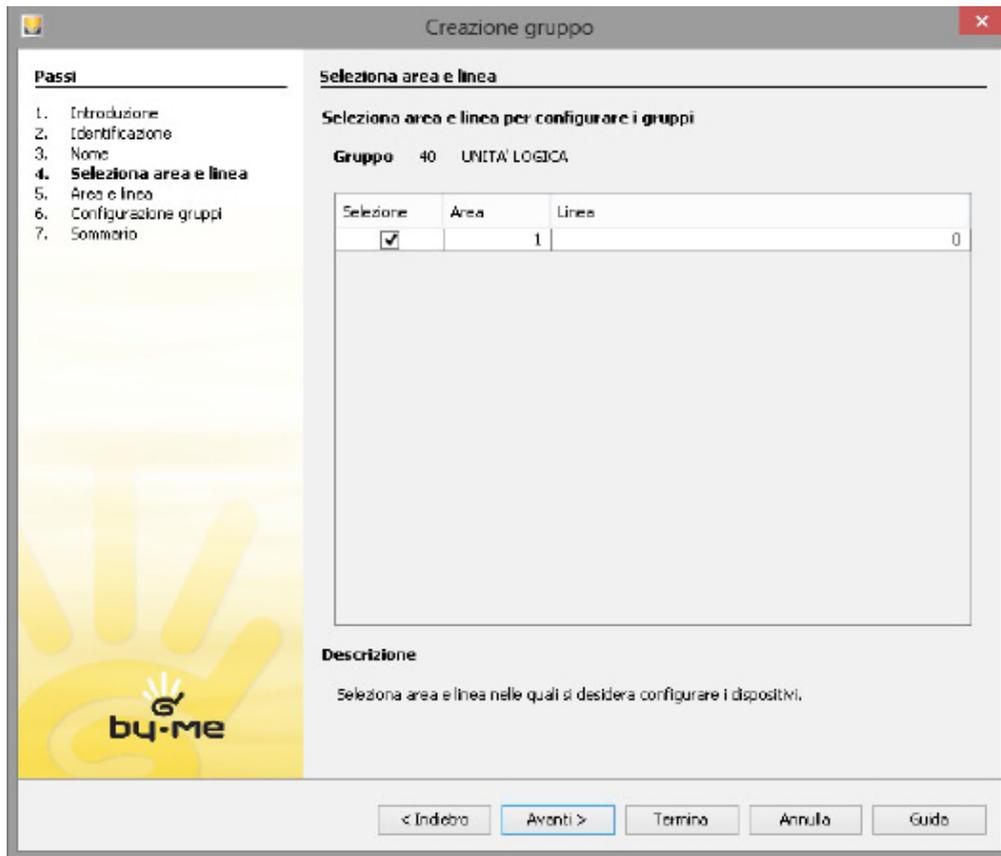
Zona

Descrizione

La composizione di tipo, numero, stanza e zona consente di definire il nome del gruppo. Le informazioni saranno visualizzate nella centrale scelta al passo precedente. (* Dato obbligatorio)

Configuration

When indicated, press the configuration button on the device and wait until the end of the configuration:



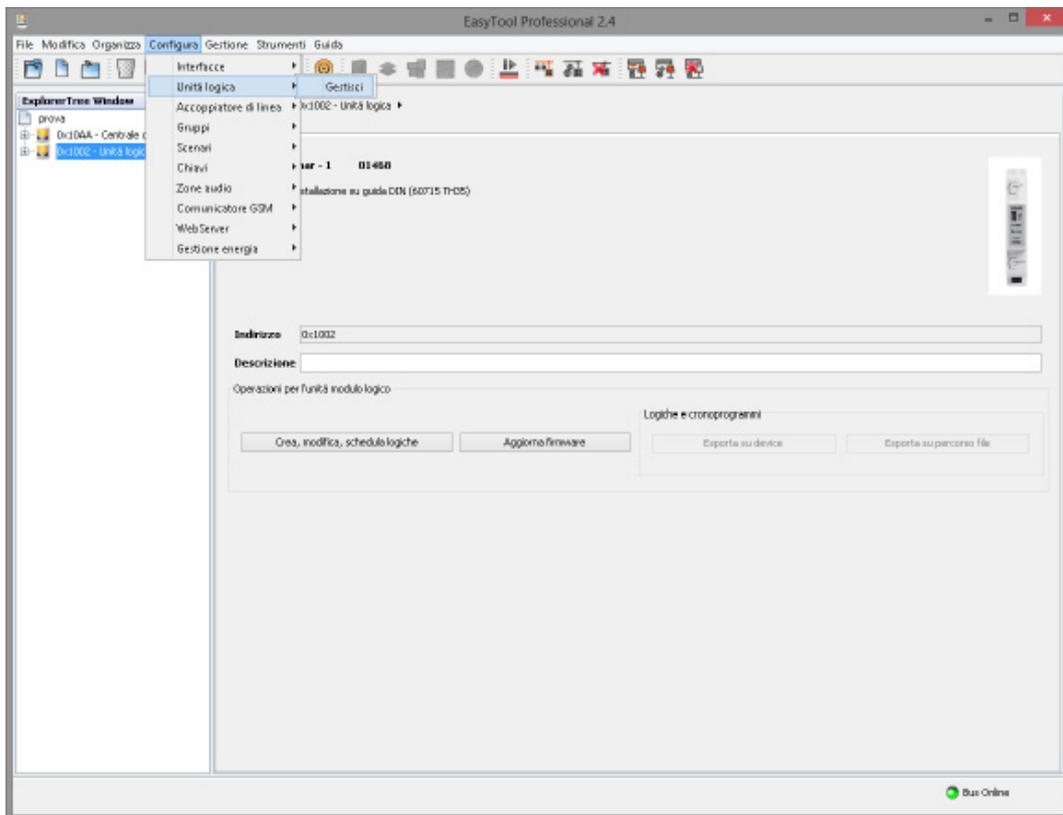
The logic unit is thus configured on the By-me side. The device then has its own physical address on the system.

Configuration

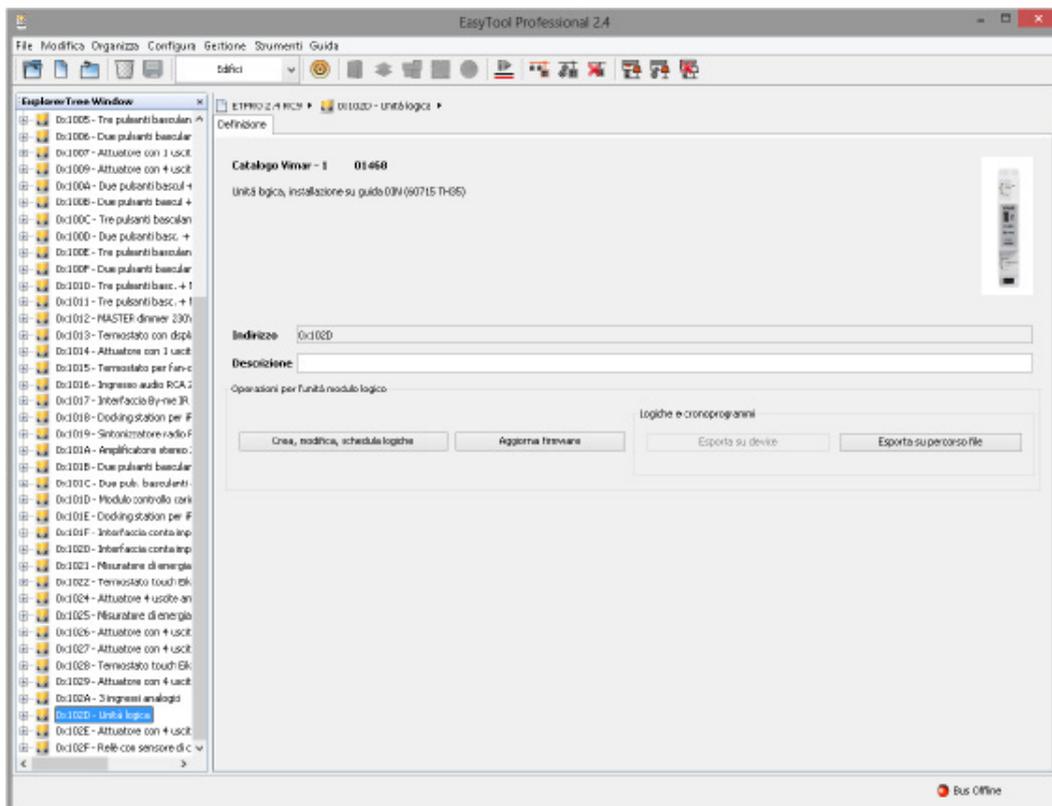
1.2 Starting the Editor

The device logic editor can be started from:

- EASYTOOL PROFESSIONAL menu item **"Menu → Configure → Logic unit → Manage"**

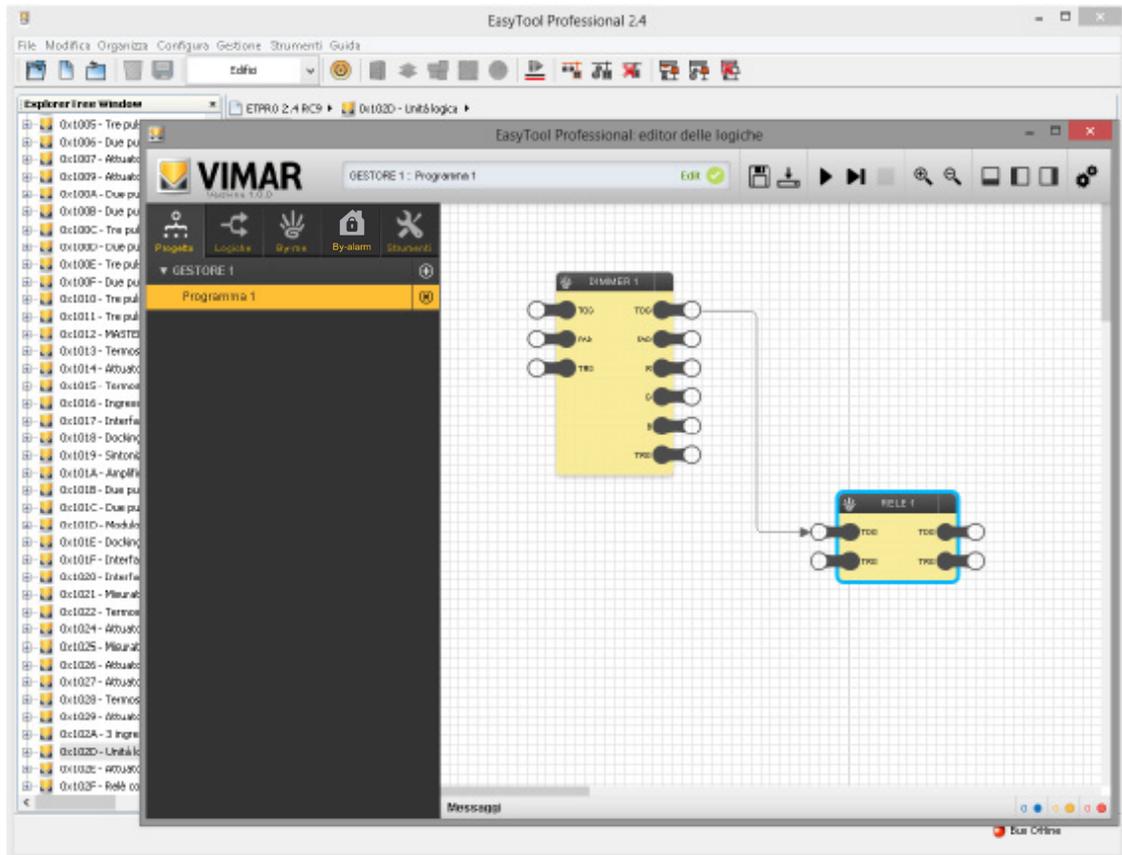


- specific page **device details**, available in the "Buildings" tree view. On the details page there is a **"Create, edit, schedule logic elements"** button:



Configuration

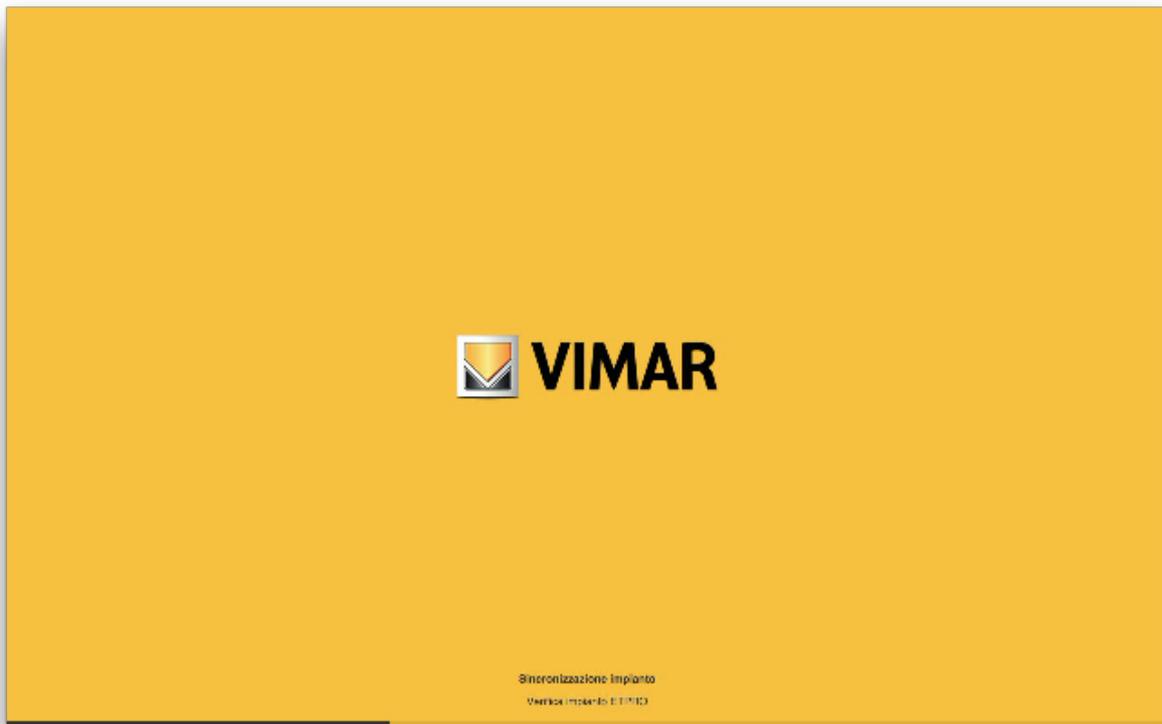
The editor must be constantly aligned with the EASYTOOL PROFESSIONAL project and is thus opened in a modal way above the main window.



To make any configuration changes to the project-system you need to close the editor window, make your changes and reopen the editor.

Configuration

On first opening the editor (within the same EASYTOOL PROFESSIONAL project), a synchronization procedure is started up with EASYTOOL PROFESSIONAL, which prepares a new project in the editor and downloads the configuration of the By-me project:



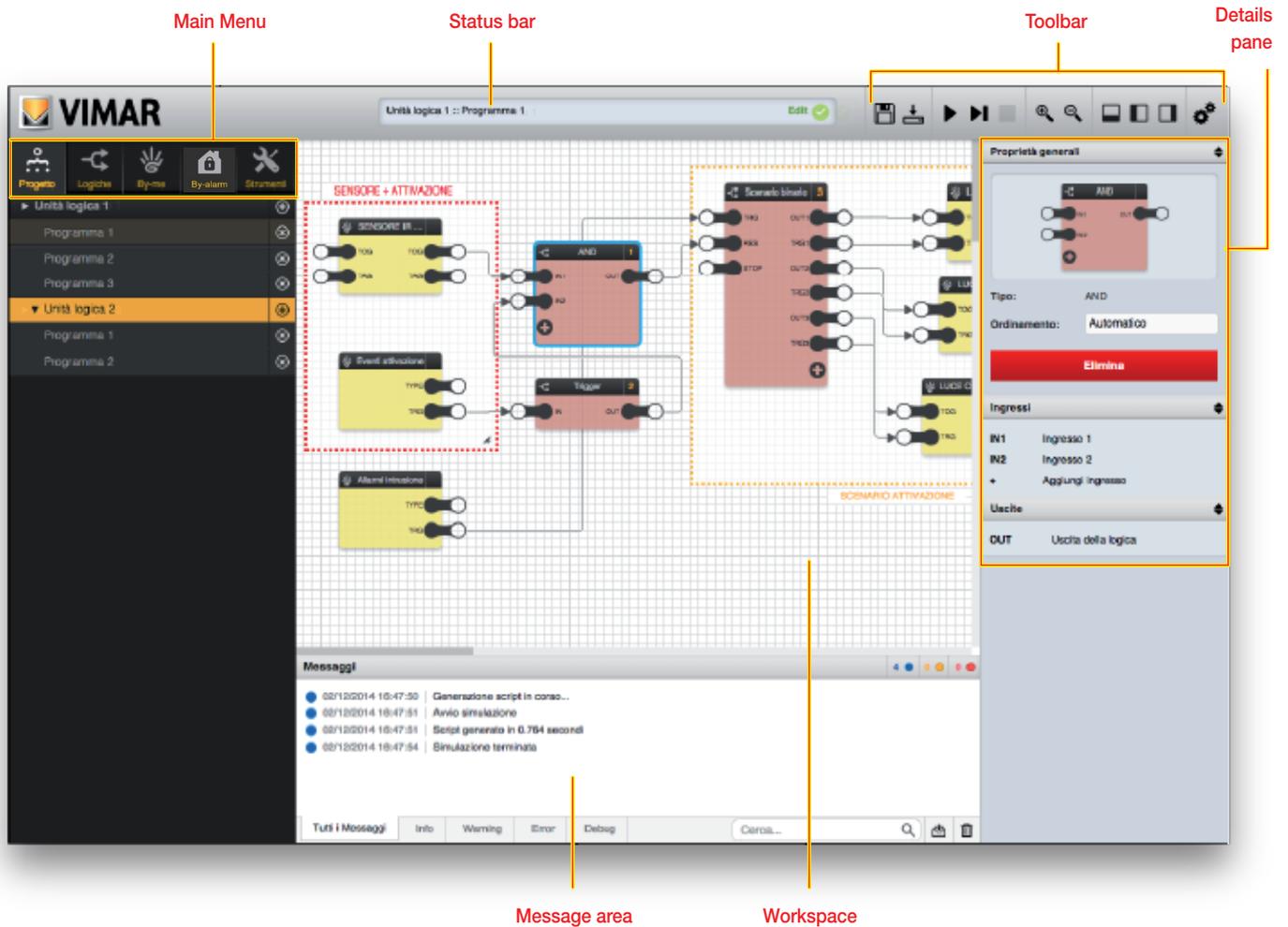
On subsequently opening the editor (within the same EASYTOOL PROFESSIONAL project), conversely, only a synchronization check is run with the By-me project; when changes are made to the project used previously, the synchronization is called up again. This operation can take up to several minutes, depending on the size and complexity of the EASYTOOL PROFESSIONAL project and the computer resources.

General overview

2. General overview

2.1 Layout

The following figure shows the structure of the editor's graphic interface, after opening its window:



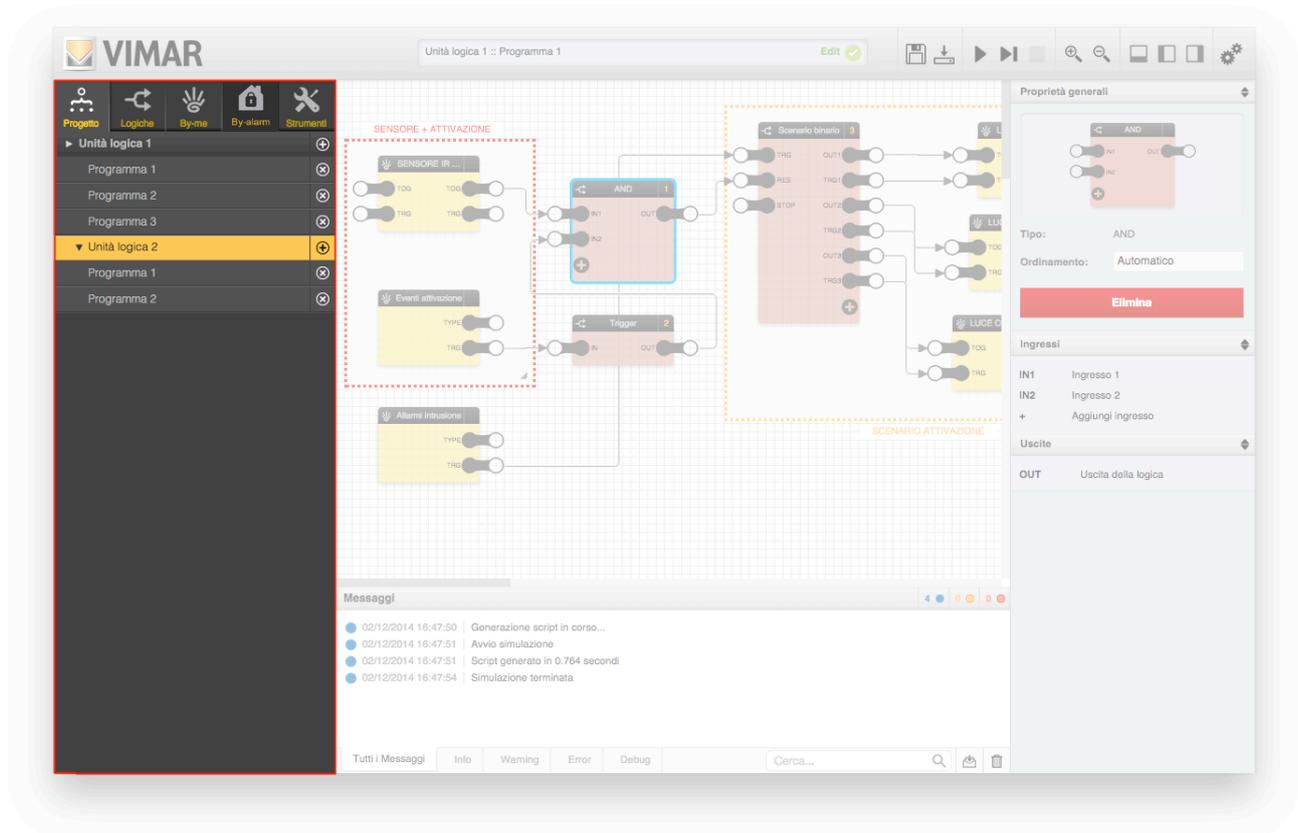
2.2 Main menu

The menu provides all the tools for creating and managing logic programs. The "tabs" at the start of the menu let you access the main sections of the menu.

General overview

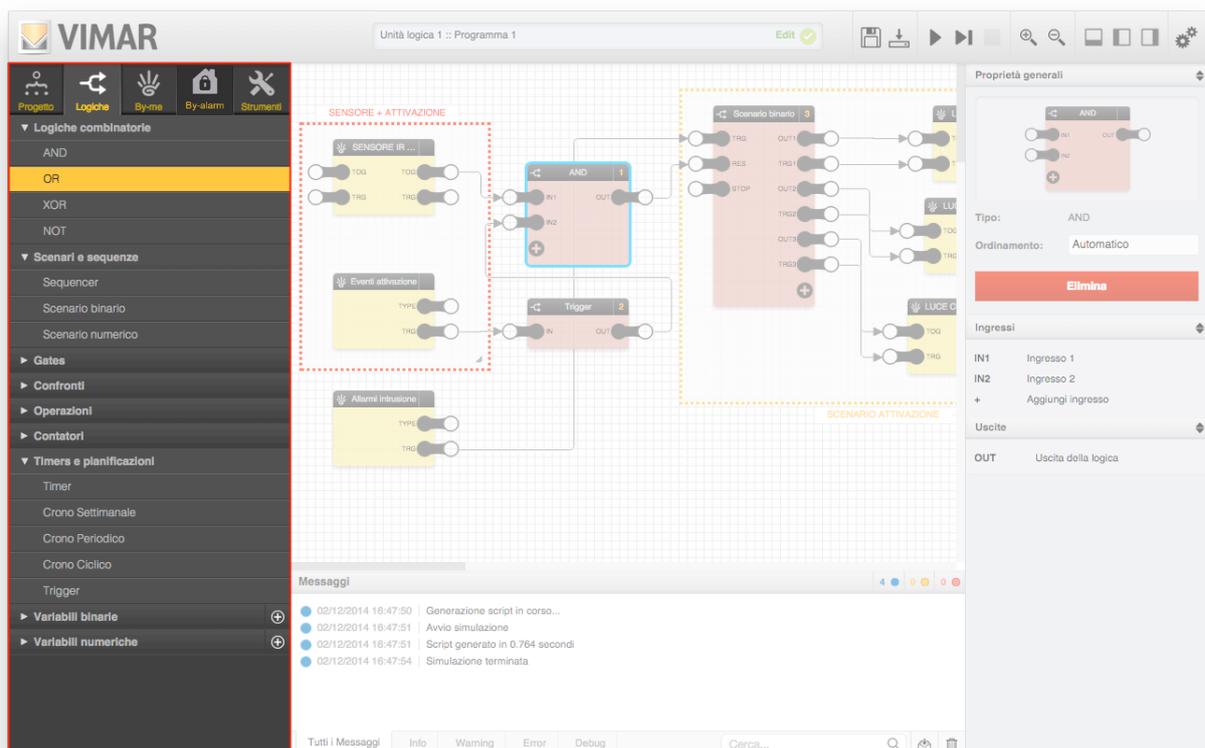
2.2.1 Project

This section contains a list of the logic units configured in EASYTOOL PROFESSIONAL; for each of them it is possible to create up to 64 logic programs. This section of the menu lets you create, modify, and delete logic programs.



2.2.2 Logics

This section contains the library of logic blocks that can be inserted in the programs. The items of the logic library can be inserted in the programs by dragging and dropping.



General overview

2.2.3 By-me

This section contains a list of all the By-me groups and scenarios in the project, subdivided by type. In this case, too, the By-me blocks can be dragged from this section into the programs to make them interact with the logic elements.

2.2.4 By-alarm

This section contains a list of all the By-alarm system areas regardless of whether they are present in the system. In this case, too, the By-alarm blocks can be dragged from this section into the programs to make them interact with the logic elements.

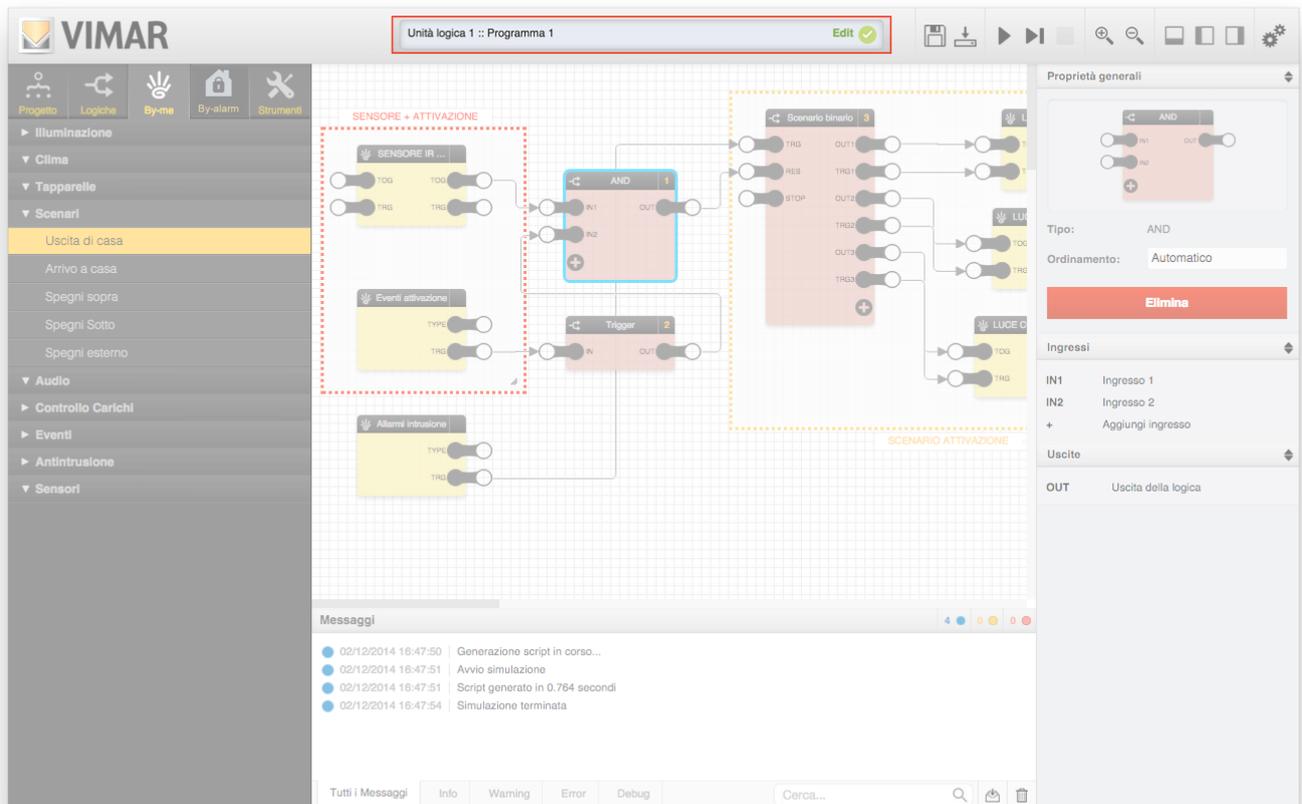
General overview

2.2.5 Instruments

This section lets you insert graphic support elements in logic programs, which can be used to include explanatory comments, notes or boxes with functional groupings, etc.

2.3 Status bar

This section of the graphical user interface shows the logic unit and the currently selected program, and highlights the current work mode (editing or simulation) as well as any error messages.



The screenshot displays the VIMAR Logic Unit software interface. At the top, a status bar shows "Unità logica 1 :: Programma 1" and an "Edit" button with a green checkmark. The main workspace is a grid-based editor for a logic program. On the left, a sidebar lists various modules: Illuminazione, Clima, Tapparelle, Scenari (highlighted), Audio, Controllo Carichi, Eventi, Antintrusione, and Sensori. The "Scenari" module is expanded, showing "Uscita di casa", "Arrivo a casa", "Spegni sopra", "Spegni Sotto", and "Spegni esterno". The logic program itself consists of several interconnected components: a "SENSORE - ATTIVAZIONE" block (highlighted with a red dashed box) containing "SENSORE IR ..." and "Eventi attivazione"; an "AND" logic gate; a "Trigger" block; and a "Scenario binario" block. The "Scenario binario" block has multiple outputs (OUT1, OUT2, OUT3, OUT4) connected to various control elements like "LUDE C" and "LUDE D". A "Proprietà generali" panel on the right shows the selected "AND" gate's properties: Tipo: AND, Ordinamento: Automatico, and an "Elimina" button. Below the workspace, a "Messaggi" panel displays a log of events: "02/12/2014 16:47:50 Generazione script in corso...", "02/12/2014 16:47:51 Avvio simulazione", "02/12/2014 16:47:51 Script generato in 0.764 secondi", and "02/12/2014 16:47:54 Simulazione terminata". At the bottom, a status bar shows "Tutti i Messaggi" and filters for Info, Warning, Error, and Debug, along with a search field.

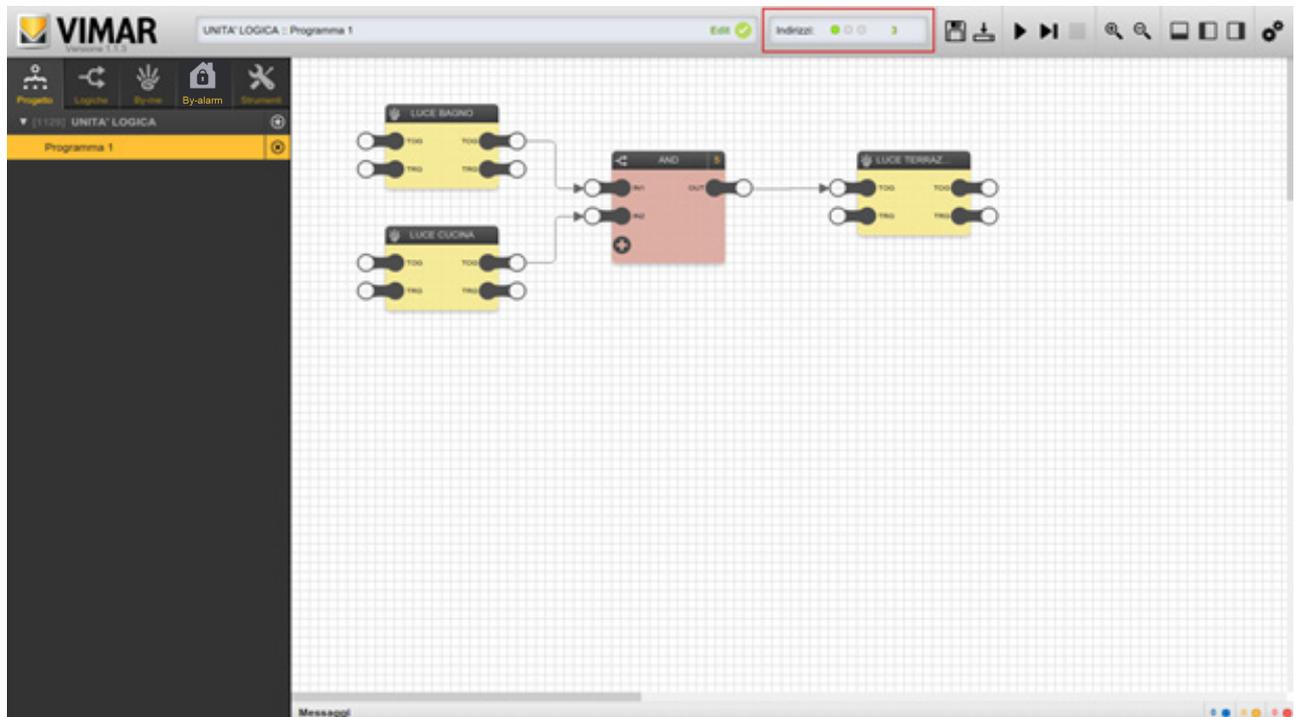
General overview

2.4 Group address bar used

This section of the graphic interface displays the group addresses used in the various programs of the selected logic unit.

Each logic unit can utilize up to a maximum of 254 group addresses.

Before compiling files for the logic unit, the calculation is done by using an approximation, the number of group addresses is incremented by one for each connected node. After compilation the group addresses actually used will be calculated.



2.5 Toolbar

The toolbar provides the following instruments, which are always available during all running phases of the logic programs (except the simulation phase, when not all of them can be used):



SAVE

Saves the configuration of the logic programs within EASYTOOL PROFESSIONAL

NOTE: Saving is performed automatically also on exiting the graphic environment of the editor.



COMPILE

Generates configuration files of the currently selected logic unit and transfers them into EASYTOOL PROFESSIONAL, for subsequent download into the device



CONTINUOUS SIMULATION

Starts simulation in real-time mode



STEP-BY-STEP SIMULATION

Starts simulation in step-by-step mode



STOP SIMULATION

Stops the current simulation



ZOOM +

Increases the zoom factor of the workspace



ZOOM -

Decreases the zoom factor of the workspace



SHOW / HIDE MESSAGES

Shows or hides the message area at the bottom



SHOW / HIDE MAIN MENU

Shows or hides the main menu on the left



SHOW / HIDE DETAILS

Shows or hides the side pane containing the details



ADVANCED OPTIONS

Used to access a drop-down menu that contains advanced options, as detailed below

General overview

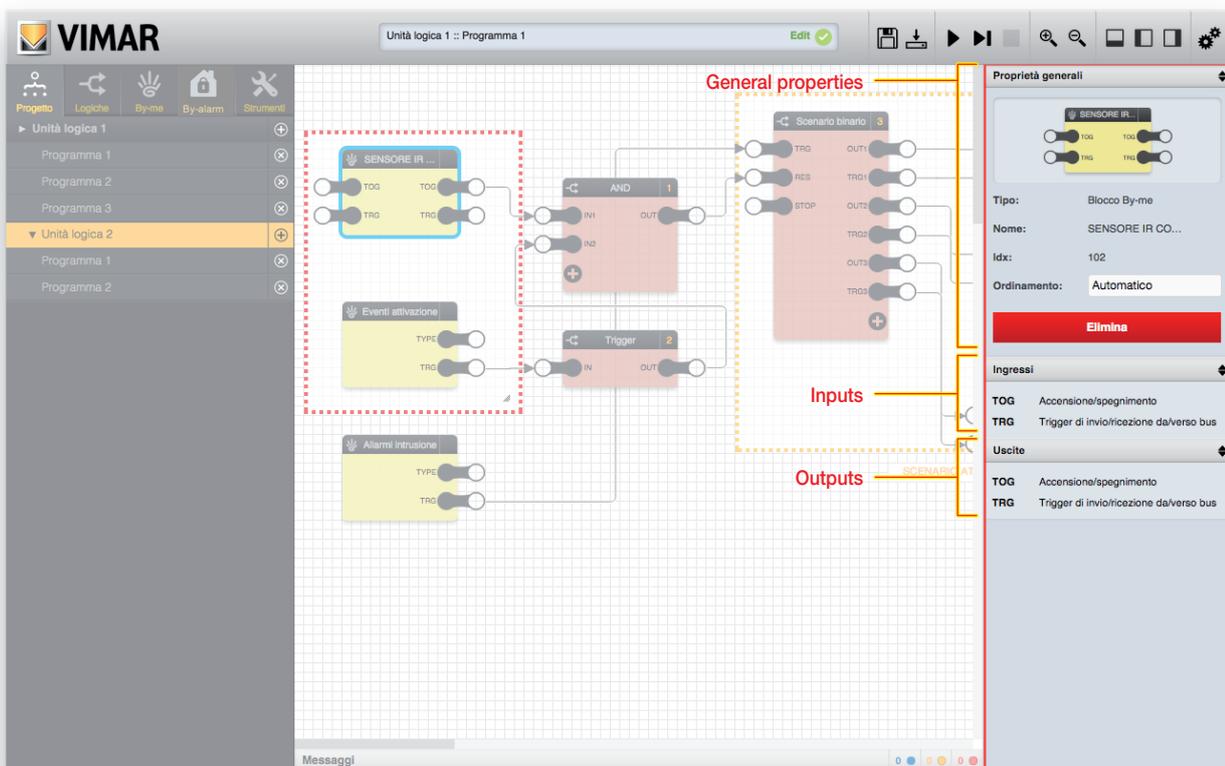
The ADVANCED OPTIONS drop-down menu provides the following items:

It re-arranges blocks automatically	It regenerates the block sorting in the open program, according to a left-to-right and top-to-bottom positioning criterion. This operation does not overwrite any manually forced sorting on the same blocks
It reloads the editor	It forces the editor graphic interface to be redrawn, an operation required in some cases when the blocks and connection lines are misaligned
Date/time settings	Sets the date/time of the Astronomical Clock block during the simulation
Save to/Load from PC	This is used to save the logic programs on your PC for later reloading and restoring the saved state. Caution: Only logic programs created by starting from the same identical EasyTool Professional project can be loaded.

2.6 Details pane

This section, normally closed (can be opened by using the appropriate button in the toolbar), contains details about the selected objects in the workspace, and lets you change their properties and options.

Depending on the type of object selected, the information can be divided into multiple sections, as exemplified in the following figure:



The sections can be closed (by clicking on the title bar) to provide easier consultation of the following ones, especially in the case of objects with many details and options.

