ZENNIO APPLICATIONS

# KES

# Progressive load swicth-off after an over-consumption alarm

Edition 1

## INDEX


1. Introductión ..... 3

2. Introductión to devices ..... 4

3. Installation performance ..... 6

   3.1. Load types ..... 6

   3.2. Progressive load switch-off system ..... 7

   3.2.1. Load management ..... 8

   3.2.2. Power overconsumption control using KES ..... 8

4. ETS configuration ..... 10

   4.1. Parameterization ..... 10

   4.1.1. KES ..... 10

   4.1.2. IRSC Plus ..... 17

   4.1.3. ACTinbox quatro ..... 18

   4.2. Topology ..... 19

   4.3. Group Addresses ..... 20

# 1. INTRODUCTIÓN

In typical electrical installations, when the maximum power consumption allowed is reached, all the working loads are typically switched-off and won't switch on again until the consumption is under the maximum again. This unexpected fault of power supply can cause damage to sensitive electronic equipment, loss of information in computers, or interruption of food preservation in refrigeration equipment.

**Zennio energy saver, KES**, allows triggering an alarm when excessive power consumption is detected. This allows the user to prevent sudden disconnections of all the loads on over-consumption situations.

The **aim** of this document is to provide guidance for integrators to enable them to easily program a progressive disconnection of loads in a KNX installation with no more help than **KES**. This system will allow the user to select the order in which the loads are disconnected and make a smooth and orderly shutdown of the facility. Furthermore, the solution described in this document will allow the orderly restoration to baseline after the overconsumption situation.

# 2. INTRODUCTIÓN TO DEVICES

As already mentioned in the introduction, the progressive load disconnection will be implemented with the help of KES, however, to complete the example other actuator devices are needed. All the devices used are listed below:

- **KNX Energy Saver, KES (Ref. ZN1IO-KES)**

  Allow instantaneous power monitor and report the alarm on overconsumption situation that starts the progressive load disconnection.

  The logic of the progressive load disconnection system will be made with the help of logical functions module that includes this device. In particular, this device by itself can control up to 4 loads.

  If you need to control more loads, you can use any of the logical functions modules that incorporate all Zennio actuators. One logical function is needed per load.

  You can find further information in the device specific manual available on the website of Zennio (www.zennio.com).


Figure 1 KNX Energy Saver, KES

- **IRSC (Ref. ZN1CL-IRSC)**

  IRSC is a KNX-IR interface that allows controlling a split HVAC system through infrareds. It allows switching on and off the unit, set the setpoint temperature, operation mode, fan speed and swing.

  In this project, we will make use of this device along with the application program **IRSC Plus** to control the operation of the HVAC system. To simplify the example, in this project we only integrate the On / Off control.

  You can find further information in the device specific manual available on the website of Zennio (www.zennio.com).


Figure 2 IRSC Plus (HVAC IR controller)

- **ACTinBOX QUATRO (Ref.ZN1I0-AB40)**

ACTinBOX QUATRO is a KNX actuator with 4 outputs and logical functions. You can find further information in the device specific manual available on the website of Zennio (www.zennio.com).

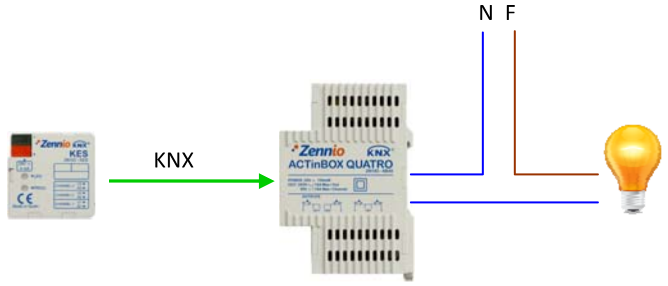

Figure 3 ACTinBOX QUATRO (4 outputs actuator)

# 3. INSTALLATION PERFORMANCE

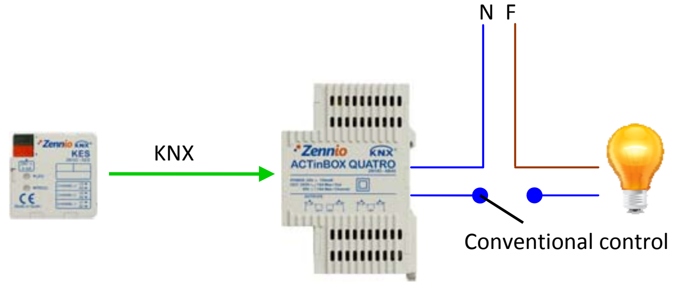
## 3.1. LOAD TYPES

We can easily establish the following classification according to the load type and how to manage them:

1. **Light loads**:
   a. **Integrated into the KNX system**: we can act on a communication object of the type "On / Off".

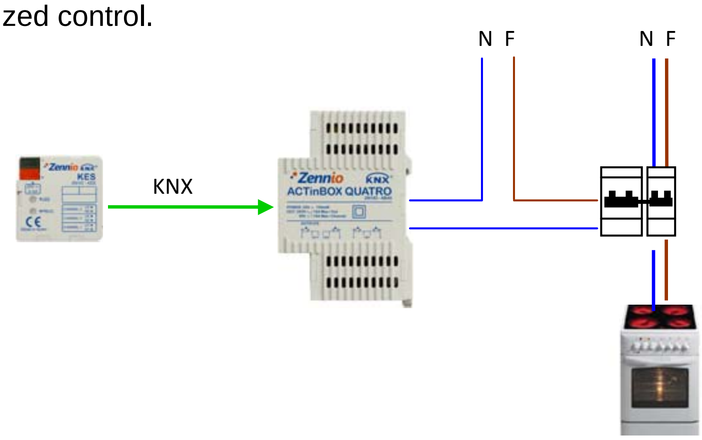



   b. **Not integrated**: the main control is conventional (a typical switch) but we can act directly on the power supply.




2. **Heavy loads**:
   a. **Integrated into the KNX system**: we can act on a communication object of the type "On / Off".

b. **Not integrated**: we will need to act on the magnetothermic circuit breaker and perform the connection/disconnection with the help of a motorized control.




## 3.2. PROGRESSIVE LOAD SWITCH-OFF SYSTEM

**Installation scheme**:

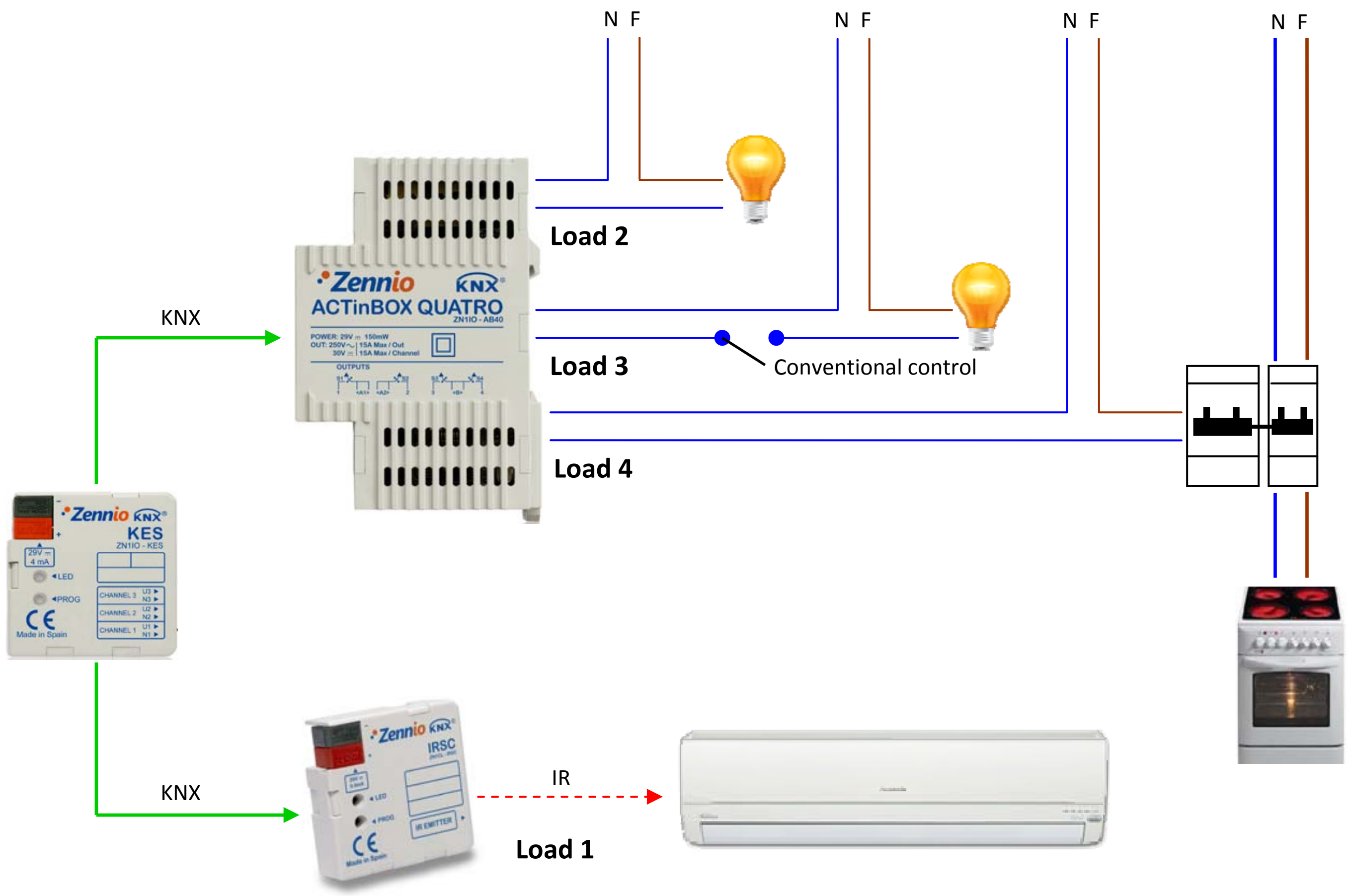



Figure 4 Installation scheme

### 3.2.1. LOAD MANAGEMENT

In the Project described in this guide we will use KES to manage 4 loads, despite the specific KNX control for these loads.

