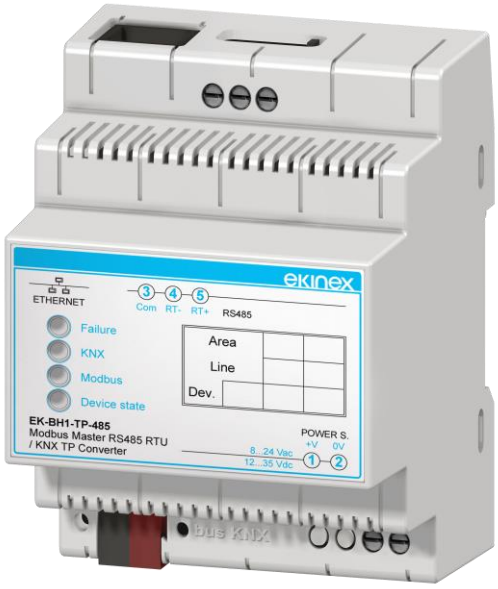
ekinex

CONTROL YOUR LIVING SPACE



Gateway configuration manual
Modbus master RTU RS485 - KNX TP
EK-BH1-TP-485

Indice

© EKINEX S.p.A. – All rights reserved

Scope of the document

This document describes the gateway (protocol converter) Modbus master RTU RS485 – KNX TP.

The gateway finds its ideal applicaton in the integration of Modbus devices over a RS485 serial network in a KNX-based automation system for homes and buildings. This product belongs to a broad line of ekinex® gateways designed to meet the needs for integration of the building automation most widely used protocols, based on serial, Ethernet or proprietary infrastructures.

For further informations about the available technical solutions, please visit www.ekinex.com.

1 Product description

The Modbus master RTU RS485 ekinex® EK-BH1-TP-485 gateway is a KNX modular unit for panel mounting. It allows you to exchange informations with one or more slave devices over a RS485 differential serial network through Modbus RTU (Remote Terminal Unit)[1] protocol. The ekinex gateway acts as Modbus Master. The informations exchanged over the Modbus network are updated over the KNX network by means of a twisted pair (TP) communication cable.

The device manages a two-way data stream: the Modbus registers can be cyclically read and their value sent as a communication object over the KNX TP network through a multicast communication to configured group addresses. The data update over the KNX network can be done cyclically and/or on event of change of the data acquired by the Modbus network.

Likewise, the ekinex gateway can make requests to cyclically readings KNX communication objects or acquire their values during data exchange over the bus. Cyclically or on event of change of the communication objects, data are written on the Modbus registers of one or more configured devices.

The ekinex gateway supports the entire Modbus RTU master protocol with the possibility of reading and writing single and multiple 1-bit registers (Coil and Status) as well as 16-bit registers (Holding and Input). It is also possible to read and write multiple registers containing 32-bit floating point values (IEEE 754 format).

As for KNX communication, 1-bit, 1-byte, 2-byte and 4-byte communication objects can be acquired: internal conversion functions allow you to convert the informations from and to 16-bit floating point values (DPT 9.xxx) starting from integer Modbus registers.

Configuration is performed through a PC application software which communicates through the integrated Ethernet port. The application software CGEKBH1TP485 is available for download at www.ekinex.com.

[1] The ekinex Modbus master RTU RS485 – KNX TP gateway does not support Modbus ASCII protocol.

© EKINEX S.p.A. – All rights reserved

1.1 Main functions

The gateway acts as a bidirectional protocol converter. Data streams are the following:

- Modbus serial network – Coil and Status registers (1-bit) as well as Holding and Input registers (16-bit) cyclical reading from one or more slaves. Refresh time starts from 100 ms, single and multiple registers reading is supported. The values of the read registers are stored in a 1440-byte volatile memory buffer ("Modbus image memory").
- KNX TP network – Sending of writing multicasting frames (APCI = write)[2] to configured group addresses. Data can be sent cyclically over the bus (configurable refresh time), on event of change of the data contained in the "Modbus memory image", or both cyclically and on change. Internal conversion functions to the most common types of KNX Datapoints are present.
- KNX TP network – Multicasting frame listening from configured group addresses (with selectable filters on the area or network of interest) or cyclical sending of read request frames (APCI = read). The values of the acquired communication objects are stored in a 1440-byte volatile memory buffer ("KNX image memory"). This buffer is independent from the "Modbus image memory" buffer.
- Modbus serial network – Writing of registers to one or more slave devices. Registers can be sent cyclically over the serial network (configurable refresh time), on event of change of the data contained in the "KNX memory image", or both cyclically and on change.

1.2 Technical data

Characteristic	Value
Power supply	8…24 Vac 12…35 Vdc
Power Absorption	At 24 Vdc: 3,5 VA
Application area	dry indoor environment
Environmental conditions	• Operating temperature: - 40 ... + 85°C • Stock temperature: - 25 ... + 55°C • Transportation temperature: - 25 ... + 70°C • Relative humidity: 93% non-condensing
Programming elements	1 pushbutton and 1 LED (red) on the front
Display elements	4 status LEDs + 1 Ethernet connector LED
Configuration elements	2 1-way microswitches • Microswitch A: OFF normal mode; ON Boot mode • Microswitch B: OFF terminaton resistance not inserted; ON termination resistance (120 Ω) inserted between RT+ ad RT- on the RS485 port.
Safety class	II
Installation	35 mm DIN rail (according to EN 60529)
Protection degree	IP20
Dimensions (WxHxD)	82 x 75 x 35 mm
Ethernet interface (IEEE 802.3)	
Connector	RJ45, minimum cable category: 5E
Modbus interface	
Communication port	RS485, electrically isolated from power supply and KNX communication port
Baud rate	Configurable, from 1200 to 115200 baud
KNX TP interface	
Communication port	KNX TP (twisted pair), 9600 baud, electrically isolated from power supply and RS485 communication port
Power supply	SELV 30 Vdc through bus KNX
Current absorption from bus	< 13 mA

[2] APCI = Application Layer Protocol Control Information. Information contained in the frame sent to the application layer of the receiving device. It is defined by the KNX standard.

© EKINEX S.p.A. – All rights reserved

1.3 Supply

The supply includes the device and terminal blocks to connect to the KNX bus. An instruction sheet is also supplied within the package.

1.4 System requirements for configuration software

Configuration and commissioning of the ekinex® gateway must be performed using the application program CGEKBH1TP485, available for download at www.ekinex.com.

The PC where the application program is installed must meet the following requirements:

- Desktop o laptop PC with Ethernet IEEE 802.3 port.
- 32/64 bit operating system, Microsoft Windows® XP, 7, 8.0, 8.1 e 10.

.NET Framework 4.0 system library installation is required.

1.5 Certifications

Compliance with the European directives is certified by the CE symbol on the product label and on the documentation.

© EKINEX S.p.A. – All rights reserved

2 Switching, display and connection elements

The device is equipped with a pushbutton and a KNX programming LED, with a status LED and terminal blocks for KNX and RS485 network connection. A port for RJ45 connector for device configuration via Ethernet as well as two 1-way microswitches are also present.