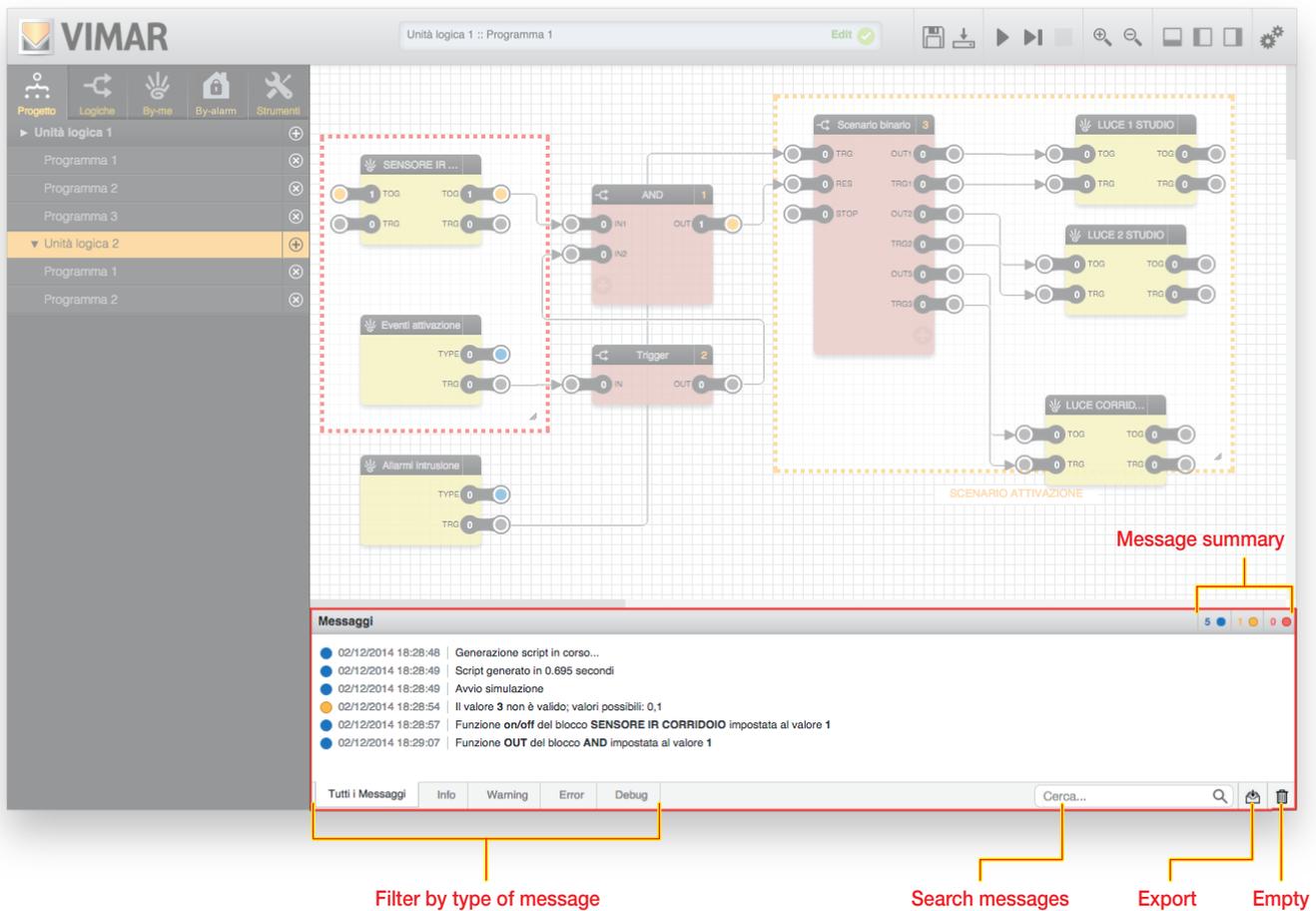
2.7 Workspace

The central part of the window is dedicated to the workspace, within which the logics are built. To expand the usable space, it is advisable to close the side panes and the message area, especially when *editing* the logic programs.

General overview

2.8 Messages area

The bottom of the window contains the messages generated by the editor during logic program creation and, especially, during simulation (as described below).



The editor-generated messages can be of different types, according to their severity and type:

- **Error:** warnings of operations or conditions that generate an error, and that typically require editing or checking by the user
- **Warning:** warnings of abnormal conditions, which moreover do not necessarily constitute an error or a situation to be changed
- **Info:** "normal" information messages, which include operations performed by the editor worth signalling to the user
- **Debug:** messages of details of operations carried out by simulation (only available in step-by-step mode, as described below)

The different types are distinguished by a colour, shown beside each message together with the date/time when the message was generated. The title bar of the message area contains a summary, on the right, of the number of messages of the different types, also visible when the message area is closed.

At the bottom of the message area there are the following commands:

- **Filter by message type:** select one of the available items to filter the messages on screen according to the corresponding type
- **Find messages:** used to filter messages based on one or more keywords
- **Export:** used to export the message log (including messages related to previous work sessions) in CSV format, which can also be read with external software (e.g. spreadsheets)
- **Empty:** used to delete video messages (the messages still remain filed in the editor and can be exported via the appropriate button for reading off-line)

3. Logic programs

3.1 Introduction

The logic units are designed to make one or more logic networks, called "programs", which typically receive information from the By-me bus, process it via logic blocks, and send the results in the form of commands over the bus.

A "Logic Program" may contain different logics or "functionalities". Theoretically all the functions required for the Logic Unit may be contained in the same Program; however, dividing the overall logic into several programs offers a series of advantages:

- From touch screen or Web Server, the Logic Programs can be enabled or disabled (if you associate a function to a Program you can then enable or pause that function remotely);
- Maintenance and any modifications after the first draft are made easier (more order).

It should be noted however that attention must be paid to potential interactions and overlaying between different Programs (e.g. use of the same By-me resources in more than one program with possible conflicts such as an actuator controlled by more than one logic program).

Each logic unit can contain up to a maximum of 64 programs.

- Each logic unit can manage up to a maximum of 254 group addresses. You must check the limit before downloading the programs on the logic unit.
If the limit is exceeded, once the program has been compiled a message error is displayed in the debug area and in the Edit box at the top where the program name appears.

The editor lets you configure the logic programs, connecting By-me blocks and logic functions using *drag&drop* and simple graphic tools, without any particular programming skills. As will be shown later, the editor also enables simulating the behaviour of logic programs, before "downloading" the programming into the logic unit itself.

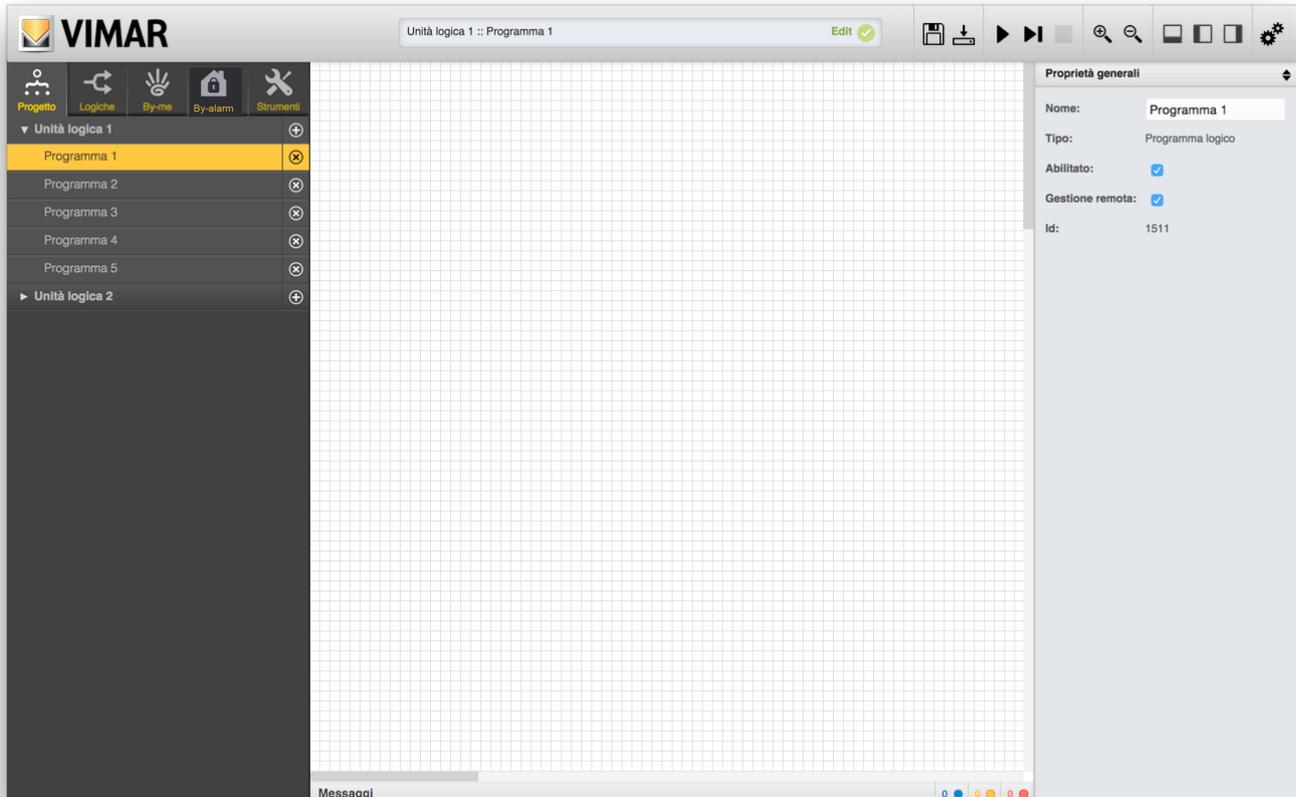
3.2 Creating a new program

To create a new program, first select the logic unit on which you want to work (if the ETPRO project contemplates more than one) in the "PROJECT" section of the main menu, then press the corresponding button "+": a new empty program is created called "Program 1".

NOTE: If the "PROJECT" item in the main menu contains no logic units, make sure you have entered at least one in the EASYTOOL PROFESSIONAL project, then open up the editor once again.

To open the new program, simply click on it: the workspace shows an empty grid, on which you can begin constructing the logic, as described below.

To change the name of the program, open the details pane and enter the new name in the text box, as shown in the following figure; the name cannot contain special characters and must have a maximum length of 16 characters.

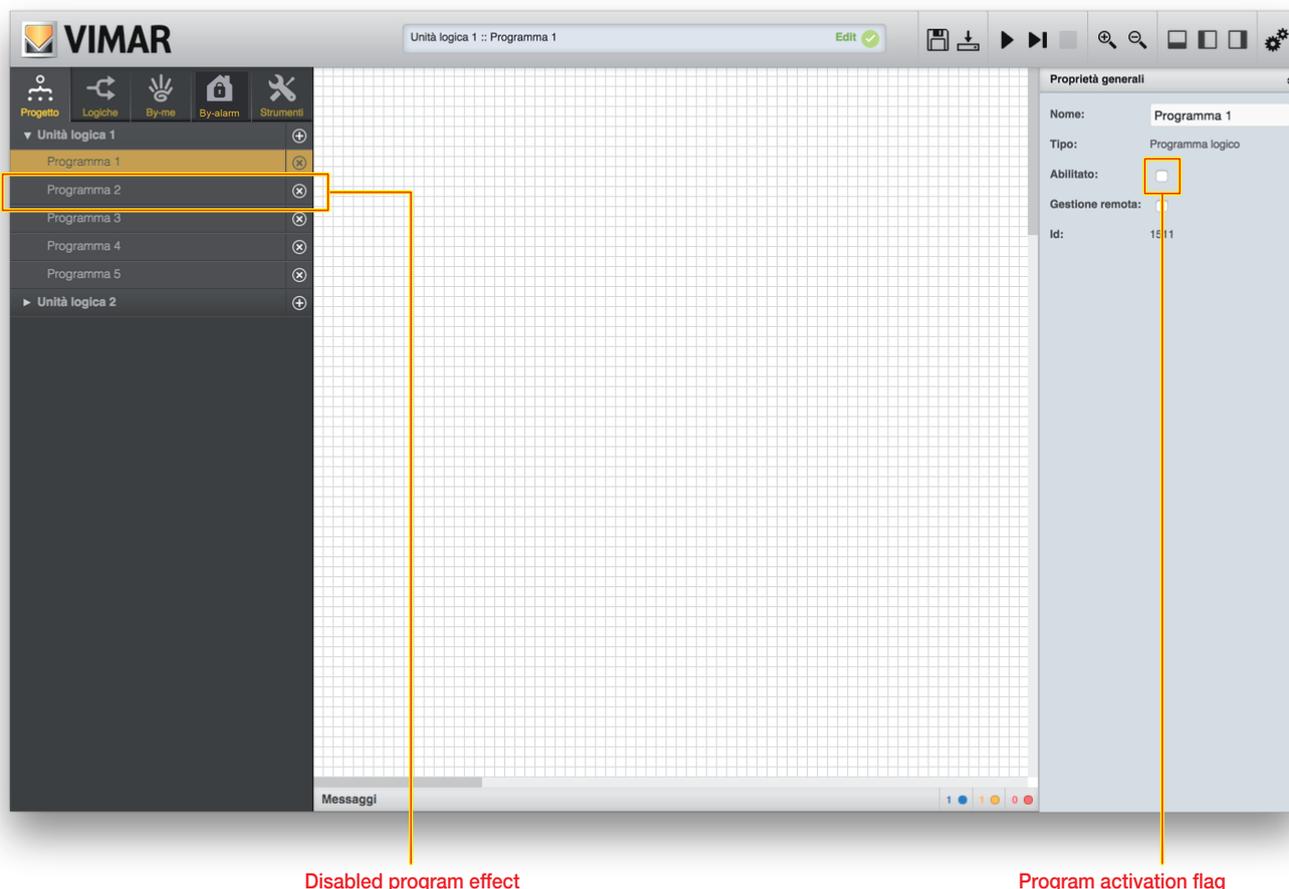


Logic programs

3.3 Removing or disabling a program

To remove an existing program, simply press the corresponding button "X" in the main menu; after confirming deletion, the program is deleted along with all the logic functions it contains. This operation cannot be undone.

If you do not want a program to be included in the logic unit, for example because it is still incomplete, you can disable it by unchecking the corresponding "ENABLED" item in the details pane; disabled programs are shown in the main menu with a semi-transparent effect.



3.4 Remote management

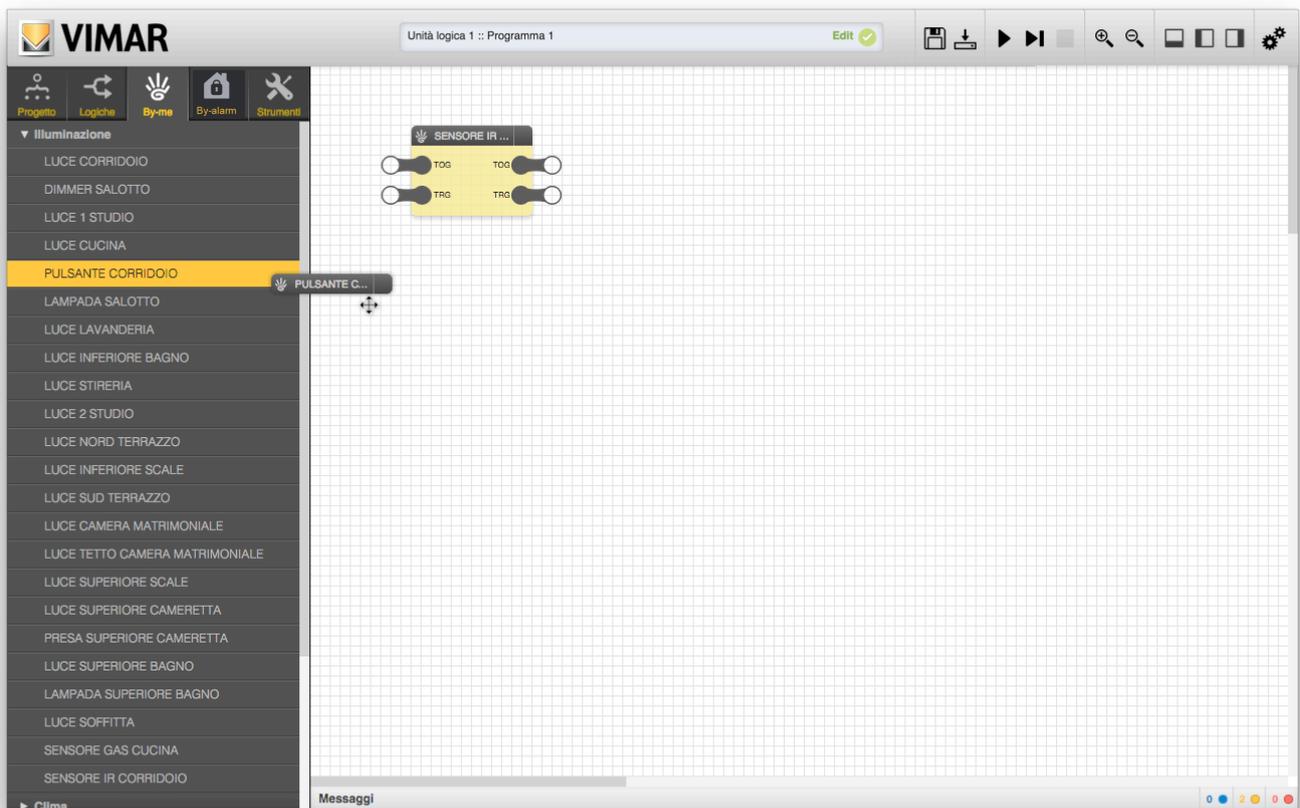
As explained below, the logic programs can be managed by the end user via the web server or touch screens; if you do not want this to be possible (for example because the program must not be disabled, or contains timer programming which must not be modified by the user) you can uncheck the "remote management" box. For more information on remote management of logic units, refer to chapter 10.

3.5 Adding blocks to a program

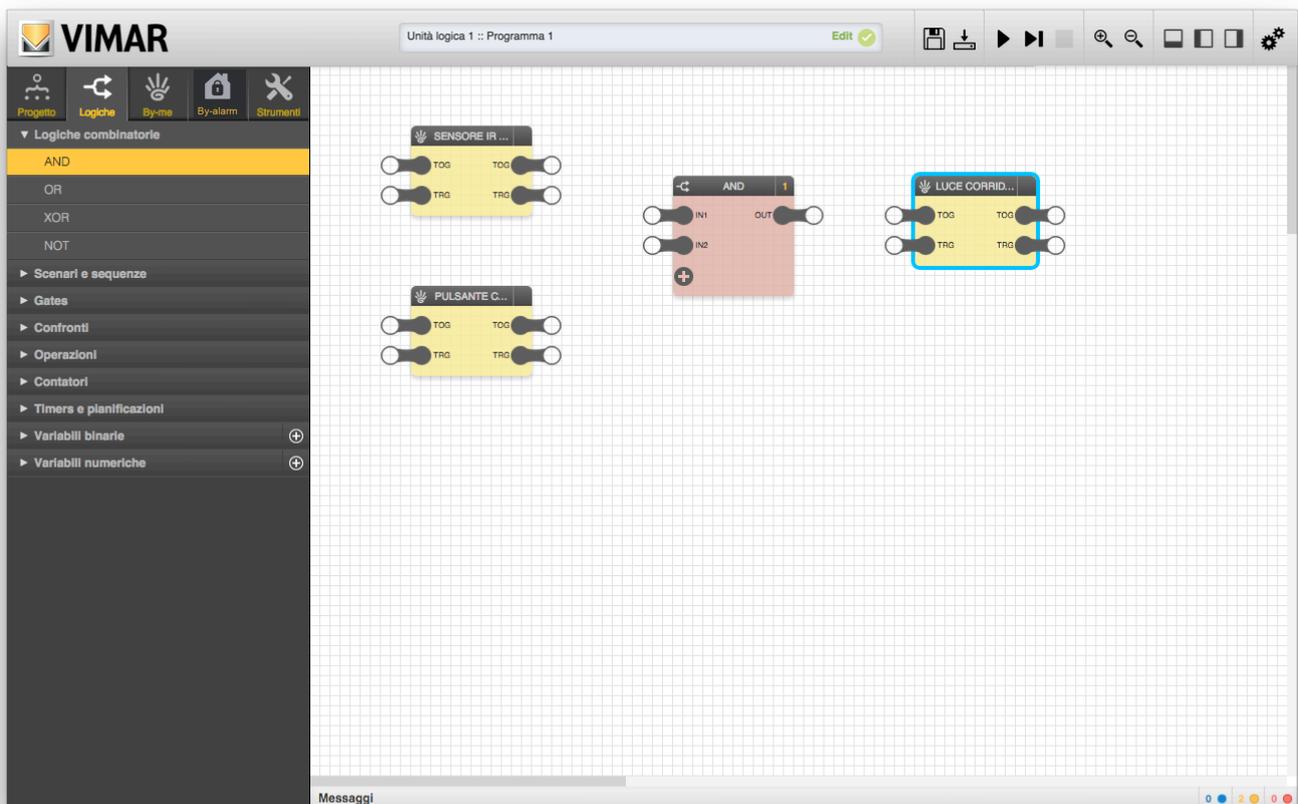
Programs provide for connecting several blocks to form a logic network. Blocks can be of the By-me or logical type; the former are required to read and/or write information on the home automation bus, the latter enable processing and combining this information.

To add a By-me block to a program you must first identify it in the relevant section of the main menu; here are listed all the supported By-me groups (for a complete list of the By-me functions supported by the logic unit, see chapter 4) in the EASYTOOL PROFESSIONAL project, divided by type.

After identifying the By-me block, simply drag it into the workspace by using *drag&drop*:



To add a logic block, similarly, you must first identify it in the "LOGIC" library, which is also organized by type (for a complete list of the available logic blocks, see chapter 5), and then drag it into the workspace:

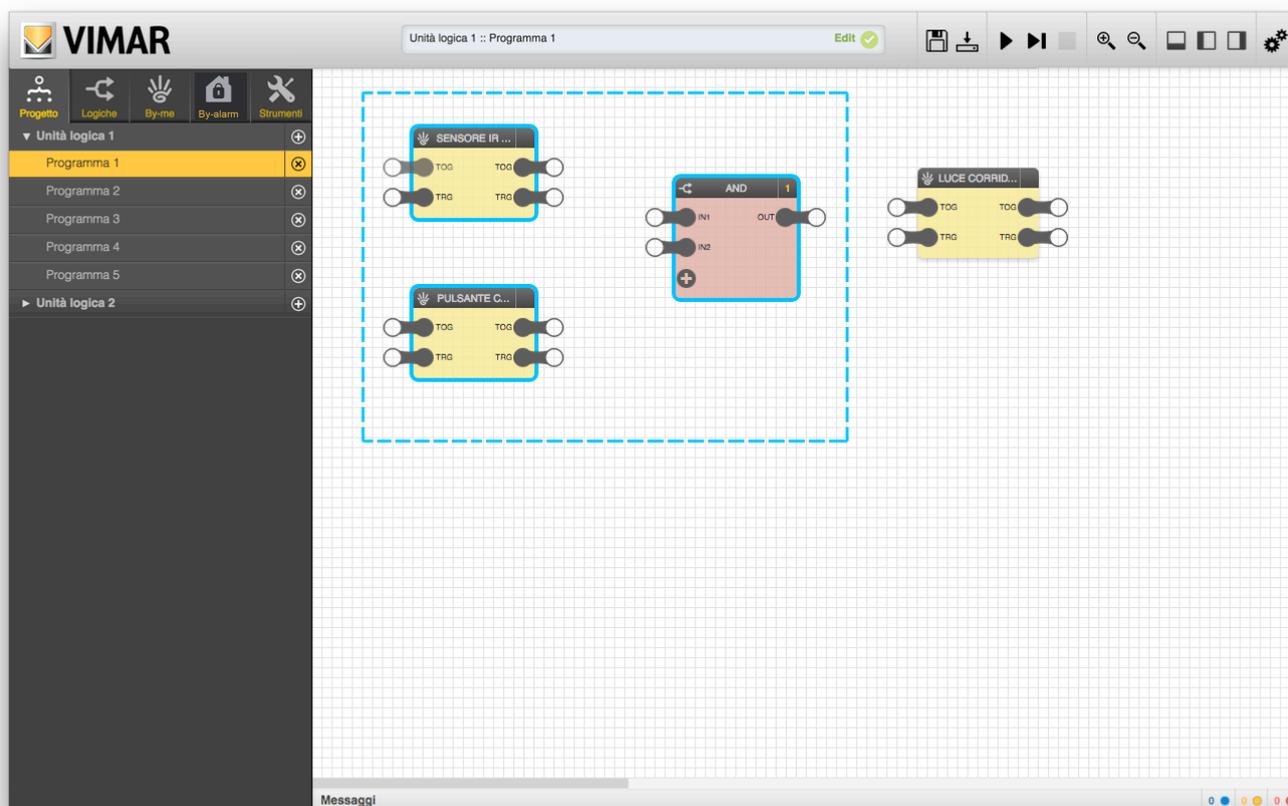


3.6 Selecting one or more blocks

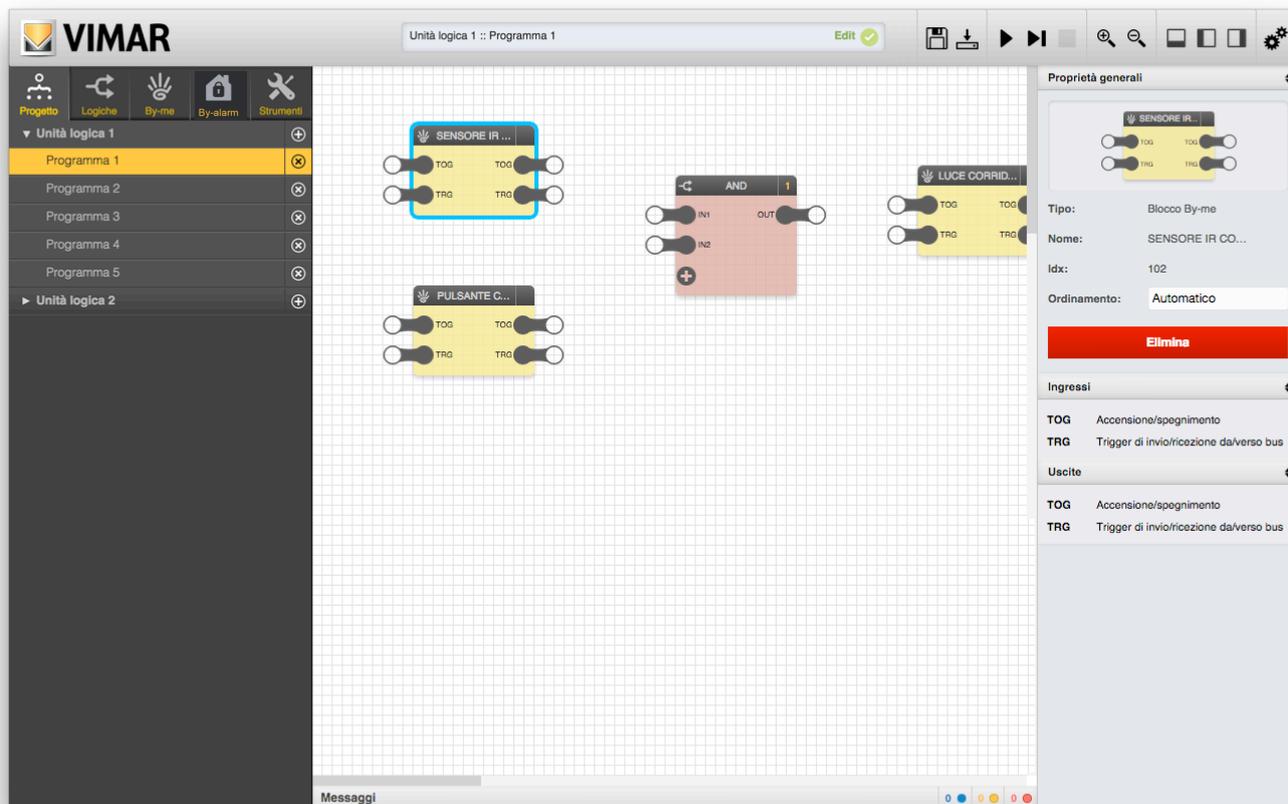
You can select one or more blocks within a program in several ways:

- Clicking on the "title" of the block (single selection)
- Clicking on the "title" of multiple blocks, while holding down the CTRL key ("scattered" multiple selection)
- Clicking on a point in the workspace and, while holding it down, moving the cursor to draw a rectangular selection area ("contiguous" multiple selection)

The selected blocks are displayed with a light blue border:



The selected blocks can be moved within the workspace by simply using *drag&drop*. Vice versa, by selecting a single block and opening the details pane you can display its properties, the list of input and output nodes, and manage any options there may be, as detailed below for each type:



NOTE: When selecting multiple blocks simultaneously you cannot see the details, as they are different for each one of them. Note that the sorting, already explained for the logic blocks, is also present for By-me objects; this advanced configuration element is available for By-me objects, but currently its use is preset and reserved for future uses.

3.7 Removing one or more blocks

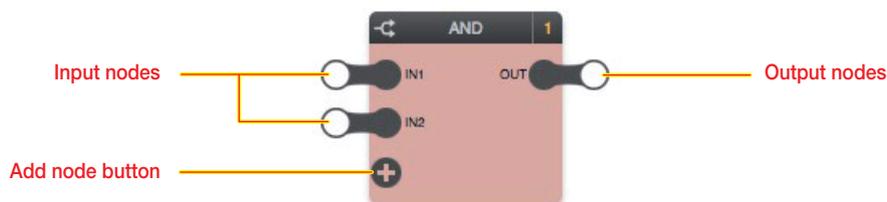
To remove one or more blocks of a program, proceed in one of the following ways:

- Select a single block, open the details pane and press the "DELETE" button
- Select one or more blocks and press the "DEL" key on the keyboard

In both cases, after a confirmation message, the selected blocks are removed from the program, as well as any connections with other blocks in the program. This operation cannot be undone.

3.8 Input and output nodes

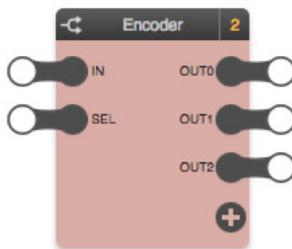
Each block contains at least one input and/or output "node", as shown in the following figure:



The input nodes are always on the left side of a block, while the output nodes are on the right. Each node is characterized by a concise tag (e.g.: "IN1", "IN2" and "OUT" in the figure above) that is given in the inputs / outputs list in the details pane, along with a brief description of each node (as well as in this manual).

3.8.1 Logic blocks

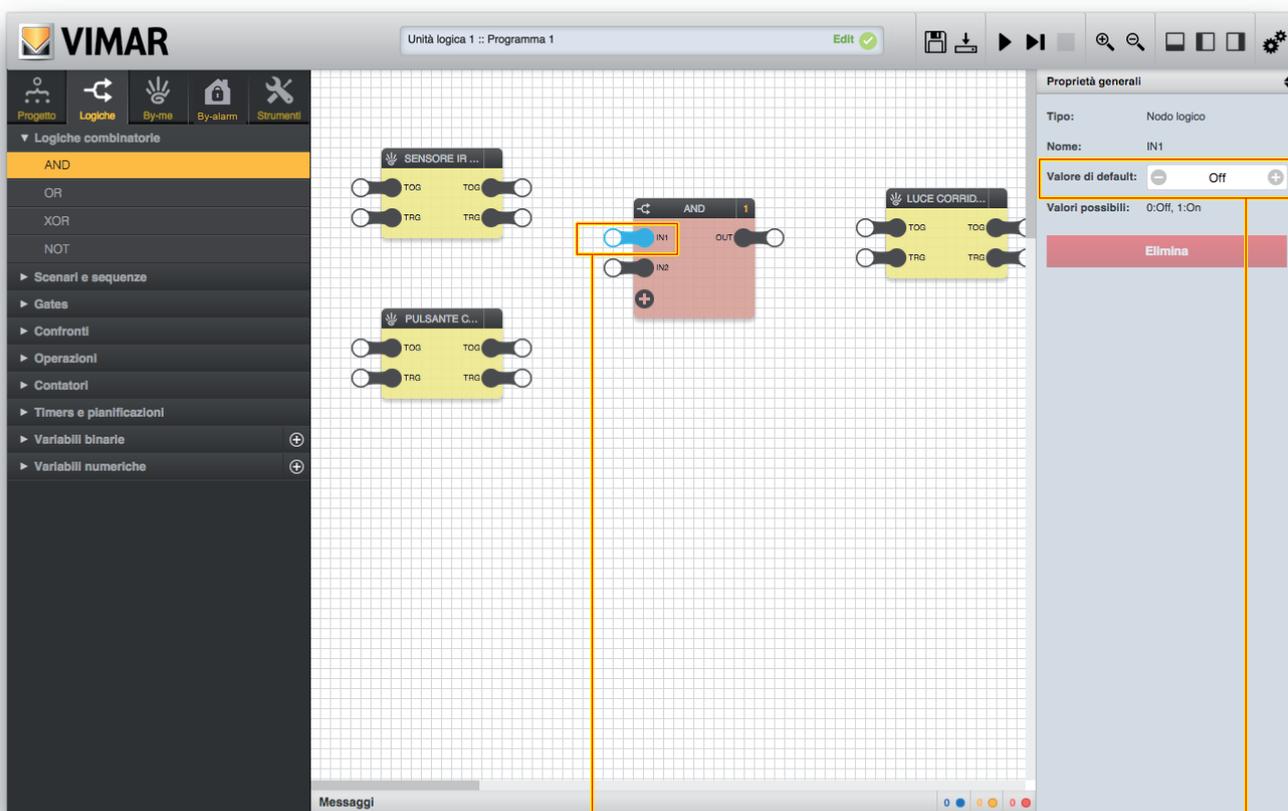
In the case of the logic blocks, the input nodes represent the "inputs" for the logic function associated to the block, while the output nodes are the "outputs":



In some cases, as in this example, the block provides for a variable number of nodes (input or output nodes); in this case, the "+" button is used to add nodes to the block, up to the maximum number.

The logic function can only be run properly if the input nodes are connected to other blocks (both logic and By-me blocks) and if the output values are "reported" on the input nodes of the same number of blocks (both logic and By-me blocks).

Not all the input nodes are strictly necessary for correctly running the logic function; if an input node is not connected, its default value is used, which can be modified by selecting the node and opening the corresponding details pane, as exemplified in the following figure:



Selected node

Default value

Logic programs

The detail pane of a node also shows the possible values it can have; this information can be useful especially for blocks that provide for specific value combinations or restrictions.

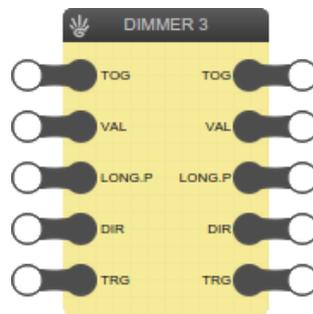
The logic blocks may also provide for outputs only, as shown in the following example (planning block):



In this case, they can only be used as an input for other logic elements, but they cannot be controlled. In the specific case of planning, for example, as detailed below (section 6.9.2), the value depends on the system clock of the logic unit, according to preset programming.

3.8.2 By-me blocks

In the case of By-me blocks, the input nodes (left side) represent the possible commands that the logic unit can send to the corresponding group (bus transmission); the outputs (right side), conversely, are the states that the logic unit can receive from the corresponding group via the bus. In the case for example of a "dimmer" group...



... two different types of data are available as either input or output:

- TOG ("toggle"): switching the dimmer on / off
- VAL ("value"): dimmer percentage
- LONG.P: long press start/end
- DIR: long press direction

If you want to send one of these two values over the bus, you need to connect the output of the corresponding logic element to the input node (left side), so that whenever the logic changes its value it gets sent, via the bus, to the corresponding dimmer actuator. Vice versa, if you want to build a logic element that is based on the state of one of these two pieces of information, it is necessary to connect the corresponding output node (right side) to one or more logic blocks, so that each change in status detected by the bus is "passed" on to the logic element.

Not all By-me blocks have the same number of input and output nodes; some information can only be read (e.g.: temperature measured by a thermostat) or, vice versa, some commands can only be sent to the devices but have no meaning as regards status (e.g.: moving or stopping a roller shutter).

3.8.3 Trigger

By-me blocks provide for, as either input or output, a special "trigger" node (TRG):

- INPUT TRIGGER (left side): used to force transmission of the values of the input nodes (connected to other logic elements) even without a change in value
- OUTPUT TRIGGER (right side): used to detect receiving data from the bus, on one of the output nodes, even if the same as the previous data (so no change in value)

In both cases, the trigger is normally 0 and goes to 1 when the trigger is active:

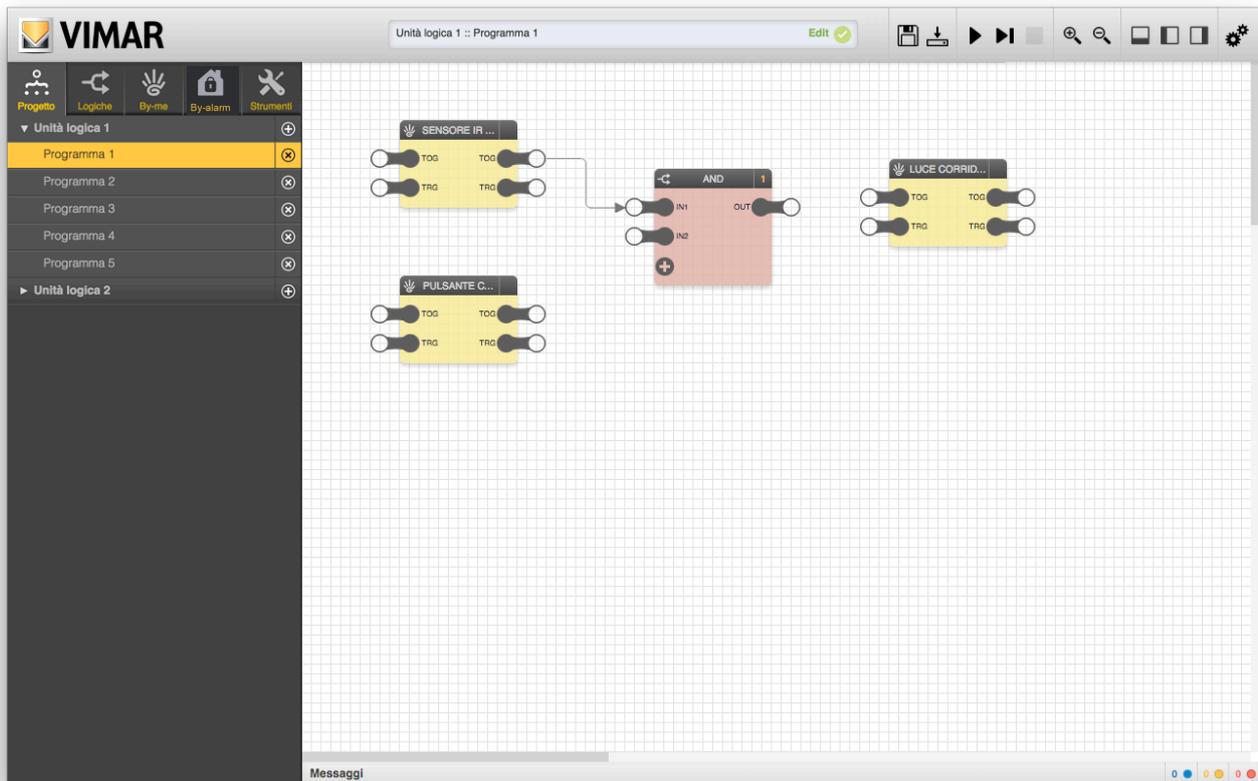
- INPUT TRIGGER: when set to 1 it forces transmission (once) until it is reset and put back to 0 (or there is a change in value)
- OUTPUT TRIGGER: set to 1 by the logic unit whenever data is received from the bus concerning a the corresponding group on the By-me block in question (on one of the output nodes of the block) and is then automatically reset with the next cycle

Under normal circumstances, the logic blocks do not provide for triggers, operating on the change in state; where necessary, however, also specific logic blocks provide for input/output trigger nodes, whose operation is entirely similar to that described above for the By-me blocks.