- **Load 1**: **Heavy load integrated in the KNX system**. E.g. HVAC system integrated by using the interface IRSC Plus.
- **Load 2**: **Light load integrated in the KNX system**. E.g. Lighting circuit controlled by the output number 1 of the ACTinBOX QUATRO.
- **Load 3**: **Light load not integrated in the KNX system**. E.g. Lighting circuit with conventional control (not KNX) controlled by the output number 2 of the ACTinBOX QUATRO.
- **Load 4**: **Heavy load not integrated in the KNX system**. E.g. Single circuit for the oven controlled by a magnetothermic circuit breaker with the help of a motorized control. It will be controlled by the output number 3 of the ACTinBOX QUATRO.

The present example show that any type of load can be controlled using the system described in this guide, no matter the type of the load.

### 3.2.2. POWER OVERCONSUMPTION CONTROL USING KES

It will be necessary to enable the **Power Limit Monitoring** in the KES and set an **Initial Value** for the **Upper Limit** slightly lower than the maximum allowed. The KES must be configured to send '**1**' through the **Over-consumption Alarm** object when the instant power reach the upper limit and '**0**' when it drops below the limit.

The progressive load switch-off system will start with the activation of the **Over-consumption Alarm**. A value of '**1**' received through this object will activate a configurable timer which after a time $\mathbf{T_0}$ will disconnect the first of the loads. If a new period $\mathbf{T_0}$ elapses and the alarm remains, the next load will be disconnected, and so on.

At the time the alarm disappears (**Over-consumption Alarm** = "**0**"), the system will begin to recover its initial status. A period of $\mathbf{T_0}$ must elapse between activation and activation.

This system is able to recover the status in which the charges were before the disconnection. If they were disconnected, they will remain disconnected and if they were active, they will be activated after the disappearance of the alarm.

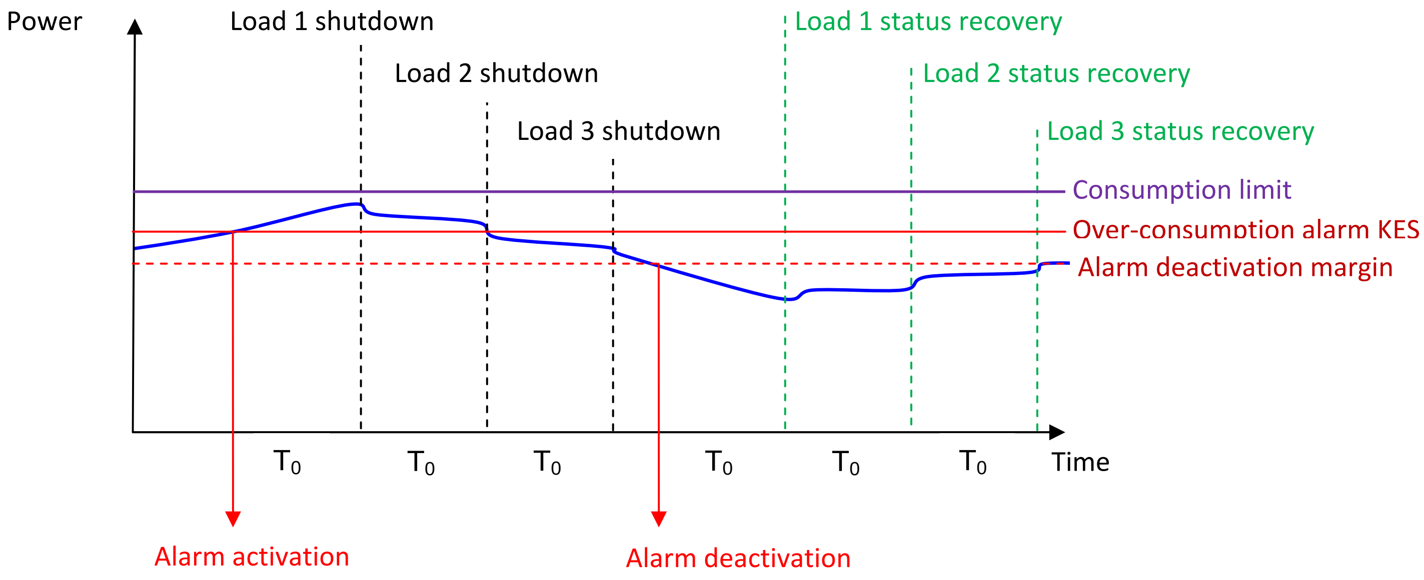



**Figure 5 System performance graph**

**Note**: If a load is manually activated during the progressive load switch-off process, it will be deactivated again. If the load is activated once the progressive load switch-off process has finished (the last load has been disconnected), it won't be deactivated until the alarm disappears.

# 4. ETS CONFIGURATION

## 4.1. PARAMETERIZATION

In the following paragraphs, the parameters for configuring the different devices in order to implement this application are detailed.

### 4.1.1. KES

**"POWER LIMIT MONITORING" CONFIGURATION**

Firstly, in the KES "**General**" tab we must set the electrical grid features: AC power supply voltage, power factor, etc., and enable the "**Channel A**" that will be monitored in this project.

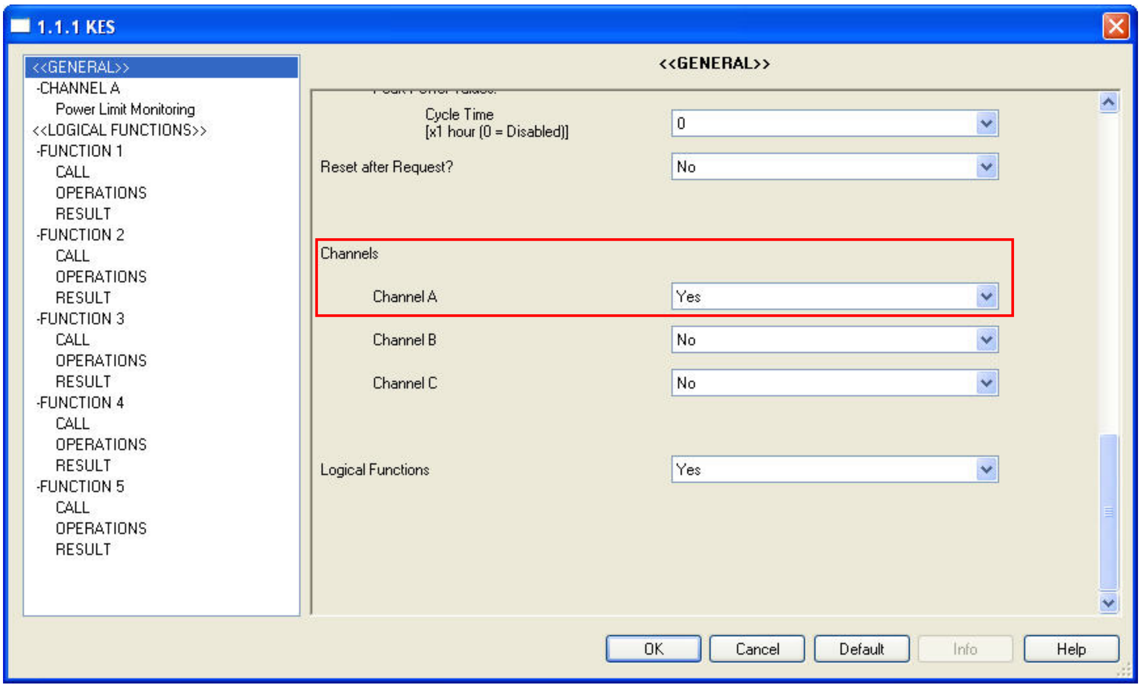



**Figure 6 Enablement of Channel A**

Within the tab that has just appeared for the channel A, we must enable the "**Power Limit Monitoring**" that should be configured as follows:

- **Upper Limit**:
  - o **Initial Value →** limit expressed in tens of watt that will trigger the overconsumption alarm. E.g. 3,000 W
  - o **When over Limit, send**… **→ 1**
  - o **Send object when under Limit? → Yes**

- **Send on Value Change** → if wanted, you can set the margin below which you want to deactivate the overconsumption alarm expressed in tens of watt. E.g. 100 W