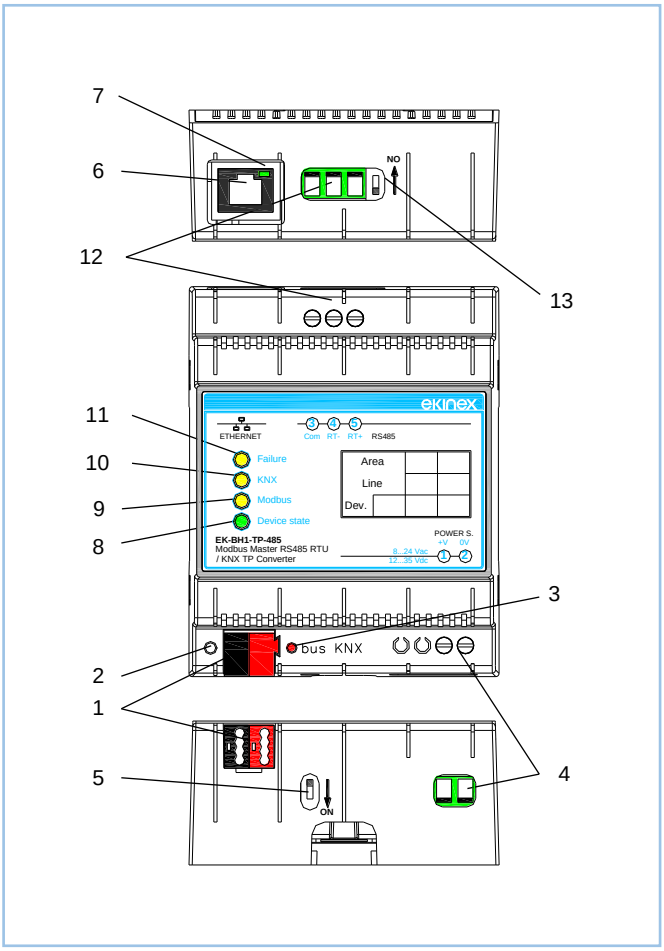


Figure 1 - Switching, display and connection elements

1)	KNX bus line terminal blocks
2)	KNX programming pushbutton
3)	KNX programming LED
4)	Power supply terminal blocks
5)	1-way microswitch A
6)	Ethernet port
7)	Ethernet port LED
8)	Device satus LED
9)	Modbus communication LED
10)	KNX communication LED
11)	Device error LED
12)	RS485 serial line terminal blocks
13)	1-way microswitch B

© EKINEX S.p.A. – All rights reserved

Command elements

- Pushbutton that switches between normal mode and KNX physical address programming.

1-way microswitches

- A - OFF: normal mode active. ON: Boot mode active
- B - OFF: open. ON: RS485 line termination inserted (120 Ω termination resistance in parallel between RT+ and RT-)

Display elements

The device can run according to two operating modes: Normal mode (configuration loaded, Modbus and KNX communication running) and Boot mode (no configuration or still loading configuration)

LED	Normal mode	Boot mode
Green LED (8) – Device status	Slow blinking (~1 Hz)	**ON**: device on **OFF**: device off
Yellow LED (9) – Modbus communication	Blinks when a frame is received on the RS485 port.	**Fast blinking**: no configuration **Very slow blinking (~0,5 Hz)**: loading configuration.
Yellow LED (10) – KNX communication	Blinks when a frame is received.	**Fast blinking**: no configuration **Very slow blinking (~0,5 Hz)**: loading configuration.
Yellow LED (11) – Device error	**ON**: at least one Modbus request did not get a correct answer **OFF**: no error	**Fast blinking**: no configuration **Very slow blinking (~0,5 Hz)**: loading configuration.
Green LED (7) – Ethernet port	**ON**: Ethernet connector plugged **OFF**: Ethernet connector unplugged	**ON**: Ethernet connector plugged **OFF**: Ethernet connector unplugged
Red LED (3) – KNX programming	**ON**: physical address programming mode on **OFF**: physical address programming mode off	**Fast blinking**: no configuration **Very slow blinking (~0,5 Hz)**: loading configuration.

In the current version of the device, both KNX physical address programming and configuration download must be performed through the configuration program: for KNX physical address please refer to “Communication parameters” paragraph, “ID Device” parameter.

© EKINEX S.p.A. – All rights reserved

3 Configuration and commissioning

The device configuration requires the following tools:

- The documentation of the Modbus products, specifically the database of each product to be integrated, containing the addresses of the registers and physical parameters of the RS485 serial communication (baud rate, parity check, delays, physical addresses of the devices to be integrated).
- CGEKBH1TP485 application software to properly configure the gateway.
- Knowledge of the ETS automation project, with particular attention to communication objects and group addresses passing on the bus during the multicast communication between sensors and actuators.

Configuration and commissioning of the ekinex® gateway require specialized skills about KNX networks and knowledge of the specific ETS automation project. In order to acquire such skills, it is essential to attend trainings and workshops organized at KNX-certified training centers. For further information: www.knx.it.

© EKINEX S.p.A. – All rights reserved

4 Modbus protocol general informations

Modbus is a master/slave protocol that manages several services encoded in the “function code” field, contained in each request telegram.

The ekinex gateway EK-BH1-TP-485 manages the Modbus protocol:

- with a twisted pair connection cable through the RS485 differential serial communication port;
- in the Master version;
- in the RTU (Remote Terminal Unit) form. The gateway, therefore, does not support ASCII format.

The module supports the following function codes:

Function Code	Description
01	Single or multiple reading of 1-bit registers (Coil)
02	Single or multiple reading of 1-bit registers (State)
03	Single or multiple reading of 16-bit registers (Holding)
04	Single or multiple reading of 16-bit registers (Input)
05	Single writing of 1-bit registers (Coil)
06	Single writing of 16-bit registers (Holding)
15	Multiple writing of 1-bit registers (Coil)
16	Multiple writing of 16-bit registers (Holding)

The data structure includes 4 different types of registers that can be read or written through the proper function code. Coil and holding registers can be both read or written. State and input registers, otherwise, can only be read.

Register type	Dimension		Access
Coil	1 Bit		Read/write
State	1 Bit		Read only
Holding	16 Bit		Read/write
Input	16 Bit		Read only

© EKINEX S.p.A. – All rights reserved

5 Configuration software

The ekinex® configuration software CG-EK-BH1-TP-485 allows you to perform the following operations:

- Selection of physical parameters of the RS485 serial communication;
- Selection of physical address of the device over the KNX TP network;
- Selection of Ethernet parameters (for configuration download only);
- KNX network: communication objects definition and relative group addresses to be acquired;
- KNX network: communication objects definition and relative group addresses to be sent over the KNX network;
- Modbus network: definition of the registers to be read from the network devices;
- Modbus network: definition of the registers to be written on the network devices;
- Firmware and/or configuration update.

The application program consists in multiple modal windows called “forms”: each form must be closed before accessing the following form. The buttons on the main form (see Figure 2 – Main form of the application program) are ordered according to the proper sequence to follow in order to perform a correct configuration.