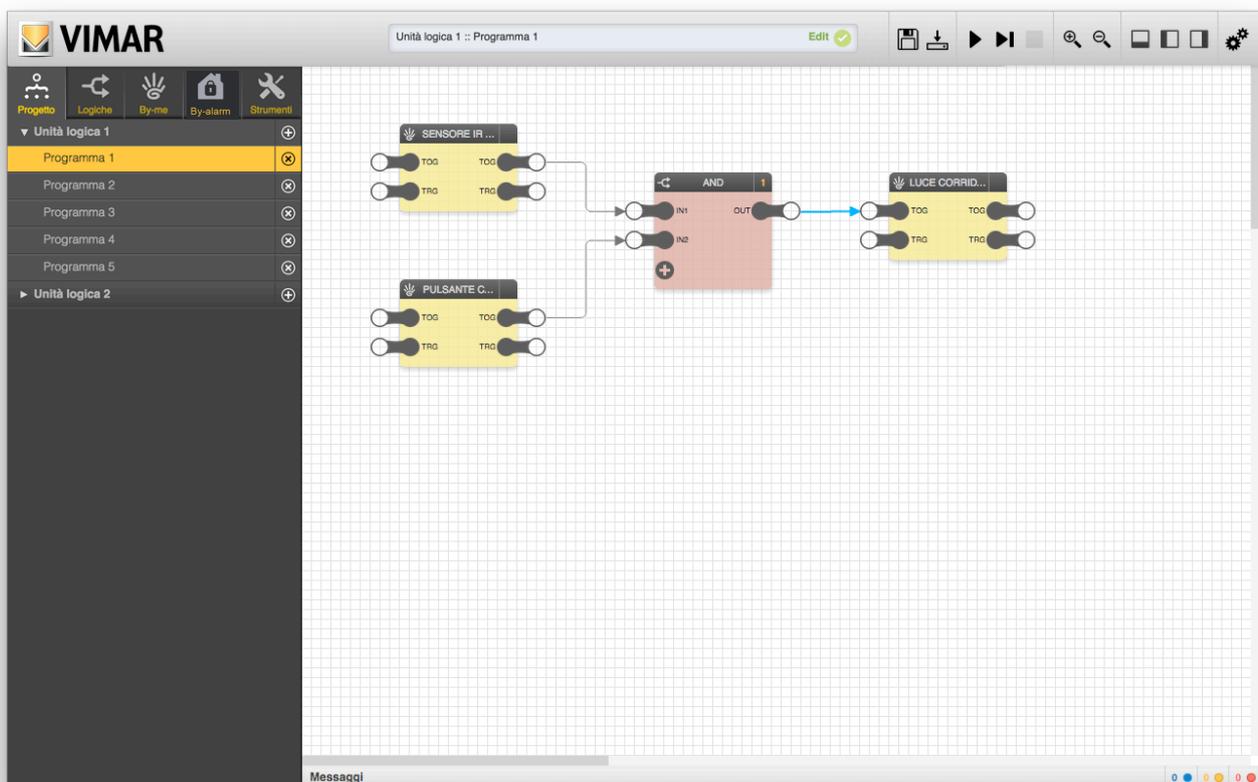
A trigger node (TRG) must be connected to a node with the same operating method (status change impulse); if this is not the case, the specific logic block must be used.

3.9 Connecting blocks

For the program to perform effectively, you must include at least one "connection" between two nodes of two blocks, so that the value of the first one ("source") is passed to the second one ("target"). To connect two nodes, simply click in the centre of the source node, hold and release in the centre of the target node:



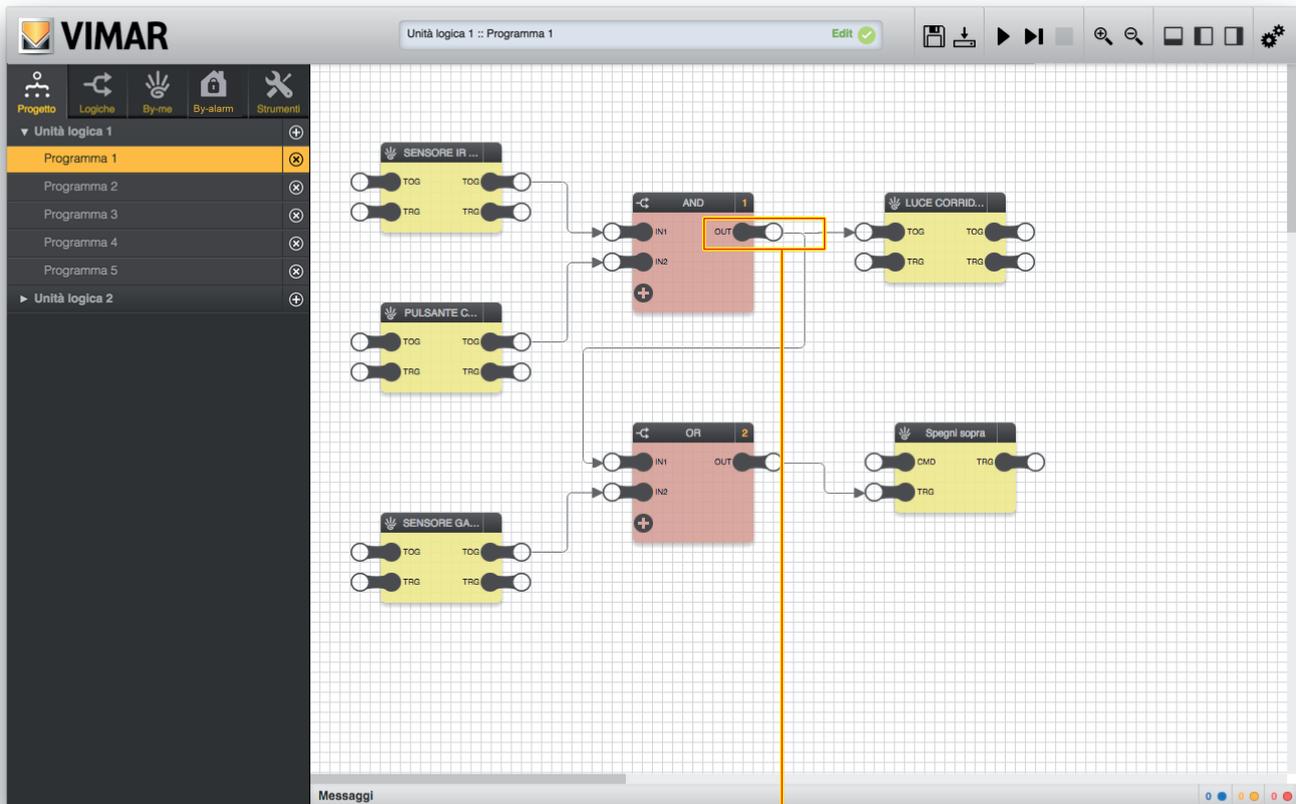
Moving the cursor over a link highlights it in red (and passes "on top" of any other connections or blocks along its route); clicking on it, conversely, selects the link:



The selected link can be deleted in either of two ways:

- Pressing the "DELETE" button in the details pane
- Directly pressing the "Del" key on the keyboard

The source of a connection must be an output node (right side of a block) while the target must be an input node (left side); an output node can be the source of several connections (with different targets), while an input node can be the target of only one connection:



Output node with more than one connection

3.10 Types of nodes

The following table gives the types of nodes.

Type of node	Description
T	TRIGGER: The change in the node value is instantaneous, the node value immediately returns to the value prior to this change.
S	STATUS: The value keeps stable until the next change in status.
M	MIXED: Independent node on status change; it can be of either the STATUS or the TRIGGER type.

When connecting nodes together it is important to pay attention to their type: you cannot directly connect a TRIGGER node to a STATUS node and vice versa. Instead, you can connect STATUS or TRIGGER nodes to MIXED nodes.

Thanks to these types the application allows you to make no connection errors.

3.11 Running order

While simulating and compiling, as detailed below, the editor, starting from the graphically designed logic networks, generates a "list" that is put into execution in a cyclic manner, from beginning to end, as quickly as possible (depending on the complexity of the project).

3.11.1 Program order

On each run cycle, the following operations are performed (the cycle time depends on the number and complexity of the programs):

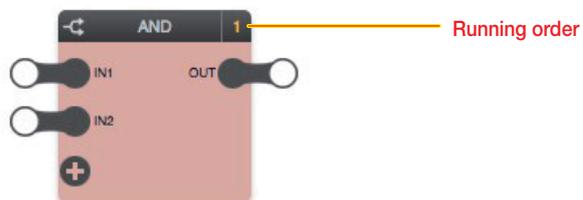
- Read inputs from bus
- Run program 1
- Run program 2
- ...
- Run program n
- Write commands on bus

The order of the programs is exactly as shown in the main menu; this implies that any interactions between the programs (such as passing values by means of variables, or multiple programs writing the same node of a By-me block) are affected by this order (and any actions taken by the programs "at the end" of the list are received by the previous ones only in the next cycle).

NOTE: If a program is disabled (cf. 3.3) or paused, it is "skipped" in the cycle; any interaction with the bus and/or with other programs in this case is suspended.

3.11.2 Block order

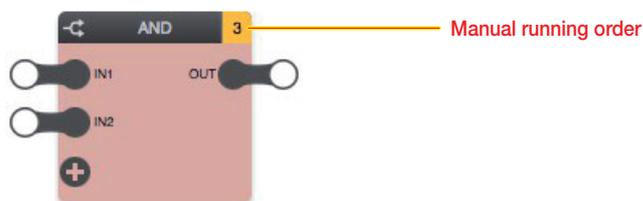
Within each program, the logic blocks also have their own running order; the logic unit processes the function associated with the logic blocks in the following order. The order of a logic block is shown at the top right, as in the following figure:



IMPORTANT: Always make sure that the order of the blocks is in line with the order of the logic performance (otherwise the logic will malfunction). Under normal circumstances, the blocks are assigned an ascending order based on the order in which they are added to the program; however, it is possible to force a different order in the following way:

- Select the block concerned
- Open the details pane
- Select "MANUAL" as sorting order
- Enter an order number, making sure to enter a number that has not yet been used

Blocks with manual sorting are highlighted as follows:



The following figure shows an example of a logic network with a block in manual sorting order and shows how to change the running order of the blocks:

Block with manual running order

Block running order settings

Logic programs

The By-me blocks do not have a running order, their sorting is not important at the moment and is reserved for future functions.

They do not represent processing by the logic unit, but only points for reading and writing via the bus; as mentioned previously, the states of the output nodes of all the By-me blocks (of all the active programs) are read at the beginning of each cycle and the commands to the input nodes of all the By-me blocks (of all the active programs) are sent over the bus at the end of the cycle, regardless therefore of the position of the blocks in the programs and of the order of the programs.

Generally, the order of the blocks in logic programs must follow a flow like the one described below:

- IN: By-me objects in reading
- PROCESSING: object logic network
- OUT: writing in By-me

this diagram is given in all the examples in the manual and must be followed as a rule to avoid logics that cannot be correctly implemented by the Logic Unit.

3.12 Passing values between programs

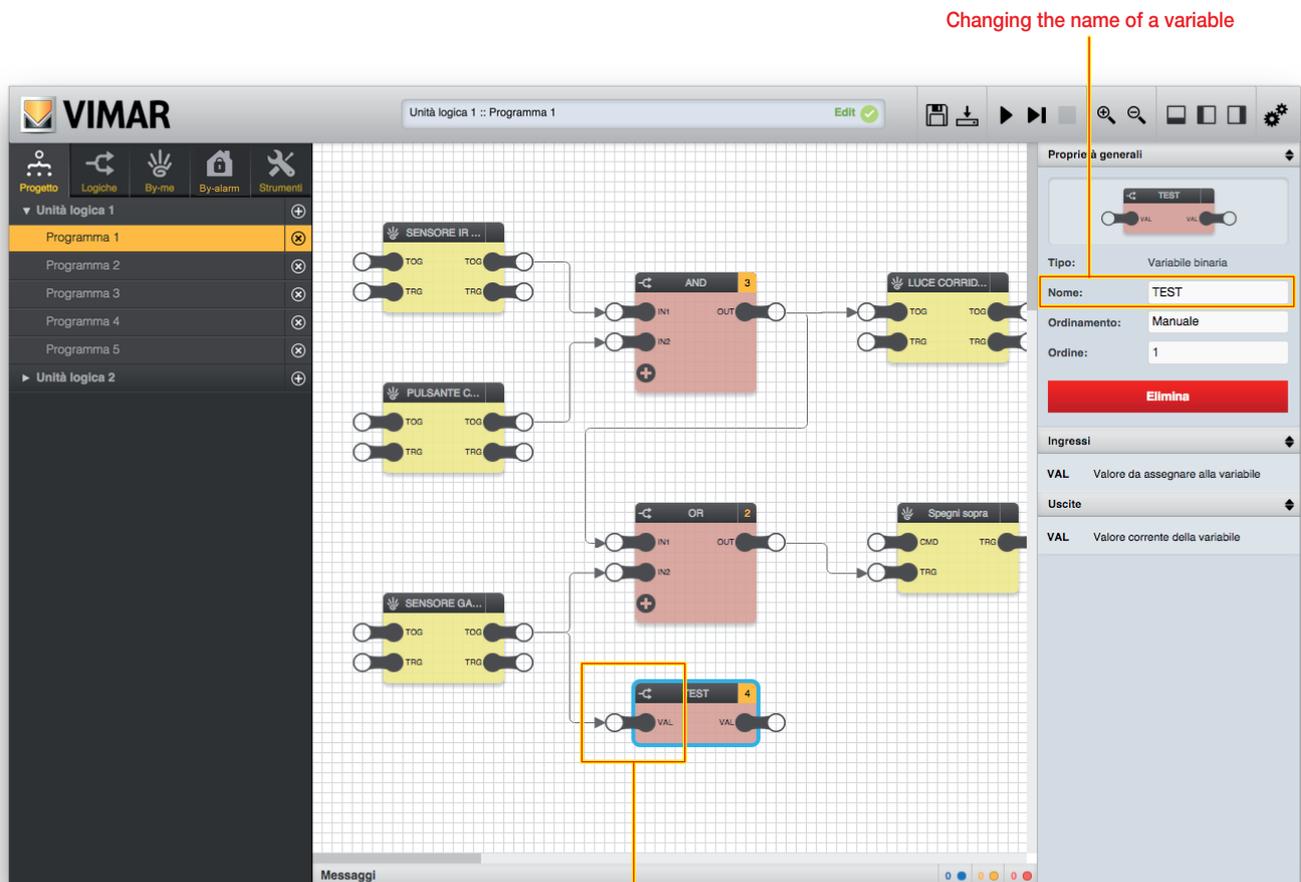
Although each program is a logic network on its own, you can pass values between different programs using special logic blocks called "variables". To create a new variable you must do the following:

- Open the "LOGIC" section of the main menu
- Identify the section called "BINARY VARIABLES" (if you want to create an ON/OFF type of variable) or "NUMERICAL VARIABLES"
- Press the corresponding "+" button and wait for the new variable to be added to the list
- Select the new variable and drag it into the first program

It is possible to assign a name to the variable in the details pane, to identify it more easily within the programs where it is to be used.

If you want to give the variable the value of an output node of a block (whether a logic or By-me block), simply connect the output node (right side) of the variable; vice versa, to use this value in other programs, connect the output node (right side) to the input node (left side) of another block (also in this case, either a logic or By-me block) as shown in the following figures.

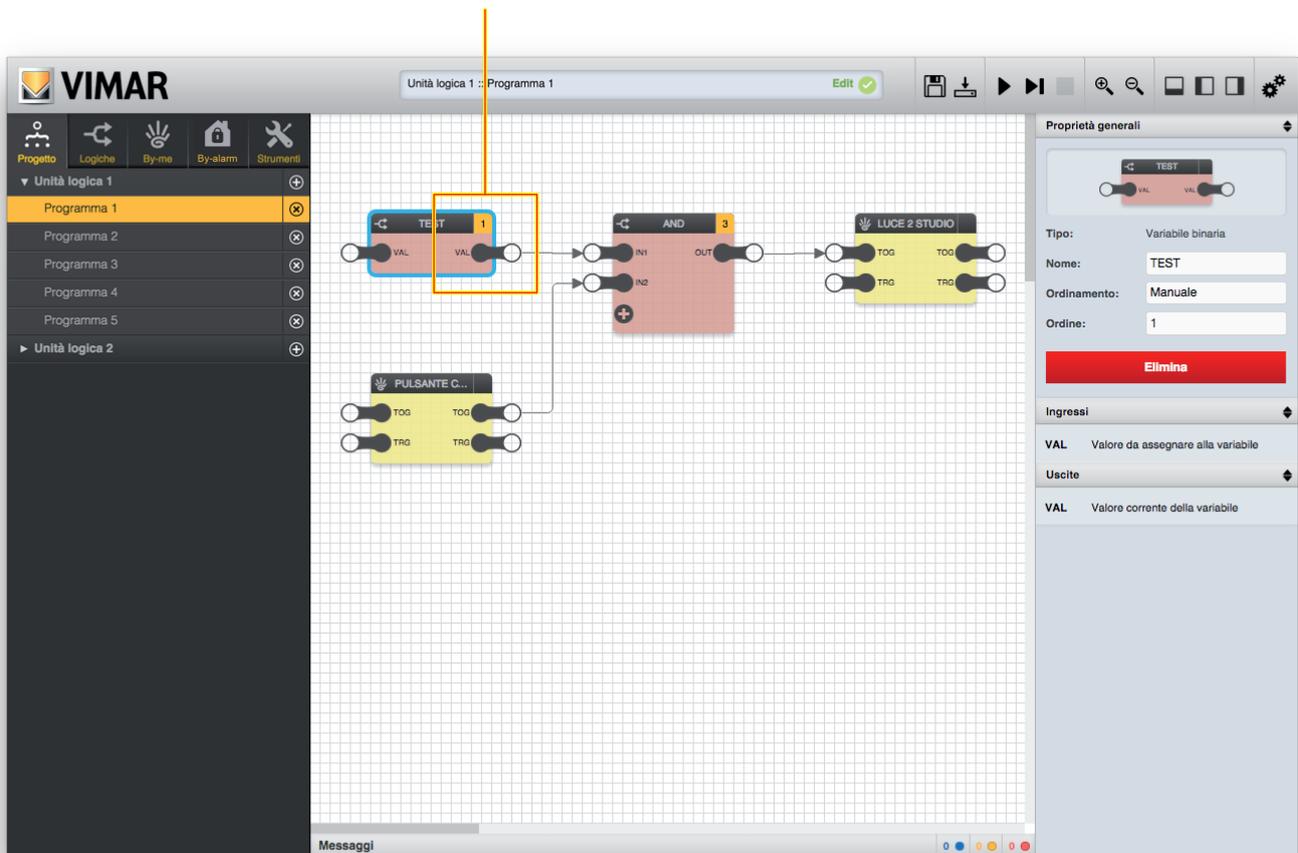
- You are advised to limit the use of variables used to transport only information retrieved from a logic network from one logic program to another.
- Use with caution: using the variables to "transport" data from By-me objects can lead to incorrect logic writing.
- IT IS NOT PERMITTED to create programs in which By-me blocks are in a different position other than IN and OUT in a logic.
- The same By-me block can be involved in more than one program as input but only in one program as output; this is to avoid malfunctions.



Changing the name of a variable

Assigning the value to a variable

Using the value of a variable



3.13 Data types

The input and output nodes of the blocks may include two types of data:

- BINARY: only the values 1 (ON) and 0 (OFF) are allowed
- NUMERICAL: any numerical value is allowed, possibly with specific restrictions depending on the block

These two data types are incompatible, so the editor prevents connecting binary nodes with numerical nodes and vice versa: as soon as you start dragging and dropping to create a connection, the incompatible nodes are made semi-transparent and cannot be released for creating the connection.

3.14 Saving

On closing the editor, the logic programs are automatically saved within the EASYTOOL PROFESSIONAL project, so you can edit them later.

You can however manually save the state of the logic programs – of all the logic units that there may be in the project – with the "SAVE" button on the toolbar; a progress screen is displayed while saving, and it is not possible to work on the logic programs.

3.15 Simulation

Before transferring the programs to the logic units, it is advisable to test them in the editor using "SIMULATION", which lets you manually enter the possible values received from the bus and check the behaviour of the logic networks, both continuously (repeated execution of the logic in real-time) and "step by step" (i.e. performing one calculation cycle at a time).

For more details on simulation, see chapter 6.

4. By-me

4.1 Introduction

As mentioned previously, the By-me blocks enable reading values from the home automation bus and sending commands to the By-me groups after the logic processing carried out in the programs that contain them.

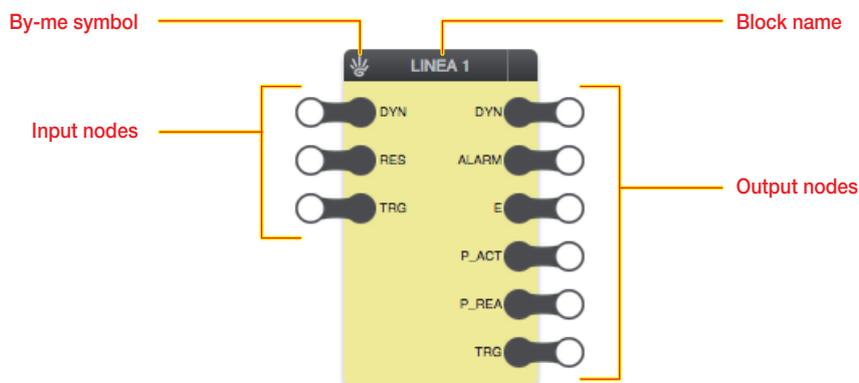
The By-me blocks available in the relevant section of the main menu are generated by an EASYTOOL PROFESSIONAL project import procedure, called up automatically when the editor comes on whenever the EASYTOOL PROFESSIONAL project changes (cf. 1.2).

CAUTION: If the timer parameter on the single functional unit is set (control or relay) in art. 01480, 01481, 01482, 01485, 01486 and 01487, the Editor of the logic unit will not import the relative group.

4.2 By-me blocks

4.2.1 Layout

As mentioned earlier, the By-me blocks appear graphically as shown in the following example:



The By-me blocks are characterized by a yellow background.

4.2.2 Input nodes

The input nodes enable sending commands over the bus after the processing carried out in the logic programs; the available nodes depend on the type of By-me group, as detailed later on in this chapter.

By selecting a node and opening the details pane, you can set the following options:

Control strategy	This determines with which criterion the node value is sent over the bus. Possible values: <ul style="list-style-type: none"> • On changing: the value is sent when it changes (unless the By-me block <i>trigger</i> is explicitly set to 1, as detailed below) • Periodic sending: the value is sent, as well as on changing, also periodically, with a settable time
Time for periodic sending	In the case of periodic sending, it determines the time between one send and the next one <i>Possible values:</i> 1 ... 600 (seconds) <i>Note: low cyclical sending times can generate excess traffic on the bus.</i>
Initial Sync	Allows you to "force" sending the value of the node over the bus at start-up. For full details see section 4.2.2.1 on the next page.

WARNING: Periodic sending can create traffic problems on the bus, particularly when they use low period values. This option must therefore be used only when it is strictly necessary to continuously underline data on the bus.

The details pane, in addition to the above options, also shows the possible values which the node can take; in the case of binary nodes the possible values are 0 (OFF) or 1 (ON), in the opposite case of numerical nodes the possible values depend on the type of node, and may have specific restrictions.

4.2.2.1 Initial Sync

The Sync function, which involves only the "actual" input nodes and not the TRG nodes, lets you force sending the value of the node over the bus on starting the logic unit (by default this feature is disabled).

If the flag ✓ is added for a given node, on starting up the logic unit will send a message over the bus with the value of the corresponding data point even if the latter has not changed compared to its default value.

This option, especially if extended to all the logic nodes, can generate higher traffic on the bus; due to this it should then only be used for nodes where it is necessary to immediately restore a value consistent with the logic gates (eg if you restart the system or logic unit after a power failure).

Caution: The Sync function should never be selected for the input nodes of the By-me blocks used as "logic inputs" (see the following figures).

Correct selection

Proprietà generali

Tipo:	Nodo By-me
Nome:	TOG
Valori possibili:	0:Off, 1:On
Strategia di co...:	Su variazione
Sync iniziale:	<input checked="" type="checkbox"/>
Id:	610

Elimina

Incorrect selection

Proprietà generali

Tipo:	Nodo By-me
Nome:	TOG
Valori possibili:	0:Off, 1:On
Strategia di co...:	Su variazione
Sync iniziale:	<input checked="" type="checkbox"/>
Id:	607

Elimina

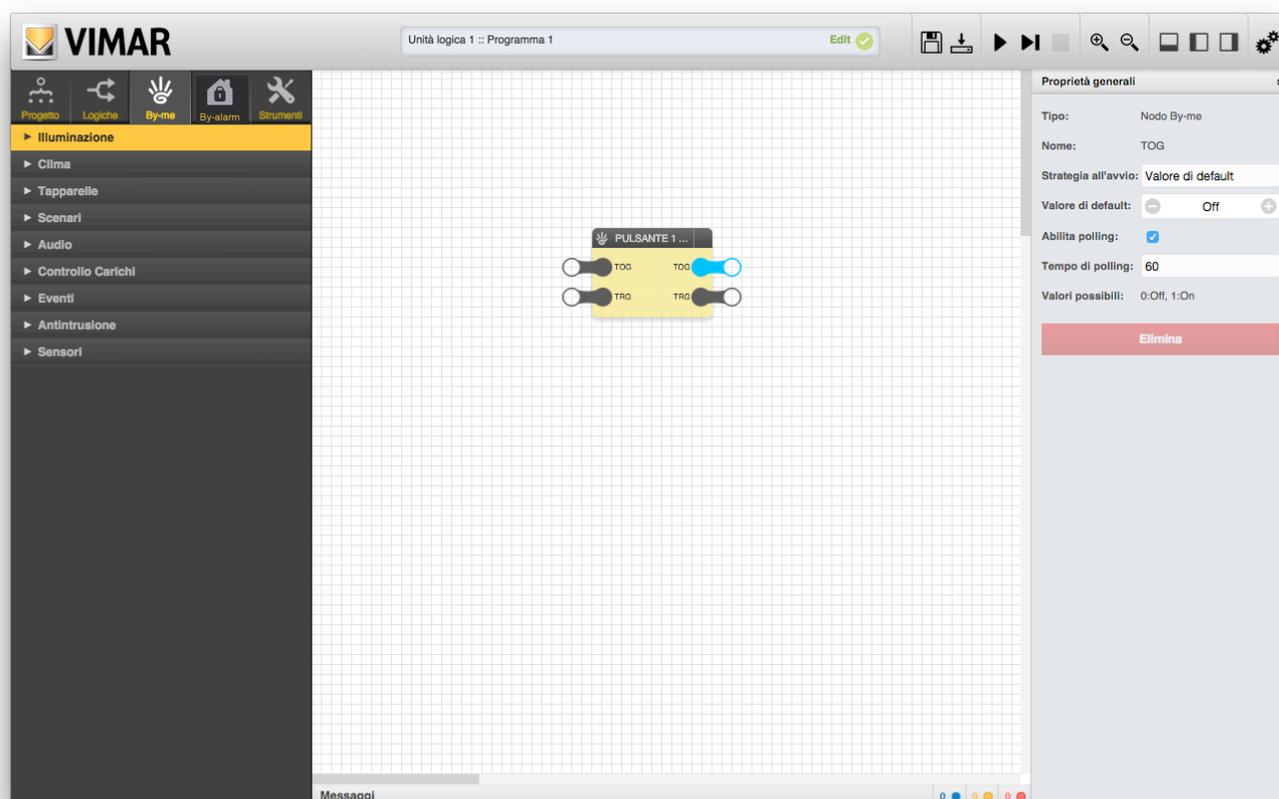
4.2.3 Output nodes

The output nodes enable receiving states from the bus and using them in the logic programs; the available nodes depend on the type of By-me group, as detailed later on in this chapter.

By selecting a node and opening the details pane, you can set the following options:

Start-up strategy	This determines which value the node must take on starting the logic unit. Possible values: <ul style="list-style-type: none"> • Default value: the "default value" set by the user is used (see below) • Last value: the last value received before switching off the logic unit is used • Reading from bus: a request for reading the status is sent to the device
Default value	This allows setting the default value of the node, which is used in the logic elements until a different value is received
Enable polling	Enables periodic reading of the node value by polling the device over the bus
Polling time	Periodic device polling time. Possible values: 1 ... 600 (seconds) Note: low polling times can generate excess traffic on the bus

WARNING: Periodic sending can create traffic problems on the bus, particularly when they use low period values. This option must therefore be used only when it is strictly necessary to continuously underline data on the bus.



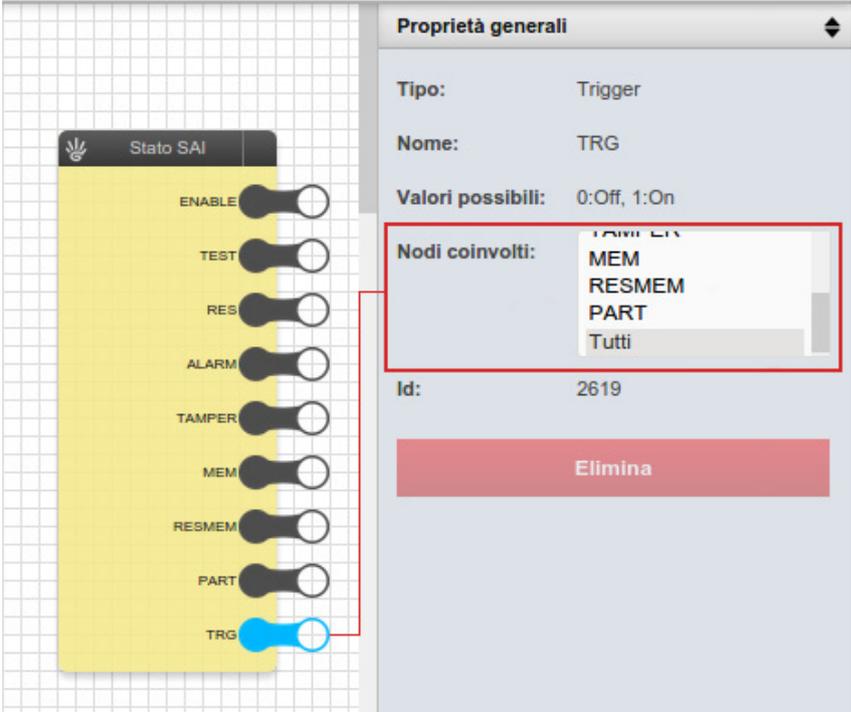
4.2.4 Trigger

As mentioned in chapter 4, the By-me blocks have two trigger nodes, one at input and one at output.

The input trigger makes it possible to force sending commands related to the input nodes (connected to other blocks), even though their value has not changed. When this node is set to 1 (through a connection starting from a logic block n in the program), the logic unit sends the commands over the bus, regardless of the current value and of any periodic sending; to repeat forced sending, it is necessary to set the trigger to 0 and then back to 1.

The trigger output, on the other hand, is set to 1 by the logic unit whenever data are received from the bus on one of the output nodes (connected to other blocks), even if the value has not changed; the trigger remains on 1 for the cycle and then goes back to 0, until data are next received from the bus.

Using the "Nodes involved" option in "General properties" in the details pane, for both triggers you can establish which nodes in the By-me block trigger the trigger signal, for output triggers, or, for input triggers, cause telegrams to be sent to the corresponding group addresses on the bus.



The image shows a configuration interface for a logic unit. On the left, a panel titled "Stato SAI" contains a vertical list of toggle switches for various states: ENABLE, TEST, RES, ALARM, TAMPER, MEM, RESMEM, PART, and TRG. The TRG switch is highlighted in blue and is currently turned on. A red line connects the TRG switch to the "Nodi coinvolti" field in the "Proprietà generali" panel on the right. The "Proprietà generali" panel displays the following information:

- Tipo:** Trigger
- Nome:** TRG
- Valori possibili:** 0:Off, 1:On
- Nodi coinvolti:** A list box containing MEM, RESMEM, PART, and Tutti.
- Id:** 2619

At the bottom of the "Proprietà generali" panel, there is a red button labeled "Elimina".

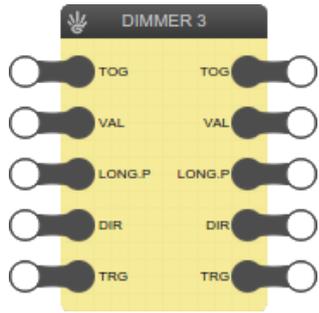
CAUTION: The images shown for the various By-me blocks are the most representative ones. They are not intended to be complete and comprehensive as the shape and the presence of nodes depends on the configuration and the type of devices in the By-me group.

4.3 Lighting

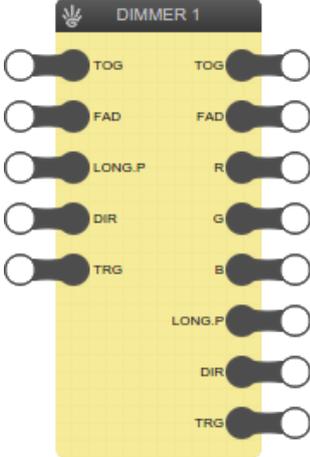
4.3.1 ON/OFF lights

Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	TOG	On/Off control <i>Possible values:</i> 0 → OFF 1 → ON	S	•	•
	TRG	Trigger for sending/receiving from/to bus	T	•	•

4.3.2 Dimmer

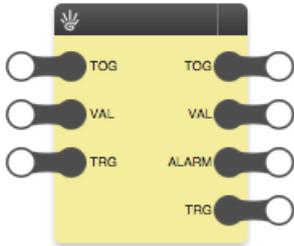
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	TOG	On/Off control <i>Possible values:</i> 0 → OFF 1 → ON	S	•	•
	VAL	Percentage adjustment <i>Possible values:</i> 0 ... 100 [%]	S	•	•
	LONG.P	Long press start/end <i>Possible values:</i> 0 → STOP long press end 1 → START long press start	S	•	•
	DIR	Long press direction <i>Possible values:</i> 0 → Down 1 → Up	S	•	•
	TRG	Trigger for sending/receiving from/to bus	T	•	•

4.3.3 RGB

Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	TOG	On/Off control <i>Possible values:</i> 0 → OFF 1 → ON	S	•	•
	FAD	Fading show ON/OFF <i>Possible values:</i> 0 → OFF 1 → ON	S	•	•
	R	Red intensity <i>Possible values:</i> 0 ... 255 [%], 255=100%	S		•
	G	Green intensity <i>Possible values:</i> 0 ... 255 [%], 255=100%	S		•
	B	Blue intensity <i>Possible values:</i> 0 ... 255 [%], 255=100%	S		•
	LONG.P	Long press start/end <i>Possible values:</i> 0 → STOP long press end 1 → START long press start	S	•	•
	DIR	Long press direction <i>Possible values:</i> 0 → Down 1 → Up	S	•	•
	TRG	Trigger for sending/receiving from/to bus	T	•	•

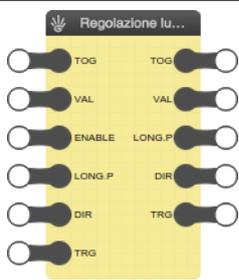
4.3.4 Proportional analogue output actuator

For example, the group must contain a device such as: Proportional analogue 4-output actuator art. 01466.

Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	TOG	On/Off control <i>Possible values:</i> 0 → OFF 1 → ON	S	•	•
	VAL	Percentage value <i>Possible values:</i> 0 ... 100 [%]	S	•	•
	ALARM	Alarm Must be set to 1 when the VAL input value exceeds a threshold <i>Possible values:</i> 0 → OFF 1 → ON	S		•
	TRG	Trigger for sending/receiving from/to bus	T	•	•

4.3.5 Brightness control

For example, the group must contain a device such as: Device with 3 analogue signal inputs art. 01467 (connecting to brightness sensor art. 01530).

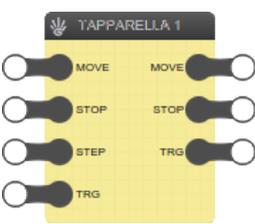
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	TOG	On/Off control <i>Possible values:</i> 0 → OFF 1 → ON	S	•	•
	VAL	Value <i>Possible values:</i> 0 ... 100 [%]	S	•	•
	ENABLE	Calibration sensor ON continuous brightness <i>Possible values:</i> 0 → OFF 1 → ON	S	•	
	LONG.P	Long press start/end <i>Possible values:</i> 0 → STOP long press end 1 → START long press start	S	•	•
	DIR	Long press direction <i>Possible values:</i> 0 → Down 1 → Up	S	•	•
	TRG	Trigger for sending/receiving from/to bus	T	•	•

4.4 Shutters

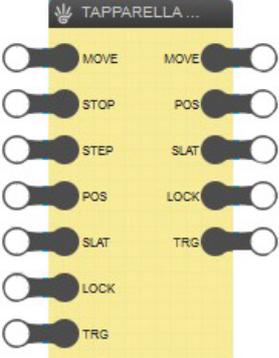
4.4.1 Shutters up/down

Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	MOVE	Movement up/down <i>Possible values:</i> OFF, 0 → Up ON, 1 → Down	S	•	•
	STOP	Stopping the movement <i>Possible values:</i> ON → Stop	S	•	•
	TRG	Trigger for sending/receiving from/to bus	T	•	•

4.4.2 Roller blinds

Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	MOVE	Movement up/down <i>Possible values:</i> 0 → Up 1 → Down	S	•	•
	STOP	Stopping the movement <i>Possible values:</i> ON → Stop	S	•	•
	STEP	Regulating laths up/down <i>Possible values:</i> 0 → Up 1 → Down	S	•	
	TRG	Trigger for sending/receiving from/to bus	T	•	•

4.4.3 Roller blinds with position

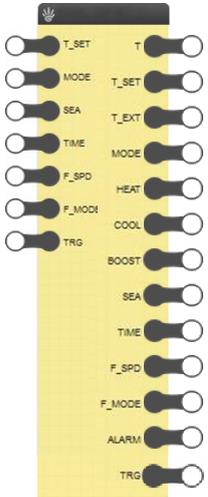
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	MOVE	Movement up/down <i>Possible values:</i> 0 → Up 1 → Down	S	•	•
	STOP	Stopping the movement <i>Possible values:</i> ON → Stop	S	•	
	STEP	Regulating slats up/down <i>Possible values:</i> 0 → Up 1 → Down	S	•	
	POS	Percentage position <i>Possible values:</i> 0 ... 100 [%] 0 =open, 100=closed	S	•	•
	SLAT	Percentage slat position <i>Possible values:</i> 0 ... 100 [%] 0 =open, 100%=closed	S	•	•
	LOCK	Roller shutters lock <i>Possible values:</i> ON, 1 → Locked OFF, 0 → Unlocked	S	•	•
	TRG	Trigger for sending/receiving from/to bus	T	•	•

By-me

4.5 Climate

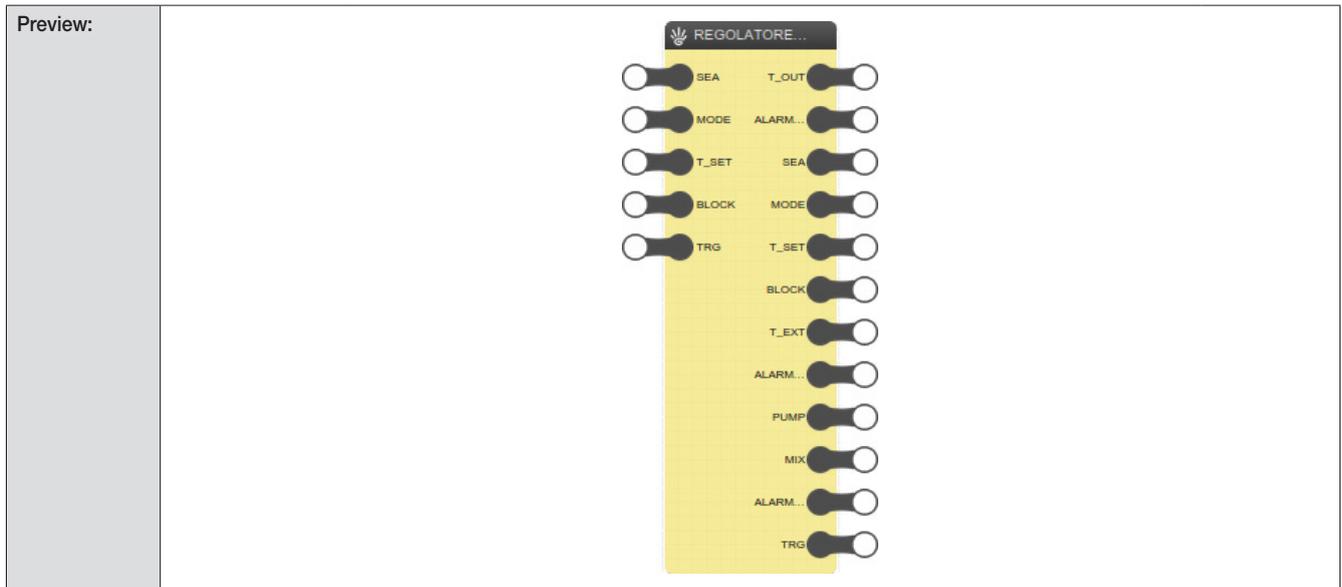
4.5.1 Thermostat/temperature probe

N.B. The logic can manage only thermostats art. 02951.