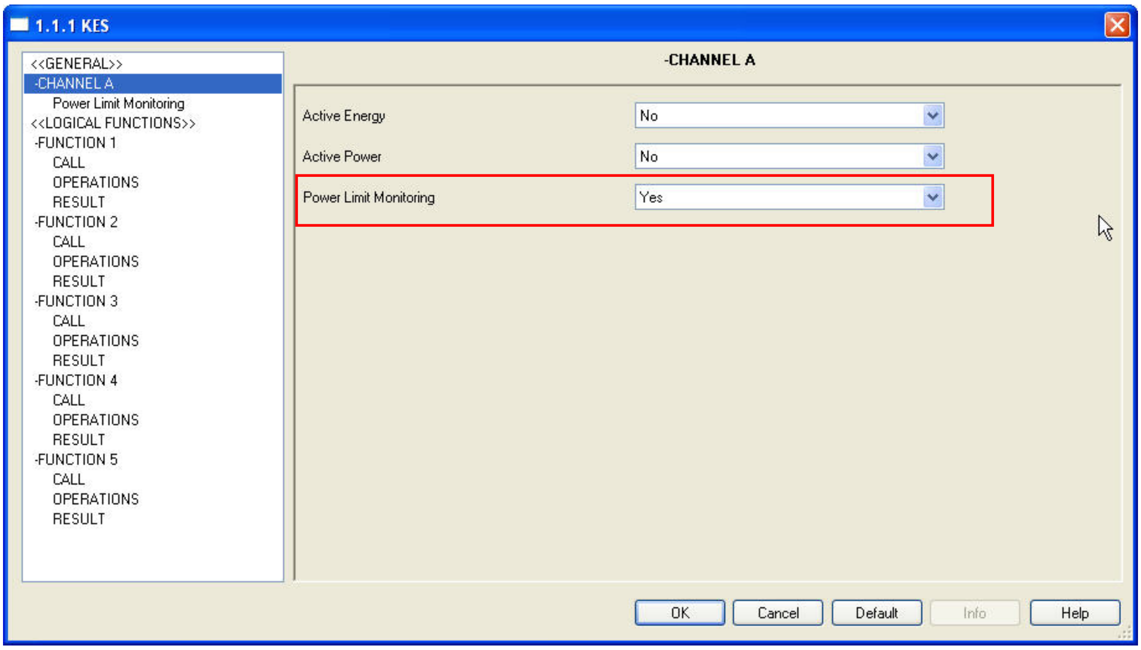



Figure 7 Enablement of the Power Limit Monitoring

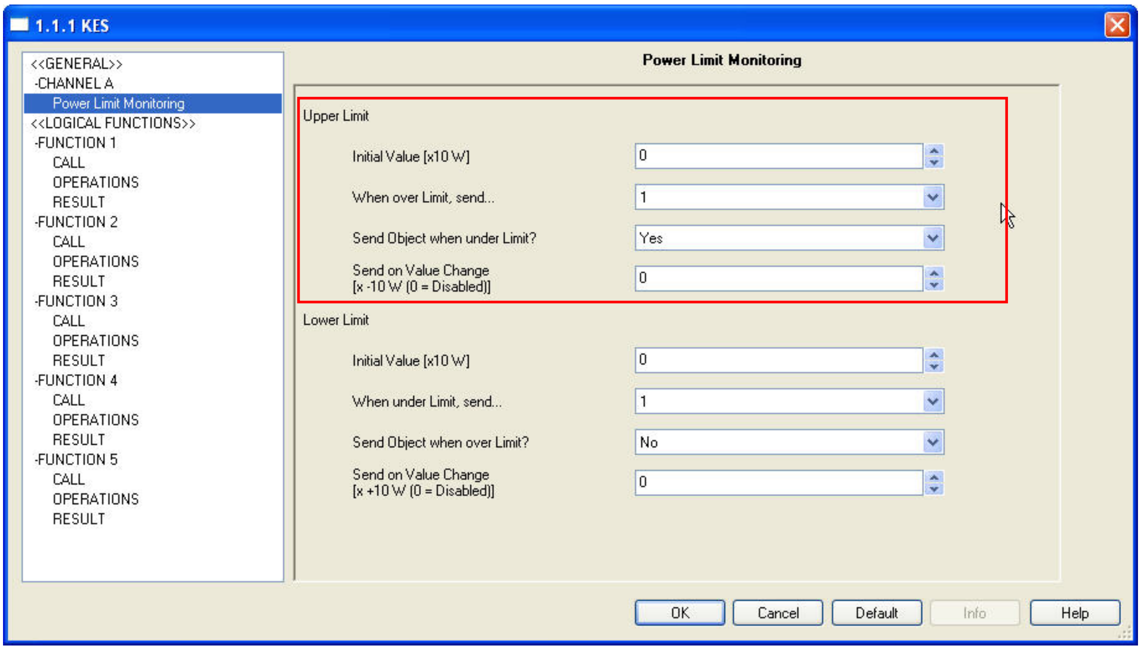



Figure 8 Upper Limit parameterization

**CONFIGURATION OF THE PROGRESSIVE LOAD SWITCH-OFF FUNCTION:**

The progressive load switch-off function comprises two types of functions:

1. **Base function or counter**: this is the function that manages the number of loads to be disconnected or connected at each time step. It is configured as a counter that is incremented each period $T_0$ while the alarm is active, and is decreased each period $T_0$ if the alarm is disabled. The counter can take values between 0 and the number of controlled loads.
2. **Secondary function**: this is the function that executes the disconnection or restoration of the load as indicated by the base function. It will receive the counter value from the base function and will decide the status of its controlled load.

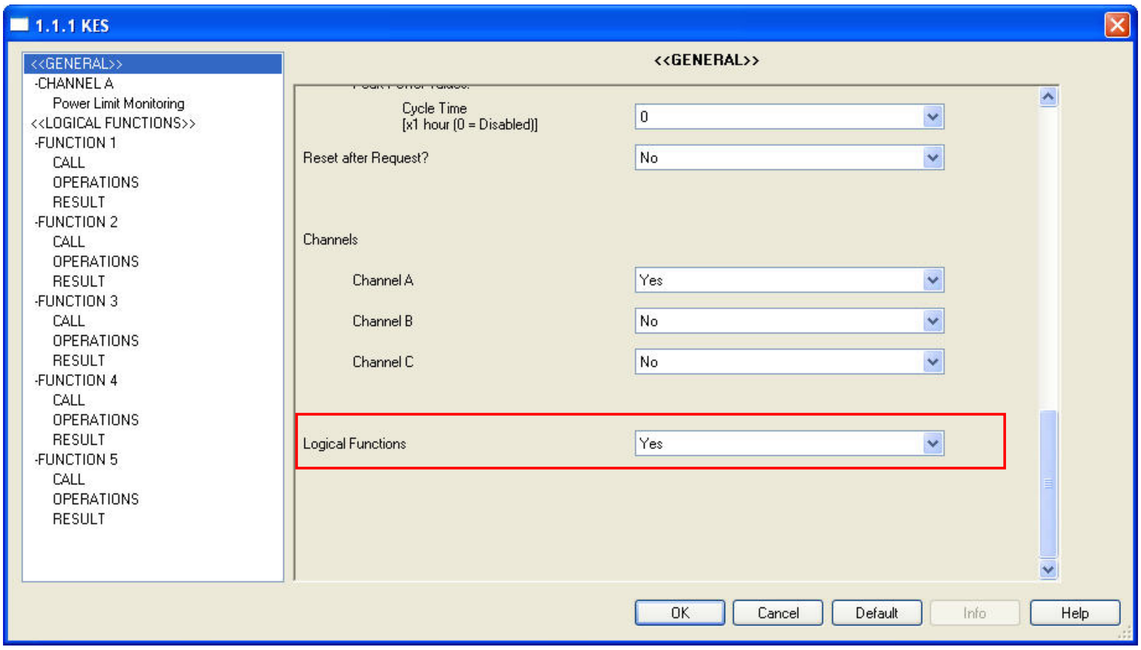
The system consists of at least one basic function, and as many secondary functions as loads we want to control. In this project we managed 4 loads so we will require 4 secondary functions.

These functions will be implemented with the help of the logical functions module included in the KES. We need one logical function per secondary function and one more for the base function, so it will be necessary to enable the five logical functions available.