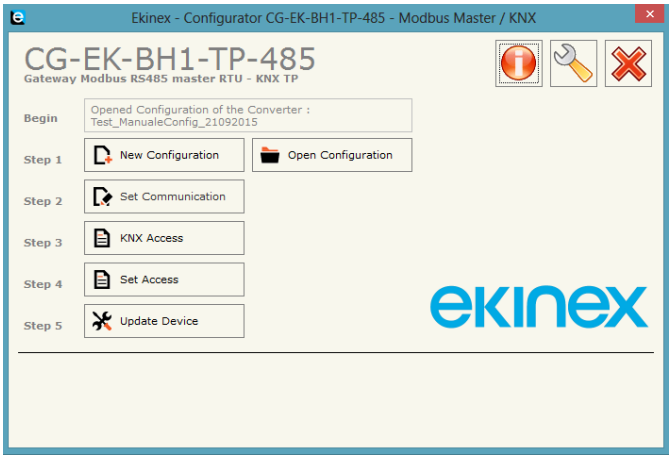


Figure 2 – Main form of the application program

Starting from the main form, by accessing the *About...* window, you can check the current version of the installed program.

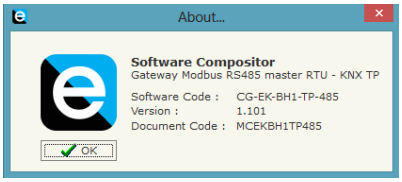


Figure 3 – About form

Please visit the section about communication gateways on www.ekinex.com in order to check the current version of the application program and download the latest version.

© EKINEX S.p.A. – All rights reserved

5.1 Memory image structure

The proper configuration of the device refers to a support volatile memory area where the acquired data are temporarily copied, both on Modbus and KNX side: this memory area is divided into 2 buffers, “Modbus image” and “KNX image”, each one composed of 1440 bytes.

Each support byte can be individually addressed (see *Position* field in *KNX Set Access* and *Set Modbus Access* forms) or you can target a specific support bit in each buffer (*Bit Mode* field in *KNX Set Access* fom and *Start Bit* field in *Set Modbus Access* form).

As shown in figure, the same address can refer to both buffers:

- “Modbus image” used in *Set Modbus Access* form, *Modbus Read* tab and in *KNX Set Access* form for writing frames.
- “KNX image” used in *KNX Set Access* form for reading frames and in form *Set Modbus Access*, *Modbus Write* tab.

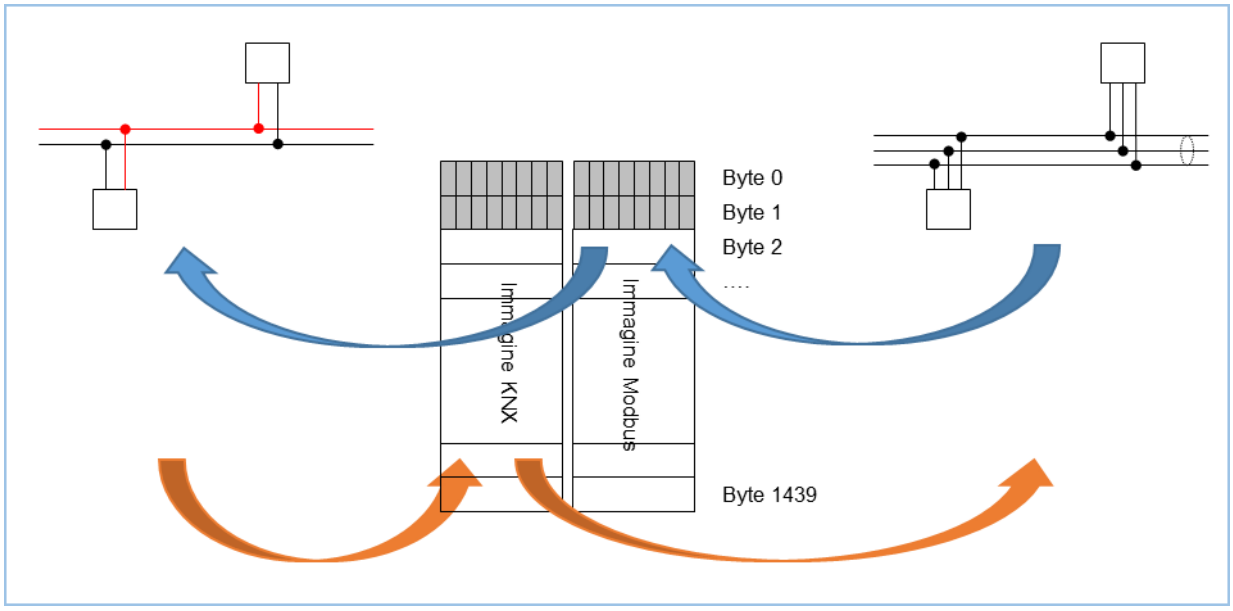


Figure 4 – Support memory with “KNX image” and “Modbus image” buffers

The proper addressing of the support buffers must be manually performed by the user, based on the size of the data to be acquired. Overlapping support data may end up in a protocol converter mulfunction.

For an assessment of the potential of the procotol converter, the “Modbus image” memory buffer allows, for example, to acquire up to 720 16-bit Holding or Input registers.

© EKINEX S.p.A. – All rights reserved

5.2 Creating a new project or modifying a saved project

The application program allows you to create a new configuration or open an existing one using the buttons called *New Configuration* and *Open Configuration* (see Figure 2 – Main form of the application program): the configuration files are stored on the hard drive in XML format.

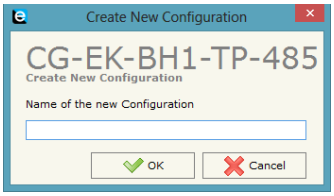


Figure 5 – Create new configuration form

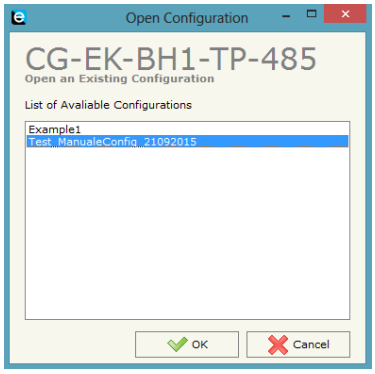


Figure 6 – Open configuration form

In order to duplicate an existing project, you must find the project folder containing the XML files and copy them in a new folder. Project files can be found by the following path:

"C:\Program Files(x86)\Ekinex\Compositor CGEKBH1TP485\Projects".

Once the project has been duplicated, simply restart the application program and open the form *Open configuration* (see Figure 6 - Open configuration form): you will see the name of the duplicated project in the list of available configurations.

© EKINEX S.p.A. – All rights reserved

5.3 Software Options

The *Software Options* form allows you to select a different language for the application program.