Preview:					
					
Nodes:	TAG	Description	TYPE	IN	OUT
	T_SET	Temperature setpoint <i>Possible values:</i> 0 ... 50 The setpoint refers to the current operating mode of the thermostat; by setting a value on this node, we therefore modify the On mode setpoint	S	•	•
	MODE	Operating mode <i>Possible values:</i> 0 → Automatic 1 → Manual 2 → Reduction 3 → Absent 4 → Protection 5 → Timed manual 6 → OFF	S	•	•
	SEA	Season (control mode) <i>Possible values:</i> 0 → Neutral zone 1 → Air conditioning 2 → Heating	S	•	•
	TIME	Timing <i>Possible values:</i> 1 ... 255 [min] When set, it represents the time during which the thermostat stays in "Timed manual" mode (thus running with the fixed setpoint and ignoring any weekly programming) before returning to automatic mode. This parameter defines only the duration of this operating mode but does not determine the passage; the mode is determined by the value in the MODE node.	S	•	•
	HUM	Humidity sensor <i>Possible values:</i> 0 ... 100 [%]	S		•
	F_SPD	Fan coil speed <i>Possible values:</i> 0 ... 100 [%] Expressed as a percentage even with fan coils controlled on 3 ON-OFF speeds; in this case the 3 speeds correspond to the values 33%, 66% and 100%	S	•	•
	F_MODE	Fan coil mode <i>Possible values:</i> 0 → Automatic 1 → Manual	S	•	•
	T	Temperature measured <i>Possible values:</i> 0...40.0 [°C]	S		•
	T_EXT	Temperature measured (external probe) <i>Possible values:</i> -20...80.0 [°C]	S		•
	HEAT	Heating main output status <i>Possible values:</i> 0 → OFF 1 → ON	S		•
	COOL	Air-conditioning main output status <i>Possible values:</i> 0 → OFF 1 → ON	S		•
	BOOST	Boost status (auxiliary heating/cooling) <i>Possible values:</i> 0 → OFF 1 → ON	S		•
	ALARM	Screed alarm <i>Possible values:</i> 0 → OFF 1 → ON	S		•
	TRG	Trigger for sending/receiving from/to bus	T	•	•

Note: The number and the type of nodes may depend on the specific configuration of the project

4.5.2 Climate controllers



Nodes:	TAG	Description	TYPE	IN	OUT
	SEA	Season (regulating mode) <i>Possible values:</i> 0 → Air conditioning 1 → Heating	S	•	•
	MODE	Operation <i>Possible values:</i> 0 → Auto 1 → Comfort 2 → Economy 3 → OFF	S	•	•
	T_SET	Setpoint <i>Possible values:</i> 10 ... 100 [°C]	S	•	•
	BLOCK	Controller block ON/OFF <i>Possible values:</i> 0 → No alarm 1 → Alarm	S	•	•
	T_OUT	Delivery probe temperature <i>Possible values:</i> -20 ... 110 [°C]	S		•
	AL.T_OUT	Delivery probe alarm <i>Possible values:</i> 0 → No alarm 1 → Alarm	S		•
	T_EXT	External temperature probe <i>Possible values:</i> -20 ... 70 [°C]	S		•
	AL.T_EXT	External probe alarm <i>Possible values:</i> 0 → No alarm 1 → Alarm	S		•
	PUMP	Pump open/closed <i>Possible values:</i> 0 → OFF 1 → ON	S		•
	MIX	Valve opening <i>Possible values:</i> 0 ... 100 [%]	S		•
	AL.PROP	Proportional output alarm <i>Possible values:</i> 0 → No alarm 1 → Alarm	S		•
	TRG	Trigger for sending/receiving from/to bus	T	•	•

Note: The number and the type of nodes may depend on the specific configuration of the project

4.6 Scenarios

4.6.1 By-me scenarios

Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	TRG	Trigger for sending/receiving from/to bus. The input trigger is used to control the scenario; the output trigger notifies that the scenario has been run on the bus	T	•	•

WARNING: It is not possible to create a logic in which groups interact with Scenarios that contain the same groups.

4.7 Audio

4.7.1 Speaker systems zones

Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	TOG	On/Off switch <i>Possible values:</i> 0 → OFF 1 → ON	S	•	•
	VOL	Volume <i>Possible values:</i> 0 ... 99 [%]	S	•	•
	CHAN	Channel (selecting the sound source from a maximum of 4 available) <i>Possible values:</i> 1 ... 4	S	•	•
	C+/T+	Channel+/Track+ <i>Possible values:</i> 0 → Track+ 1 → Channel+	S	•	•
	SLEEP	Timed ON/OFF switch <i>Possible values:</i> 0 → OFF 1 → ON	S	•	•
	TRG	Trigger for sending/receiving from/to bus	T	•	•

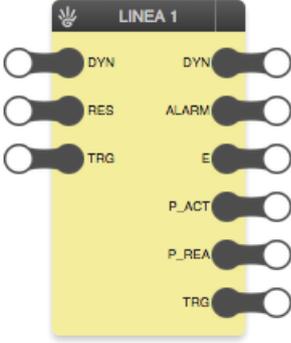
4.8 Energy management

4.8.1 Loads

Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	ON	Forcing ON <i>Possible values:</i> 0 → OFF 1 → ON	S	•	
	OFF	Forcing OFF <i>Possible values:</i> 0 → OFF 1 → ON	S	•	
	MODE	Operating mode <i>Possible values:</i> 0 → Automatic ON 1 → Automatic OFF 2 → Forced ON 3 → Forced OFF	S		•
	TRG	Trigger for sending/receiving from/to bus	T	•	•

4.8.2 Line controllers

This object represents part of the device 01455 linked to a single line in the system. There will therefore be as many "Line Manager" objects as there are lines configured in the system. According to this configuration, each line will measure (or not) consumption or production. See the notes concerning individual nodes for more information.

Preview:					
					
Nodes:	TAG	Description	TYPE	IN	OUT
	DYN	Dynamic mode <i>Possible values:</i> 0 → OFF 1 → ON Dynamic Mode of meters. Present in the line manager if the system configuration allows a meter in the line manager index. This parameter is useful when in ON the active power measurement is sent to the device for a time equal to the parameter "Measurement refresh duration" and a frequency equal to the parameter "Measurement refresh frequency" (see for example Controller Installation Manual supplied with 21509 para. 3.4 section Energy Management). It can be used by a display device to show the value in real time, such as for example on opening a page in the touch screen; after the set time the transmission ends.	S	•	•
	RES	Partial reset <i>Possible values:</i> 0... 2147483648 [Wh] Used to set the partial energy measurement of the meter in the line manager index to a given value; it is present if the system configuration allows a meter in the line manager index. This parameter forces the partial energy value to the value set here and is useful for aligning the energy value calculated by the Vimar device to that of an external meter. The measurement MUST be of the same time, which depends on how the system has been configured and how the current sensors are positioned: exchanged (if for production), produced (photovoltaic meter) or consumer (without production).	S	•	
	ALARM	At least one load on the line off <i>Possible values:</i> 0 → OFF 1 → ON	S		•
	T_MIN (*)	Minimum threshold <i>Possible values:</i> 2...135 [kW] It represents the minimum energy threshold for the load control logic. Threshold value 1 refers to the value set in the line manager device in question.	S		•
	T_MAX (*)	Maximum threshold <i>Possible values:</i> 2...135 [kW] It represents the maximum energy threshold for the load control logic. Threshold value 2 refers to the value set in the line manager device in question.	S		•
	E (*)	Partial energy <i>Possible values:</i> 0... 2147483648 [Wh] It represents the energy measured from the last reset.	S		•
	P_ACT (*)	Power on <i>Possible values:</i> 0...135 [kW] It represents the measured power. It refers to the active power read by the meter in the line manager. Depending on the system configuration this power can have different meanings (see table on the next page).	S		•
	P_REA (*)	Power reactive <i>Possible values:</i> 0...135 [kVAR] It represents the reactive quota of the measured power. It refers to the reactive power read by the meter in the line manager.	S		•
	TRG	Trigger for sending/receiving from/to bus	T	•	•

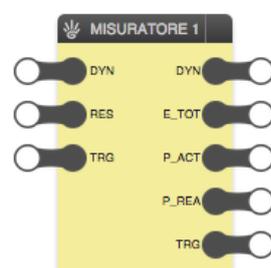
(*) Some nodes might not be present depending on the configuration of the system. In particular, the energy and power data are available only when an external meter is associated to the line.

Note 1: The P_ACT values depend on the type of system: max current supported by the cable on which the measurement is taken and the power output from the energy distributor. For example, in a domestic system with a standard Enel contract, this can reach 3.3kW.

Note 2: The P_REA values depend upon the inductive/capacitive absorption characteristics of the appliances in the system.

4.8.3 Meters

Like the previous block, this object represents part of the device 01455 linked to a single meter in the system.

Preview:					
					
Nodes:	TAG	Description	TYPE	IN	OUT
	DYN	Dynamic mode <i>Possible values:</i> 0 → OFF 1 → ON Dynamic Mode of meters. Present in the line manager if the system configuration allows a meter in the line manager index. This parameter is useful when in ON the active power measurement is sent to the device for a time equal to the parameter "Measurement refresh duration" and a frequency equal to the parameter "Measurement refresh frequency" (see for example Controller Installation Manual supplied with 21509 para. 3.4 section Energy Management). It can be used by a display device to show the value in real time, such as for example on opening a page in the touch screen; after the set time the transmission ends.	S	•	•
	RES	Partial reset <i>Possible values:</i> 0... 2147483648 [Wh] Used to set the partial energy measurement of the meter in the line manager index to a given value; it is present if the system configuration allows a meter in the line manager index. This parameter forces the partial energy value to the value set here and is useful for aligning the energy value calculated by the Vimar device to that of an external meter. The measurement MUST be of the same time, which depends on how the system has been configured and how the current sensors are positioned: exchanged (if for production), produced (photovoltaic meter) or consumer (without production).	S	•	
	E_TOT	Total energy <i>Possible values:</i> 0... 2147483648 [Wh] It represents the overall energy measured; for plants with production, it is the difference between energy drawn off (consumed) and fed in (produced)	S		•
	E_IN (*)	Energy drawn (from the mains) <i>Possible values:</i> 0... 2147483648 [Wh] It represents the total energy consumed, whatever the production	S		•
	E_OUT (*)	Energy fed (into the mains) <i>Possible values:</i> 0... 2147483648 [Wh] It represents the total energy produced (if production is available), whatever the consumption	S		•
	P_ACT	Power on <i>Possible values:</i> 0..135 [kW] It represents the measured power. It refers to the active power read by the meter in the line manager. Depending on the system configuration this power can have different meanings (see table below).	S		•
	P_REA	Power reactive <i>Possible values:</i> 0..135 [kVAR] It represents the reactive component of the measured power.	S		•
	TRG	Trigger for sending/receiving from/to bus	T	•	•

(*) Some nodes might not be available depending on the firmware version of the device

Note 1: The P_ACT values depend on the type of system: max current supported by the cable on which the measurement is taken and the power output from the energy distributor. For example, in a domestic system with a standard Enel contract, this can reach 3.3 kW.

Note 2: The P_REA values depend upon the inductive/capacitive absorption characteristics of the appliances in the system.

EXAMPLES OF USE OF THE ENERGY MANAGEMENT FUNCTIONS BY MEANS OF THE LOGIC UNIT

<p>Single-phase system with no generation</p> <p>For the diagram see page 262 of the Controller Installation Manual supplied with art. 21509</p>	<p>Load control module 01455</p> <p>BLOCK By-me Meter 1: Node P_ACT = Power Exchanged Note: > 0 = Drawn off; < 0 Fed in</p> <p>The By-me block Line 1 is not used for metering.</p>
<p>Single-phase system with “local” generation</p> <p>For the diagram see page 263 of the Controller Installation Manual supplied with art. 21509</p>	<p>Load control module 01455</p> <p>BLOCK By-me Meter 1: Node P_ACT = Power Exchanged Note: > 0 = Drawn off; < 0 Fed in</p> <p>BLOCK By-me Meter 2: Node P_ACT = Power Generated Note: must be ≥ 0</p> <p>The By-me block Line 1 and Line 2 are not used for metering.</p>
<p>Single-phase system with “remote” generation</p> <p>For the diagram see page 264 of the Controller Installation Manual supplied with art. 21509</p>	<p>Load control module 01455</p> <p>BLOCK By-me Line 1: Node P_ACT = Power Generated Note: must be ≥ 0</p> <p>BLOCK By-me Meter 1: Node P_ACT = Power Exchanged Note: > 0 = Drawn off; < 0 Fed in</p>
	<p>Power meter 01450</p> <p>BLOCK By-me Meter 1: Node P_ACT = Power Generated Note: must be ≥ 0</p> <p>This is the same as indicated above, use one block or the other according to the needs of the logic program</p>
<p>Three-phase system with no generation</p> <p>For the diagram see page 265 of the Controller Installation Manual supplied with art. 21509</p>	<p>Load control module 01455</p> <p>BLOCK By-me Meter 1: Node P_ACT = Power Exchanged Note: > 0 = Drawn off; < 0 Fed in</p> <p>The By-me block Line 1 is not used for metering.</p> <p>BLOCK By-me Meter 2: Node P_ACT = Power Exchanged Note: > 0 = Drawn off; < 0 Fed in</p> <p>The By-me block Line 2 is not used for metering.</p> <p>BLOCK By-me Meter 3: Node P_ACT = Power Exchanged Note: > 0 = Drawn off; < 0 Fed in</p> <p>The By-me block Line 3 is not used for metering.</p>
<p>Three-phase system with generation (from one to three phases)</p> <p>For the diagram see page 266 of the Controller Installation Manual supplied with art. 21509</p>	<p>Load control module 01455</p> <p>BLOCK By-me Line 1-2-3: Node P_ACT = Power Generated Note: must be ≥ 0</p> <p>BLOCK By-me Meter 1-2-3: Node P_ACT = Power Exchanged Note: > 0 = Drawn off; < 0 Fed in</p>
	<p>Power meter 01450</p> <p>BLOCK By-me Meter 1-2-3: Node P_ACT = Power Generated Note: must be ≥ 0</p> <p>This is the same as indicated above, use one block or the other according to the needs of the logic program</p>

4.8.4 Counters

Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	RES	Partial reset <i>Possible values: 0... 4294967296</i>	S	•	
	VAL	Counter (the description depends on the type of meter) <i>Possible values: 0... 4294967296</i>	S		•
	TRG	Trigger for sending/receiving from/to bus	T	•	•

Note: The number and the type of nodes may depend on the specific configuration of the project

4.9 Anti-intrusion alarm

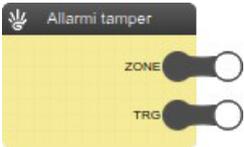
4.9.1 SAI activation events

Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	TRG	Trigger for receiving from bus Set to 1 whenever an activation event is received from the SAI system	T		•

4.9.2 SAI intrusion alarms

Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	ZONES	Alarm zone Identification <i>Possible values: 1... 31 with steps of 1</i>	S		•
	TRG	Trigger for receiving from bus Set to 1 whenever an intrusion alarm is received from the SAI system	T		•

4.9.3 SAI tamper alarms

Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	ZONES	Alarm zone Identification <i>Possible values: 1... 31 with steps of 1</i>	S		•
	TRG	Trigger for receiving from bus Set to 1 whenever a tamper alarm is received from the SAI system	T		•

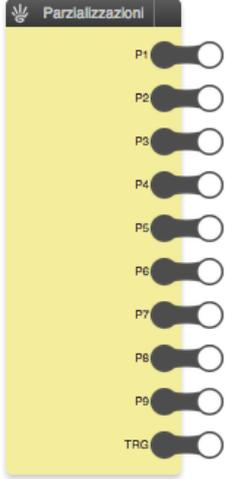
4.9.4 SAI technical alarms

Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	ZONES	Alarm zone Identification <i>Possible values:</i> 1... 31 with steps of 1	S		•
	TRG	Trigger for receiving from bus Set to 1 whenever a technical alarm is received from the SAI system	T		•

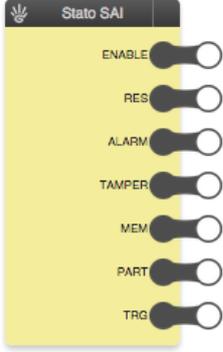
4.9.5 SAI troubleshooting

Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	ZONES	Alarm zone Identification <i>Possible values:</i> 1... 31 with steps of 1	S		•
	TRG	Trigger for receiving from bus Set to 1 whenever a fault is received (not included in the above-described types of events/alarms) by the SAI system	T		•

4.9.6 SAI partitioning

Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	P1 ... P9	State of partitioning 1 ... 9 <i>Possible values:</i> 0 → Off 1 → Active	S		•
	TRG	Trigger for sending/receiving from/to bus	T		•

4.9.7 SAI status

Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	ENABLE	System on <i>Possible values:</i> 0 → OFF 1 → ON	S		•
	RES	Siren reset When it goes to 1, the siren is switched off <i>Possible values:</i> 0 → OFF 1 → ON	S		•
	ALARM	System in alarm mode <i>Possible values:</i> 0 → OFF 1 → ON	S		•
	TAMPER	Tamper alarm <i>Possible values:</i> 0 → OFF 1 → ON	S		•
	MEM	Alarm memory <i>Possible values:</i> 0 → OFF 1 → ON	S		•
	PART	Partitioned This node is 1 when at least one partitioning is entered, but the system is not completely entered <i>Possible values:</i> 0 → OFF 1 → ON	S		•
	TRG	Trigger for sending/receiving from/to bus	T		•

4.10 Sensors

4.10.1 Display-only sensors

Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	VAL	Value <i>Possible values:</i> any numerical value	S		•
	ALARM	Alarm <i>Possible values:</i> 0 → No alarm 1 → Alarm	S		•
	TRG	Trigger for sending/receiving from/to bus	T		•

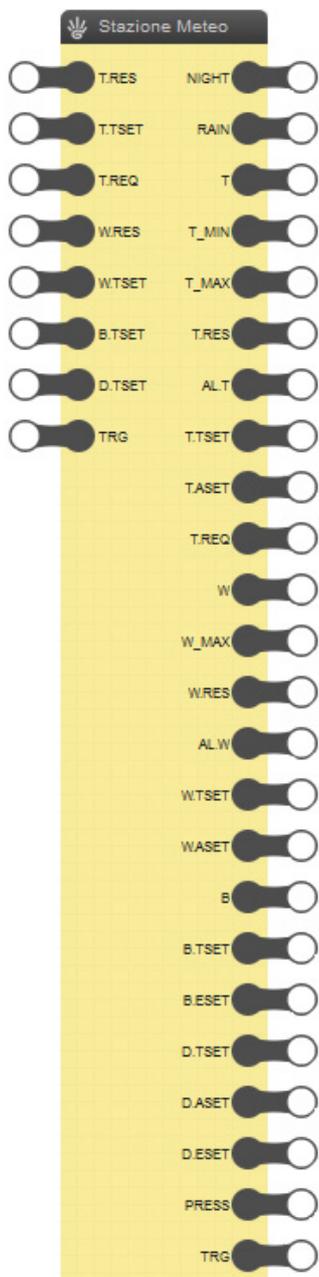
Note: The number of nodes and the corresponding data type might depend on the EASYTOOL PROFESSIONAL configuration

4.10.2 Sensors with control

Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	ENABLE	Sensor activation <i>Possible values:</i> 0 → OFF 1 → ON	S	•	
	SET	Setpoint Used to set the threshold above which the output associated to the sensor is enabled <i>Possible values:</i> any numerical value	S	•	
	TOG	ON/OFF <i>Possible values:</i> 0 → OFF 1 → ON	S		•
	VAL	Value <i>Possible values:</i> any numerical value	S		•
	TRG	Trigger for sending/receiving from/to bus	T		•

Note: The number of nodes and the corresponding data type might depend on the EASYTOOL PROFESSIONAL configuration

4.10.3 Weather station

Preview:					
					
Nodes:	TAG	Description	TYPE	IN	OUT
	NIGHT	Day/Night <i>Possible values:</i> 0 → day 1 → night	S		•
	RAIN	Rain/No rain <i>Possible values:</i> 0 → no rain 1 → rain	S		•
	T	Temperature measured <i>Possible values:</i> -273°C...670760.96°C	S		•
	T_MIN	Minimum temperature <i>Possible values:</i> -273°C...670760.96°C	S		•
	T_MAX	Maximum temperature <i>Possible values:</i> -273°C...670760.96°C	S		•
	T.RES	Reset temperature <i>Possible values:</i> -273°C...670760.96°C	S	•	•

Nodes:	TAG	Description	TYPE	IN	OUT
	AL.T	Temperature alarm <i>Possible values:</i> 0 → OFF 1 → ON	S		•
	T.TSET	Temperature target setpoint <i>Possible values:</i> -273°C...670760.96°C	S	•	•
	T.ASET	Temperature current setpoint <i>Possible values:</i> -273°C...670760.96°C	S		•
	T.REQ	Minimum/maximum temperature request <i>Possible values:</i> 0 → OFF 1 → ON	S	•	•
	W	Wind speed <i>Possible values:</i> 0 m/s...670760.96 m/s	S		•
	W_MAX	Maximum wind speed <i>Possible values:</i> 0 m/s...670760.96 m/s	S		•
	W.RES	Reset maximum wind speed <i>Possible values:</i> 0 → OFF 1 → ON	S	•	•
	AL.W	Wind speed alarm <i>Possible values:</i> 0 → OFF 1 → ON	S		•
	W.TSET	Wind speed target setpoint <i>Possible values:</i> 0 m/s...670760.96 m/s	S	•	•
	W.ASET	Wind speed current setpoint <i>Possible values:</i> 0 m/s...670760.96 m/s	S		•
	B	Brightness <i>Possible values:</i> 0 lx...670760.96 lx	S		•
	B.ESET	Brightness setpoint enabling <i>Possible values:</i> 0 → OFF 1 → ON	S	•	•
	B.TSET	Brightness target setpoint <i>Possible values:</i> 0 lx...670760.96 lx	S	•	•
	D.TSET	Dawn target setpoint <i>Possible values:</i> 0 lx...670760.96 lx	S	•	•
	D.ASET	Current dawn setpoint <i>Possible values:</i> 0 lx...670760.96 lx	S		•
	D.ESET	Dawn setpoint enabling <i>Possible values:</i> 0 → OFF 1 → ON	S		•
	PRESS	Pressure <i>Possible values:</i> 0 hPa...670760.96 hPa	S		•
	TRG	Trigger <i>Possible values:</i> 0 → OFF 1 → ON	T	•	•

Note: The number of nodes and the corresponding data type might depend on the EASYTOOL PROFESSIONAL configuration

4.11 KNX integration

Given the structural compatibility of By-me and KNX, you can create mixed systems.

Via the programming performed with EasyTool Professional (ver. 2.10 or later) you can export some control widgets for KNX device communication objects (configured via ETS) to By-me supervisors, such as web servers and touchscreens.

In addition, through the By-me logic unit, you can create integrated logic elements involving the devices of the two systems.

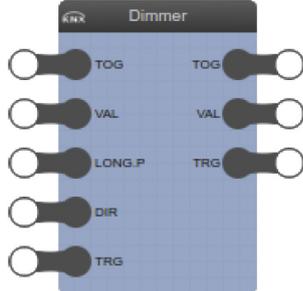
The following blocks illustrate how it is possible to achieve this integration by connecting nodes that allow sending/receiving commands or writing/reading values on datapoints shared between By-me and KNX installations.

4.11.1 Actuator

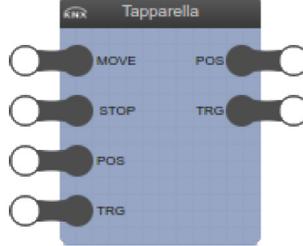
Nodes:	TAG	Description	TYPE	IN	OUT
	TOG	On/Off switch <i>Possible values:</i> 0 → OFF 1 → ON	S	•	•
	TRG	Trigger for sending/receiving from/to bus	T	•	•



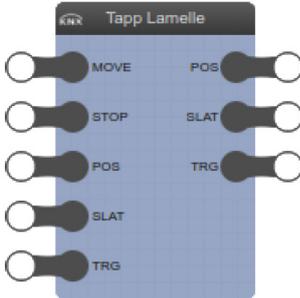
4.11.2 Dimmer

Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	TOG	On/Off switch <i>Possible values:</i> 0 → OFF 1 → ON		•	•
	VAL	Percentage adjustment <i>Possible values:</i> 0 ... 100 [%]	S	•	•
	LONG.P	Long press start/end <i>Possible values:</i> 0 → STOP long press end 1 → START long press start	S	•	•
	TRG	Trigger for sending/receiving from/to bus	T	•	•

4.11.3 Shutters up/down

Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	MOVE	Movement up/down <i>Possible values:</i> 0 → Up 1 → Down	S	•	
	STOP	Stopping the movement <i>Possible values:</i> 1 → Stop	S	•	
	POS	Percentage position <i>Possible values:</i> 0 ... 100 [%]	S	•	•
	TRG	Trigger for sending/receiving from/to bus	T	•	•

4.11.4 Roller blinds

Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	MOVE	Movement up/down <i>Possible values:</i> 0 → Up 1 → Down	S	•	
	STOP	Stopping the movement <i>Possible values:</i> 1 → Stop	S	•	
	STEP	Regulating slats up/down <i>Possible values:</i> 0 → Up 8 → Down	S	•	
	POS	Percentage position <i>Possible values:</i> 0 ... 100 [%]	S	•	•
	SLAT	Slat percentage position <i>Possible values:</i> 0 ... 100 [%]	S	•	•
	TRG	Trigger for sending/receiving from/to bus	T	•	•

4.11.5 Boolean Generic

Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	TOG	Boolean value <i>Possible values:</i> 0 → OFF 1 → ON	S	•	•
	TRG	Trigger for sending/receiving from/to bus	T	•	•

CAUTION: For the block to function correctly, it is necessary to set a default value that is consistent with the actual operation.

4.11.6 Numerical Generic

Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	VAL	Numerical value <i>Possible values:</i> Variable range depending on the DPT represented	S	•	•
	TRG	Trigger for sending/receiving from/to bus	T	•	•

CAUTION:

- For the block to function correctly, it is necessary to set a default value that is consistent with the actual operation.
- In order for the blocks "Generic Boolean" and "Generic numerical" to function correctly, the nodes must be connected to other nodes of the same type.

By-alarm

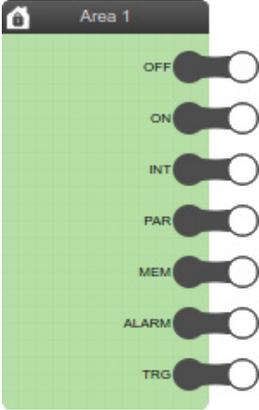
5. By-alarm

5.1 Introduction

The By-alarm blocks enable reading values from the intrusion detection alarm system and sending commands to the By-me groups after the logic processing carried out in the programs that contain them.

Caution: The By-alarm blocks are always visible even if not available in your system.

5.1.1 Area

Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	OFF	Area OFF <i>Possible values: 0...1</i>	S		•
	ON	Area on ON <i>Possible values: 0...1</i>	S		•
	INT	Area on INT <i>Possible values: 0...1</i>	S		•
	PAR	Area on PAR <i>Possible values: 0...1</i>	S		•
	MEM	Area in alarm memory <i>Possible values: 0...1</i>	S		•
	ALARM	Area in alarm mode <i>Possible values: 0...1</i>	S		•
	TRG	Trigger for sending/receiving from/to bus	T		•

Logic functions

6. Logic functions

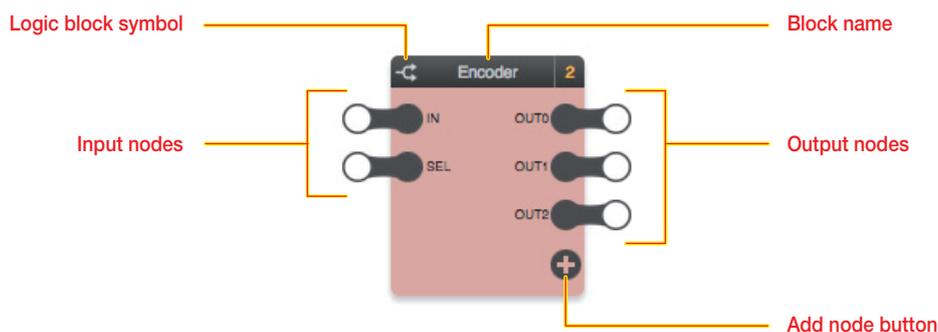
6.1 Introduction

The logic blocks enable performing operations on one or more input values and they return one or more output values, which can be connected with other logic blocks or By-me blocks.

6.2 Logic blocks

6.2.1 Layout

As mentioned earlier, the logic blocks appear graphically as shown in the following example:

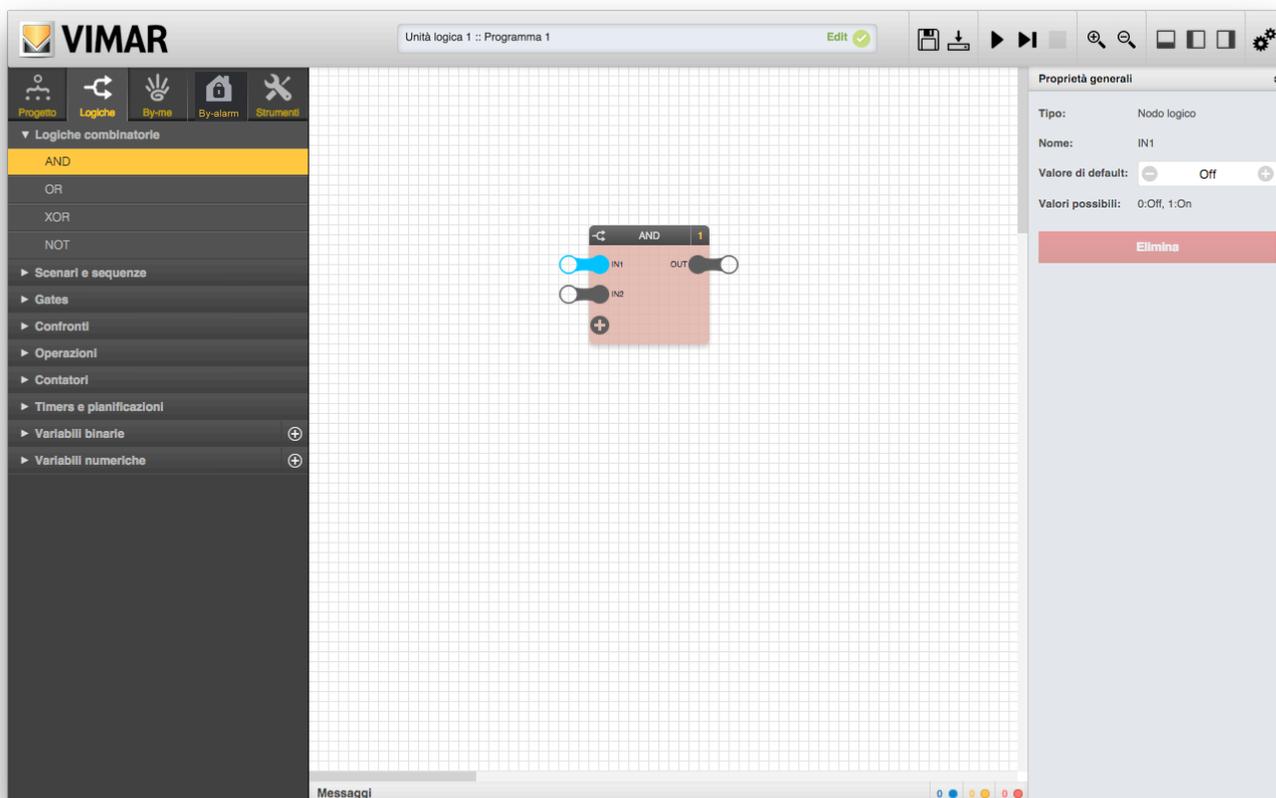


The logic blocks are characterized by an amber background.

6.2.2 Input nodes

The input nodes enable passing values to the logic functions. By selecting an input node and opening the details pane, you can set the following options:

Default value	This enables setting the value of the node to be used when starting to run, until a different value is received, or if the node is not connected to another block
----------------------	---



The details pane, in addition to the above options, also shows the possible values which the node can take; in the case of binary nodes the possible values are 0 (OFF) or 1 (ON), in the opposite case of numerical nodes the possible values depend on the type of node, and may have specific restrictions.

Logic functions

6.2.3 Output nodes

The output nodes return the results of the logic function paired with the block and enable passing them to other blocks, of both the logic and By-me type. There is no option for the output nodes of logic blocks.

6.2.4 Adding and removing nodes

Some blocks have a variable number of nodes; in these cases, the block, once dragged from the side menu, typically contains a minimal set of nodes, which may be increased up to a maximum number of nodes, by pressing the "+" button.

To remove a previously added node, perform the following steps:

- Select the node
- Open the details pane
- Press the "DELETE" button

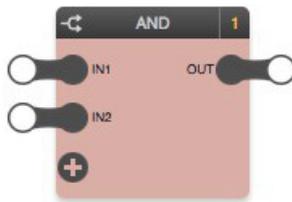
Any connections associated with the node will be deleted.

6.2.5 Types of blocks and nodes

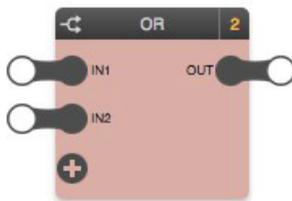
In some cases Logic Blocks (or nodes of Logic Blocks) are separated into "binary" and "numerical". The former are designed to manage boolean signals, i.e. can only have True/False (or "ON/OFF") values. The latter can hand numerical data. The Editor controls the correspondence of these types and prevents connections between nodes of different types.

6.3 Combinational logic gates

6.3.1 AND

Description:	Performs the AND logic function between two or more binary inputs (up to a maximum of 10)				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	IN1 ... IN10	Input 1 ... 10	Possible values: 0 → OFF 1 → ON	M	•
	OUT	Output	Possible values: 0 → OFF 1 → ON	M	•
	+	Add node		•	

6.3.2 OR

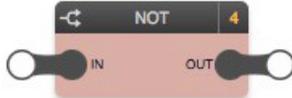
Description:	Performs the OR logic function between two or more binary inputs (up to a maximum of 10)				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	IN1 ... IN10	Input 1 ... 10	Possible values: 0 → OFF 1 → ON	M	•
	OUT	Output	Possible values: 0 → OFF 1 → ON	M	•
	+	Add node		•	

Logic functions

6.3.3 XOR

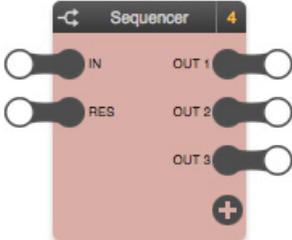
Description:	Performs the XOR logic function between two or more binary inputs (up to a maximum of 10)				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	IN1 ... IN10	Input 1 ... 10 <i>Possible values:</i> 0 → OFF 1 → ON	M	•	
	OUT	Output <i>Possible values:</i> 0 → OFF 1 → ON	M		•
	+	Add node		•	

6.3.4 NOT

Description:	Performs the NOT logic function of the input				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	IN	Input <i>Possible values:</i> 0 → OFF 1 → ON	M	•	
	OUT	Output <i>Possible values:</i> 0 → OFF 1 → ON	M		•

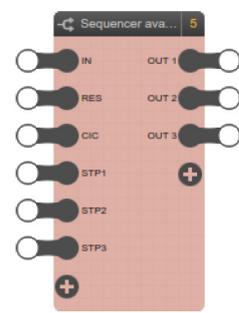
6.4 Scenarios and sequences

6.4.1 Sequencer

Description:	According to the input status IN, it toggles in sequence up to 10 boolean outputs, keeping each of them active for a settable time				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	IN	Start sequence <i>Possible values:</i> 0 → OFF 1 → ON	M	•	
	RES	Reset sequence <i>Possible values:</i> 0 → OFF 1 → ON	M	•	
	OUT1 ... OUT10	Output 1... 10 <i>Possible values:</i> 0 → OFF 1 → ON	M		•
	+	Add node			•
Options:	Cyclical sequence	Determines whether the sequence should be repeated once completed <i>Possible values:</i> TRUE/FALSE			
	Step duration 1 ... 10	Waiting time between step X and the next <i>Possible values:</i> from 1 second to 12 hours The step is 1 second and can be specified in the format HH:MM:SS (hours, minutes, seconds)			

Logic functions

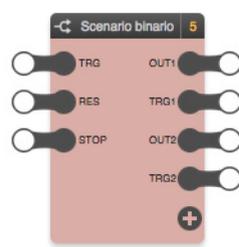
6.4.2 Advanced Sequencer

Description:	The function is similar to that of the Sequencer with the difference that some options have been converted into nodes so that they can be changed dynamically via the virtual datapoints				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	IN	Start sequence <i>Possible values:</i> 0 → OFF 1 → ON	M	•	
	RES	Reset sequence <i>Possible values:</i> 0 → OFF 1 → ON	M	•	
	CIC	Cyclic sequence <i>Possible values:</i> 0 → OFF 1 → ON	S	•	
	STP1...STP10	Step duration 1 ... 10 <i>Possible values:</i> any	S	•	
	OUT1...OUT10	Output 1 ... 10 <i>Possible values:</i> any	M		•

CAUTION:

The values to provide to the Advanced Sequencer logic unit to define the enabling time of each output it consists of (via nodes STP1.. STP10) must always be expressed in seconds.