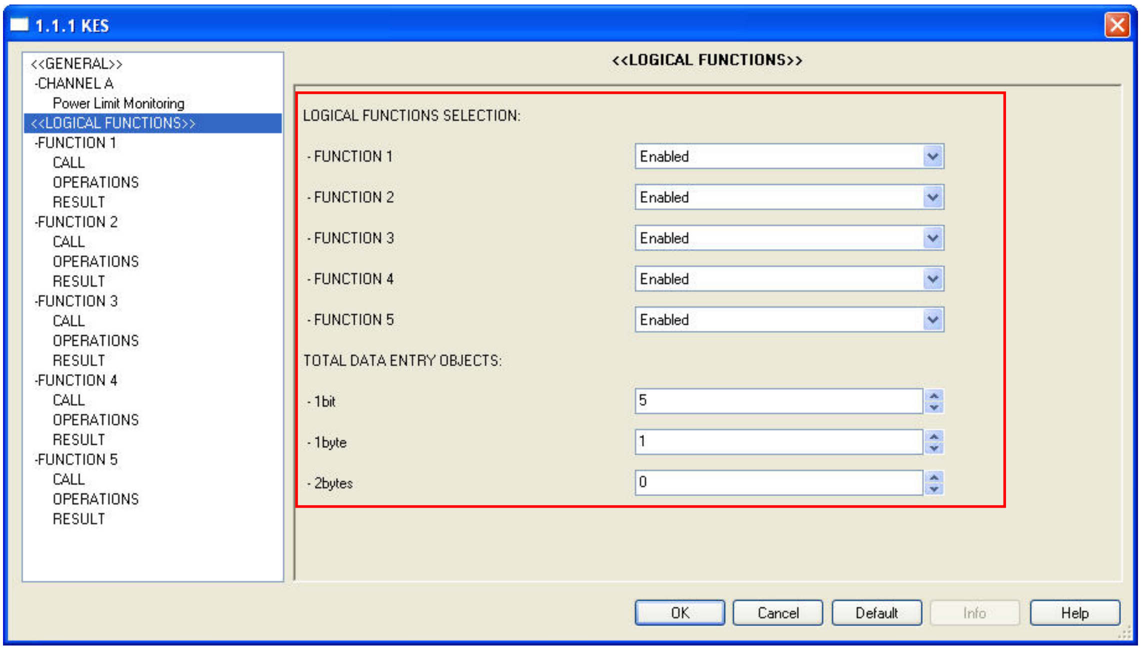
Number of data entry objects needed:

1. **Base function or counter**: it needs one 1-bit object for the status of the overconsumption alarm, and one 1-byte object to store the current value of the counter.
2. **Secondary function**: each secondary function needs one 1-bit object to store the status of the load previous to the disconnection, and one 1-byte, shared with the other secondary functions, to receive the value of the counter from the base function.

To sum up, the system will need five 1-bit objects and only one 1-byte object.




## Figure 9 Enablement of the Logical Functions module




## Figure 10 Configuration of Logical Functions and Data Entry Objects

**CONFIGURATION OF THE BASE FUNCTION OR COUNTER:**

**Call objects**:

- (1 bit) Data Entry 1 → Overconsumption alarm status
- (1 byte) Data Entry 1 → Counter current value

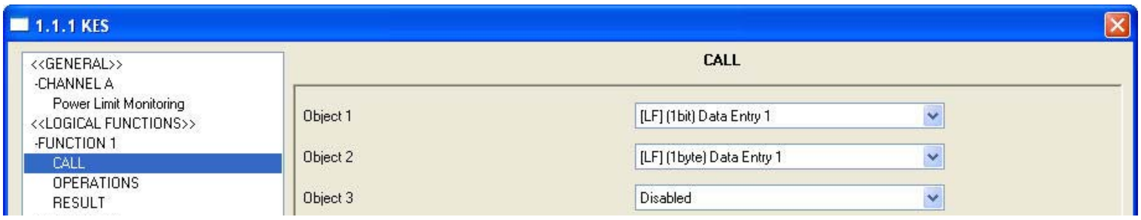
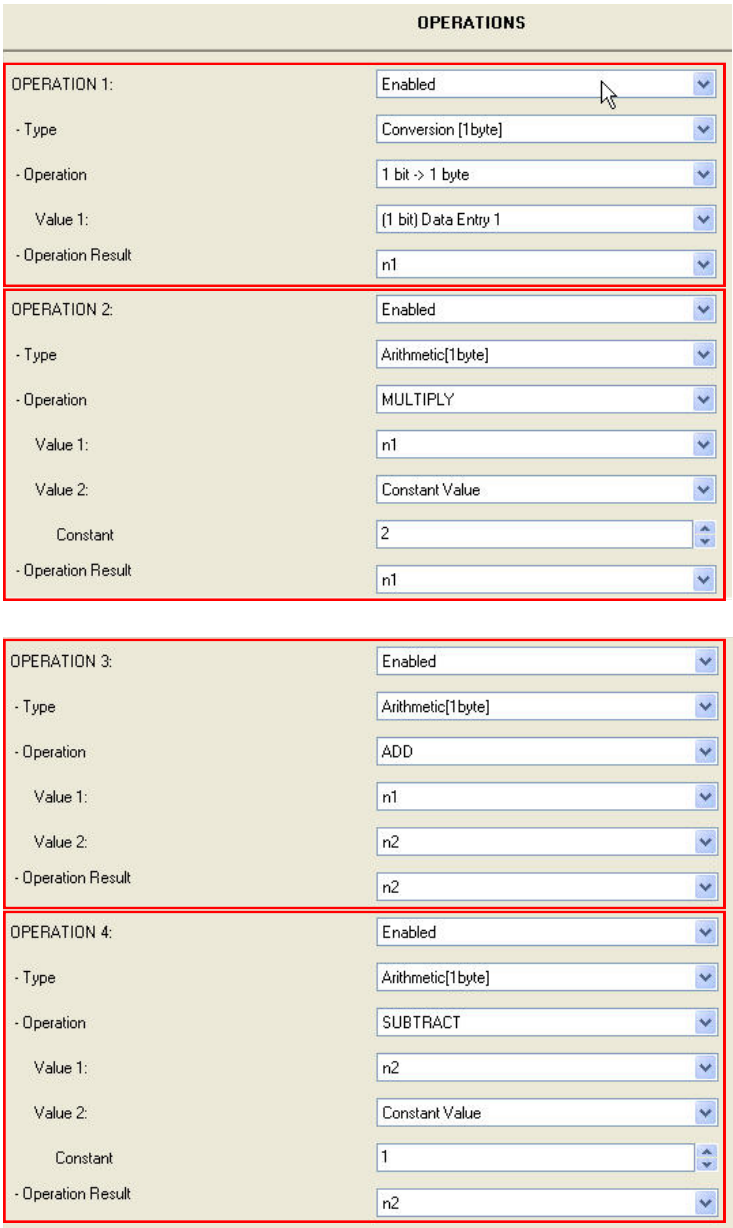



**Figure 11 Configuration of the Base Function call**

**Operations**:




**Operation 1**: it converts the 1-bit data entry (alarm status) into a 1-byte value.

**Operation 2**: it multiplies the result value of the previous operation by 2. n1 can take the values 0 (no alarm) or 2 (alarm) according to the status of the alarm.

**Operations 3 and 4**: it sums the result value of the previous operation to the current value of the counter (n2) and subtracts 1. In this way, we achieve to increment (+1) or decrement (-1) the counter value according to the status of the alarm.

**Figure 12 Operations of the Base Function**

**Result**:

- Type: 1 byte
- Value: n2 → Current counter value
- Sending: Result is different from the last sent
- Restriction: Values lower than reference one: 5 → Number of controlled loads (4) + 1.
- Delay: 10 seconds → Desired value for To (delay between disconnections)

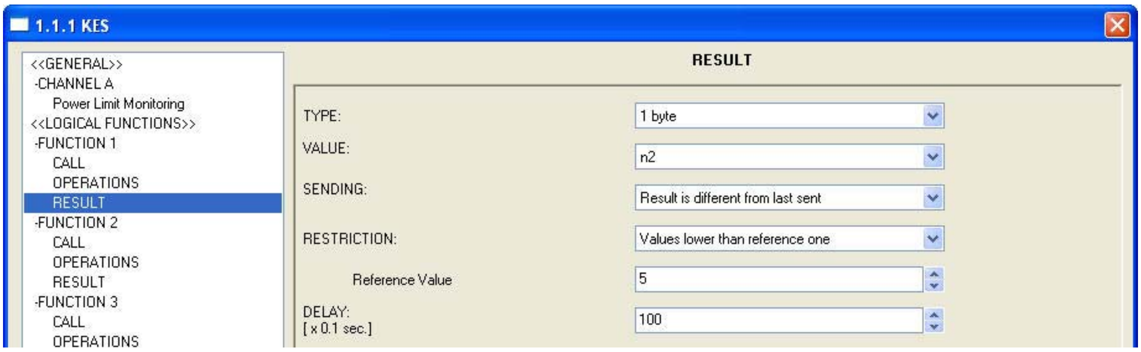



**Figure 13 Configuration of the Base Function result**

**Note**: the minimum time required by the KES to measure and update its variables is approximately 2 seconds, so the delay To set should be greater than this value.