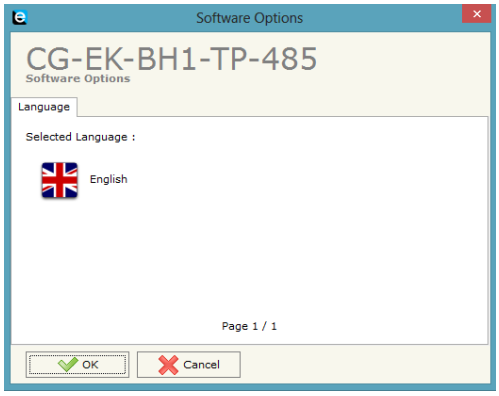


Figure 7 – Options form, Language tab

© EKINEX S.p.A. – All rights reserved

5.4 Communication parameters

In this section we define the basic communication parameters for the KNX TP network, for the Modbus network and for Ethernet connection. Ethernet connection is required in order to perform the configuration update on the device.

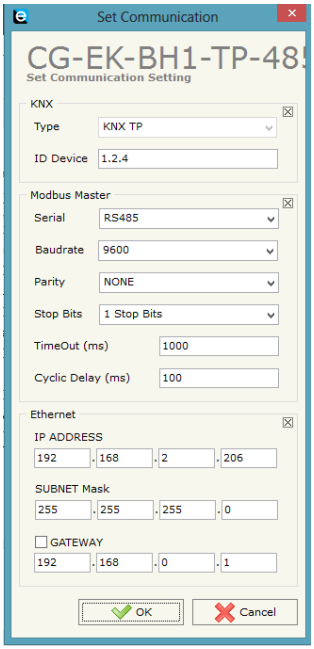


Figure 8 – Set communication form

You can access the form by pressing the *Set Communication* button in the main form (see Figure 2 – Main form of the application program).

Description of fields in *Set communication* form.

Parameter name	*Values*	*Description*
KNX		
Type	**KNX TP**	Type of connection used for KNX communication. The parameter has a constant value “KNX TP”. The device supports KNX communicaton over a twisted pair communication cable.
ID Device		This parameter identifies the physical address assigned to the KNX device. The format requires the use of a dot “.” as a separator beween the 3 fields: area, line and device address. Here are the conventions used for physical addressing and the values used for each field: Area field: = 0 reserved for backbone, values [1...15] Line field: = 0 reserved for main line, values [1...15] Device address field: = 0 reserved for coupler, values [1...255], range [1..64] for devices belonging to the line, above 64 for device belonging to extensions or other segments of the line. Example: 1.3.5: Area = 1; Line = 3; Device address = 5.

© EKINEX S.p.A. – All rights reserved

Parameter name	*Values*	*Description*
Modbus Master		
Serial	**RS232** RS485	Type of serial port used for communication.
Baudrate	1200 2400 4800 **9600** 19200 38400 57600 115200	Baudrate of the serial communication.
Parity	**NONE** ODD EVEN	Parity check.
Stop Bits	**1 Stop Bits** 2 Stop Bits	Number of stop bits added to the transmission/reception of a byte.
TimeOut (ms)		Maximum waiting time (in milliseconds) for a response frame from a polled slave, after the master sends a request.
Cyclic Delay (ms)		Minimum delay between requests (in milliseconds) performed by the master.
Ethernet		
IP ADDRESS		IP Address (4-octet format) assigned to the device. Each octet is set in an Edit box. Default IP Address is: **192.168.2.205**. This is the address assigned to the device before the first configuration or after a complete restore.
SUBNET Mask		Subnet mask assigned to the device.
GATEWAY		Gateway address used for Ethernet communication. The gateway can be enabled or disabled through the control check-box placed at the right side of the field.

Please refer to the technical documentation of the slave device in order to set the correct parameters of the serial communication. Incompatible values of these parameters may prevent the correct exchange of frames.

© EKINEX S.p.A. – All rights reserved

5.5 KNX communication object configuration

In this section we define communication objects sent or acquired over the KNX network. You can access the form by pressing the *KNX Access* button in the main form (see Figure 2 – Application program main form).

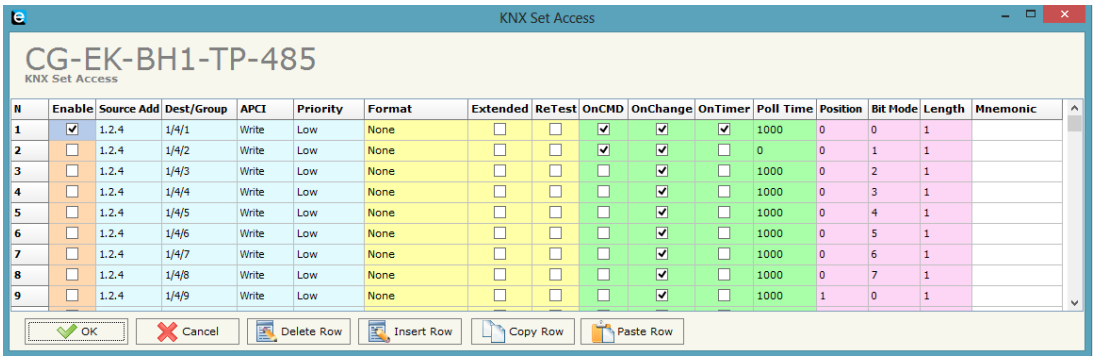


Figure 9 – KNX Set Access form

The form contains a configurable grid. Each record allows you to assign the properties for each communication object exchanged over the KNX network. In order to make the management of a significant number of data easier, after selecting a record it is possible to delete it from the project, insert a new record in a specific position and perform copy/paste of a previously configured record.

Description of fields in *KNX Set Access* form

Field name	***Values***	***Description***
N		Progressive number of the configuration record
Enable	**checked** / unchecked	Configuration record enabling. If a record is disabled, the corresponding data points will not be acquired or changed over the KNX bus
Source Address		In case of writing frames (field APCI=write) the physical address may correspond to the physical address of the gateway (*Device ID* field in the *Set Communication* form), in the format Area.Line.Address (each field must be separated by a dot). In case of reading frames (field APCI=read), *Source Address* acts as a filter. Through this field you can acquire datapoints of all lines over the KNX bus (0.0.0 value) or you can select one specific line (e.g. 4.3.0) or a single device identified by a specific physical address (e.g. 4.3.1).
Dest/Group		A Group Address (2-level, 3-level or free structure) or a Physical Address can be set. In case of a group address the fields must be separated through a "/", while in case of physical address the separator will be a ".".
APCI	read / write	The "read" option is used to send a request in order to read a communication object over the KNX bus. The "write" option must be selected if you want to change the value of a communication object over the KNX bus. Other services can be configured by editing the value of the corresponding service. The name used in the field refers to a 4-bit code (APCI = Application Layer Protocol Control Information) which defines the type of service required in KNX communication standard.
Priority	System/ Urgent / Normal / Low	KNX frames priority. In multicast communication (exchange of frames from/to group addresses), the default priority is Low.
Format	None / Swap16 / Swap32 / Swap All / Int to Float / Float to Int / Float 16 to Float 32	In case of a frame containing a data (in response to a reading request frame APCI = read), the Format field determines the data type conversion from the received frame to the support internal memory area. In case of a writing frame (APCI = write), the Format field determines the data type conversion from the support internal memory area to the frame.
Extended	**checked** / unchecked	Enables extended frame format for KNX communication (cEMI = Common Extended Message Interface)