6.4.3 Binary scenario

Description:	When an impulse is received on the TRG input, it runs a sequence of boolean commands, each of which can be set, possibly spacing each command with a preset time, common to all the outputs				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	TRG	Input trigger <i>Possible values:</i> 0 → OFF 1 → ON	T	•	
	RES	Reset scenario Returns all the scenario outputs to the initial status (default). <i>Possible values:</i> 0 → OFF 1 → ON	M	•	
	STOP	Stop scenario If on, the scenario is stopped. The scenario is resumed when the STOP signal is disabled (useful above all when setting time > 0 as an interval between the activation of the scenario outputs). <i>Possible values:</i> 0 → OFF 1 → ON	M	•	
	OUT1 ... OUT10	Output 1 ... 10 <i>Possible values:</i> 0 → OFF 1 → ON	S		•
	TRG1 ... TRG10	Trigger 1 ... 10 <i>Possible values:</i> 0 → OFF 1 → ON	T		•
	+	Add node (and its trigger)			•
Options:	Output interval	Waiting time between output command <i>Possible values:</i> 1 ... 60 (seconds)			
	Set output 1 ... 10	Value to set for output 1 ... 10 <i>Possible values:</i> 0 → False (OFF) 1 → True (ON)			

Logic functions

6.4.4 Numerical scenario

Description:	When an impulse is received on the TRG input, it runs a sequence of numerical commands, each of which can be set, possibly spacing each command with a preset time, common to all the outputs				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	TRG	Input trigger <i>Possible values:</i> 0 → OFF 1 → ON	T	•	
	RES	Reset scenario Returns all the scenario outputs to the initial status (default). <i>Possible values:</i> 0 → OFF 1 → ON	M	•	
	STOP	Stop scenario If on, the scenario is stopped. The scenario is resumed when the STOP signal is disabled (useful above all when setting time > 0 as an interval between the activation of the scenario outputs). <i>Possible values:</i> 0 → OFF 1 → ON	M	•	
	OUT1 ... OUT10	Output 1... 10 <i>Possible values:</i> any numerical value	S		•
	TRG1 ... TRG10	Trigger 1 ... 10 <i>Possible values:</i> 0 → OFF 1 → ON	T		•
	+	Add node + trigger			•
Options:	Output interval	Waiting time between output command <i>Possible values:</i> 1 ... 60 (seconds)			
	Set output 1 ... 10	Value to be set on output 1 ... 10 <i>For outputs from 1 to 10 it is possible to set any value.</i>			

6.5 Gates

6.5.1 Binary selector

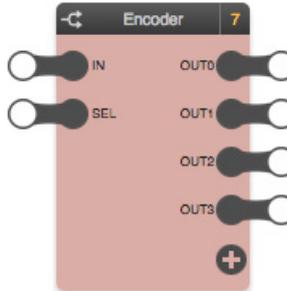
Description:	Returns the value of one of the inputs according to the value of the SEL input which acts as a selector. If SEL=False → OUT=IN0 If SEL=True → OUT=IN1				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	SEL	Input selector <i>Possible values:</i> 0 → OFF (in this case OUT = IN0) 1 → ON (in this case OUT = IN1)	M	•	
	IN0 IN1	Input 0, input 1 <i>Possible values:</i> 0 → OFF 1 → ON	M	•	
	OUT	Output <i>Possible values:</i> 0 → OFF 1 → ON	M		•

Note: This block performs a similar function to that of a "binary" decoder.

6.5.2 Numerical selector

Description:	Restores the value of one of the inputs according to the value of the SEL input which acts as a selector If SEL=False → OUT=IN0 If SEL=True → OUT=IN1				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	SEL	Input selector <i>Possible values:</i> 0 → OFF (in this case OUT = IN0) 1 → ON (in this case OUT = IN1)	M	•	
	IN0 IN1	Input 0, input 1 <i>Possible values:</i> any numerical value	M	•	
	OUT	Output <i>Possible values:</i> any numerical value	M		•

6.5.3 Encoder

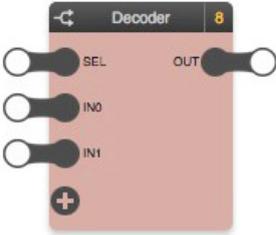
Description:	Sets the value of IN at one of its outputs according to the value of the SEL input which acts as a selector <i>Number of outputs: from 2 to 10</i>				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	IN	Input <i>Possible values:</i> 0 → OFF 1 → ON	S	•	
	SEL	Output selector <i>Possible values:</i> 1 ... 10	S	•	
	OUT0 ... OUT9	Output 0... 9 <i>Possible values:</i> 0 → OFF 1 → ON	S		•
	+	Add output			•

Example:

OUT nodes such as enable can be used for logic networks according to the SEL value.

Logic functions

6.5.4 Decoder

Description:	Restores in output the value of one of the inputs according to the input value of the SEL which acts as a selector				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	SEL	Input selector <i>Possible values: 0 ... 9</i>	S	•	
	IN0 ... IN9	Input 0 ... 9 <i>Possible values: 0 → OFF 1 → ON</i>	S	•	
	OUT	Output <i>Possible values: 0 → OFF 1 → ON</i>	S		•
	+	Add input		•	

Example:

The OUT node can be used to control, or not, an actuator using IN nodes, according to the SEL value

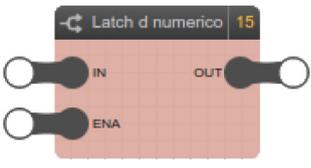
Note: The binary decoder function, i.e. a decoder with 2 binary inputs and binary selection node, is accomplished by the "Binary Switch" logic block.

6.5.5 Binary Latch D

Description:	In this block the input signal IN is propagated to the output OUT if the enable ENA signal is enabled (1). If the enable ENA signal is disabled, the last state remains at the output OUT. When enable ENA is again active then (transition 0--> 1) the last value read in the input node IN is sent to the output OUT. Basically, with ENA=0, the Latch block saves the last value read to send it to the output at the moment when ENA is again enabled. The IN and OUT data format is binary.				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	IN	Input <i>Possible values: 0 ... 1</i>	M	•	
	ENA	Enable <i>Possible values: 0 ... 1</i>	M	•	
	OUT	Output <i>Possible values: 0 ... 1</i>	M		•

Logic functions

6.5.6 Numerical Latch D

Description:	<p>In this block the input signal IN is propagated to the output OUT if the enable ENA signal is enabled (1). If the enable ENA signal is disabled, the last state remains at the output OUT.</p> <p>When enable ENA is again active then (transition 0--> 1) the last value read in the input node IN is sent to the output OUT.</p> <p>Basically, with ENA=0, the Latch block saves the last value read to send it to the output at the moment when ENA is again enabled. The IN and OUT data format is numerical.</p>				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	IN	Input <i>Possible values:</i> any numerical value	M	•	
	ENA	Enable <i>Possible values:</i> 0 ... 1	M	•	
	OUT	Output <i>Possible values:</i> any numerical value	M		•

6.5.7 Flip-flop T

Description:	<p>T flip-flop</p> <p>Works as a step-step relay. Whenever a rising edge is proposed to the input (TRG) the output (OUT) changes status. If the LCK input (Block) is 1 (True) the effect of the TRG is inhibited and the output never changes. If the input PRT (Priority) is 1 the output takes on the value set in the parameter "Priority value".</p> <p>It may be used for example to control the lights in a corridor. The light can be controlled normally only if a brightness threshold is satisfied (this condition would end up in the LCK) and if it is in any case always on at night (Flag connected to the PRT input)</p>				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	TRG	Trigger <i>Possible values:</i> 0 → OFF 1 → ON	T	•	
	LCK	Locks the current state <i>Possible values:</i> 0 → OFF 1 → ON	M	•	
	PRT	Priority flag <i>Possible values:</i> 0 → OFF 1 → ON	M	•	
	OUT	Output signal <i>Possible values:</i> 0 → OFF 1 → ON	S		•
Options:	Priority value	Value to assign to the output for a priority flag	<i>Possible values:</i> TRUE/FALSE		

Truth table:

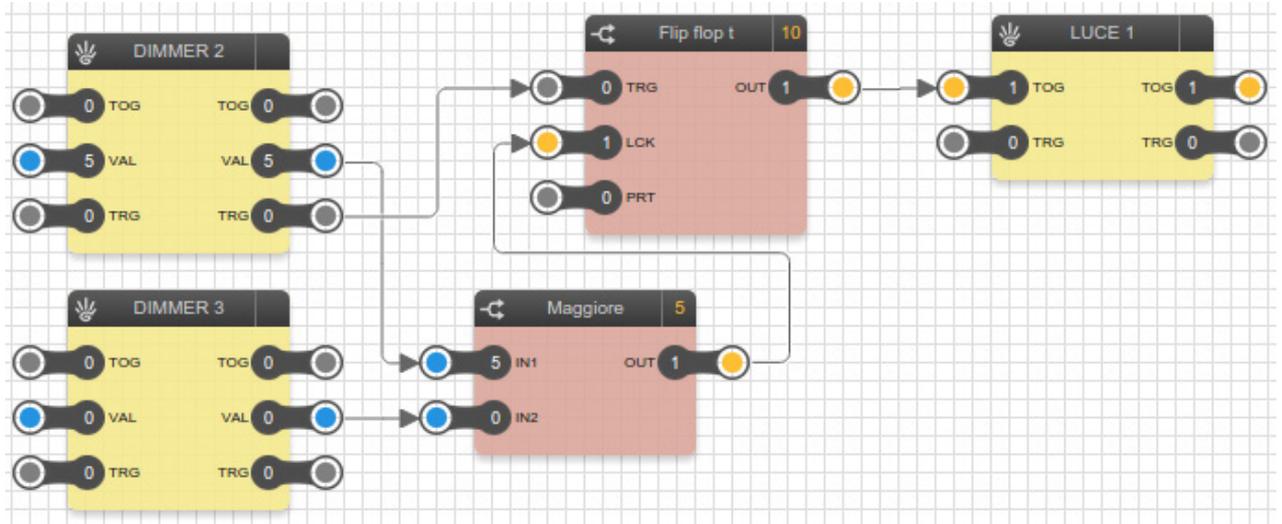
	TRG	OUT
With LCK=0	0>1	NOT OUT
With LCK=1	0>1	Does not change

Note: See also the Priority flag parameter.

Logic functions

Example:

A light is controlled via a Flip Flop T which measures the change in status of a dimmer and stops it from being switched on if the value of the first dimmer is greater than the second.



6.5.8 Flip-flop RS

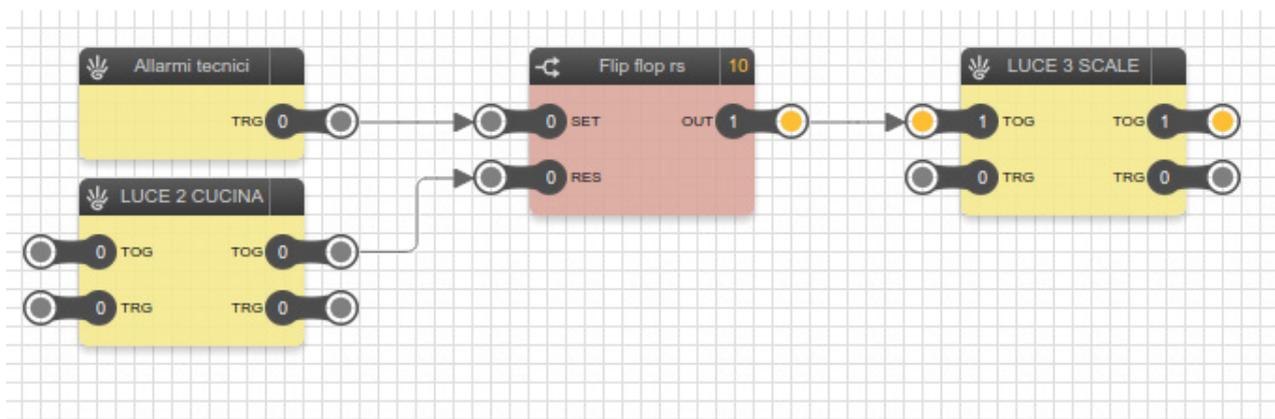
Description:	RS flip-flop An elementary memory block which is "loaded" with the SET input and reset with the RES input (reset). If both inputs are 1 the one specified by the parameter "Selection priority" will prevail. It may for example be used to manage an alarm signal. An alarm contact is connected to the SET. When it has gone to 1 the Flip-Flop maintains the output on 1 until it has been reset by the RES. In this way even if the alarm is "normalised" (moving to 0) the information is maintained.				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	SET	Set <i>Possible values:</i> 0 → OFF 1 → ON	M	•	
	RES	Reset <i>Possible values:</i> 0 → OFF 1 → ON	S	•	
	OUT	Output signal <i>Possible values:</i> 0 → OFF 1 → ON	M		•
Options:	Selection priority	<i>Possible values:</i> Set / Reset			

Truth table:

S	R	OUT
0	0	Does not change
0	1	0
1	0	1
1	1	See selection priority parameter

Example:

A light is switched on if there is an alarm signal. The light connected to the RES input is used to reset the Flip Flop RS status.



6.5.9 Flip-flop D

Description:	Flip-flop type D Operation is similar to that of Latch D with the difference that the Flip-Flop D operates on the change on the edge of CLK. The DAT data is reported to OUT only on the rising edge of the CLK signal and maintains it until the next rising edge of CLK (this block is basically a memory cell).				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	DAT	Data <i>Possible values: 0 ... 1</i>	M	•	
	CLK	Clock <i>Possible values: 0 ... 1</i>	T	•	
	OUT	Output <i>Possible values: 0 ... 1</i>	M		•

6.5.10 Binary encoder

Description:	Sets one of the two outputs to the input value IN according to the value of the input SEL, which acts as a selector				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	IN	Input <i>Possible values: 0 → OFF 1 → ON</i>	S	•	
	SEL	Output selector <i>Possible values: 0 → OFF 1 → ON</i>	S	•	
	OUT0 OUT1	Output 1 (if selector 0) Output 2 (if selector 1) <i>Possible values: 0 → OFF 1 → ON</i>	S		•

Logic functions

6.6 Comparisons

6.6.1 Comparison operators

Description:	<p>Compares the value of the two inputs, and gives as output a TRUE/FALSE value according to the specified operator</p> <p style="text-align: right;"><i>Available operators:</i></p> <ul style="list-style-type: none"> • Greater than • Greater than or equal to • Less than • Less than or equal to • Equal to • Different 				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	IN1 IN2	Input 1, input 2 <i>Possible values:</i> any numerical value	S	•	
	OUT	Result of comparison <i>Possible values:</i> 0 → OFF 1 → ON	S		•

6.7 Operations

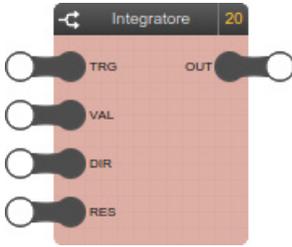
6.7.1 Mathematical operators

Description:	<p>Performs a mathematical operation on the inputs, according to the type of operator</p> <p style="text-align: right;"><i>Available operators:</i></p> <ul style="list-style-type: none"> • Maximum • Minimum • Average • Addition • Subtraction • Multiplication • Division • Absolute value • Log10 				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	IN1 IN2 (*) ...	Input 1, input 2 ... <i>Possible values:</i> any numerical value	S	•	
	OUT	Value (result of the operation) <i>Possible values:</i> any numerical value	S		•

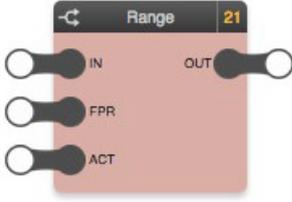
(*) The number of outputs may be limited depending on the operation (e.g., division max 2, absolute value max 1)

Logic functions

6.7.2 Integrator

Description:	<p>A logic block which performs an integration function on a numerical value. For each edge on the TRG node it adds the content of the input VAL node (positive or negative).</p> <p>The output numerical node VAL carries the integrated value. A DIR node will also be added that adds or subtracts the input value VAL (to be considered as a value with a sign)</p>				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	TRG	Trigger <i>Possible values:</i> 0 → OFF 1 → ON	T	•	
	VAL	Value <i>Possible values:</i> any numerical value	S	•	
	DIR	Counter direction <i>Possible values:</i> 0 → ADDITION 1 → SUBTRACTION	M	•	
	RES	Reset <i>Possible values:</i> 0 → OFF 1 → ON (resets the counter)	M	•	
	OUT	Value (result of the operation) <i>Possible values:</i> any numerical value	S		•

6.7.3 Ranges

Description:	<p>Performs linear interpolation of the input value IN according to the assigned mapping, also known as "characteristic", defined by two pairs of values (X,Y). The value IN is a ratio of between X0 and X1, and this ratio is in turn calculated between Y0 and Y1 to determine the output value.</p> <p>If priority mode is set, a predetermined value is returned.</p> <p>The typical field of application of this block is the conversion of values between different variables.</p>				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	IN	Input <i>Possible values:</i> any numerical value	M	•	
	FPR	Priority enabling <i>Possible values:</i> 0 → No priority 1 → Priority (the priority value is returned)	M	•	
	ACT	Direct/reverse operation <i>Possible values:</i> 0 → Direct operation 1 → Inverse operating	M	•	
	OUT	Output <i>Possible values:</i> any numerical value	M		•
Options:	X0 Y0 X1 Y1	Characteristics of the linear interpolation <i>Possible values:</i> any numerical value			
	Priority value	Value to be returned in case of priority enabling			

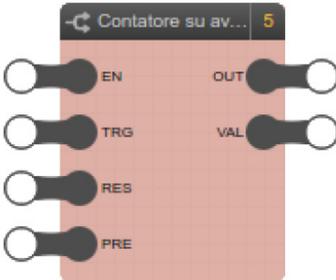
Logic functions

6.8 Counters

6.8.1 Up counter, down counter

Description:	Counts the number of pulses received at the input (trigger), increasing or decreasing its value each time (depending on the type of counter). <i>Types of counter:</i> up counter, down counter.				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	EN	Enable <i>Possible values:</i> 0 → Not enabled 1 → Enabled	S	•	
	TRG	Trigger <i>Possible values:</i> 0 → OFF 1 → ON (the counter is incremented)	T	•	
	RES	Reset <i>Possible values:</i> 0 → OFF 1 → ON (resets the counter)	M	•	
	OUT	Output <i>Possible values:</i> 0 → OFF 1 → ON	M		•
	VAL	Current value <i>Possible values:</i> any numerical value	S		•
Options:	Preset	Default value, set when resetting or starting the logic element. In the case of a counter UP the count starts from 0 and must reach Preset to enable OUT, while in Counter DOWN the count starts from Preset and must reach 0 to enable OUT. <i>Possible values:</i> any numerical value			

6.8.2 Advanced UP/Advanced DOWN Counter

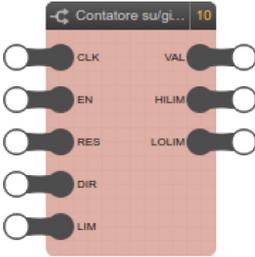
Description:	The function is similar to that of the UP Counter, DOWN Counter with the difference that some options have been converted into nodes so that they can be changed dynamically via the virtual datapoints				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	EN	Enable <i>Possible values:</i> 0 → OFF 1 → ON	S	•	
	TRG	Trigger <i>Possible values:</i> 0 → OFF 1 → ON	T	•	
	RES	Reset <i>Possible values:</i> 0 → OFF 1 → ON (resets the counter)	M	•	
	PRE	Preset <i>Possible values:</i> 0 ... 32767	S	•	
	OUT	Output <i>Possible values:</i> 0 → OFF 1 → ON	M		•
	VAL	Current value <i>Possible values:</i> any numerical value	S		•

Logic functions

6.8.3 UP/DOWN counter

Description:	Counts the number of pulses received at the input (trigger), increasing or decreasing its value each time (depending on the type of counter). <i>Types of counter:</i> up counter, down counter.				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	CLK	Clock <i>Possible values:</i> 0 → OFF 1 → ON (the counter is incremented)	T	•	
	EN	Enable <i>Possible values:</i> 0 → Not enabled 1 → Enabled	S	•	
	RES	Reset <i>Possible values:</i> 0 → OFF 1 → ON (resets the counter)	M	•	
	DIR	Counter direction <i>Possible values:</i> 0 → UP 1 → DOWN	S	•	
	VAL	Current value <i>Possible values:</i> any numerical value	S		•
	HILIM	High limit reached. Trigger signal	T		•
	LOLIM	Low limit reached. Trigger signal	T		•
Options:	Maximum limit	Default value, set when resetting or starting the logic element. In the case of an UP counter the count starts from 0 and must reach Maximum Limit to enable OUT, while in a DOWN counter the count starts from Maximum Limit and must reach 0 to enable OUT. <i>Possible values:</i> 32767			

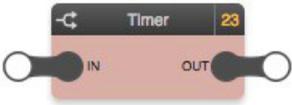
6.8.4 Advanced UP/DOWN counter

Description:	The function is similar to that of the UP/DOWN Counter with the difference that some options have been converted into nodes so that they can be changed dynamically via the virtual datapoints				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	CLK	Clock <i>Possible values:</i> 0 → OFF 1 → ON	T	•	
	EN	Enable <i>Possible values:</i> 0 → OFF 1 → ON	S	•	
	RES	Reset <i>Possible values:</i> 0 → OFF 1 → ON (resets the counter)	M	•	
	DIR	Counter direction <i>Possible values:</i> 0 → UP 1 → DOWN	S	•	
	LIM	Maximum limit <i>Possible values:</i> any numerical value	S	•	
	VAL	Current value <i>Possible values:</i> any numerical value	S		•
	HILIM	High limit reached <i>Possible values:</i> 0 → OFF 1 → ON	T		•
	LOLIM	Low limit reached <i>Possible values:</i> 0 → OFF 1 → ON	T		•

Logic functions

6.9 Timers and planning

6.9.1 Timer

Description:	Delays the value received at the input by a preset time When a 1 is received in the input IN (rising edge), an internal counter starts until the time specified as "rising delay", after which the output is moved to 1; vice versa, on receipt of an input 0 (falling edge), the block waits for the time set as "falling delay" before setting the output to 0.				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	IN	ON/OFF signal at input <i>Possible values:</i> 0 → OFF 1 → ON	M	•	
	OUT	ON/OFF signal at output, delayed by the timer <i>Possible values:</i> 0 → OFF 1 → ON	M		•
Options:	Rising delay	Delay in propagating the rising received in input <i>Possible values:</i> from 1 second to 12 hours in steps of 1 second.			
	Falling delay	Delay in propagating the falling edge received in input <i>Possible values:</i> from 1 second to 12 hours in steps of 1 second.			

6.9.2 Advanced Timer

Description:	The function is similar to that of the Timer with the difference that some options have been converted into nodes so that they can be changed dynamically via the virtual datapoints				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	IN	Start sequence <i>Possible values:</i> 0 → OFF 1 → ON	M	•	
	D.UP	Up delay <i>Possible values:</i> any numerical value	S	•	
	D.DWN	Down delay <i>Possible values:</i> any numerical value	S	•	
	OUT	Output delayed by timer <i>Possible values:</i> 0 → OFF 1 → ON	M		•

CAUTION:

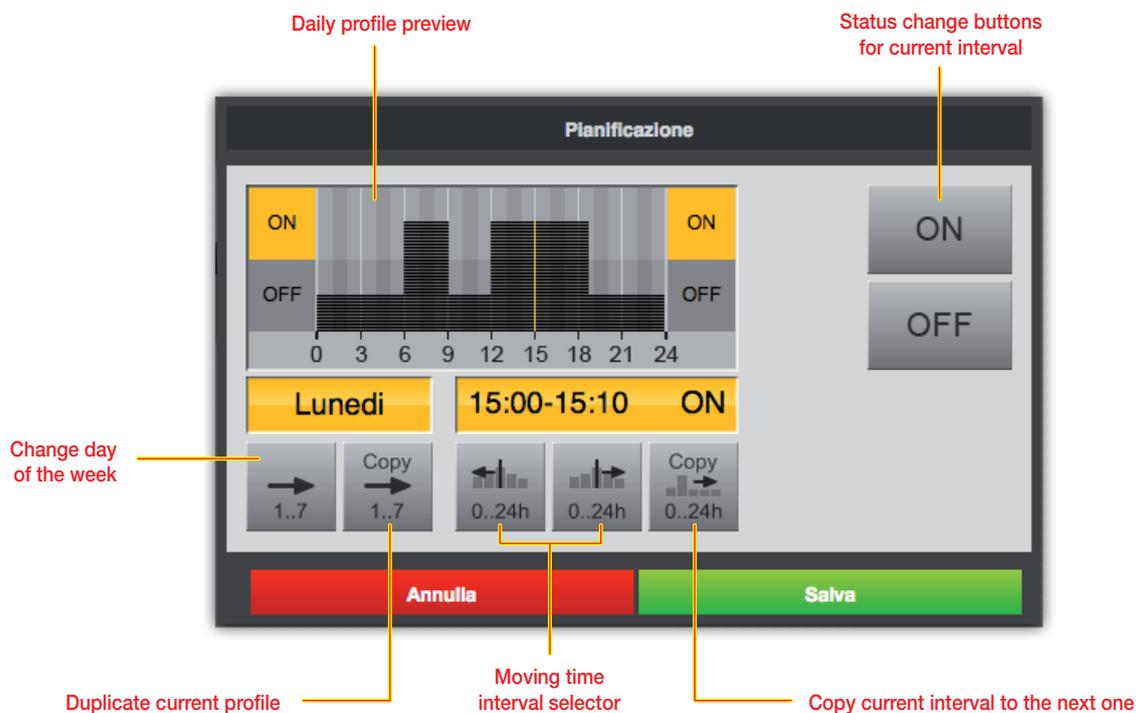
The value to provide to the Advanced Timer logic unit to define the up or down delay to apply (via nodes D.UP and D.DWN) must always be expressed in seconds.

Logic functions

6.9.3 Weekly timer

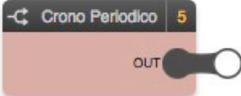
Description:	Used to set weekly planning The block has value 1 or 0 depending on the time and day of the week, according to the programming set in the editor or by the end user via the webserver or touch devices				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	OUT	Planning status <i>Possible values: 0 → OFF 1 → ON</i>	S		•
Options:	Plan	Planning button, used to open the planning pop-up to set when the output has to be set to ON			

Clicking the PLAN button opens the pop-up that is used to establish, for each day of the week, at what times the lock should be ON, in steps of 10 minutes:

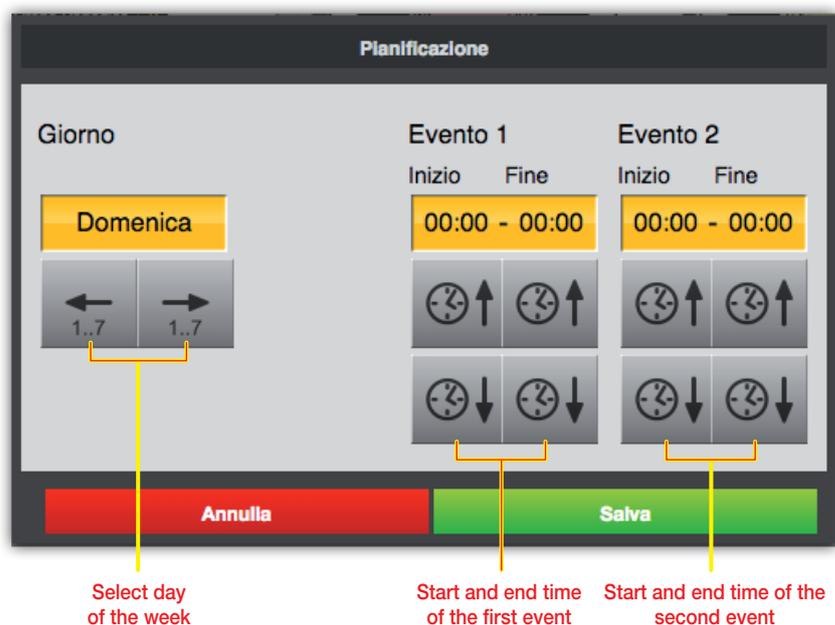


Logic functions

6.9.4 Periodic timer

Description:	Used to set periodic planning, consisting of one or two intervals for each day of the week The block has value 1 or 0 depending on the time and day of the week, according to the programming set in the editor or by the end user via the webserver or touch devices				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	OUT	Planning status <i>Possible values: 0 → OFF 1 → ON</i>	S		•
Options:	Plan	Planning button, used to open the planning pop-up to set when the output has to be set to ON			

Clicking the PLAN button opens the pop-up that is used to establish, for each day of the week, one or two intervals when the planning is active:

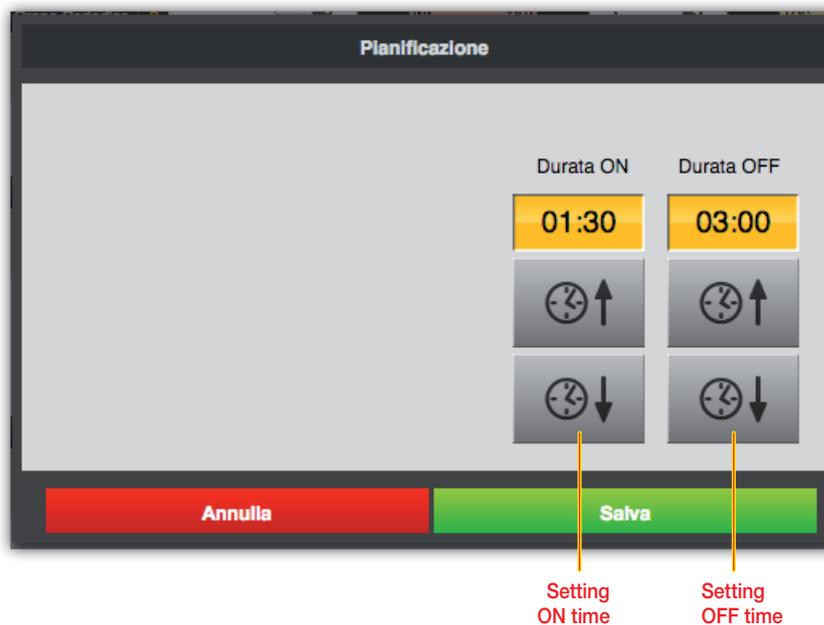


In addition to the increase and decrease buttons, you can also change the times by clicking on the time indicators: a pop-up prompts you to enter the start or end time of the event.

6.9.5 Cyclic timer

Description:	Used to set cyclic planning, based on an ON time and an OFF time The block has value 1 or 0 depending on the duration of the cycle set in the editor or by the end user via the webserver or touch devices				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	OUT	Planning status <i>Possible values: 0 → OFF 1 → ON</i>	S		•
Options:	Plan	Planning button; used to open the planning pop-up to set the ON and OFF times			

Clicking the PLAN button opens the pop-up which is used to determine the ON and OFF time:



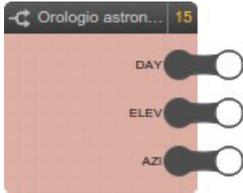
As in the case of periodic programs, in this case, too, clicking on the time indicators will open a pop-up where you can enter the ON/OFF time, instead of increasing/decreasing it with the buttons.

Logic functions

6.9.6 Trigger

Description:	Generates a trigger (pulse of the duration of one cycle) on an edge detected at the input When it receives a 1 on input or a 0 (depending on the value set in the parameter "Edge"), the output set to 1 for the duration of a single processing cycle, then the output is once again set to 0. In this way it is possible to generate an "impulse" for logic blocks which require one (e.g. scenarios, sequencer etc.) on the input rising edge.				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	IN	Input edge	T	•	
	OUT	Pulse of the duration of one cycle. The logic is run repeatedly over time, the impulse generated by the trigger lasts only for one run cycle, on the next step if a new edge is not measured on input no impulse will be generated.	T		•
Options:	Front view	Rising or falling edge to be detected at the input			

6.9.7 Astronomical clock

Description:	Returns the status of day/night, the degree of elevation of the sun and the angle to the North according to the current date/time and the coordinates given via parameters.				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	DAY	Day <i>Possible values:</i> 0 → Night 1 → Day	S		•
	ELEV	Elevation of the sun Returns the height of the sun over the horizon. The value 0° indicates the horizon. Positive values indicate day, negative ones night. <i>Possible values:</i> -90° ... +90°	S		•
	AZI	Azimuth Returns the angular distance of the sun from the North. The value 0° indicates the North, 90° East, 180° South and 270° West. <i>Possible values:</i> 0° ... 360°	S		•
Options:	Latitude	Latitude (-90 / 90) with up to 7 decimal digits. Latitude coordinate example: Rome 41.9100711			
	Longitude	Longitude (-180 / 180) with up to 7 decimal digits. Longitude coordinate example: Rome 12.5359979			
	Threshold	Threshold related to the sun's elevation to determine the day/night output (default 0°, in this case when the sun passes the horizon the DAY output will be day)			
	Time zone	Can be selected in the drop down menu			
	Daylight saving time change	<i>Possible values:</i> disabled, automatic, and manual. If manual is possible, set the standard time/daylight saving time.			

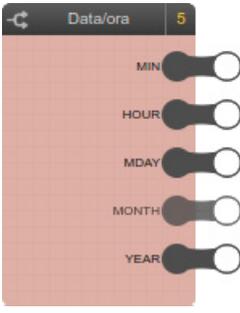
CAUTION: This logic block needs the system clock in the home automation system.

If the system clock is configured to automatically send the daylight saving time (as for example in the control panel 21509), it is necessary to set the "Change daylight saving time" parameter in the device.

It is possible to simulate the Astronomical Clock block changing the date/time settings on the advanced options menu.

For a correct simulation, the "Change daylight saving time" parameter must be set to automatic; at the end of the simulation, then restore this parameter to disabled (according to the previous note).

6.9.8 Date/Time

Description:	Returns the current time of the By-me system.				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	MIN	Minutes <i>Possible values: 0...59</i>	S		•
	HOUR	Hours <i>Possible values: 0...23</i>	S		•
	MDAY	Day of the month <i>Possible values: 1...31</i>	S		•
	MONTH	Month <i>Possible values: 1...12</i>	S		•
	YEAR	Year <i>Possible values: 2015...2099</i>	S		•

6.9.9 Command repetition

Description:	It features two trigger nodes; when a trigger reaches the TRG input it generates N+1 triggers on the output TRG node that are distanced from each other by a configurable parameter.				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	TRG	Trigger for sending/receiving from/to bus	T	•	•
	TRG	Trigger for sending/receiving from/to bus	T	•	•
Options:	Interval (s)	Time in seconds between the generation of one output trigger and the next one			
	Repetitions	Number of triggers to be generated after the first one			

Logic functions

6.10 Variables

6.10.1 Introduction

As explained in section 3.11, the variables enable passing values between different programs. The variables must first be created with the "+" button in the special section of the main menu, then they can be dragged into the programs that need to use them

6.10.2 Binary variables

Description:	Used to transfer a boolean value between different programs.				
Category:	Binary variables				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	VAL	Value to assign to the variable <i>Possible values:</i> 0 → OFF 1 → ON	M	•	
	VAL	Current value of the variable <i>Possible values:</i> 0 → OFF 1 → ON	M		•

6.10.3 Numerical variables

Description:	Used to transfer a boolean value between different programs.				
Category:	Binary variables				
Preview:					
Nodes:	TAG	Description	TYPE	IN	OUT
	VAL	Value to assign to the variable <i>Possible values:</i> any numerical value	M	•	
	VAL	Current value of the variable <i>Possible values:</i> any numerical value	M		•

Simulation

7. Simulation

7.1 Introduction

After creating a logic program, you can simulate its operation in the editor, manually entering the state of the inputs and verifying the output processing in real time, also with the logic blocks that entail changing the outputs over time.

7.2 Types of simulation

Two different types of simulation are available:

- **Simulazione continua:** each program cycle must be launched manually, between cycles it is possible to change the state of the nodes
- **Step-by-step simulation:** each program cycle must be launched manually, between cycles it is possible to change the state of the nodes

The first type provides a more realistic assessment of the logic networks created, the second one permits a thorough check on every single passage of values between blocks and offers a higher level of diagnosis.

7.3 Graphical simulation environment

On pressing one of the simulation buttons (continuous or step-by-step), the editor window is modified as follows:

- The main menu is limited to PROJECT view only, allowing only passing between logic programs. It is not possible to create or delete programs.
- The details pane is closed to provide the most workspace for the simulation.
- Each drag&drop operation, connection, change or deletion of the content of the logic programs is locked.
- The nodes are coloured according to their state and allow forcing the value manually (as detailed below).

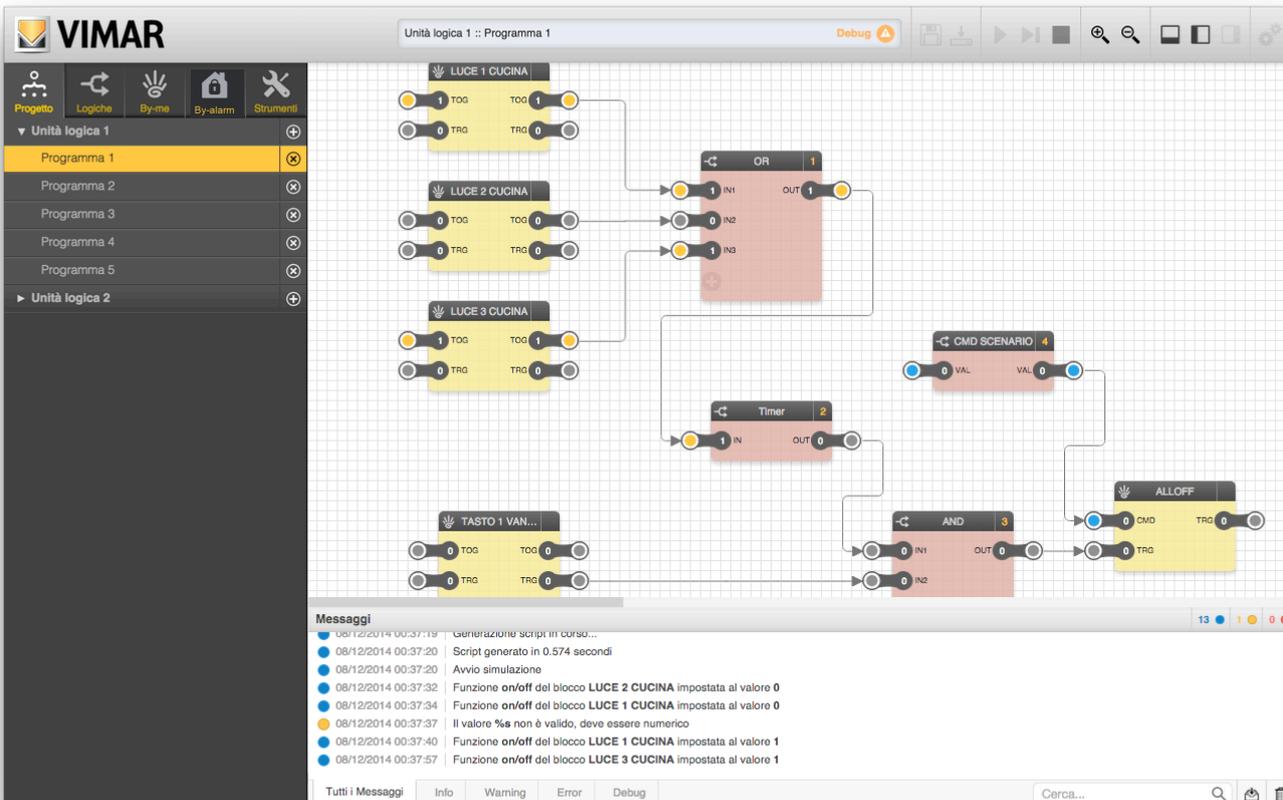
The colour of the nodes follows this convention:

Binary nodes	Grey	Value 0 (OFF)
	Yellow	Value 1 (ON)
Numerical nodes	Blue	Any value of those permitted

During simulation, the editor shows a variety of information in the message area related to the running of the programs, the manual state changes (made by the user) and automatic state changes (detected by the logic blocks). In addition, during step-by-step simulation, many "debug" level messages are reported, allowing a thorough analysis of the programs running, especially useful in the event of errors or malfunctions.