**CONFIGURATION OF THE SECONDARY FUNCTION:**

**Call object**:

- (1 byte) Data Entry 1 → Current value of the counter

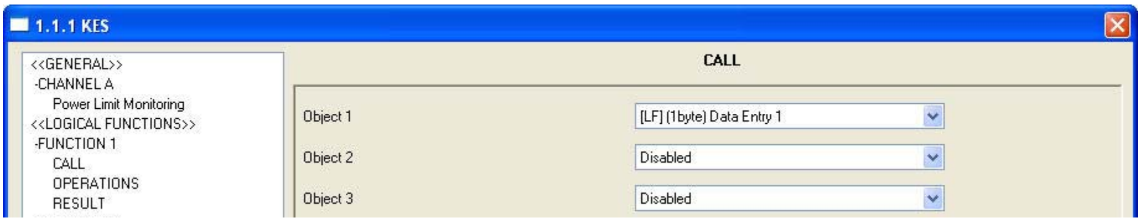
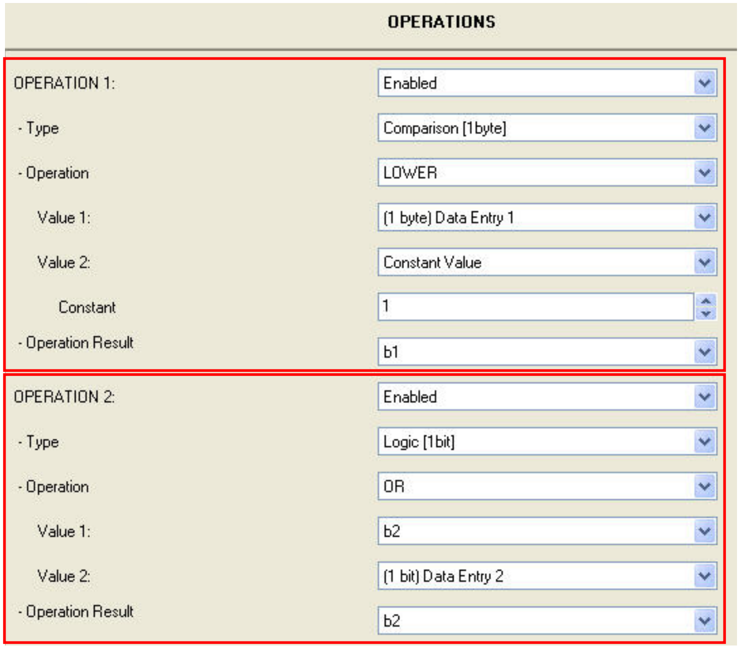


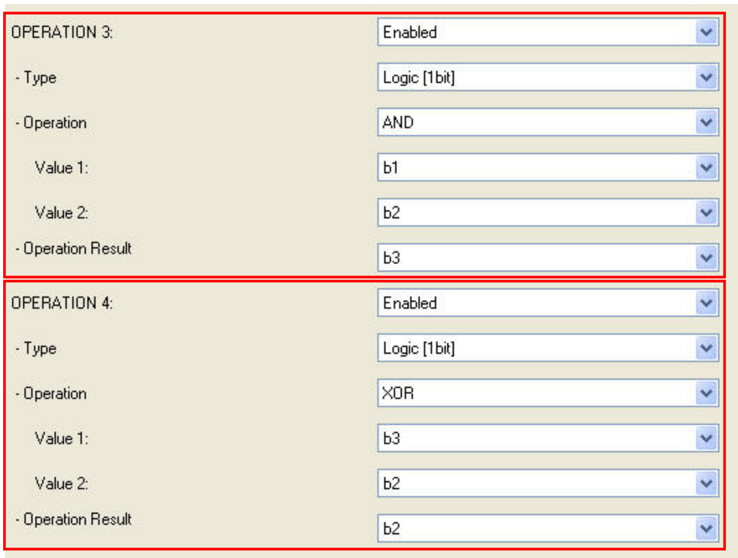

**Figure 14 Configuration of the Secondary Function call**

**Operations**:




**Operation 1**: comparison between the counter value sent by the base function ((1 byte) Data entry) and the value of the load controlled by this secondary function (1). If the counter value is greater or equal to the load value, the load will be disconnected, if the load is lower, it will be restored.

**Operation 2**: this operation stores (in b2) the status of the load previous to the disconnection ((1 bit) Data entry 2).




**Operation 3**: this operation decides the result of the function (b3) according to the result of the first operation (b1): disconnect the load (b3=0) or restore the load (b3=b2).

**Operation 4**: it resets the stored value for the status of the load if this load has been restored (b2=0) or it keeps the status of the load if it remains disconnected (b2=b2).

**Figure 15 Operations of the Secondary Function**

**Note**: the constant value defined in the first operation identifies the load and the order in which they will be disconnected. This value should be modified in each secondary function in a sequential way.

**Result**:

- Type: 1 bit
- Value: b3 → Function result
- Sending: Whenever function is executed
- Restriction: No restriction
- Delay: 0 seconds

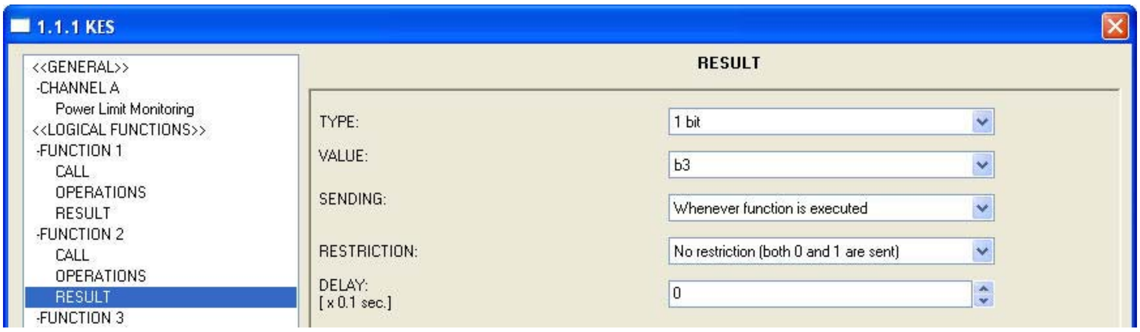



**Figure 16 Configuration of the Secondary Function result**

### 4.1.2. IRSC PLUS

In the general parameters tab of the IRSC Plus, it will be necessary to define the model of the AC unit that is going to be controlled. To do this, you should consult the correspondences table available in the download area of the Zennio website.

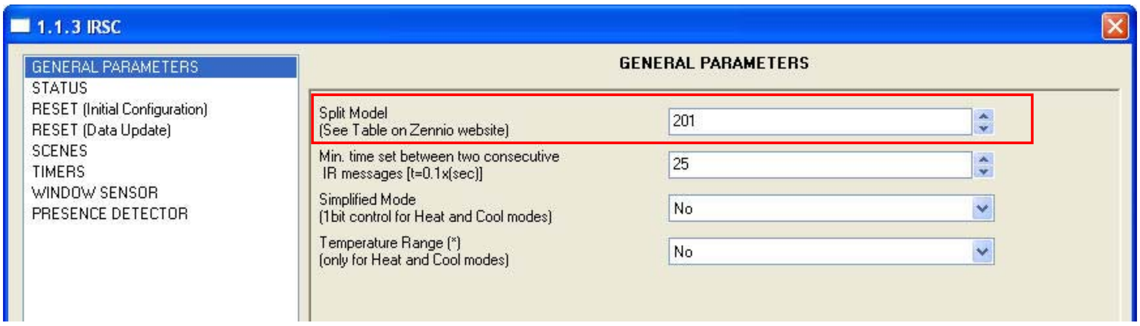



**Figure 17 Configuration of the AC Split Model in the IRSC Plus**

In the present example, we are only going to use the On/Off control of the AC unit. The "On/Off" object appears by default but it will be necessary to enable the status object "On/Off (status)" in the STATUS tab.

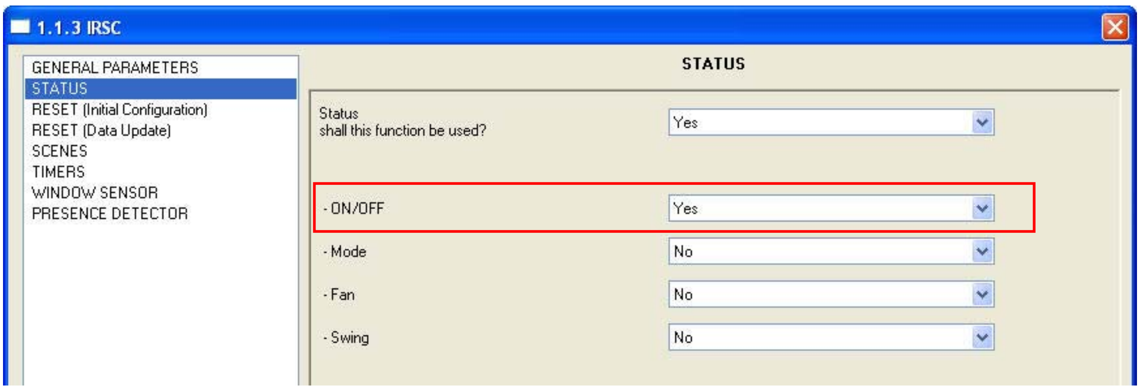



**Figure 18 Enablement of the status object for the On/Off control of the AC unit**

### 4.1.3. ACTINBOX QUATRO

The loads 2, 3 and 4 of the present project are going to be controlled using the outputs of this actuator so we need to enable 3 of the 4 outputs available in this actuator with their corresponding status objects. To do this, we need to enable the outputs in the GENERAL tab.