© EKINEX S.p.A. – All rights reserved

Field name	*Values*	*Description*
ReTest	**checked** / unchecked	Enables the re-send of a frame in case of wrong response message
OnCMD	**checked** / unchecked	Not used
OnChange	**checked** / unchecked	Event which enables the automatic sending of command frames over the KNX bus when the data on the Modbus device changes their values.
OnTimer	**checked** / unchecked	Event which enables the cyclical sending of command frames over the KNX bus.
Poll Time		Cyclic poll time (in ms) when OnTimer event is enabled.
Position	Value in range [0…1439]	Position of the first byte where a data is stored, in the internal support memory buffer. In case of a record where APCI=read, *Position* refers to the "KNX image" buffer; in case of APCI=write, *Position* refers to the "Modbus image" buffer. Please refer to the paragraph concerning the structure of the memory image to perform a correct addressing and avoid overlaps between the two data buffers.
Bit Mode	No / 0 / 1 / 2 / 3 / 4 / 5 / 6 / 7	Position, inside the first byte of the internal support memory buffer, where a 1-bit data is stored.
Length		Size (in number of bytes) of the data stored inside the internal memory.
Mnemonic		Text to comment the record and/or the datapoint over the KNX bus.

If *OnChange* field is selected, *OnTimer* in selected and *Poll Time* ≠ 0, the gateway will send the commands both cyclically and on change of the data acquired over the Modbus network.

If *OnChange* and *OnTimer* fields are not selected, the gateway will only store the communication objects exchanged through the multicast frames over the KNX network ("sniffer" function).

Conversion types of internal data selectable through *Format* field:

Conversion	APCI = read	APCI = write
None	The value of the communication object is transferred in raw mode to the "KNX image" buffer and sent as register to the Modbus network.	The value of the communication object acquired over the Modbus network and stored in the "Modbus image" buffer is transferred in raw mode as communication object over the KNX network.
Swap16	16-bit swap inside the stored data	16-bit swap inside the stored data
Swap32	32-bit swap inside the stored data	32-bit swap inside the stored data
Swap All	All bit swap inside the stored data	All bit swap inside the stored data
Int to Float		The integer value acquired over the Modbus network is converted to a 2-byte (DPT 9.xxx) floating point value in order to be sent as communication object over the KNX network.
Float to Int	The 2-byte (DPT 9.xxx) floating point communication object value acquired over the Modbus network is converted to integer in order to be sent as Holding register over the Modbus network.	
Float 16 to Float 32	The 2-byte (DPT 9.xxx) floating point communication object value acquired over the Modbus network is converted to a 32-bit floating point value (according to standard IEEE 754) in order to be sent as double Holding register over the Modbus network.	

© EKINEX S.p.A. – All rights reserved

5.6 Modbus registers configuration

In this section we define the registers read or written over the Modbus network. You can access the form by pressing the *Set Access* button in the main form (see Figure 2 – Application program main form).

The form is divided into 2 tabs, *Modbus Read* tab (see Figure 10 - Set Modbus Access form, Modbus Read tab) and *Modbus Write* tab (see Figure 11 - Set Modbus Access form, Modbus Write tab). The *Read* tab contains the configuration grid of the registers whose values are acquired over the Modbus network and made available over the KNX network. The *Write* tab, instead, contains the configuration grid of the registers whose values are acquired over the KNX network and must be written over the Modbus network.

In order to make the management of a significant number of data easier, after selecting a record it is possible to delete it from the project, insert a new record in a specific position and perform copy/paste of a previously configured record.

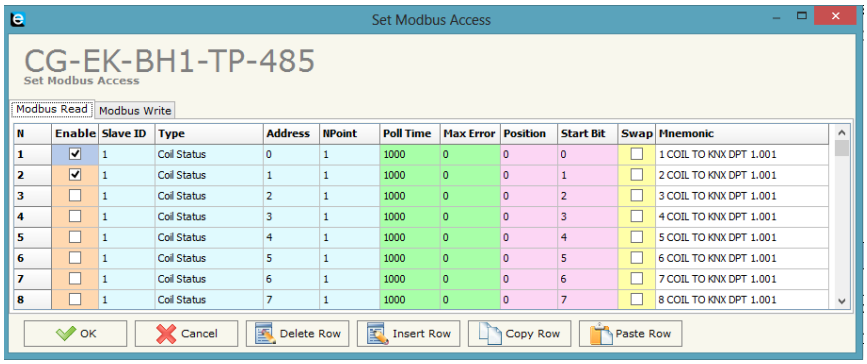


Figure 10 – Set Modbus Access form, Modbus Read tab

Description of fields in *Set Modbus Access* form*, Modbus Read* tab

Field name	***Values***	***Description***
N		Progressive number of the configuration record
Enable	**checked** / unchecked	Configuration record enabling. If a record is disabled, the corresponding data point will not be acquired by the Modbus device.
Slave ID		Slave device address
Type	Coil Status Input Status Holding Register Input Register	Coil Status size (for support buffer) = 1 Bit, R/W Input Status size = 1 Bit, R Holding Register size = 2 Bytes, R/W Input Register size = 2 Bytes, R
Address		Register address within the mapping of the slave device. According to the used convention, the address value can differ of ± 1 compared to the address indicated in the slave device mapping documentation.
NPoint		It defines the number of consecutive registers to be read from the slave device. With NPoint=1, the gateway enables only single register reading commands, while with NPoint>1, the gateway enables multiple registers reading commands.
Poll Time		Cyclic poll time (in ms) of the reading commands sent over the serial bus.
Max Error		Number of reading errors detected by the gateway before suspending the cyclical reading until next reboot. If MaxError=0, this function is disabled.
Position	Value in range [0…1439]	Position of the first byte where a data is stored, in the internal support memory buffer. Please refer to the paragraph concerning the structure of the memory image to perform a correct addressing and avoid overlaps with the “KNX image” buffer.
Start Bit	0 / 1 / 2 / 3 / 4 / 5 / 6 / 7 / 8	Position, inside the first byte of the internal support memory buffer, where a 1-bit data is stored.
Swap	**checked** / unchecked	If checked, the 16-bit register is stored in the support buffer with its bits

© EKINEX S.p.A. – All rights reserved

Field name	Values	Description
		swapped.
Mnemonic		Text to comment the register read over the Modbus network.