The message area, normally closed to provide the most space for the simulation, can be opened to view these messages, the number of which – depending on the type – is summarized on the right of the message bar, visible even when closed. For more details on the message area, see section 2.7.

The following figure shows an example of simulation with the message area open:



The screenshot displays the VIMAR logic editor in simulation mode. The main workspace contains a logic network with several blocks: three 'LUCE 1 CUCINA' blocks (yellow), an 'OR' block (red), a 'Timer' block (red), a 'CMD SCENARIO' block (blue), an 'AND' block (red), and an 'ALLOFF' block (yellow). The 'LUCE' blocks have TOG and TRG inputs and outputs. The 'OR' block has three IN inputs and one OUT output. The 'Timer' block has one IN input and one OUT output. The 'CMD SCENARIO' block has one VAL input and one VAL output. The 'AND' block has two IN inputs and one OUT output. The 'ALLOFF' block has one CMD input and one TRG output. The message area at the bottom shows a log of simulation events, including script generation and state changes for the LUCE blocks.

Simulation

7.4 Manual data input

To set the status of a node manually, do the following:

- Double click on the value of the node (the label becomes editable)
- Delete the current value and enter the new value
- Press ENTER

The colour of the node (if digital) changes as a function of the new value, and is passed to the simulator, which propagates it instantly (in the case of continuous simulation) or at the next cycle (in step-by-step mode).

The output values from the blocks may be modified but not the inputs.

The inputs of a non-connected block (e.g. the input of a "Greater than" comparison block used as a threshold) cannot be modified during simulation.

During simulation they maintain the default value set during logic editing.

7.5 Simulating sending a signal from a trigger node

In both simulation modes it is possible to generate a rising edge from a trigger node by double clicking on the node itself. As the trigger signal stays on 1 only for the duration of the run cycle, the visual feedback, above all in continuous simulation, may be very short.

7.6 Stopping the simulation

You can stop the simulation at any time by pressing the simulation stop button on the toolbar (normally not accessible outside simulation).

Compiling

8. Compiling

After verifying with the simulation that the logic programs meet the requirements, you can transfer them into the logic units via the "COMPILATION" procedure:

- If there are multiple logic units, select the one on which you want to work, opening one of its logic programs (the status bar shows the logic unit you are working on)
- Press the "COMPILE FOR LOGIC MODULE" button in the toolbar at the top

Compilation can take up to several minutes, depending on the complexity of the EASYTOOL PROFESSIONAL project and of the logic programs, in which the following operations are performed:

- In-depth analysis of the EASYTOOL PROFESSIONAL project and construction of the data structures for two-way communication with the By-me bus
- Generation of the configuration file for the logic unit, containing the list of By-me groups, the types of related data, calendar schedules, etc.
- Generation of the "list" of logic programs, set in operation by the logic unit

Once file generation has finished, the files are transferred to EASYTOOL PROFESSIONAL, which takes them for subsequent transfer to the logic unit via a USB connection.

During the process of compilation, a screen shows its progress, as exemplified in the following figure. At this stage it is not possible to work with the logic programs.

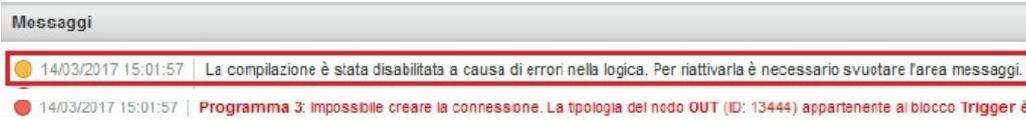


After compilation, you can then again make changes to the logic programs or close the editor (and save the configuration of the logic programs in EASYTOOL PROFESSIONAL).

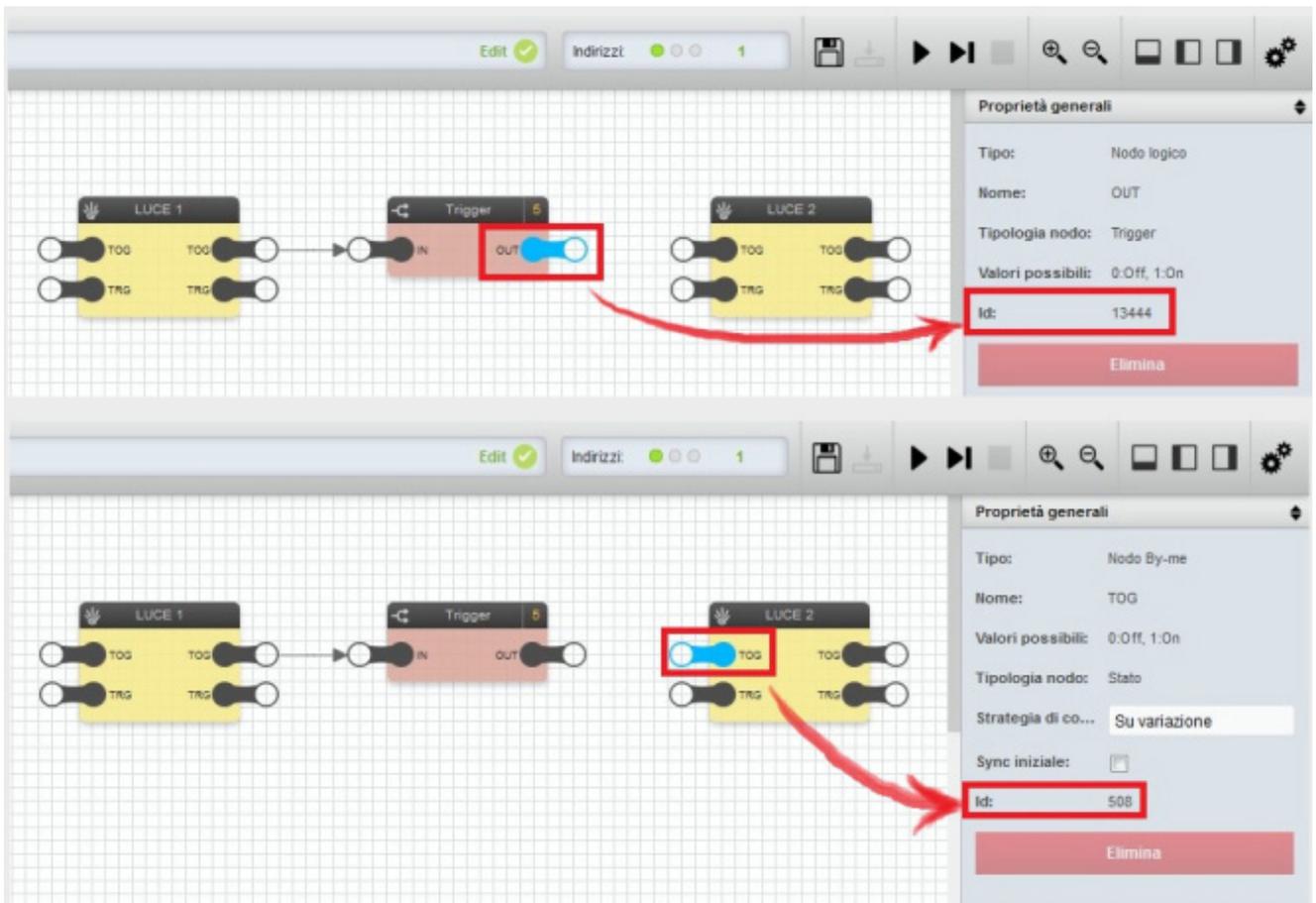
Compiling

8.1 Compilation disabled

If you are importing a program within which there are errors, the Messages Area will open and the application will display the following:

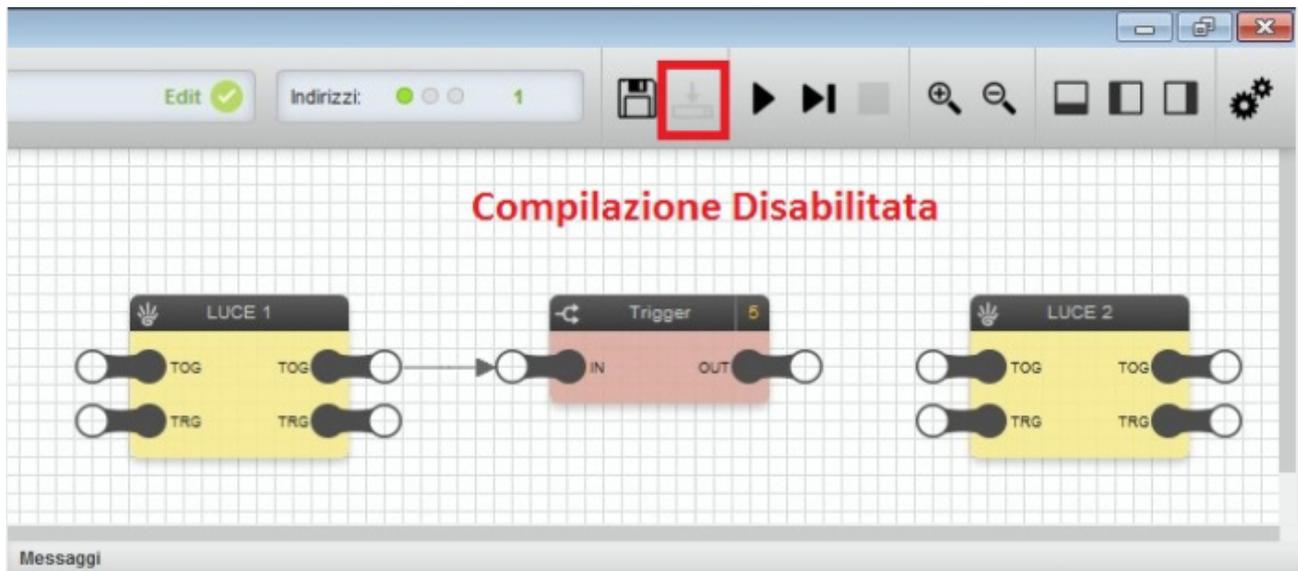


- a *Warning* message that, paired with the  icon, informs that compilation is disabled due to an error; to continue you need to correct it and then empty the Messages Area by clicking on the  icon.
- an *Error* message (red) that, paired with the  icon, informs that you cannot create the connection between the nodes involved; the message indicates the name of the program, the name of the block and the node ID in order to identify the cause of the error immediately.



Compiling

The compilation function is disabled (its icon cannot be selected) and remains so until all the errors have been corrected.



Note: When the Messages Area is emptied of error messages, compilation is enabled again.

Drawing tools

9. Drawing tools

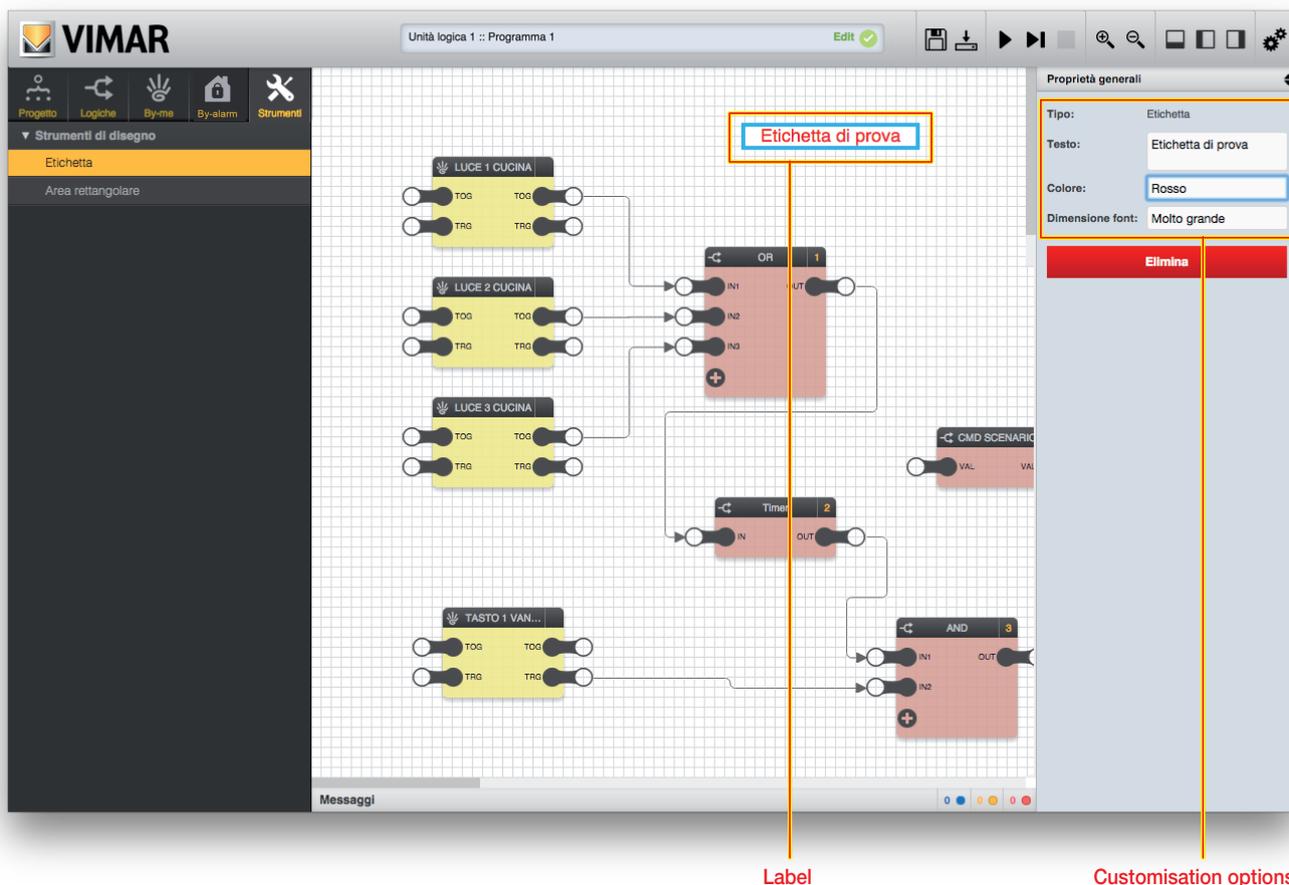
9.1 Introduction

In order to increase the readability of the logic programs, especially with complex logic networks, the editor provides several drawing tools with which the user can add notes and highlight areas of the program.

These tools are available in the "TOOLS" area of the main menu, in the "DRAWING TOOLS" section; using *drag&drop* they can be dragged into the logic programs in the same way as other types of objects, as seen previously.

9.2 Labels

Labels let you enter free text in programs; you can enter an unlimited number of labels for each logic program.



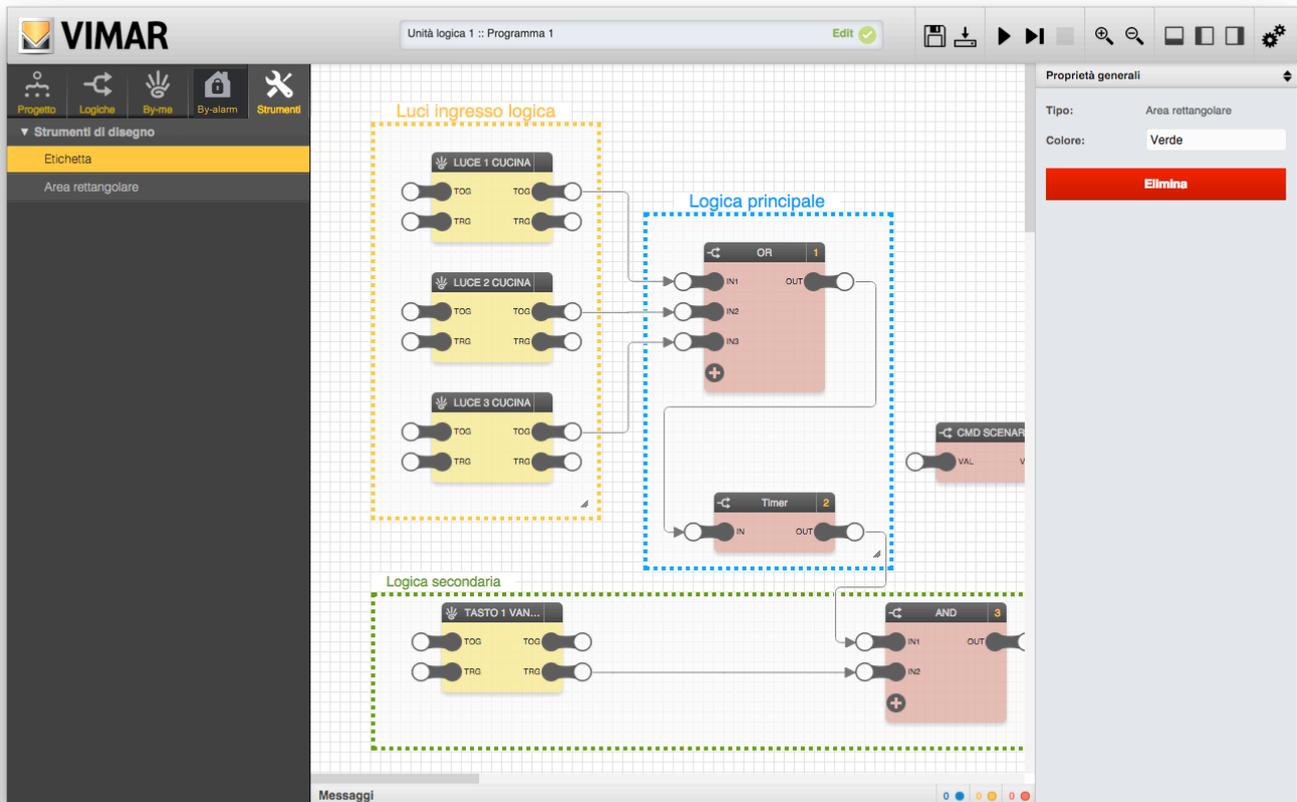
After dragging a label into the logic program and placing it where you want, it can be customised by opening the details pane (after selecting it); the following options are available:

Text	Text displayed in the logic program
Colour	Lets you choose the colour of the text
Font size	Lets you choose the font size

The labels can be removed from the logic programs by using the corresponding "DELETE" button in the details pane, or by pressing the DEL key on the keyboard after selecting them.

9.3 Rectangular areas

You can highlight one or more portions of the logic program by dragging from the main menu the same number of coloured rectangular areas, as exemplified in the following figure:



After dragging a rectangular area into a program, you can:

- Resize it by dragging the slider in its corner at bottom right
- Change its border colour, by using the "colour" selector in the details pane

The rectangular areas are always drawn underneath the blocks and their connections; moreover, they do not support multiple selection (like blocks or labels) and to customize them or remove them from the program it is necessary to click on them, one at a time, and use the tools in the details pane (change colour and "DELETE" button to remove them from the program).

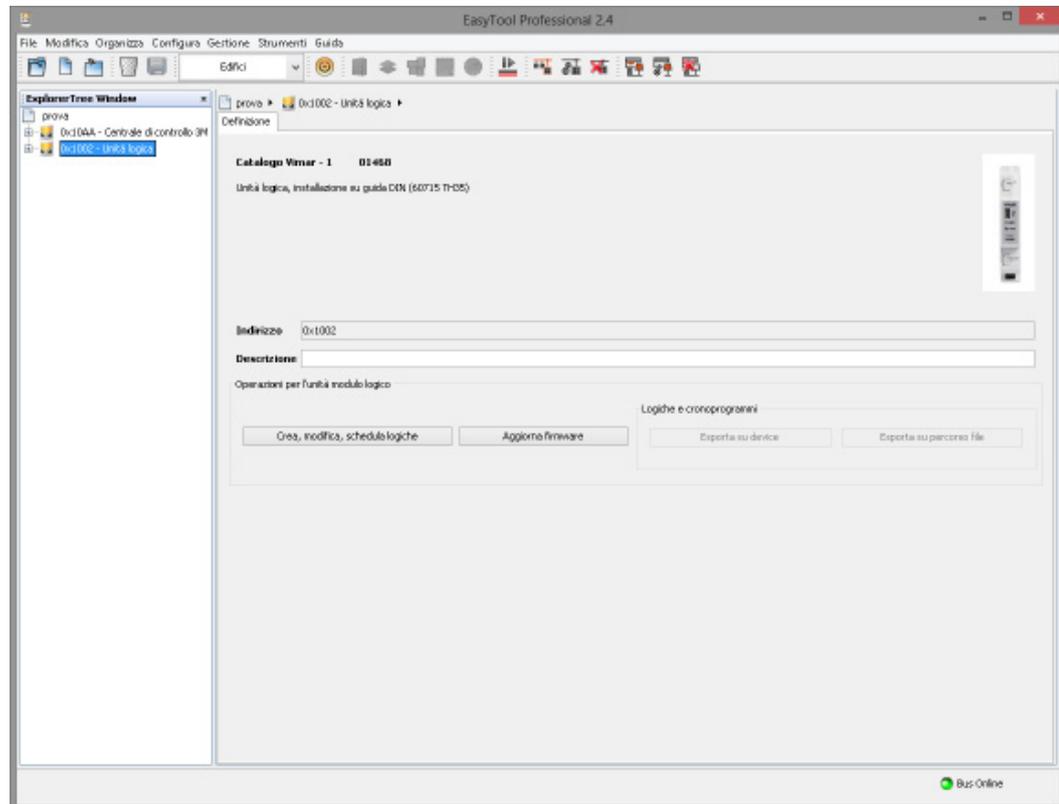
Device management

10. Device management

10.1 Introduction

After configuring and creating logic elements and timelines, you can transfer them to the unit by connecting it via USB. Logic elements and programs are constrained to some library and firmware versions and may require updating the software on the device, again via USB.

You can access the advanced management features of the Logic Unit from the tab for the device in the tree view of the buildings. Click on view "Buildings" and then select the desired device from the tree:



From this tab you can:

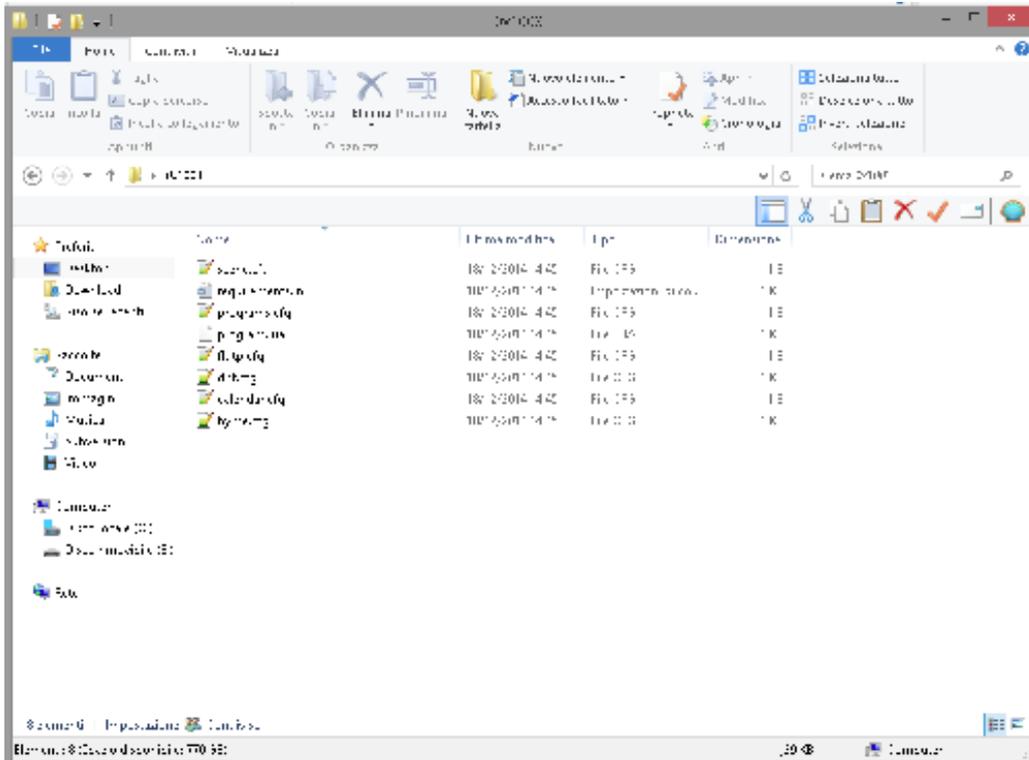
- open the **logic editor** (refer to the appropriate section of the manual);
- open the tool to **update the firmware**;
- **export the programs and timelines to the device or file system.**

Device management

10.2 Exporting programs and timelines

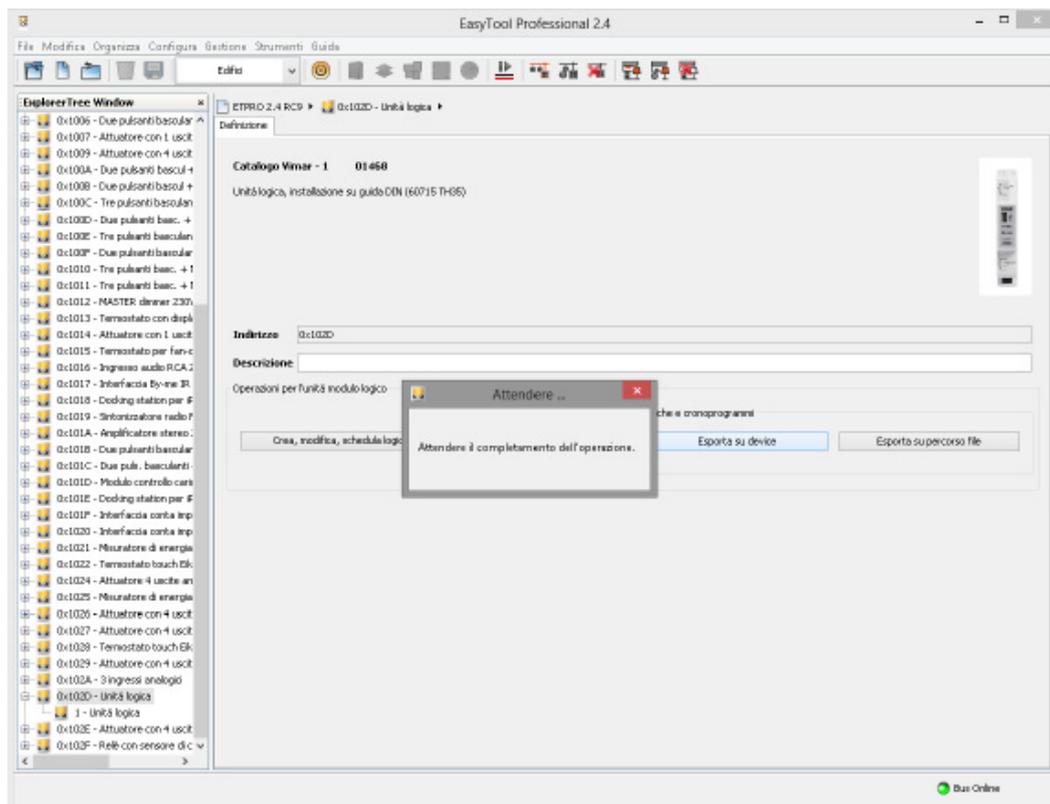
10.2.1 To file

After saving the logic files and their scheduling in the logic editor, the "export to user file path" button will be available. On clicking the button you will be prompted to select the export path. Exporting is fast and will be immediately notified by EasyTool Professional. The result will be a folder located on the chosen path, named with the address of the chosen device and containing a set of logic files:

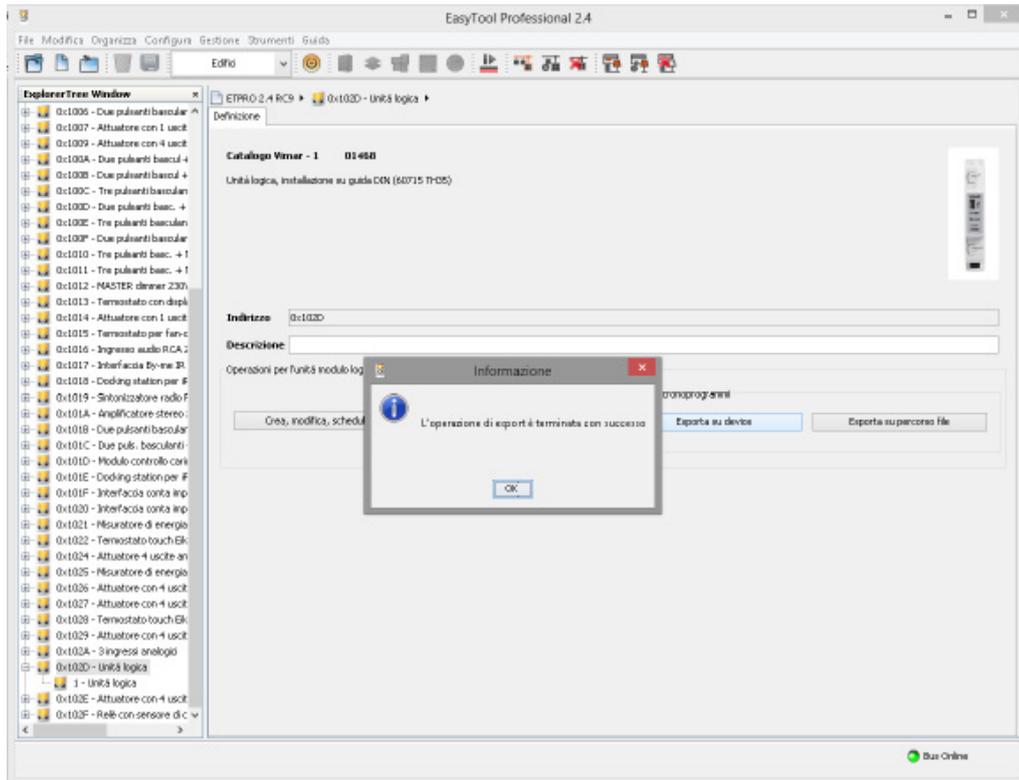


10.2.2 To device

After connecting the selected device to the USB port and if the files are available, the "export to device" function is also available. Clicking on the button will start the export procedure, which may take some time:



After finishing export, EasyTool Professional will notify the user:



CAUTION: Do not disconnect the power supply when the green LED is on steady.

10.3 Updating firmware

10.3.1 Driver prerequisites

The EasyTool Professional installation program not only installs the application but also installs the drivers used to manage the communication between the program and the logic unit. During installation the same or a more recent version of the driver is already installed in Windows it will not be reinstalled, leaving the most updated version active.

When first connecting the device to the computer the operating system will in any case run an internal update of the driver to make sure it is using the most recent version.

Once installed, therefore, connect the device to the PC before running the EasyTool Professional firmware update procedure to prevent the driver refresh (done by the operating system) from interfering with the firmware upgrade operations (by EasyTool Professional).

This operation is required only after the first installation of EasyTool Professional or after an update. In all other cases the device can be connected more easily, as described in the instructions below.

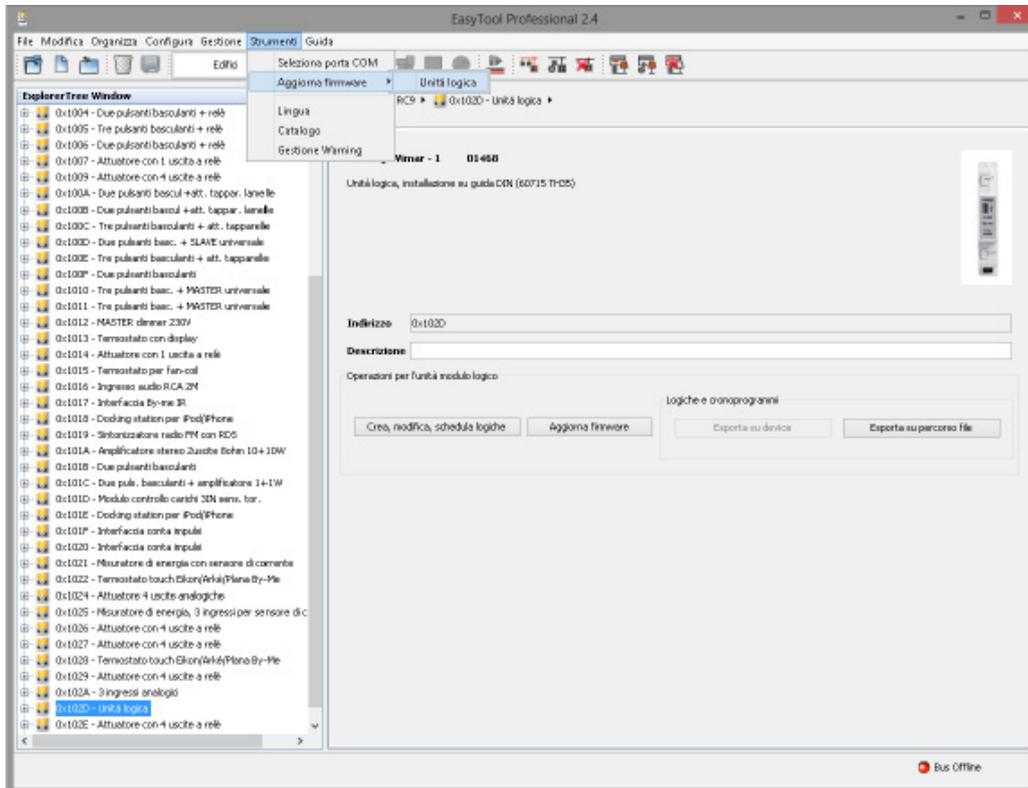
If this operation is not done in the above-described order, the EasyTool Professional firmware updating function could be blocked by the Windows operating system. In this case end the EasyTool Professional program from task manager.

Device management

10.3.2 Update tool

Remove the By-me Bus connection from the Logic Unit to update.

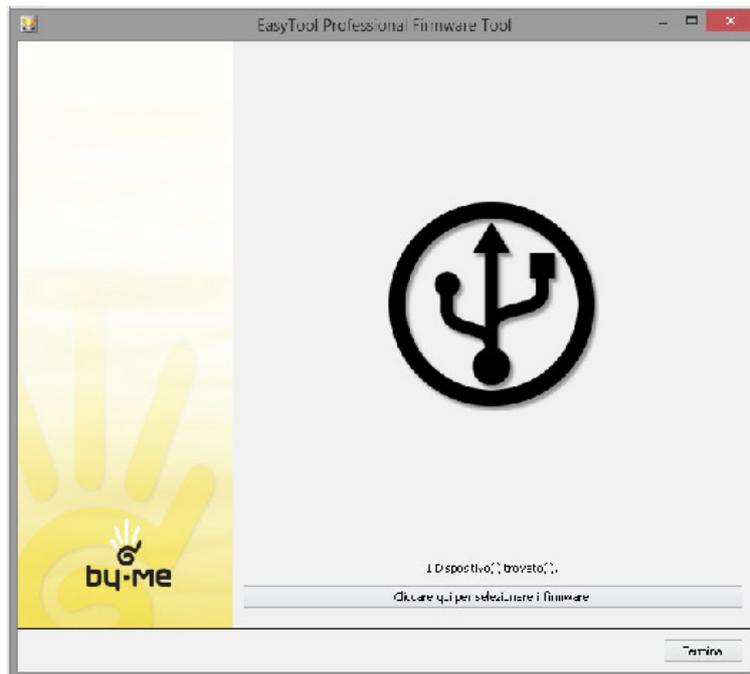
From the detail tab of the logic unit device ("Update firmware" button), or the appropriate menu item (*Tools* → *Update Firmware* → *Logic Unit*) you can run the firmware updating tool:



Start the firmware update tool, which will be waiting for the device:

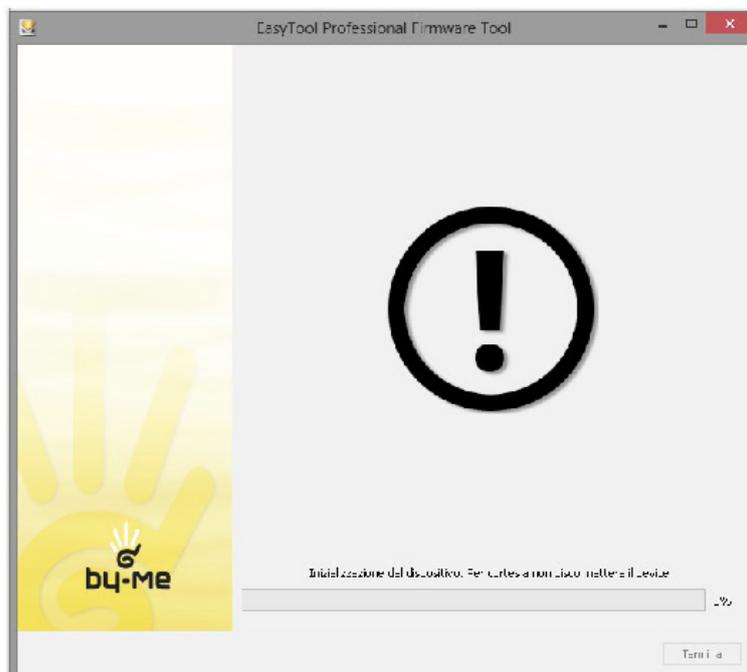


When the software is requested press the configuration button on the device which at that moment is disconnected from both the BUS and the PC; wait a couple of seconds and then connect the device to the computer via a cable to the USB port, **holding down the configuration button**. After plugging the cable into the port, release the button:

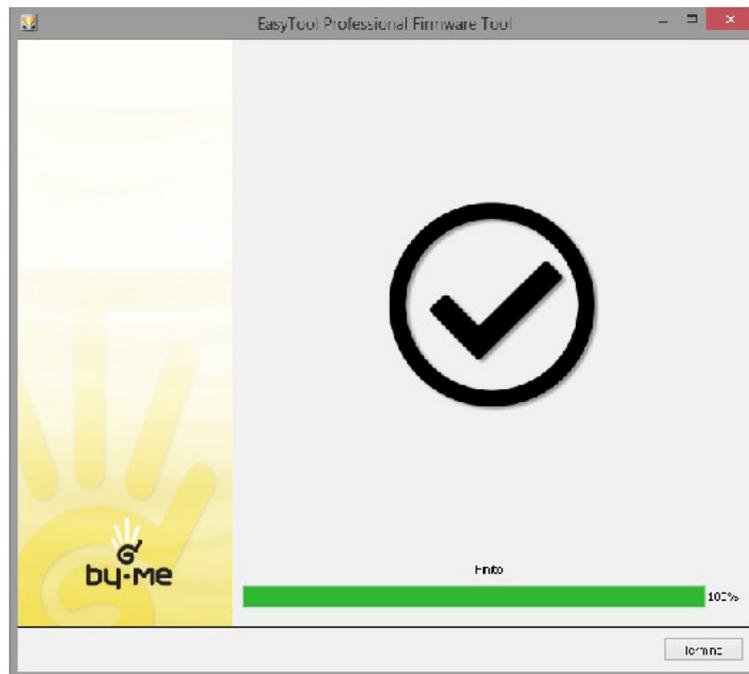


Select the firmware by clicking on the appropriate control. The update will start immediately. **Do not touch the device, do not disconnect it, do not turn off the computer or close the running program:** the device might not work.