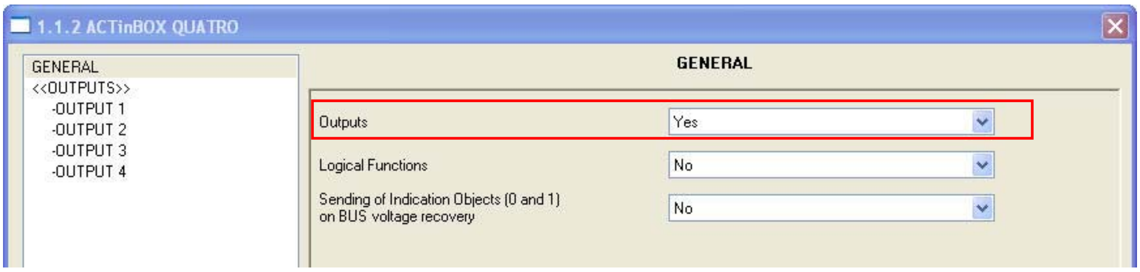



Figure 19 Enablement of the outputs of the actuator ACTinBOX QUATRO

In the OUTPUTS tab that has just appeared we must set the channels A and B as "individual outputs" and enable the outputs 1, 2 and 3 with the default configuration.

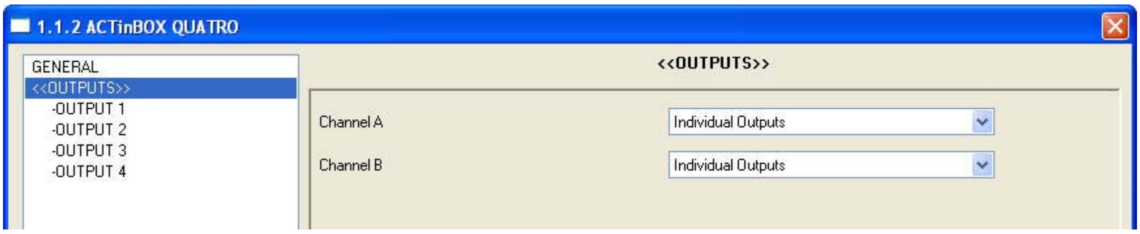



Figure 20 Configuration of the individual outputs in the ACTinBOX QUATRO

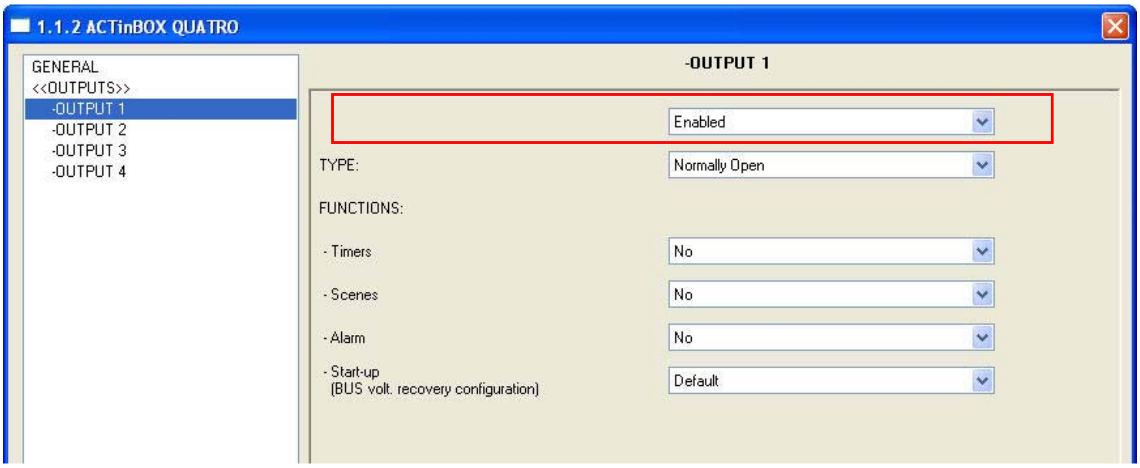
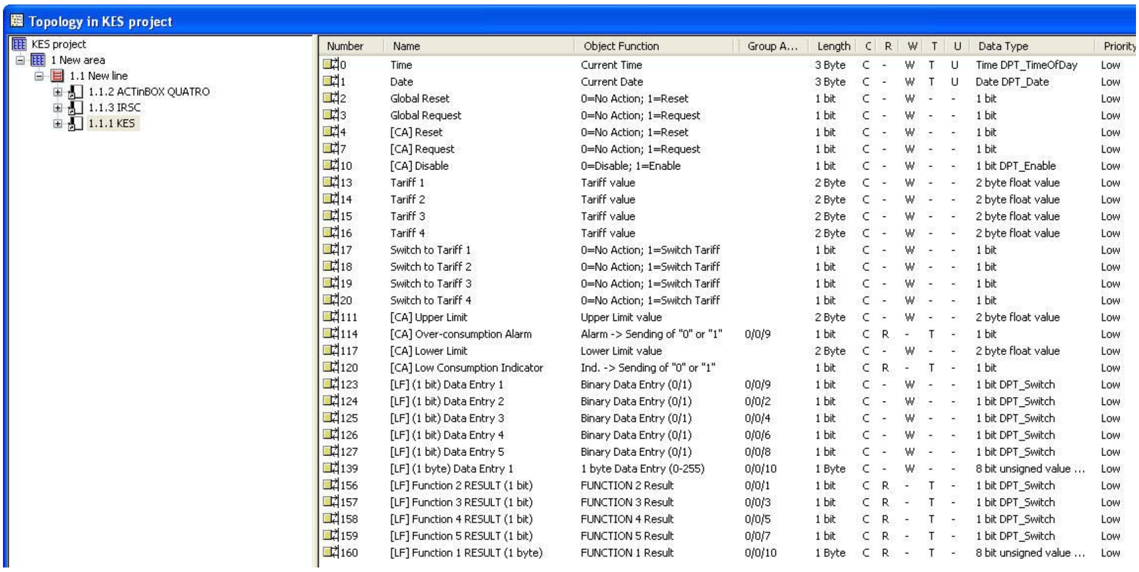



Figure 21 Enablement of one of the outputs of the actuator ACTinBOX QUATRO

## 4.2. TOPOLOGY

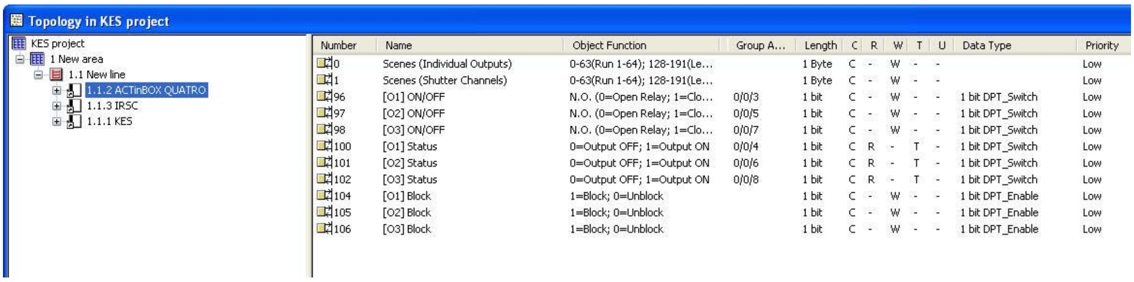
Next, it is shown the topology for the programming explained in the previous section.