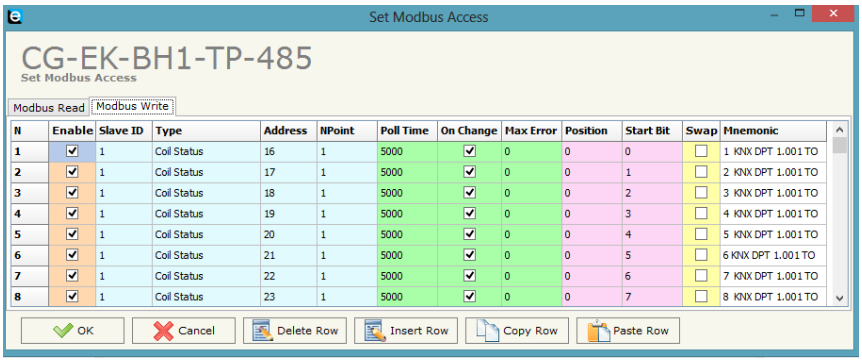


Figure 11 - Set Modbus Access form, Modbus Write tab

Description of fields in *Set Modbus Access* form, *Modbus Write* tab

Nome campo	Valori	Descrizione
N		Progressive number of the configuration record
Enable	**checked** / unchecked	Configuration record enabling. If a record is disabled, the corresponding data point will not be acquired by the Modbus device.
Slave ID		Slave device address
Type	Coil Status Holding Register	Coil Status size (for support buffer) = 1 Bit, R/W Holding Register size = 2 Bytes, R/W
Address		Register address within the mapping of the slave device. According to the used convention, the address value can differ of ± 1 compared to the address indicated in the slave device mapping documentation.
NPoint		It defines the number of consecutive registers to be written to the slave device. With NPoint=1, the gateway enables only single register writing commands, while with NPoint>1, the gateway enables multiple registers writing commands.
Poll Time		Cyclic poll time (in ms) of the writing commands sent over the serial bus.
OnChange	**checked** / unchecked	If checked, every change of value of the data acquired over the KNX networks causes the corresponding writing frames to be sent over the serial bus.
Max Error		Number of writing errors detected by the gateway before suspending the cyclical reading until next reboot. If MaxError=0, this function is disabled.
Position	Value in range [0…1439]	Position of the first byte where a data is stored, in the internal support memory buffer. Please refer to the paragraph concerning the structure of the memory image to perform a correct addressing and avoid overlaps with the “KNX image” buffer.
Start Bit	0 / 1 / 2 / 3 / 4 / 5 / 6 / 7 / 8	Position, inside the first byte of the internal support memory buffer, where a 1-bit data is stored.
Swap	**checked** / unchecked	If checked, the 16-bit register is stored in the support buffer and sent as a writing frame with its bits swapped.
Mnemonic		Text to comment the register written over the Modbus network.

© EKINEX S.p.A. – All rights reserved

If *OnChange* field is selected and *Poll Time* = 0, the registers will be written over the Modbus network only if the corresponding data acquired over the KNX network change their value.

If *OnChange* field is selected and *Poll Time* ≠ 0, the registers will be written over the Modbus network both cyclically and on change of the corresponding data acquired over the KNX network.

© EKINEX S.p.A. – All rights reserved

5.7 Configuration update

The implemented configuration and possibly the updated firmware can be downloaded by pressing the *Update Device* button in the main form of the application program (see Figure 2 – Main form of the application program).

There can be 2 possible update sequences, the first in case the IP address assigned to the device is unknown, the second in case the IP address is known.

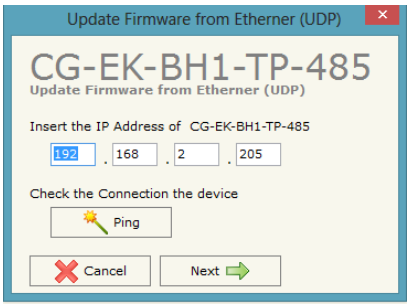


Figure 12 - Update configuration form

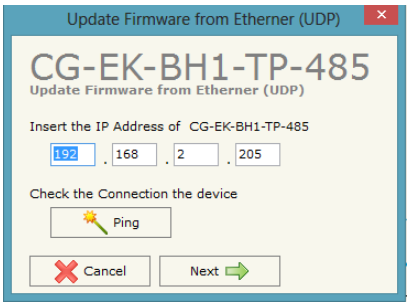


Figure 13 – Download options form

Sequence to follow in case of unassigned or unknown IP address:

- Power off the device
- Set the 1-way microswitch A (see Figure 1 – Switching, display and connection elements) to ON position
- Power on the device
- Connect PC and device by means of an Ethernet cable. Make sure that the PC's network parameters are consistent with the IP address assigned to the device in Boot Mode **192.168.2.205.** Otherwise, change the PC's network settings
- Write the IP address **192.168.2.205** inside the Update Configuration form (see Figure 12 – Update configuration form)
- Press *Ping* button; if you correctly applied the procedure, the text "*Device found!*" will appear
- Press *Next* button
- Select the desired options (see Figure 13 – Download options form): firmware update, configuration update or both
- Press *Execute update firmware* button
- When all operations are completed (see Figure 14 – Update in progress) shut down the device
- Set the 1-way microswitch A (see Figure 1 – Switching, display and connection elements) to OFF position
- Power on the device

© EKINEX S.p.A. – All rights reserved

If the sequence is successful, this means that firmware and/or configuration has been correctly downloaded on the device.

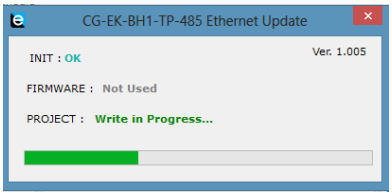


Figure 14 - Update in progress

Sequence to follow in case of known IP address:

- Power on the device with PC and device connected by means of an Ethernet cable
- Provide the device IP address (see Figure 12 – Update configuration form). Make sure that the PC's network parameters are consistent with the IP address assigned to the device**.** Otherwise, change the PC's network settings
- Press *Ping* button; if you correctly applied the procedure, the text "*Device found!*" will appear (see Figure 12 – Update configuration form)
- Press *Next* button (see Figure 12 – Update configuration form)
- Select the desired options (see Figure 13 – Download options form): firmware update, configuration update or both
- Press *Execute update firmware* button
- When all operations are completed (see Figure 14 – Update in progress) the device automatically switches back to Normal mode.

If the sequence is successful, this means that firmware and/or configuration has been correctly downloaded on the device.

It is recommended to update the firmware when a new version of the application program is installed or when configuring the device for the first time.

In case the update procedure goes into PROTECTION mode (see Figure 15 – Update error, "Protection" mode), you may want to check the following:

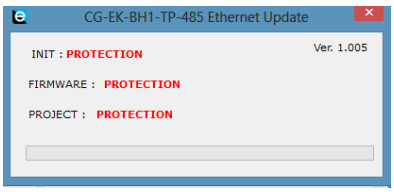


Figure 15 – Update error, "Protection" mode

© EKINEX S.p.A. – All rights reserved

- Repeat the update sequence
- Reboot your PC
- When running the program on a Virtual Machine, close it and rerun the program using the primary OS
- When using Windows 7 or later, make sure the user has administrator privileges
- Pay attention to firewall settings
- Check LAN configuration

In case of manual firmware update, replace “Sim67812.sim” file in the system folder “C:\Program Files (x86)\Ekinex\Compositor CGEKBH1TP485\Master”. After replacing, open *Update configurazione* form (see Figure 12 – Update configuration form) in the application program and start the proper sequence.