The interface may appear frozen even for several minutes. Wait for it to finish:



Once the operation is finished, EasyTool Professional will notify the user:



You can now close the firmware update wizard.

10.4 Fault indication

Device operating faults are indicated by the green LED; the following table shows the types and their descriptions.

Green LED	Description
Off	No faults.
On steady	<ul style="list-style-type: none">• With the USB cable disconnected, it signals start-up of the Logic Unit; DO NOT CUT OFF THE POWER SUPPLY.• With the USB cable connected, it signals a successful connection.
Slow flashing at start-up (1 s)	Signals operation during the phase of switching on the Logic Unit; the flashing could last for several seconds.
Fast flashing (0.5 s)	An operating fault indicating failure to execute the programmed logic.
Very fast flashing (0.5 s)	Signals a lack of configuration; the By-me device must then be reconfigured.

Remote control

11. Remote control

11.1 Introduction

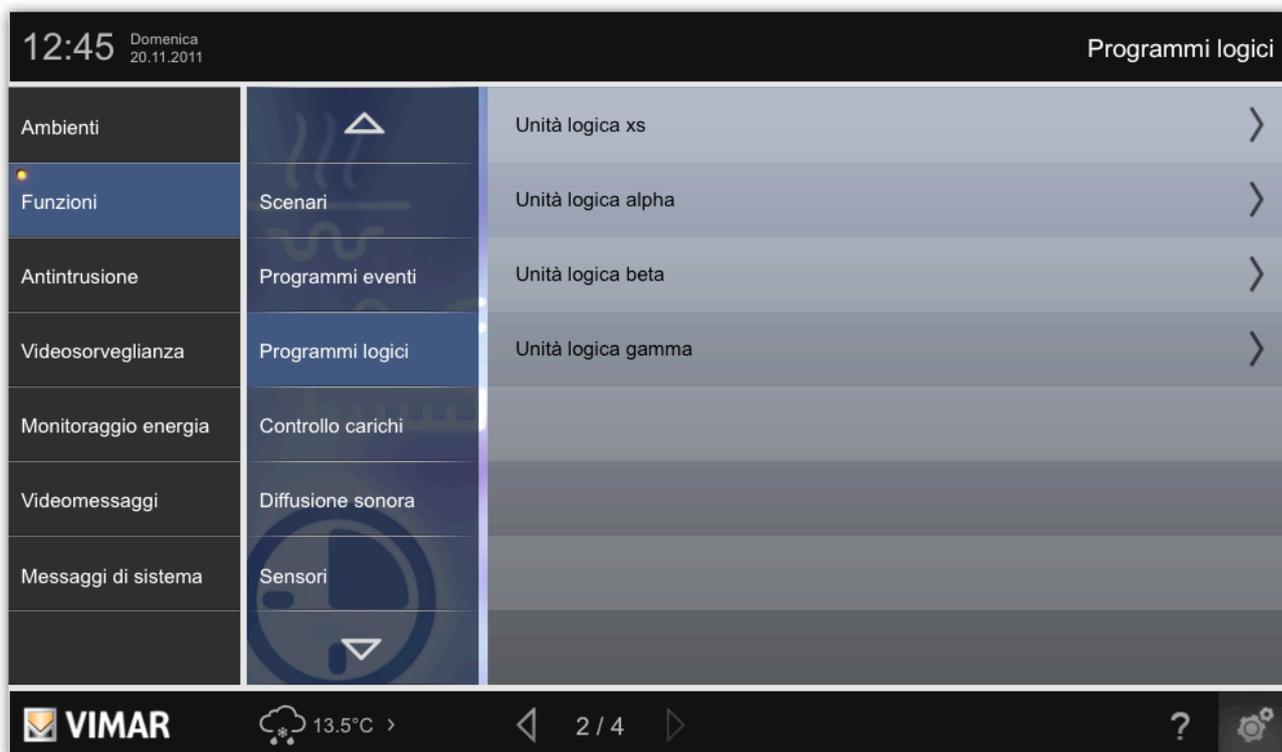
It is possible to interact with the programs created with the editor, once they are loaded in the logic units, via other By-me devices, such as the Web Server or touch screen. In particular, it is possible to pause one or more programs (suspending their execution) or change the scheduling (weekly, periodic or cyclic) planned in the editor (cf. 6.9).

Please note that the Programs which can be viewed and edited from Web Server or touch screen are those for which the "Remote Management" flag was kept enabled during editing.

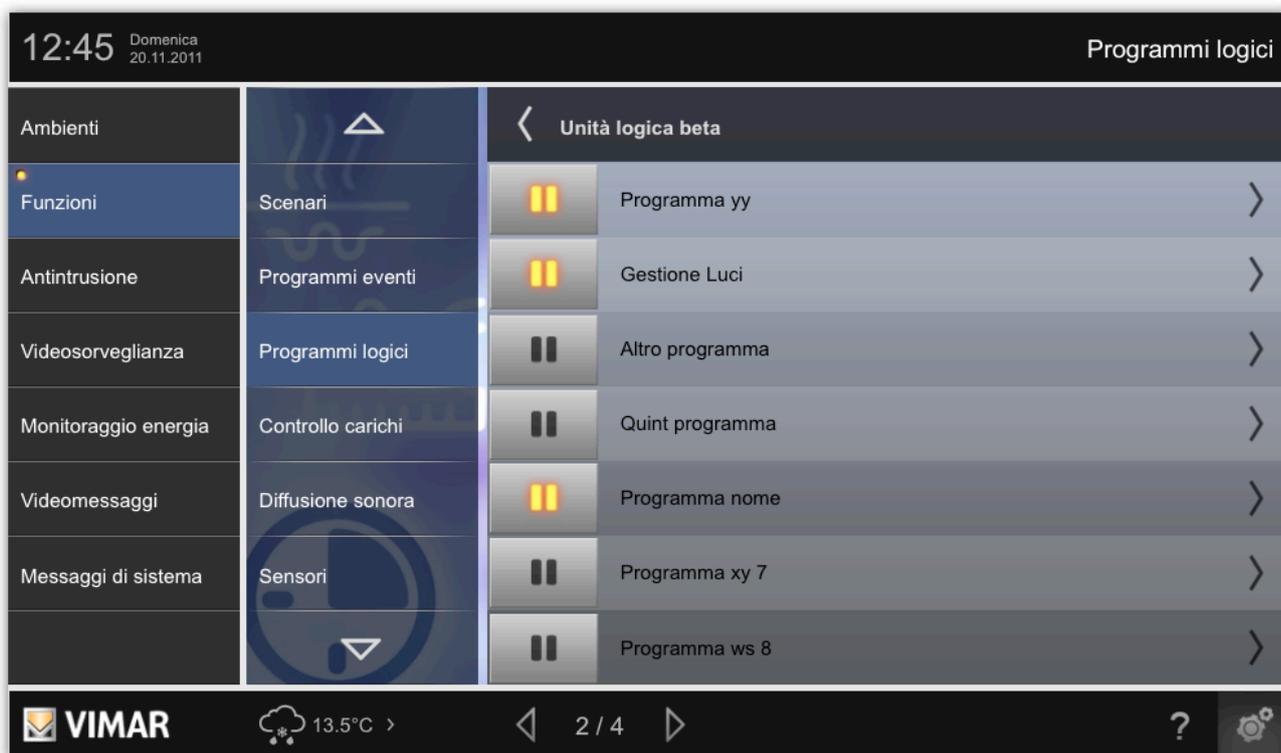
The information contained in this chapter refers specifically to the web server; for other devices, similar analogue graphic screenshots are available, to which you should refer in any case to the relevant documentation.

11.2 State of program running

Entering the "LOGIC PROGRAMS" section in the "FUNCTIONS" area will display a list of the available logic units:



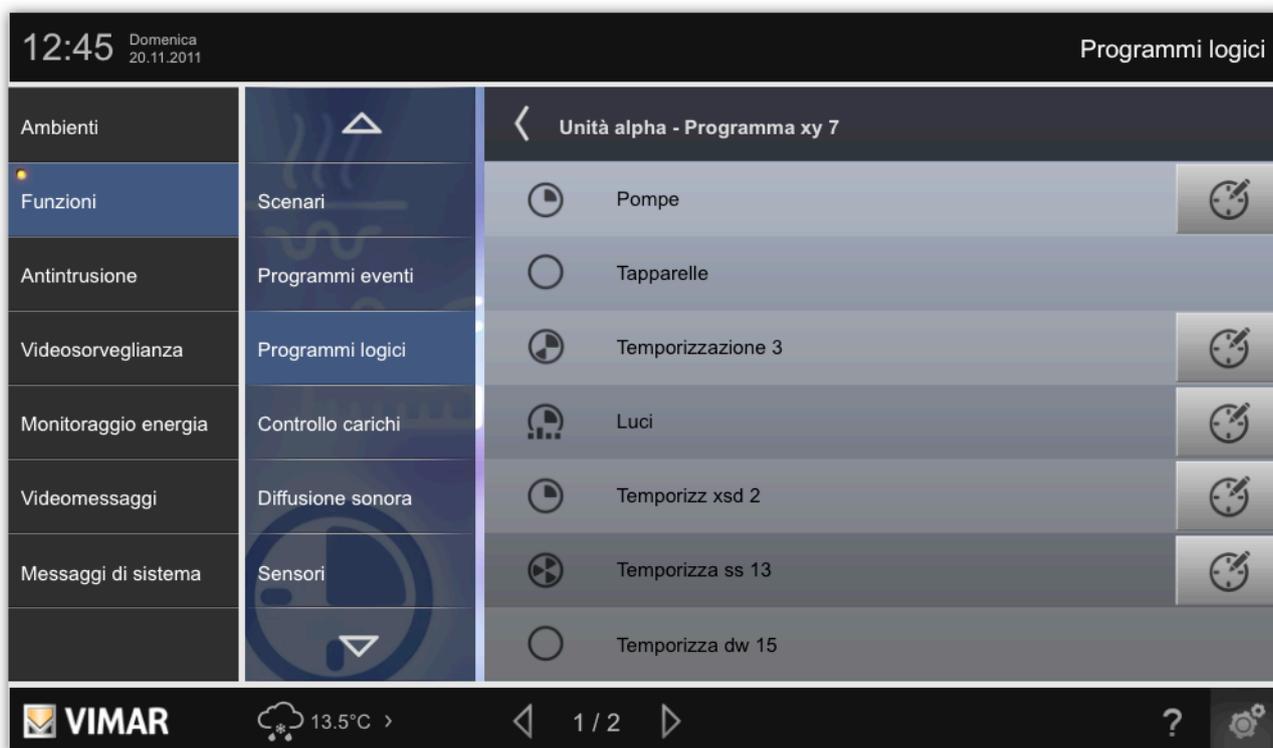
Pressing the  button of the relevant logic unit (right side of the graphic screen) will poll the device to obtain the list of available logic programs it contains (this can take several seconds, depending on the number of programs):



To pause a program, simply press the corresponding button ; when the button is lit, the program is not running, though it is still compiled in the logic unit. To unlock execution of a paused program, press its button again.

11.3 Calendar schedules

On going into the details of the single logic program (with the corresponding "arrow" button on the right of the graphic screen), the logic unit is polled again to obtain the list of scheduled programs it contains:



To edit a program, press the corresponding button , and interact with the configuration pop-up, similarly to how events programs are managed on the control panel, and as seen in the editor in section 6.9 in this manual.

Appendix

12. Appendixes

12.1 Glossary

Logic unit	VIMAR By-me device mod. <i>01468</i> <i>able to run one or more</i> logic programs configured via the editor. The catalogue name is the "Logic Unit".
Logic program	Logic network composed of one or more interconnected logic and By-me <i>blocks</i> <i>Each</i> logic unit can contain up to 64 logic programs.
Logic block	A block that can be inserted in a <i>logic program</i> to perform a specific function, interacting with other blocks via input and/or output <i>nodes</i>
By-me block	A block that can be added to a <i>logic program</i> to read and/or write information on the home automation bus, interacting with other <i>blocks</i> via input and/or output <i>nodes</i>
Node	Single element of a logic or By-me block which provides specific information at the input or output; nodes can be interconnected with other nodes via the same number of connections
Connection	Connection between two <i>nodes</i> of the same number of <i>blocks</i> . The connection has a "direction" that determines the order in which information is exchanged between the <i>nodes</i> ; in particular, the status of the <i>source</i> node is passed to the <i>target node</i>
Editor	Graphic environment for configuring the <i>logic programs</i> . Used to build <i>logic programs</i> for the <i>logic units</i> in the project and to download into the latter the information necessary for running them.
EASYTOOL PROFESSIONAL	EasyTool Professional. Software for configuring the By-me system in which the editor operates for the logic unit described in this manual.

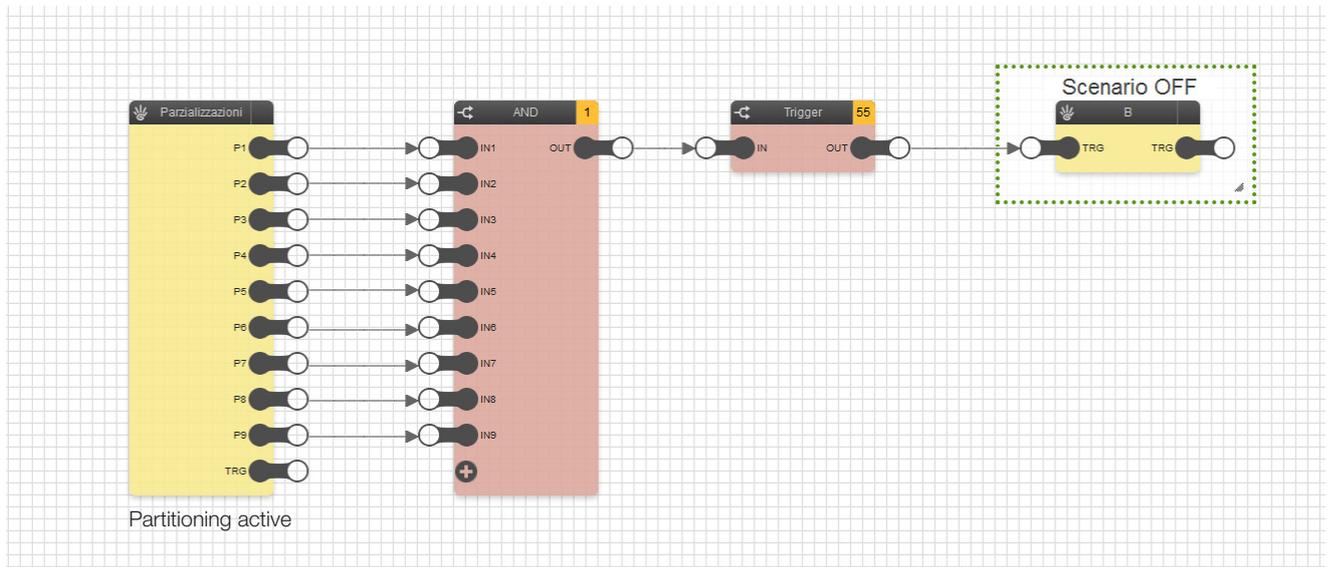
Application examples

13. Application examples

This section illustrates, as an example, the construction of some logic programs to implement typical functions of the By-me system.

13.1 Activating a scenario via the anti intrusion system

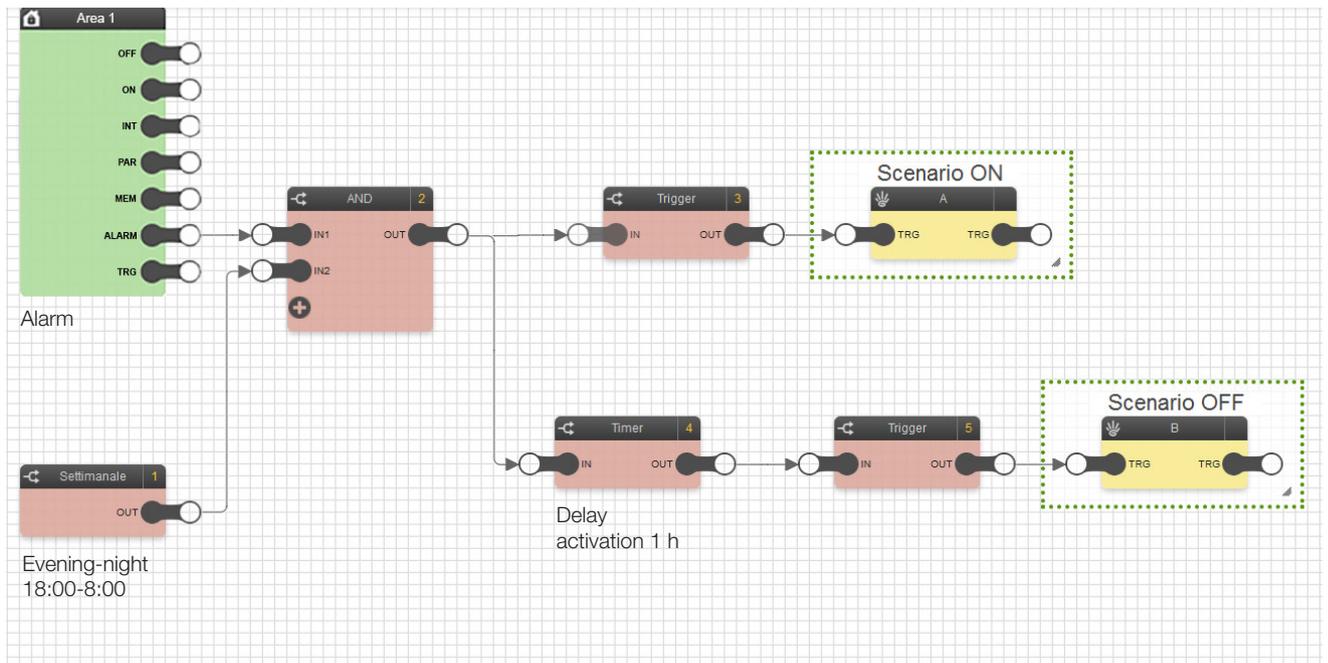
If the user FULLY ENABLES the SAI intrusion detection system (all partitioning on), the logic program enables a lights off SCENARIO. If the activation of the intrusion detection system is partitioned, the scenario will not be activated.



- The Partitioning block is used, the outputs of which (P1...P9) are connected to the respective inputs of the AND logic.
- When all the outputs (P1...P9) are on 1 (ON) the output OUT of the AND logic switches on the SCENARIO OFF block (which must always be preceded by the TRIGGER block).

13.2 Activating a scenario following an intrusion alarm

If there is an evening or night-time alarm (between 18:00 and 8:00 am the following day), the logic program enables a lights on SCENARIO which is then switched off 1 hour later via a lights off scenario.

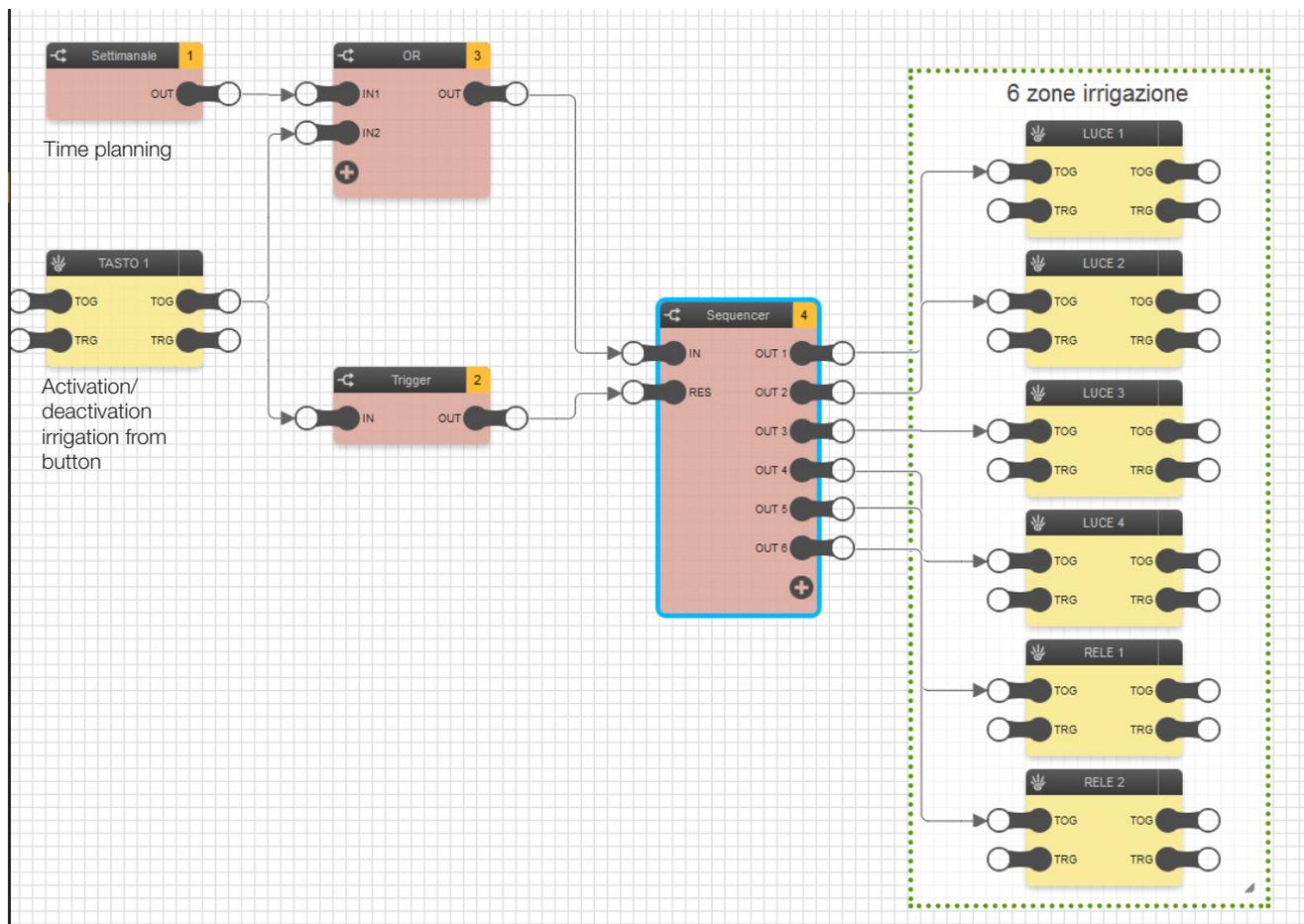


- The SAI intrusion detection alarm block and the WEEKLY TIMER block are used, and must be connected to the respective inputs of the AND logic; a planning is set in the WEEKLY TIMER in which the block is ON between 18:00 and 23:59 of the current day and between 0:00 and 8:00 of the next day.
- When an intrusion alarm is triggered, during the time set in the WEEKLY TIMER, the OUT output of the AND logic switches on the SCENARIO ON block (lights on) and the TIMER block which, after 1 h (value set as the rising delay), enables the SCENARIO OFF (lights off).
- The SCENARIO blocks are always preceded by TRIGGER blocks with the EDGE parameter set to TRUE.

Application examples

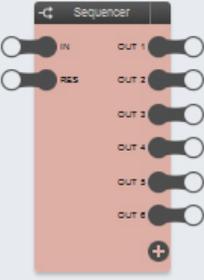
13.3 Sequential and timed irrigation with start/lock control from button

The logic program switches on the irrigation in 6 differentiated zones of the garden, switching them on in sequence with 10 minutes in each zone. In addition to automatic switch on, the irrigation can also be controlled manually via a button; this button can also be used to block the irrigation prior to the natural end of the sequence.



- Use the BUTTON block and the WEEKLY TIMER block, connected respectively to inputs IN1 and IN2 in the OR logic; the WEEKLY TIMER has a planning in which the block is ON every day at a set time (for example, 3 pm) for the length of time needed to run the various programs (60 min. in this case) since we want to control 6 zones for 10 minutes each.
- The BUTTON block can switch on the irrigation whatever the set planning.
- The OR logic is connected to the SEQUENCER block input, the outputs of which (OUT1...OUT6) are connected to the respective ON/OFF blocks with enable the irrigation zones.
- The BUTTON block is also connected through the TRIGGER block to the RES input to stop the activation sequence in the 6 zones; the TRIGGER is on with the parameter EDGE set to FALSE.
- To enable the 6 zones in sequence (zone 1 comes on for 10 minutes, zone 2 comes on for 10 minutes after zone 1 is switched off, and so on) the SEQUENCER must be set as follows:

Proprietà generali



Tipo: Sequencer

Ordinamento: Manuale

Ordine:

Id: 7007

Sequenza ciclica: Falso

Durata passo 1: 00:10:00

Durata passo 2: 00:10:00

Durata passo 3: 00:10:00

Durata passo 4: 00:10:00

Durata passo 5: 00:10:00

Durata passo 6: 00:10:00

Durata passo 7: hh:mm:ss

Durata passo 8: hh:mm:ss

Durata passo 9: hh:mm:ss

Durata passo 10: hh:mm:ss

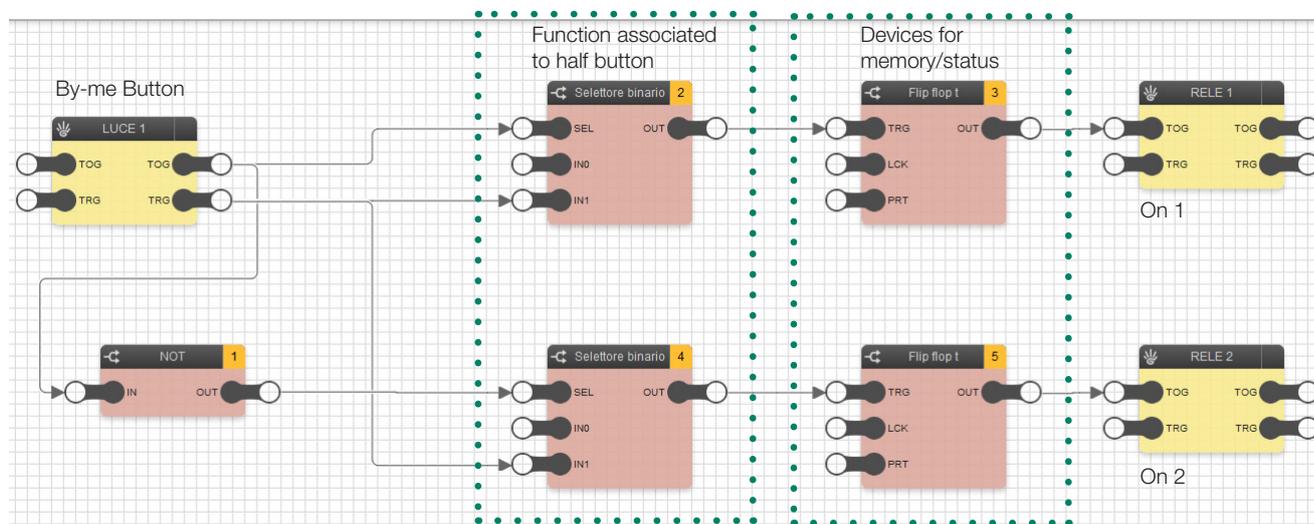
Elimina

Application examples

13.4 By-me rocker switch used for 2 separate ON/OFF functions.

Using a By-me rocker switch, the logic program can manage 2 separate on/off switches on the same button. The upper button is used for ON-OFF of a utility and the lower button for the ON-OFF of the other.

This solution expands on the functions of the By-me controls which, with conventional configuration, cannot implement the application illustrated here.



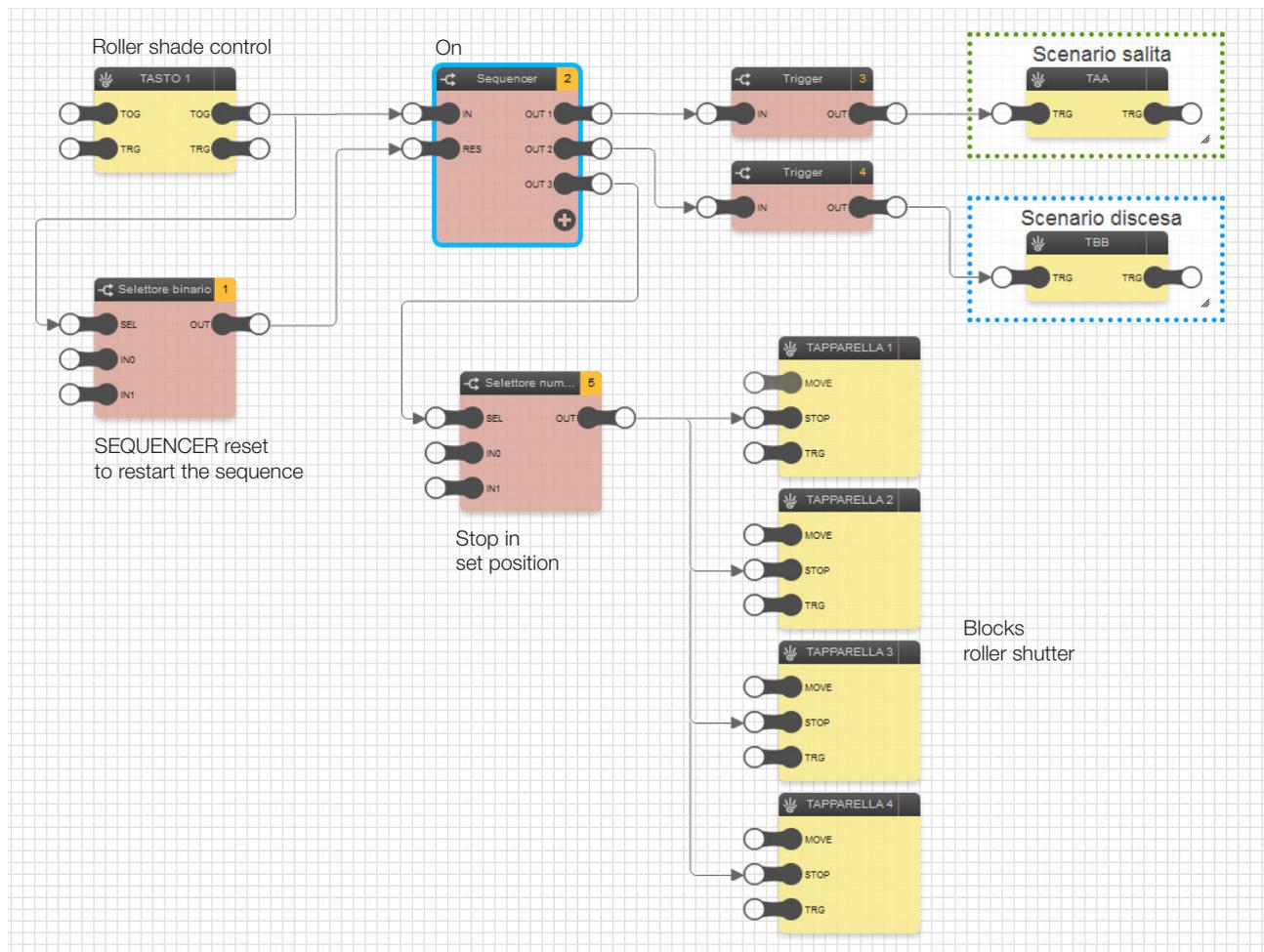
- Each time the button identified by the block LIGHT 1 is pressed, the control issues a TRIGGER impulse on the BUS and a control which, according to whether the top or bottom of the button is pressed, may be respectively for group ON or OFF of the group it was configured in.
- The Flip flop T switches the output status whenever it receives an input impulse and so to keep enabled or disable when an impulse is received; in this case there must however be a control which ensures that the impulse ONLY arrives if generated by pressing the correct part of the button (top or bottom).
- For this purpose Binary Selector blocks are used, which allow the impulse to pass ONLY if the group control generated with the impulse is correct.

Example: By pressing the top of the button, selector 2 allows the impulse to transit as with TOG to 1 SEL also goes to 1 thus collecting the TRG impulse which was set off at the same time and connected to IN1 on the Binary selector; when it reaches the Flip Flop T it will then switch to the previous status.

Application examples

13.5 Opening/closing shutters to preset positions.

Via a scenario, the logic program raises the shutters to an intermediate position (different from shutters full up or full down).



WARNING: This program functions correctly only if all the shutters concerned have the same up/down times.

- When the input group is in ON (BUTTON 1 block) the SEQUENCER IS ENABLED; to operate, it enables the output and keeps it active for the time set in the general properties of the block, and then passes to the next until the last output is disabled.
- As outputs OUT 1 and 2 on the SEQUENCER must in any case control the scenarios, the command sent by them cannot be a stable output but rather an impulse; by inserting a TRIGGER block between the outputs OUT 1 and OUT 2 and the respective scenario blocks, this condition is assured. (In the TRIGGER the parameter Edge is set to TRUE because it must intercept the output activation).

IMPORTANT: When setting the general properties of the SEQUENCER, the time value entered on the OUT1 and OUT 2 step duration is very important, as this determines:

- Length of step 1: the time taken by the shutters to be raised COMPLETELY must be entered.
- Length of step 2: the time taken by the shutters to reach the REQUIRED POSITION must be entered.

Proprietà generali

Tipo: Sequencer

Ordinamento:

Ordine:

Id: 53688

Sequenza ciclica:

Durata passo 1:

Durata passo 2:

Durata passo 3:

Durata passo 4:

Durata passo 5:

Durata passo 6:

Durata passo 7:

Durata passo 8:

Durata passo 9:

Durata passo 10:

Elimina

Ingressi

IN Inizio sequenza

RES Reset sequenza

Uscite

OUT 1 Uscita 1

OUT 2 Uscita 2

OUT 3 Uscita 3

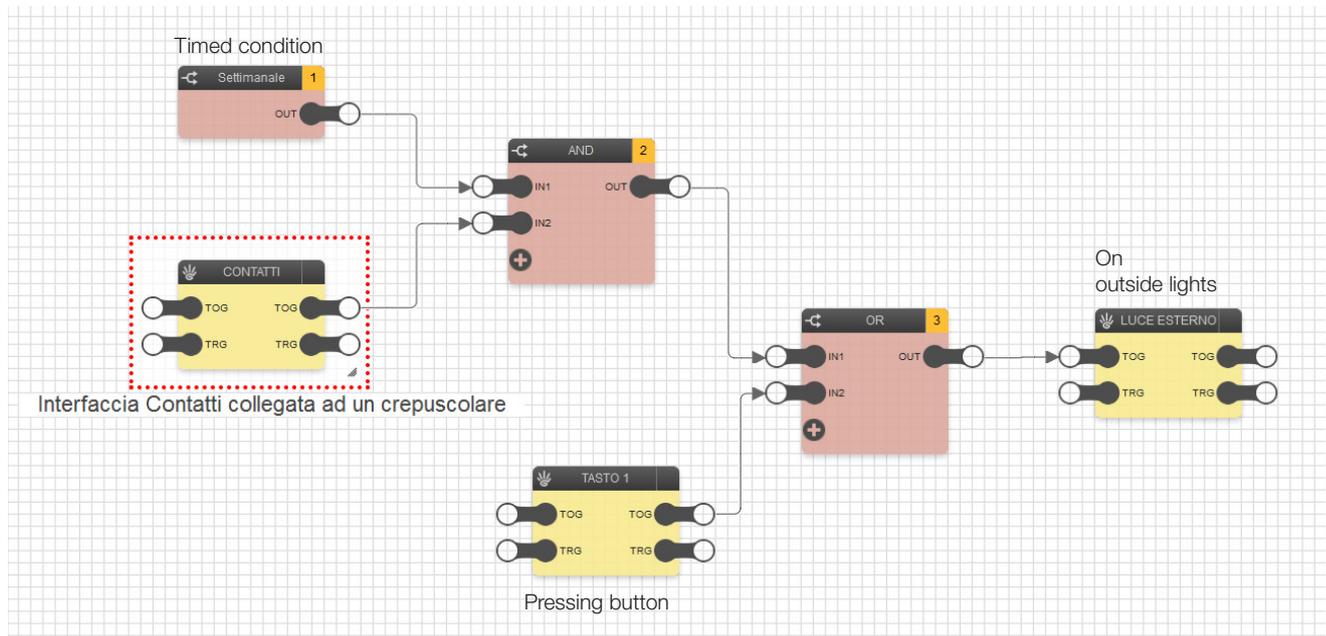
+ Aggiungi uscita

- The output from OUT3 of the SEQUENCER controls the STOP of all shutters involved in the scenario (in this case 3 groups); between OUT3 and the STOP inputs of the shutter blocks is the NUMERICAL SELECTOR as the STOP node on the shutter block requires a numerical and not a binary 1. When entering "1" as the IN1 value of the Selector, when the control reaches SEL it produces that output value. Length of step 3 (OUT3) can be set to 1 sec as an impulse is sufficient for the shutters to be blocked.
- The BINARY SELECTOR connected to the RES input of the SEQUENCER ensures that when the input group goes to OFF, the SEQUENCER is reset and restarts (for example, if the user wishes to cancel the program after having ordered it to start). Without the Binary selector, the Sequencer would stop the sequence and would RESUME IT IN THE SAME POINT at the next ON signal (which could cause malfunctions).
- On the BINARY SELECTOR IN0=1 and IN1=0.

Application examples

13.6 Switching on outdoor lights from the twilight sensor and button control

The logic program switches on the outside lights between 21:00 and 5:00, only when triggered by a twilight sensor. We can also switch on the lights manually from the button.



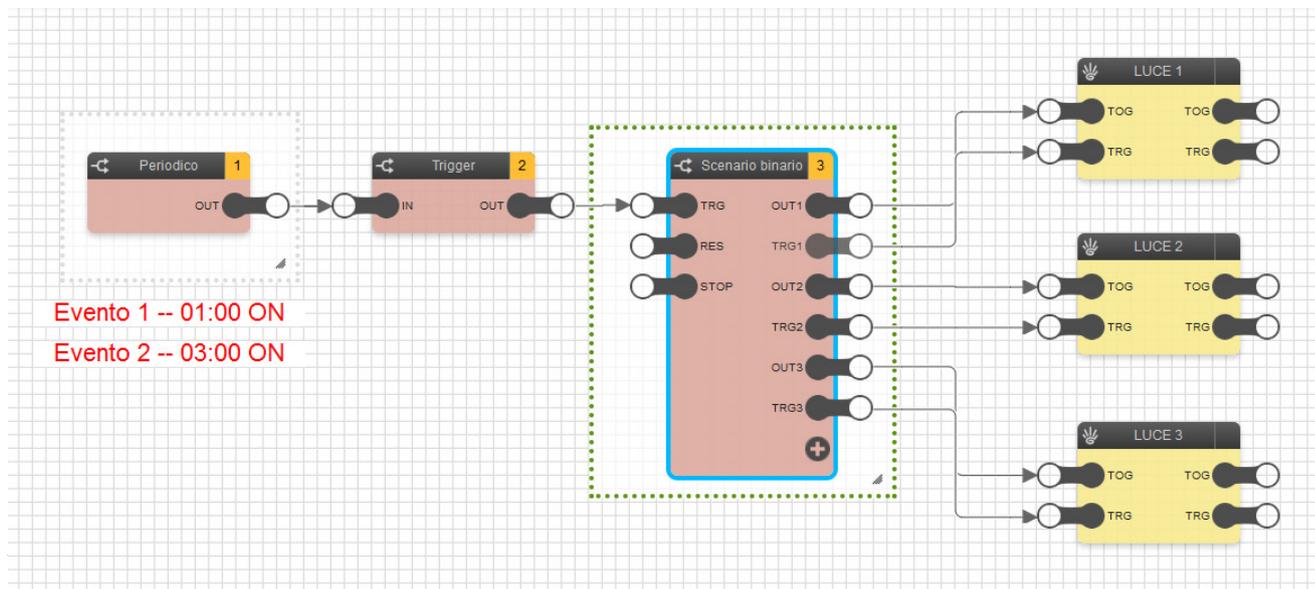
- We use the CONTACTS block (which represents the interface to which the twilight sensor is connected) and the WEEKLY TIMER block, connected respectively to inputs IN1 and IN2 of the AND logic; a planning is set in the WEEKLY TIMER in which the block is ON every day from 21:00 to 5:00.
- The BUTTON 1 block represents the button used to switch the outside lights on whatever the status of the AND logic output (and therefore the reading of the twilight sensor or the time setting).
- The OR logic is connected to the OUTSIDE LIGHT block input and ensures that the lights can be switched on from the BUTTON 1 block whatever the status of the AND logic.