| DEVICE | PHYSICAL ADDRESS |
|---|---|
| KES | **1.1.1** |
| ACTinBOX QUATRO | **1.1.2** |
| IRSC Plus | **1.1.3** |



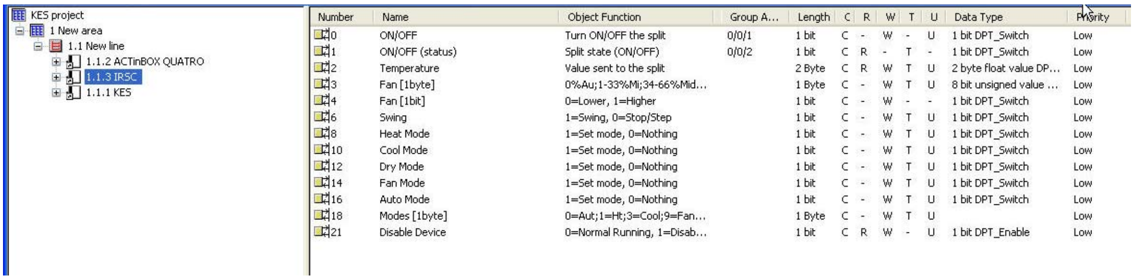

| Number | Name | Object Function | Group A... | Length | C | R | W | T | U | Data Type | Priority |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Time | Current Time | | 3 Byte | C | - | W | T | U | Time DPT_TimeOfDay | Low |
| 1 | Date | Current Date | | 3 Byte | C | - | W | T | U | Date DPT_Date | Low |
| 2 | Global Reset | 0=No Action; 1=Reset | | 1 bit | C | - | W | - | - | 1 bit | Low |
| 3 | Global Request | 0=No Action; 1=Request | | 1 bit | C | - | W | - | - | 1 bit | Low |
| 4 | [CA] Reset | 0=No Action; 1=Reset | | 1 bit | C | - | W | - | - | 1 bit | Low |
| 7 | [CA] Request | 0=No Action; 1=Request | | 1 bit | C | - | W | - | - | 1 bit | Low |
| 10 | [CA] Disable | 0=Disable; 1=Enable | | 1 bit | C | - | W | - | - | 1 bit DPT_Enable | Low |
| 13 | Tariff 1 | Tariff value | | 2 Byte | C | - | W | - | - | 2 byte float value | Low |
| 14 | Tariff 2 | Tariff value | | 2 Byte | C | - | W | - | - | 2 byte float value | Low |
| 15 | Tariff 3 | Tariff value | | 2 Byte | C | - | W | - | - | 2 byte float value | Low |
| 16 | Tariff 4 | Tariff value | | 2 Byte | C | - | W | - | - | 2 byte float value | Low |
| 17 | Switch to Tariff 1 | 0=No Action; 1=Switch Tariff | | 1 bit | C | - | W | - | - | 1 bit | Low |
| 18 | Switch to Tariff 2 | 0=No Action; 1=Switch Tariff | | 1 bit | C | - | W | - | - | 1 bit | Low |
| 19 | Switch to Tariff 3 | 0=No Action; 1=Switch Tariff | | 1 bit | C | - | W | - | - | 1 bit | Low |
| 20 | Switch to Tariff 4 | 0=No Action; 1=Switch Tariff | | 1 bit | C | - | W | - | - | 1 bit | Low |
| 111 | [CA] Upper Limit | Upper Limit value | | 2 Byte | C | - | W | - | - | 2 byte float value | Low |
| 114 | [CA] Over-consumption Alarm | Alarm -> Sending of "0" or "1" | 0/0/9 | 1 bit | C | R | - | T | - | 1 bit | Low |
| 117 | [CA] Lower Limit | Lower Limit value | | 2 Byte | C | - | W | - | - | 2 byte float value | Low |
| 120 | [CA] Low Consumption Indicator | Ind. -> Sending of "0" or "1" | | 1 bit | C | R | - | T | - | 1 bit | Low |
| 123 | [LF] (1 bit) Data Entry 1 | Binary Data Entry (0/1) | 0/0/9 | 1 bit | C | - | W | - | - | 1 bit DPT_Switch | Low |
| 124 | [LF] (1 bit) Data Entry 2 | Binary Data Entry (0/1) | 0/0/2 | 1 bit | C | - | W | - | - | 1 bit DPT_Switch | Low |
| 125 | [LF] (1 bit) Data Entry 3 | Binary Data Entry (0/1) | 0/0/4 | 1 bit | C | - | W | - | - | 1 bit DPT_Switch | Low |
| 126 | [LF] (1 bit) Data Entry 4 | Binary Data Entry (0/1) | 0/0/6 | 1 bit | C | - | W | - | - | 1 bit DPT_Switch | Low |
| 127 | [LF] (1 bit) Data Entry 5 | Binary Data Entry (0/1) | 0/0/8 | 1 bit | C | - | W | - | - | 1 bit DPT_Switch | Low |
| 139 | [LF] (1 byte) Data Entry 1 | 1 byte Data Entry (0-255) | 0/0/10 | 1 Byte | C | - | W | - | - | 8 bit unsigned value ... | Low |
| 156 | [LF] Function 2 RESULT (1 bit) | FUNCTION 2 Result | 0/0/1 | 1 bit | C | R | - | T | - | 1 bit DPT_Switch | Low |
| 157 | [LF] Function 3 RESULT (1 bit) | FUNCTION 3 Result | 0/0/3 | 1 bit | C | R | - | T | - | 1 bit DPT_Switch | Low |
| 158 | [LF] Function 4 RESULT (1 bit) | FUNCTION 4 Result | 0/0/5 | 1 bit | C | R | - | T | - | 1 bit DPT_Switch | Low |
| 159 | [LF] Function 5 RESULT (1 bit) | FUNCTION 5 Result | 0/0/7 | 1 bit | C | R | - | T | - | 1 bit DPT_Switch | Low |
| 160 | [LF] Function 1 RESULT (1 byte) | FUNCTION 1 Result | 0/0/10 | 1 Byte | C | R | - | T | - | 8 bit unsigned value ... | Low |

**Figure 22 Communication Objects View of the device 1.1.1**




| Number | Name | Object Function | Group A... | Length | C | R | W | T | U | Data Type | Priority |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | Scenes (Individual Outputs) | 0-63(Run 1-64); 128-191(Le... | | 1 Byte | C | - | W | - | - | | Low |
| 1 | Scenes (Shutter Channels) | 0-63(Run 1-64); 128-191(Le... | | 1 Byte | C | - | W | - | - | | Low |
| 96 | [O1] ON/OFF | N.O. (0=Open Relay; 1=Clo... | 0/0/3 | 1 bit | C | - | W | - | - | 1 bit DPT_Switch | Low |
| 97 | [O2] ON/OFF | N.O. (0=Open Relay; 1=Clo... | 0/0/5 | 1 bit | C | - | W | - | - | 1 bit DPT_Switch | Low |
| 98 | [O3] ON/OFF | N.O. (0=Open Relay; 1=Clo... | 0/0/7 | 1 bit | C | - | W | - | - | 1 bit DPT_Switch | Low |
| 100 | [O1] Status | 0=Output OFF; 1=Output ON | 0/0/4 | 1 bit | C | R | - | T | - | 1 bit DPT_Switch | Low |
| 101 | [O2] Status | 0=Output OFF; 1=Output ON | 0/0/6 | 1 bit | C | R | - | T | - | 1 bit DPT_Switch | Low |
| 102 | [O3] Status | 0=Output OFF; 1=Output ON | 0/0/8 | 1 bit | C | R | - | T | - | 1 bit DPT_Switch | Low |
| 104 | [O1] Block | 1=Block; 0=Unblock | | 1 bit | C | - | W | - | - | 1 bit DPT_Enable | Low |
| 105 | [O2] Block | 1=Block; 0=Unblock | | 1 bit | C | - | W | - | - | 1 bit DPT_Enable | Low |
| 106 | [O3] Block | 1=Block; 0=Unblock | | 1 bit | C | - | W | - | - | 1 bit DPT_Enable | Low |

**Figura 23 Communication Objects View of the device 1.1.2**




| Number | Name | Object Function | Group A... | Length | C | R | W | T | U | Data Type | Priority |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | ON/OFF | Turn ON/OFF the split | 0/0/1 | 1 bit | C | - | W | - | U | 1 bit DPT_Switch | Low |
| 1 | ON/OFF (status) | Split state (ON/OFF) | 0/0/2 | 1 bit | C | R | - | T | - | 1 bit DPT_Switch | Low |
| 2 | Temperature | Value sent to the split | | 2 Byte | C | R | W | T | U | 2 byte float value DP... | Low |
| 3 | Fan [1byte] | 0%Au;1-33%Mi;34-66%Mid... | | 1 Byte | C | - | W | T | U | 8 bit unsigned value ... | Low |
| 4 | Fan [1bit] | 0=Lower, 1=Higher | | 1 bit | C | - | W | - | - | 1 bit DPT_Switch | Low |
| 6 | Swing | 1=Swing, 0=Stop/Step | | 1 bit | C | - | W | T | U | 1 bit DPT_Switch | Low |
| 8 | Heat Mode | 1=Set mode, 0=Nothing | | 1 bit | C | - | W | T | U | 1 bit DPT_Switch | Low |
| 10 | Cool Mode | 1=Set mode, 0=Nothing | | 1 bit | C | - | W | T | U | 1 bit DPT_Switch | Low |
| 12 | Dry Mode | 1=Set mode, 0=Nothing | | 1 bit | C | - | W | T | U | 1 bit DPT_Switch | Low |
| 14 | Fan Mode | 1=Set mode, 0=Nothing | | 1 bit | C | - | W | T | U | 1 bit DPT_Switch | Low |
| 16 | Auto Mode | 1=Set mode, 0=Nothing | | 1 bit | C | - | W | T | U | 1 bit DPT_Switch | Low |
| 18 | Modes [1byte] | 0=Aut;1=Ht;3=Cool;9=Fan... | | 1 Byte | C | - | W | T | U | | Low |
| 21 | Disable Device | 0=Normal Running, 1=Disab... | | 1 bit | C | R | W | - | U | 1 bit DPT_Enable | Low |