© EKINEX S.p.A. – All rights reserved

5.8 Configuration example

The figure shows the connection between a Modbus device and a KNX-based building automation system via ekinex gateway: the purpose of this example is the integration of the functions by exchanging informations between different communication systems and the unification of the user interface on a KNX-based supervision system.

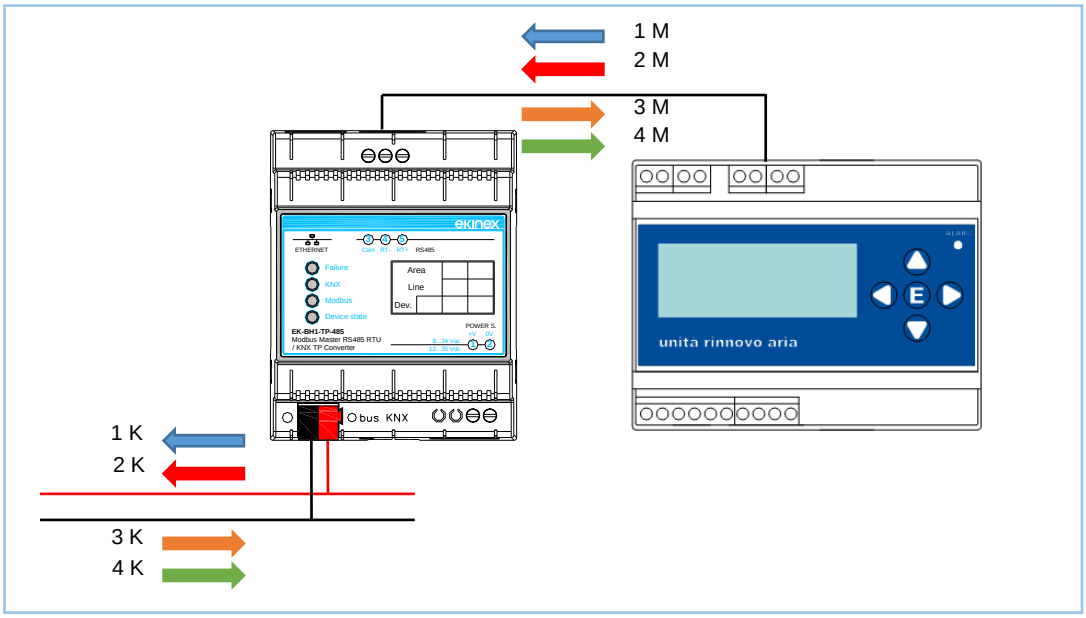


Figure 16 - Modbus Slave RTU device integration example

The arrows highlight the 4 data streams managed by the gateway between the Modbus device and the KNX network:

- 1 M: states such as alarms or position of the machine actuators controlled by the Modbus device. The gateway cyclically "reads" the "coil" registers containing the state informations and converts them into 1-Bit communication objects (DPT 1.001 switch). The communication objects, represented by the flow 1 K, are sent over the bus on change of their Modbus state (OnChange event). This example provides 16 “coil" registers read from Modbus.
- 2 M: numerical data such as temperature of air treated by the machine controlled by the Modbus device. The gate cyclically "reads" the "holding registers" and converts them into 2-byte communication objects (DPT 9,001 temperature). The communication objects, which represent the flow 2 K, are sent over the bus on change of their Modbus value. This example provides 16 "holding" registers read from Modbus.
- 3 K: 1-Bit communication objects (DPT 1,001 switch) representing states such as mode of operation (heating or cooling), discrete fan speed (speed 1 or speed 2 or speed 3) processed by a thermostat over the KNX network. The gateway performs a cyclical read request and converts the communication objects in as many states to be sent as 1-Bit "coil" registers to the Modbus device. The register set "coil" that need to be written to the Modbus device is the flow 3 M. This example provides 16 "coil" registers to be witten on Modbus.
- 4K: 2-Byte communication objects (DPT 9,001 temperature) representing temperature setpoints displayed on a KNX-based supervision system. The gateway performs a cyclical read request and converts the communication objects in as many values to be sent to the 2-Byte "holding" registers in the Modbus device. The holding register set to be written to the Modbus device is represented by the flow 4 M. This example provides 16 "holding" registers to be written on Modbus.

© EKINEX S.p.A. – All rights reserved

In order to properly commission the system, you need to configure the gateway with the proper RS485 communication parameters (baud rate, parity check, number of stop bits, timeout after a request and delay between requests). These data are usually available on the documentation of the Modbus device.

It is also necessary to have the database containing the mapping of the read-only and read-write communication registers. In this example, the following Modbus database is used:

1-bit coil registers, read-only: addresses from 0 to 15 (decimal format);

1-bit coil registers, read/write: addresses from 16 to 31;

2-byte holding registers, read-only: addresses from 32 to 47;

2-byte holding registers, read/write: addresses from 48 to 63;

The group addresses to be associated to the communication objects must be properly planned and coordinated with the KNX project realized using ETS. This example adoptes the following group addresses (3-levels convention):

1 K stream: DPT 1.001 switch communication objects, group addresses from 1/4/1 to 1/4/16;

3 K stream: DPT 1.001 switch communication objects, group addresses from 1/4/17 to 1/4/31;

2 K stream: DPT 9.001 temperature communication objects, group addresses from 1/4/32 to 1/4/47;

4 K: DPT 9.001 temperature communication objects, group addresses from 1/4/48 to 1/4/63;

Please refer to the documentation about the Modbus database of the used device. The address of each register can have an offset from the address shown in the "Address" field on the configuration grid. For example, Address = 0 in the configuration grid can correspond to address = 1 in the mapping.

Configuration grid, *Set Modbus Access* form, *Modbus Read* tab.

N	En	SlaveID	Type	Addr	NPoint	Poll Time	Max Error	Position	Start Bit	Swap	Mnemonic
1	y	1	Coil Status	0	1	1000	0	0	0	n	1 M – Coil 1
2	y	1	Coil Status	1	1	1000	0	0	1	n	1 M – Coil 2
3	y	1	Coil Status	2	1	1000	0	0	2	n	1 M – Coil 3
4	y	1	Coil Status	3	1	1000	0	0	3	n	1 M – Coil 4
5	y	1	Coil Status	4	1	1000	0	0	4	n	1 M – Coil 5
6	y	1	Coil Status	5	1	1000	0	0	5	n	1 M – Coil 6
7	y	1	Coil Status	6	1	1000	0	0	6	n	1 M – Coil 7
8	y	1	Coil Status	7	1	1000	0	0	7	n	1 M – Coil 8
9	y	1	Coil Status	8	1	1000	0	1	0	n	1 M – Coil 9
10	y	1	Coil Status	9	1	1000	0	1	1	n	1 M – Coil 10
11	y	1	Coil Status	10	1	1000	0	1	2	n	1 M – Coil 11
12	y	1	Coil Status	11	1	1000	0	1	3	n	1 M – Coil 12
13	y	1	Coil Status	12	1	1000	0	1	4	n	1 M – Coil 13
14	y	1	Coil Status	13	1	1000	0	1	5	n	1 M – Coil 14
15	y	1	Coil Status	14	1	1000	0	1	6	n	1 M – Coil 15
16	y	1	Coil Status	15	1	1000	0	1	7	n	1 M – Coil 16
17	y	1	Holding Register	32	1	1000	0	2	0	y	2 M – Holding Register 1
18	y	1	Holding Register	33	1	1000	0	4	0	y	2 M – Holding Register 2
19	y	1	Holding Register	34	1	1000	0	6	0	y	2 M – Holding Register 3