Application examples

13.7 Switching on single lights at set times.

The logic program switches on the outside lights every night at 1 and at 3 o'clock.

This application is useful, for example, to switch off lights which could remain on all night by mistake.



- To create the event in the required time interval, use the PERIODIC TIMER block, to set a planning in which Event 1 starts at 1:00 (and Ends, for example, at 2.00) and Event 2 starts at 3.00 (and Ends, for example, at 4.00).
- The PERIODIC TIMER block is connected to a TRIGGER block; this is used to obtain an impulse at the BINARY SCENARIO block input, the outputs of which OUT and TRG are used to control the respective LIGHT blocks.
- The BINARY SCENARIO blocks must be set as follows to control the LIGHT blocks:



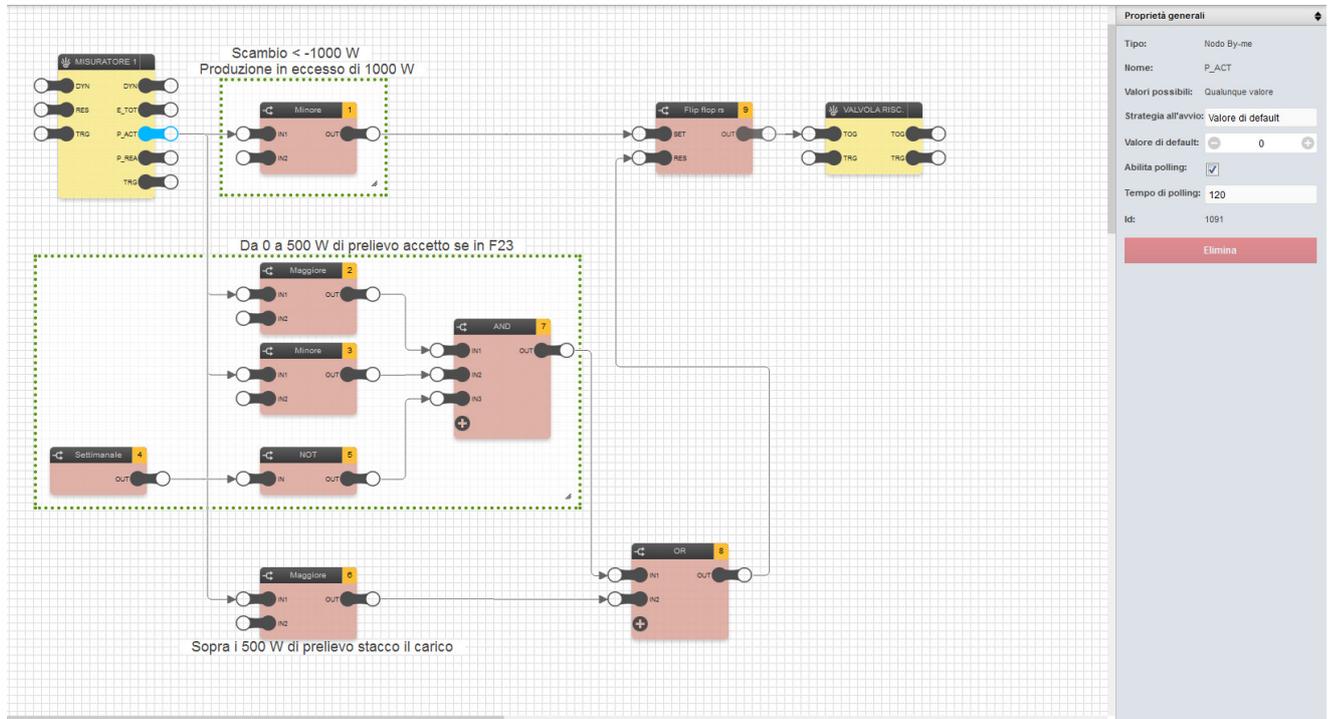
Application examples

13.8 Consumption management for activating the heat pump.

When the photovoltaic system produces and feeds into the network 1 kW more than the total consumption, the logic program switches on the load (heat pump) controlling the climate in a room.

Disable as follows:

- if it falls in the band F23 of the contract, the load is kept active for as long as the consumption remains below 500 W;
- the load is switched off if this value is exceeded.

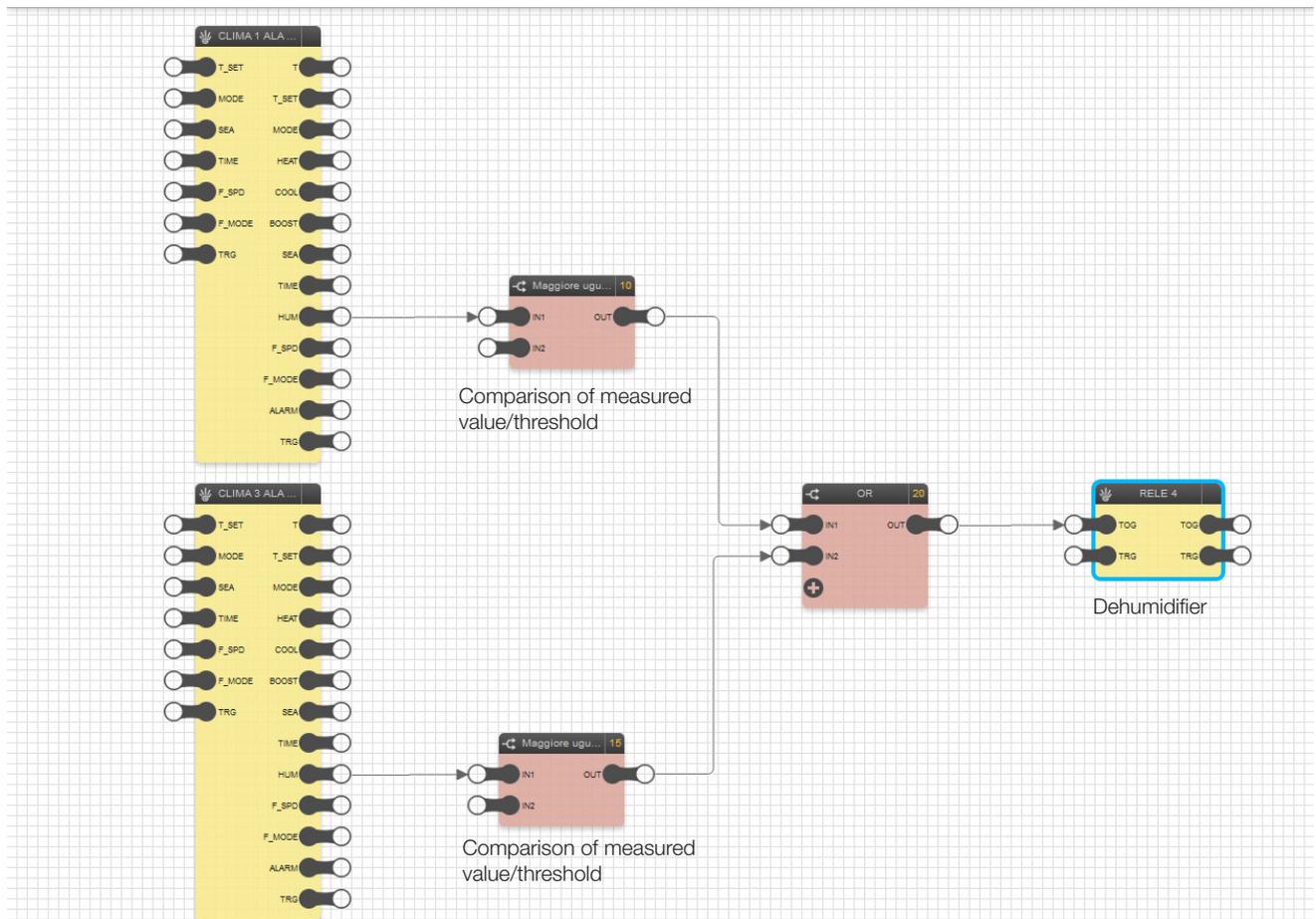


- When a value lower than -1000 W is measured on the node P_ACT, which is the node which views the power exchanged on the electricity supplier meter, there is in fact an excess production of over 1000 W; if this condition occurs, the block LESS THAN 1, in which the other node is set to -1000, sets the OUT output to TRUE. The latter, controlling the SET input of the FLIP FLOP, sets OUT to 1 and therefore controls the VALVOLARISC load.
- The FLIP FLOP remains on until TRUE is given on the RES input; then all the other logics which define the behaviour if there is no excess production come into play (i.e. P_ACT >= 0).
- The logic blocks contained in the larger box mean that, if the consumption is between 0 and 500 W in the low-cost time band F23, the load is in any case kept on.
The block GREATER THAN 2 in fact becomes TRUE as soon as P_ACT exceeds 0, LESS THAN 3 becomes TRUE if P_ACT stays below 500 W while the WEEKLY connected to the NOT block sets a TRUE value only OUTSIDE of the F23 time band, otherwise it would be necessary to switch off the load (we would be in band F1, which has a higher cost); thus, if all three conditions are TRUE, the AND block sends the TRUE to the OR block which in turn sends it to the RES node on the FLIP FLOP which resets the output and switches off the load.
The comparison blocks can be set as follows:
 - LESS THAN 1: IN2=-1000
 - GREATER THAN 2: IN2=0.
 - LESS THAN 3: IN2=500.
 - GREATER THAN 6: IN2=500
- The block GREATER THAN 5, which becomes true if P_ACT exceeds 500 W on which the load is immediately switched off, is connected directly to the OR block which if TRUE, as above, sets the RES node to TRUE, which resets the FLIP FLOP output.

Application examples

13.9 Dehumidifier management via several humidity probes

Using more than one humidity probes in the same system, the logic program controls a single dehumidifier according to the values read by each probe. As By-me systems are designed to be managed by only one humidity probe, this limit is overcome using the logic program.

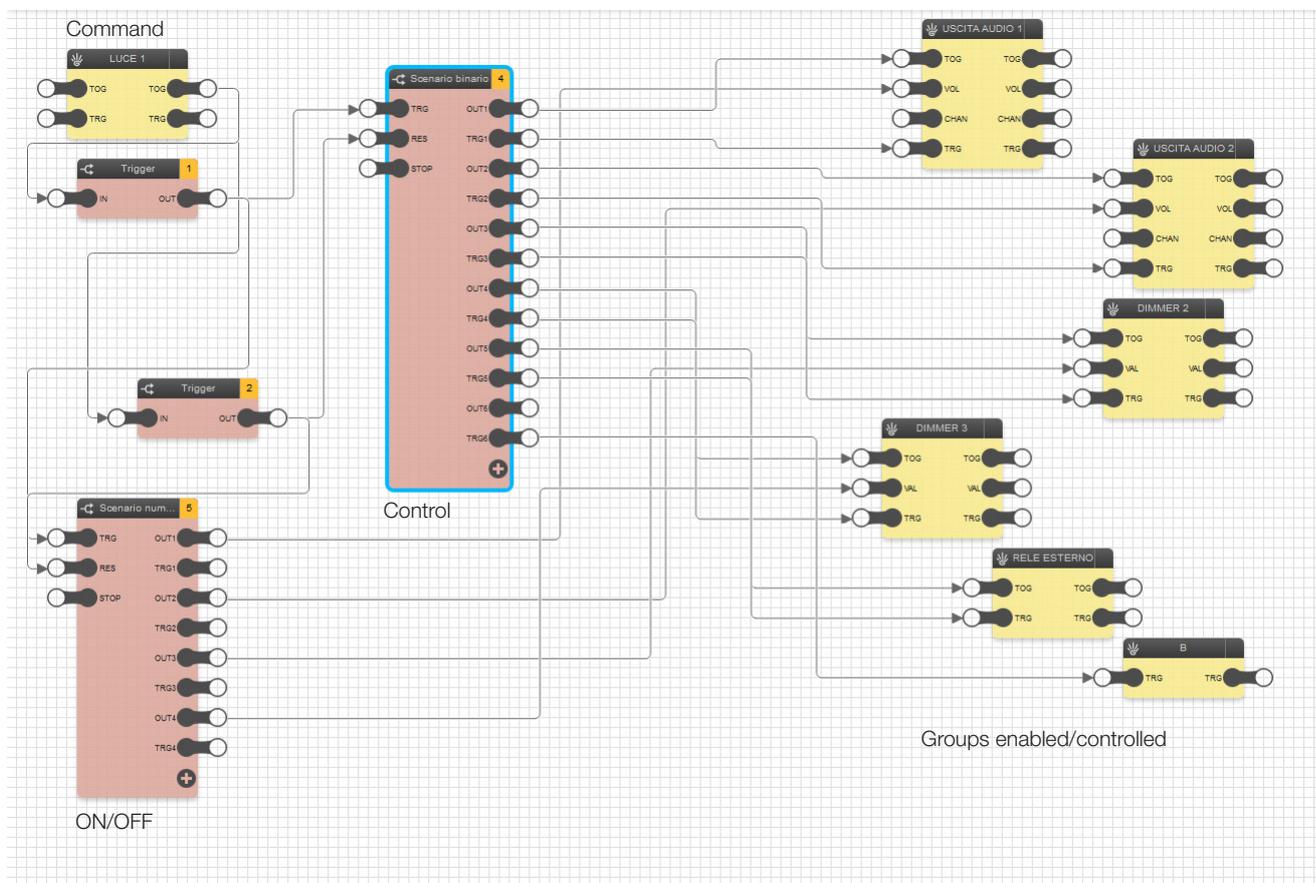


- The CLIMATE block is used to represent each of the probes present in the system (HUM output). In this example the humidity data is provided by a climate zone in which, as well as the thermostat, an input is configured in the group connected to a humidity sensor.
- Each HUM output is connected to the comparison block GREATER THAN; if the value at the input to input IN1 is greater than or equal to that set in IN2, 1 will be obtained in OUT (ON).
- All the comparison blocks are connected to the OR logic block as, to switch on the RELAY block representing the dehumidifier, it is sufficient for even only one of the probes measures a humidity value greater than or equal to that set.

Application examples

13.10 Multiple activations from a single control.

Via a single control, the logic program performs multiple activations, which not only control the group ON/OFF modes but also control the groups (for example, audio output with volume at 90% and lights at 50%).



- When a command is sent from the LIGHT 1 block the TRIGGER 1 block transmits an impulse only if on input it has received 1 while TRIGGER 2 sends the impulse only if it has received a 0.
- TRIGGER 1 transfers the impulse to the TRG input of the BINARY SCENARIO and NUMERICAL SCENARIO blocks, activating the various outputs of both scenarios; TRIGGER 2 on the other hand sends an impulse to the RES input of the two SCENARIO blocks, restoring all output values from them to 0.
- Some blocks have the possibility to be controlled by both scenarios as there is a need both to activate them and control their operation (for example, volume in the audio zone, brightness of the dimmer); therefore, at the same time commands sent from the NUMERICAL SCENARIO and the BINARY SCENARIO will reach the inputs of the same block at the same time.
- The BINARY SCENARIO block is set as follows to command the blocks AUDIO OUTPUT 1 and 2, DIMMER, etc.:

Proprietà generali

Scenario binario

<input checked="" type="checkbox"/>	TRG	<input checked="" type="checkbox"/>	OUT1
<input checked="" type="checkbox"/>	RES	<input checked="" type="checkbox"/>	TRG1
<input checked="" type="checkbox"/>	STOP	<input checked="" type="checkbox"/>	OUT2
		<input checked="" type="checkbox"/>	TRG2
		<input checked="" type="checkbox"/>	OUT3
		<input checked="" type="checkbox"/>	TRG3
		<input checked="" type="checkbox"/>	OUT4
		<input checked="" type="checkbox"/>	TRG4
		<input checked="" type="checkbox"/>	OUT5
		<input checked="" type="checkbox"/>	TRG5
		<input checked="" type="checkbox"/>	OUT6
		<input checked="" type="checkbox"/>	TRG6
		<input checked="" type="checkbox"/>	OUT7
		<input checked="" type="checkbox"/>	TRG7
		<input checked="" type="checkbox"/>	OUT8
		<input checked="" type="checkbox"/>	TRG8
		<input checked="" type="checkbox"/>	OUT9
		<input checked="" type="checkbox"/>	TRG9
		<input checked="" type="checkbox"/>	OUT10
		<input checked="" type="checkbox"/>	TRG10

Tipo: Scenario binario

Ordinamento: Manuale

Ordine:

Id: 65751

Intervallo Uscite:

Set Uscita 1:

Set Uscita 2:

Set Uscita 3:

Set Uscita 4:

Set Uscita 5:

Set Uscita 6:

Set Uscita 7:

Set Uscita 8:

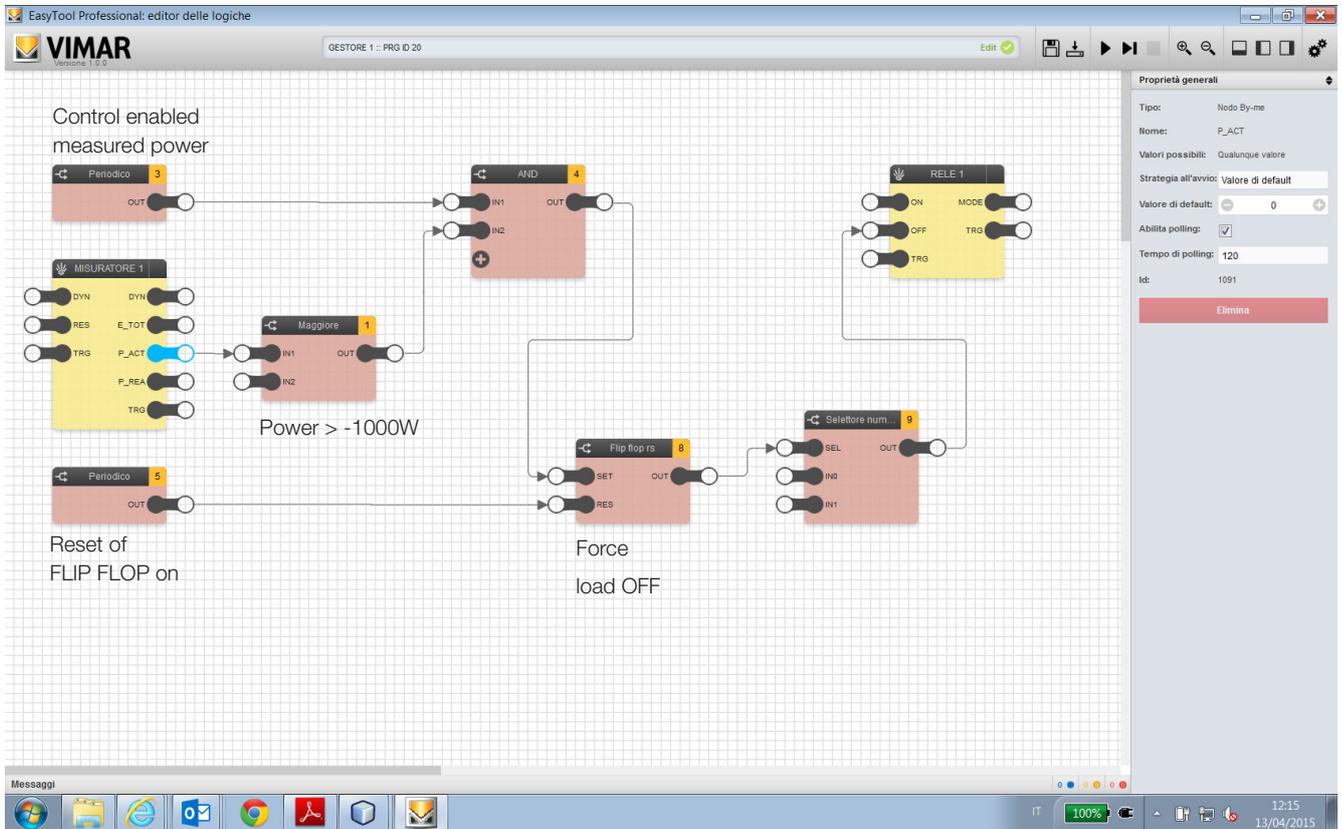
Set Uscita 9:

Set Uscita 10:

Application examples

13.11 Disabling of loads with delayed enabling (planned manually on the load in time bands with probable PHV production) when the available energy is not sufficient to power them.

Modern household appliances can be programmed to come on automatically at a given time; these programs are however cancelled if the power to the appliance is switched off for a certain period of time. If at the time the appliance is switched on there is not at least 1000 W of excess power, the logic program illustrated forces the relay connected to the appliance to OFF so that the set program does not run, thus avoiding consumption of energy from the mains electricity.

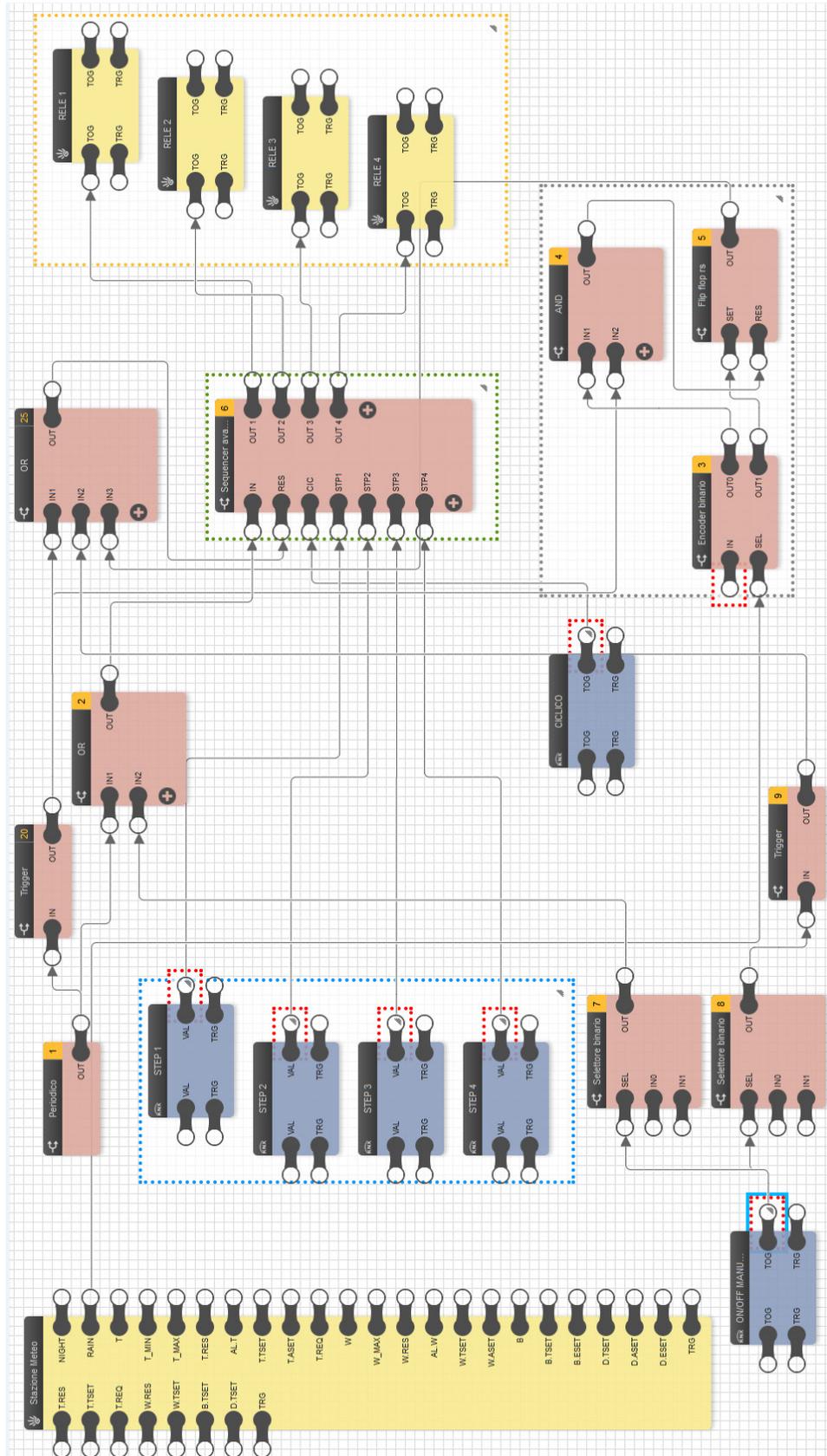


- A METER block is used in which, setting a polling time of 120 s on the output P_ACT, it updates the power data measured on the GREATER THAN block input every 2 minutes.
- The GREATER THAN block is set so that its OUT is TRUE only if the value measured by P_ACT is greater than -1000 W (for example -900) and indicates that the power produced but not used on the system does not exceed 1000 W.
- The AND block output becomes ON only if this condition is valid in the time interval set on the block PERIODIC 3.
- The planning on the block PERIODIC 3 is set to obtain activation only for 5 minutes prior to the appliance being switched on; in this way, the control is run only in this time interval, while outside of this it does not interfere with the load function.
- When the block AND goes to ON, the OUT on the FLIP FLOP RS goes to ON and remains in that state even after the block AND returns to OFF; this means that the load remains forced OFF (the OFF input of the load is kept on by means of the NUMERICAL SELECTOR) until the FLIP FLOP RS block is reset.
- The FLIP FLOP RS is reset by sending an ON to the RES node via the block PERIODIC 5; the planning sends ON to a time after that of the block PERIODIC 3 and in a sufficiently large time interval to allow the load to have been switched off and the programs set on the appliance deleted.

Application examples

13.12 Sequential irrigation, dependent on the clock and rain sensor, forced start-up and with duration parameters that can be changed via the user interfaces.

The logic program, within the time frame set in the PERIODIC clock block, irrigates 4 different zones by switching them on in sequence. The duration of each zone, identified by the respective STEP block, can be set by the user dynamically via the supervision interfaces (Web server and APP, video touchscreen, touch-screen). Cyclical start-up of the program can be forced manually as well as its reset; in the event of rain, automatic start-up is inhibited.



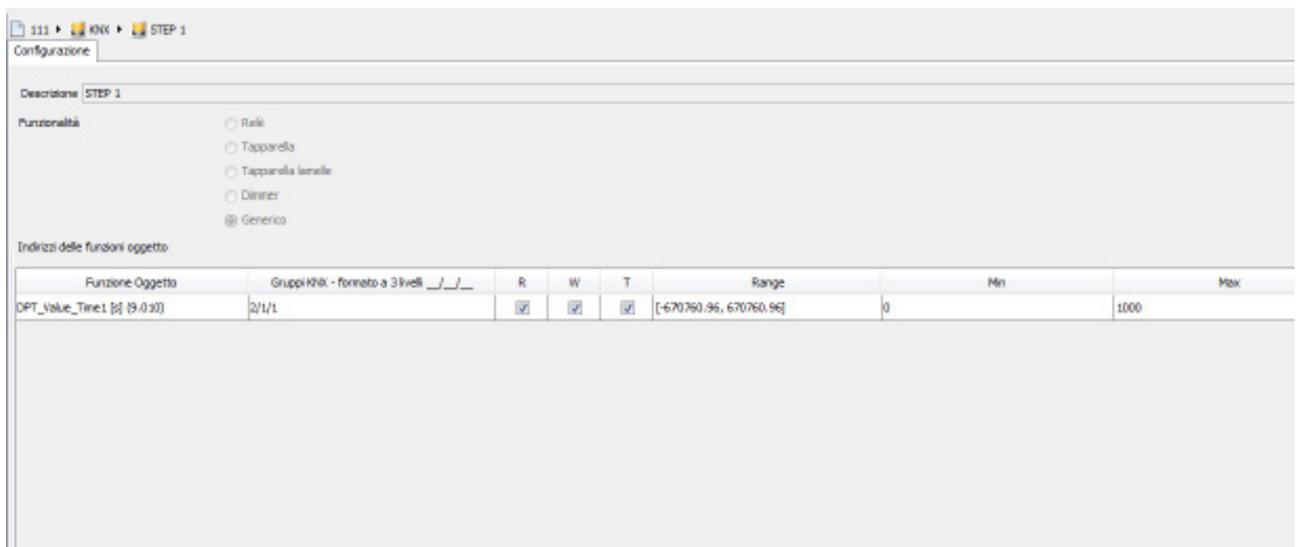
Application examples

- The PERIODIC block initiates activation of the irrigation sequence in the set time interval; at the end of that time the TRIGGER block performs an automatic reset, turning off the outputs (RELAY blocks) associated with each area.
IMPORTANT: For the program to work properly, set "Edge=False" on the TRIGGER block.
- The MANUAL ON/OFF block lets you:
 - manually activate the irrigation sequence outside the time range set in the PERIODIC block.
 - manually force deactivation of the outputs inside the time range set in the PERIODIC block.
 Downstream of the MANUAL ON/OFF block there are the BINARY SELECTOR blocks that allow the dual function of ON and OFF.
- The CYCLIC block allows the user to repeat the output activation sequence multiple times in the time range set in the PERIODIC block.
- The OR blocks guarantee the correct interpretation of the commands sent by the three blocks described above.
- The 4 zones are identified by the respective STEP blocks in which the user can set the activation duration in s via the Web Server and/or video touchscreen (ZONE 1 TIME for STEP 1 block, and so on).
- In the event of rain, the WEATHER STATION block inhibits activation of the irrigation sequence; the BINARY ENCODER, AND and FLIP FLOP RS blocks enable storing the rain event so that the sequence is not activated in the time range set in the PERIODIC block. Normal operation is then restored with the next ON command sent by the PERIODIC block (for example, the day following the one when it rained).

The procedure to perform to make the logic program operational is as follows:

1- Using EasyTool Professional, create a new KNX group for each zone (STEP 1, STEP 2, etc.):

- Select the drop-down menu **Configure -> KNX third-party integration -> New KNX Group**.
- Enter the data for the new group: **Description** (mandatory) and **Generic**; this latter option enables a drop-down menu that lets you select the generic (DPT) communication object.
Note that you can select only one DPT for each generic group.



CAUTION:

- For the STEP blocks use virtual numeric DPT 9.010.
- For the MANUAL ON/OFF and CYCLIC blocks use virtual numeric DPT 1.001.

2- Create the logic program and download it onto the logic unit.

3- Import the .xml file on the Web Server (or create the respective pages on the video touchscreen) and configure the environments/groups involved.

The logic program can be run by the Web Server and/or by the video touchscreen with the logic unit connected to the bus. Here, by way of example, are screens that allow managing the logic program:

Application examples

Web Server



Video touchscreen

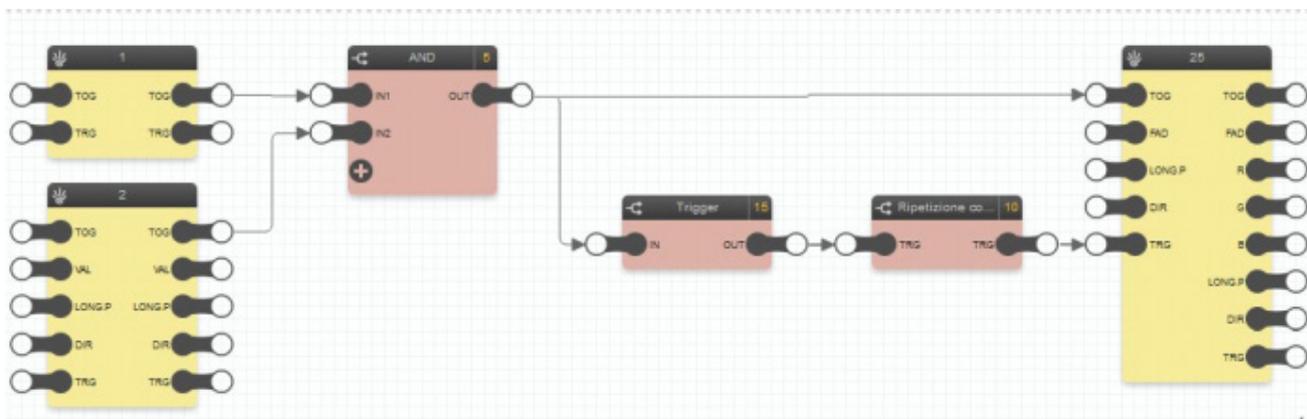


Application examples

13.13 Forced OFF command repeated 5 times.

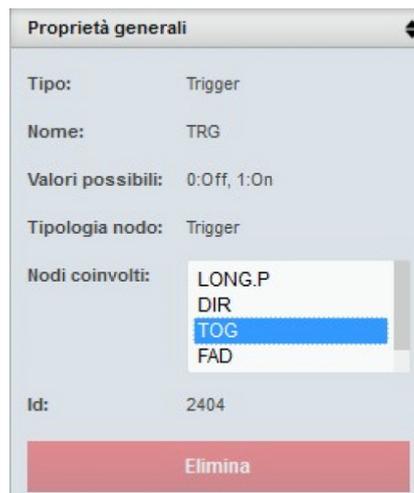
The logic program enables RGB dimmer management with a By-me command and a traditional dimmer.

With the REPEAT COMMAND block you send the OFF command to the RGB dimmer 5 times; this override is accomplished with the certainty that this command is received (thus avoiding losing the message).



You use the block 1 (BY-ME COMMAND) and block 2 (DIMMER) to control the RGB block by connecting their inputs to the AND block.

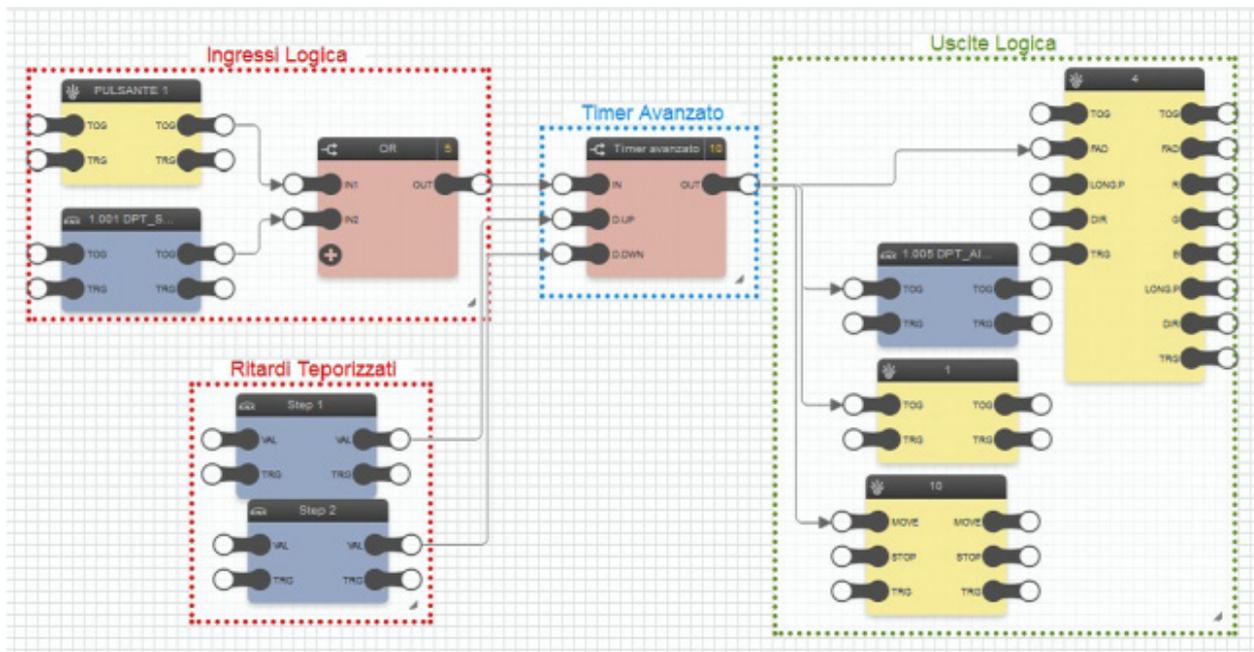
- The REPEAT COMMAND block, preceded by the TRIGGER block that allows a pulse to be obtained at its TRG input, enables repeating the OFF command 5 times at intervals 1 s apart from each other.
- The REPEAT COMMAND block is set as follows:



Application examples

13.14 Parameterization by user of up and down times for actuators (e.g., staircase light function).

The example shows how from a logic input (By-me or KNX DPT buttons) you can activate an output (By-me or KNX DPT actuators) via a delay logic (timer) and dynamically change their on/off delays via the supervision interfaces (Web server and APP, video touchscreen, touchscreen). One example of using this function is the management of timed lights, such as staircase lights.



Inputs:

IN = Input ON/OFF signal (MIXED)

D.UP = Up Delay (STATUS)

D.DWN = Down Delay (STATUS)

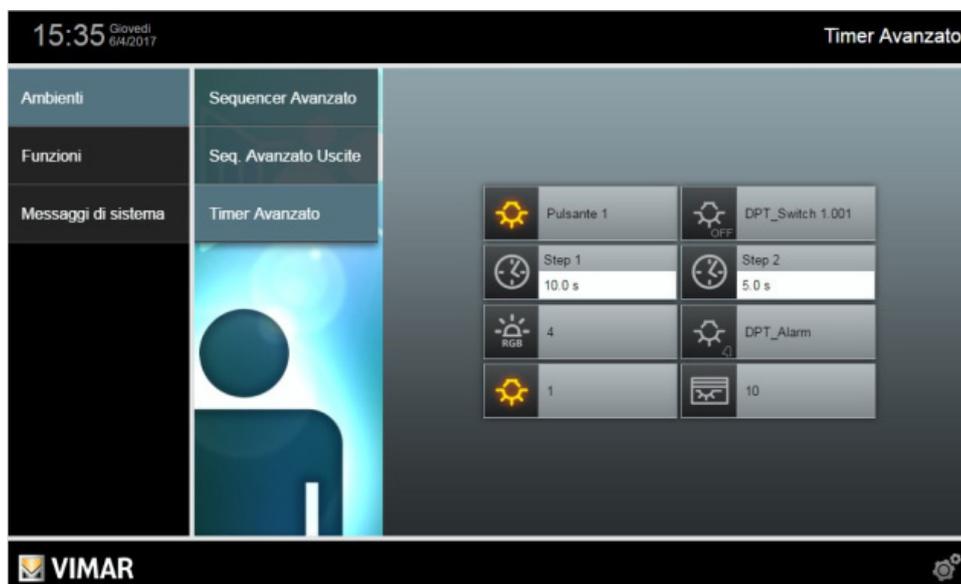
Outputs:

OUT = Output ON/OFF signal (STATUS)

The procedure to perform to make the logic program operational is as follows:

- 1- Using EasyTool Professional, create the virtual DPT datapoint groups (the procedure is similar to the one illustrated in example 13.12).
- 2- Create the logic program described above and download it onto the logic unit.
- 3- Import the .xml file on the Web Server (or create the respective pages on the video touchscreen) and configure the environments/groups involved.

The logic program can be run by the Web Server and/or by the video touchscreen with the logic unit connected to the bus. Here, by way of example, is the Web Server screen that allows managing the above-mentioned logic program:





01468 IEN 14 1907



VIMAR

Viale Vicenza, 14
36063 Marostica VI - Italy
www.vimar.com