Figura 24 Communication Objects View of the device 1.1.3

## 4.3. GROUP ADDRESSES

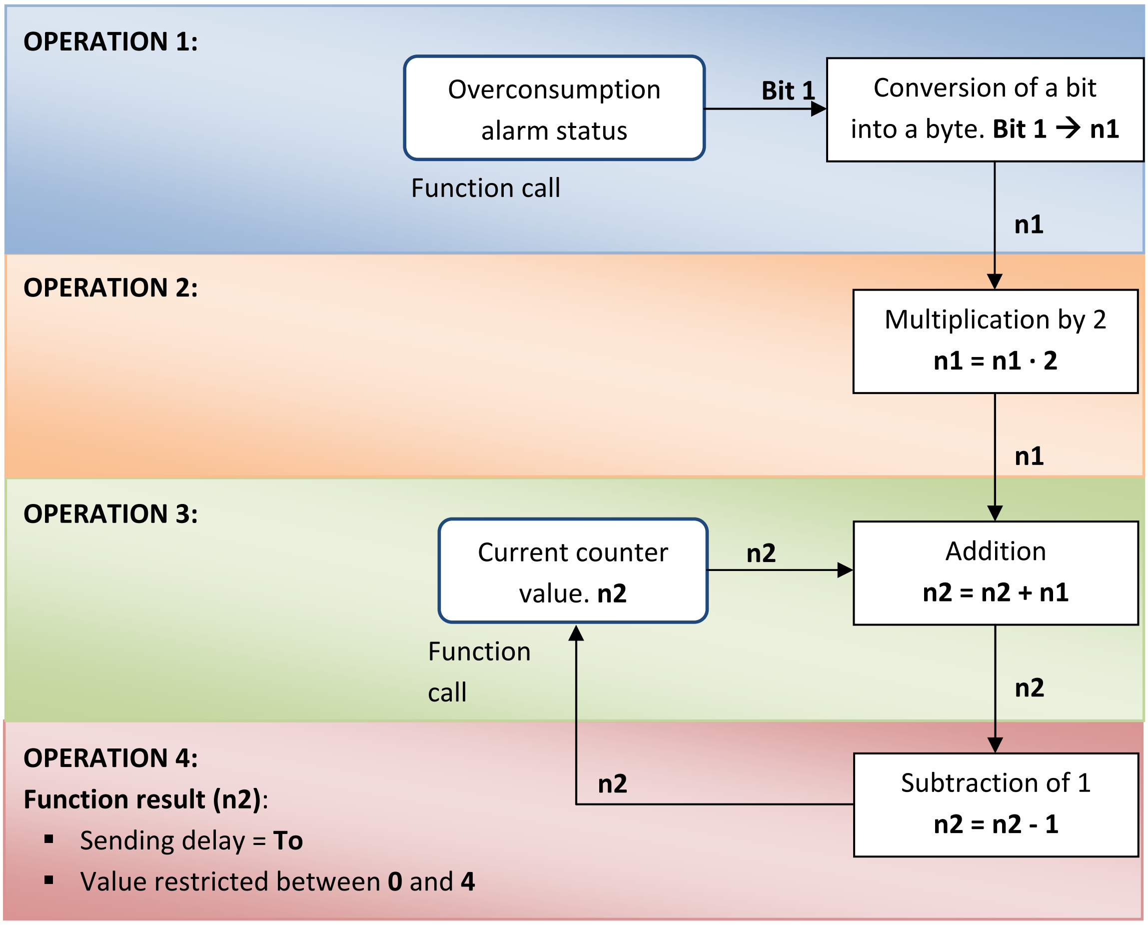
In this section, it is related the group addresses used for the implementation of the present application, the association of the different communication objects to the group addresses, and a brief description of their functions.

| DIRECCIÓN | NOMBRE | OBJETO | DISPOSITIVO | DESCRIPCIÓN |
|---|---|---|---|---|
| 0/0/1 | **Load 1 – On/Off – AC Unit** | 156 | 1.1.1 | Switch-on/off of load 1: Air Conditioning unit. |
| | | 0 | 1.1.3 | |
| 0/0/2 | **Load 1 – Status – AC unit** | 124 | 1.1.1 | Load 1 status: Air Conditioning unit. |
| | | 1 | 1.1.3 | |
| 0/0/3 | **Load 2 – On/Off – KNX lighting** | 157 | 1.1.1 | Switch-on/off of load 2: Lighting circuit integrated in the KNX system. |
| | | 96 | 1.1.2 | |
| 0/0/4 | **Load 2 – Status – KNX lighting** | 125 | 1.1.1 | Load 2 status: Lighting circuit integrated in the KNX system. |
| | | 100 | 1.1.2 | |
| 0/0/5 | **Load 3 – On/Off – Conventional lighting** | 158 | 1.1.1 | Switch-on/off of load 3: Lighting circuit with conventional control, not integrated in the KNX system. |
| | | 97 | 1.1.2 | |
| 0/0/6 | **Load 3 – Status – Conventional lighting** | 126 | 1.1.1 | Load 3 status: Lighting circuit with conventional control, not integrated in the KNX system. |
| | | 101 | 1.1.2 | |

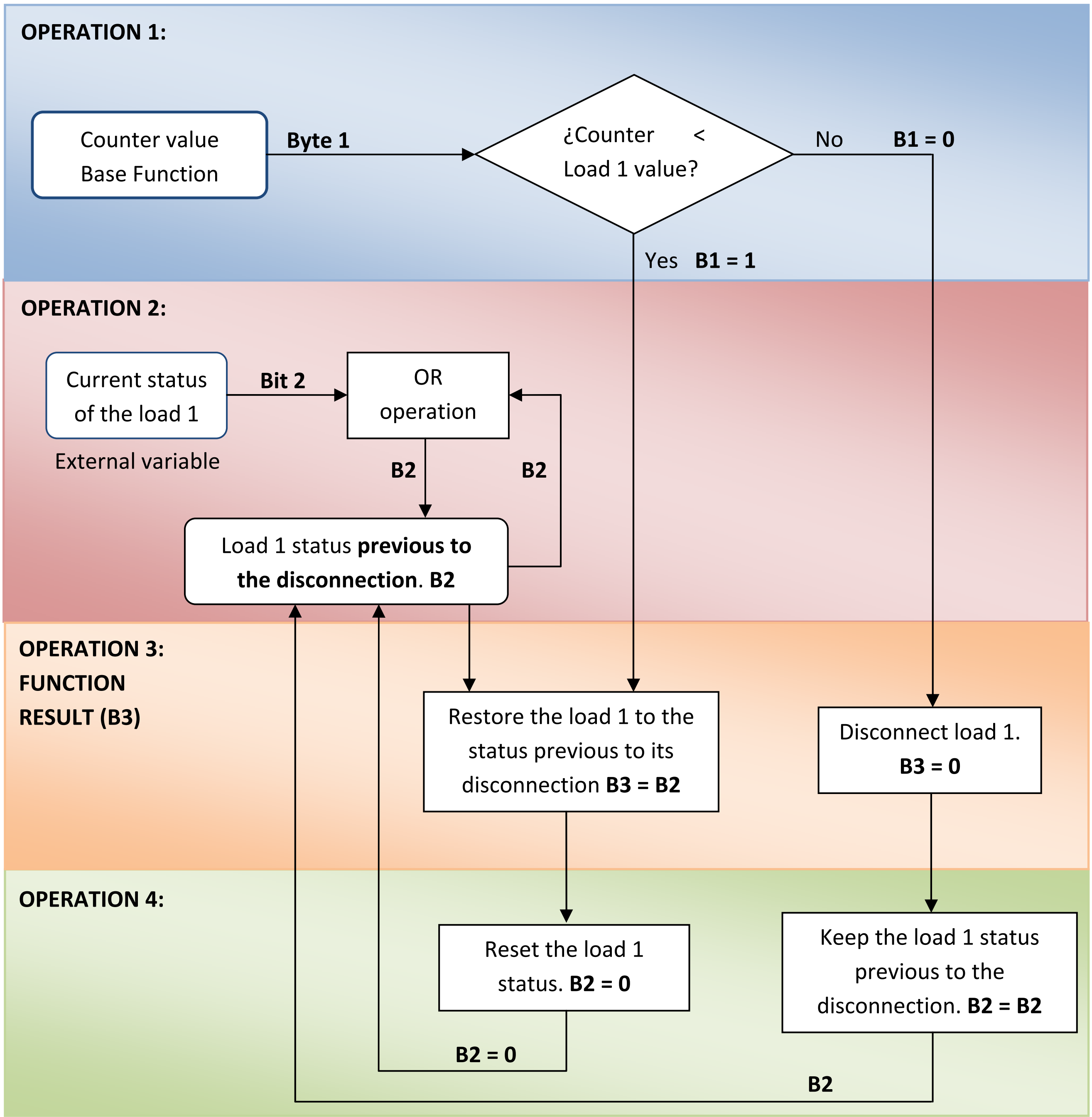
<table>
<tr><th>DIRECCIÓN</th><th>NOMBRE</th><th>OBJETO</th><th>DISPOSITIVO</th><th>DESCRIPCIÓN</th></tr>
<tr><td rowspan="2">0/0/7</td><td rowspan="2">Load 4 – On/Off – Oven</td><td>159</td><td>1.1.1</td><td rowspan="2">Switch-on/off of load 4: Oven circuit.</td></tr>
<tr><td>98</td><td>1.1.2</td></tr>
<tr><td rowspan="2">0/0/8</td><td rowspan="2">Load 4 – Status – Oven</td><td>127</td><td>1.1.1</td><td rowspan="2">Load 4 status: Oven circuit</td></tr>
<tr><td>102</td><td>1.1.2</td></tr>
<tr><td rowspan="2">0/0/9</td><td rowspan="2">[CA] Over-consumption alarm</td><td>114</td><td>1.1.1</td><td rowspan="2">Overconsumption alarm status</td></tr>
<tr><td>123</td><td>1.1.1</td></tr>
<tr><td rowspan="2">0/0/10</td><td rowspan="2">[CA] Loads switch-off counter</td><td>160</td><td>1.1.1</td><td rowspan="2">Counter of the loads to be disconnected.</td></tr>
<tr><td>139</td><td>1.1.1</td></tr>
</table>

# ANNEX I: FLOW DIAGRAMS FOR THE LOGICAL FUNCTIONS

## BASE FUNCTION OR COUNTER



## SECONDARY FUNCTION



ZENNIO APPLICATIONS

BECOME USER!

http://zennioenglish.zendesk.com

TECHNICAL SUPPORT