© EKINEX S.p.A. – All rights reserved

N	En	SlaveID	Type	Addr	NPoint	Poll Time	Max Error	Position	Start Bit	Swap	Mnemonic
20	y	1	Holding Register	35	1	1000	0	8	0	y	2 M – Holding Register 4
21	y	1	Holding Register	36	1	1000	0	10	0	y	2 M – Holding Register 5
22	y	1	Holding Register	37	1	1000	0	12	0	y	2 M – Holding Register 6
23	y	1	Holding Register	38	1	1000	0	14	0	y	2 M – Holding Register 7
24	y	1	Holding Register	39	1	1000	0	16	0	y	2 M – Holding Register 8
25	y	1	Holding Register	40	1	1000	0	18	0	y	2 M – Holding Register 9
26	y	1	Holding Register	41	1	1000	0	20	0	y	2 M – Holding Register 10
27	y	1	Holding Register	42	1	1000	0	22	0	y	2 M – Holding Register 11
28	y	1	Holding Register	43	1	1000	0	24	0	y	2 M – Holding Register 12
29	y	1	Holding Register	44	1	1000	0	26	0	y	2 M – Holding Register 13
30	y	1	Holding Register	45	1	1000	0	28	0	y	2 M – Holding Register 14
31	y	1	Holding Register	46	1	1000	0	30	0	y	2 M – Holding Register 15
32	y	1	Holding Register	47	1	1000	0	32	0	y	2 M – Holding Register 16

Configuration grid, *Set Modbus Access* form, *Modbus Write* tab.

N	En	SlaveID	Type	Addr	NPoint	Poll Time	On Change	Max Error	Position	Start Bit	Swap	Mnemonic
1	y	1	Coil Status	16	1	5000	y	0	0	0	n	3 M – Coil 17
2	y	1	Coil Status	17	1	5000	y	0	0	1	n	3 M – Coil 18
3	y	1	Coil Status	18	1	5000	y	0	0	2	n	3 M – Coil 19
4	y	1	Coil Status	19	1	5000	y	0	0	3	n	3 M – Coil 20
5	y	1	Coil Status	20	1	5000	y	0	0	4	n	3 M – Coil 21
6	y	1	Coil Status	21	1	5000	y	0	0	5	n	3 M – Coil 22
7	y	1	Coil Status	22	1	5000	y	0	0	6	n	3 M – Coil 23
8	y	1	Coil Status	23	1	5000	y	0	0	7	n	3 M – Coil 24
9	y	1	Coil Status	24	1	5000	y	0	1	0	n	3 M – Coil 25
10	y	1	Coil Status	25	1	5000	y	0	1	1	n	3 M – Coil 26
11	y	1	Coil Status	26	1	5000	y	0	1	2	n	3 M – Coil 27
12	y	1	Coil Status	27	1	5000	y	0	1	3	n	3 M – Coil 28
13	y	1	Coil Status	28	1	5000	y	0	1	4	n	3 M – Coil 29
14	y	1	Coil Status	29	1	5000	y	0	1	5	n	3 M – Coil 30
15	y	1	Coil Status	30	1	5000	y	0	1	6	n	3 M – Coil 31
16	y	1	Coil Status	31	1	5000	y	0	1	7	n	3 M – Coil 32
17	y	1	Holding Register	48	1	5000	y	0	2	0	n	4 M – Hold Reg 17
18	y	1	Holding Register	49	1	5000	y	0	4	0	n	4 M – Hold Reg 18
19	y	1	Holding Register	50	1	5000	y	0	6	0	n	4 M – Hold Reg 19
20	y	1	Holding Register	51	1	5000	y	0	8	0	n	4 M – Hold Reg 20
21	y	1	Holding Register	52	1	5000	y	0	10	0	n	4 M – Hold Reg 21
22	y	1	Holding Register	53	1	5000	y	0	12	0	n	4 M – Hold Reg 22
23	y	1	Holding Register	54	1	5000	y	0	14	0	n	4 M – Hold Reg 23
24	y	1	Holding Register	55	1	5000	y	0	16	0	n	4 M – Hold Reg 24
25	y	1	Holding Register	56	1	5000	y	0	18	0	n	4 M – Hold Reg 25
26	y	1	Holding Register	57	1	5000	y	0	20	0	n	4 M – Hold Reg 26
27	y	1	Holding Register	58	1	5000	y	0	22	0	n	4 M – Hold Reg 27
28	y	1	Holding Register	59	1	5000	y	0	24	0	n	4 M – Hold Reg 28
29	y	1	Holding Register	60	1	5000	y	0	26	0	n	4 M – Hold Reg 29
30	y	1	Holding Register	61	1	5000	y	0	28	0	n	4 M – Hold Reg 30
31	y	1	Holding Register	62	1	5000	y	0	30	0	n	4 M – Hold Reg 31
32	y	1	Holding Register	63	1	5000	y	0	32	0	n	4 M – Hold Reg 32

Configuration grid, *KNX Set Access* form.

N	En	Source Addr	Dest Group	APCI	Prio	Format	Ext	ReTest	On CMD	On Change	On Timer	Poll Time	Pos	Bit Mode	Len
1	y	1.2.4	1/4/1	write	Low	None	n	n	n	y	n	1000	0	0	1
2	y	1.2.4	1/4/2	write	Low	None	n	n	n	y	n	1000	0	1	1
3	y	1.2.4	1/4/3	write	Low	None	n	n	n	y	n	1000	0	2	1
4	y	1.2.4	1/4/4	write	Low	None	n	n	n	y	n	1000	0	3	1
5	y	1.2.4	1/4/5	write	Low	None	n	n	n	y	n	1000	0	4	1

© EKINEX S.p.A. – All rights reserved

N	En	Source Addr	Dest Group	APCI	Prio	Format	Ext	ReTest	On CMD	On Change	On Timer	Poll Time	Pos	Bit Mode	Len
6	y	1.2.4	1/4/6	write	Low	None	n	n	n	y	n	1000	0	5	1
7	y	1.2.4	1/4/7	write	Low	None	n	n	n	y	n	1000	0	6	1
8	y	1.2.4	1/4/8	write	Low	None	n	n	n	y	n	1000	0	7	1
9	y	1.2.4	1/4/9	write	Low	None	n	n	n	y	n	1000	1	0	1
10	y	1.2.4	1/4/10	write	Low	None	n	n	n	y	n	1000	1	1	1
11	y	1.2.4	1/4/11	write	Low	None	n	n	n	y	n	1000	1	2	1
12	y	1.2.4	1/4/12	write	Low	None	n	n	n	y	n	1000	1	3	1
13	y	1.2.4	1/4/13	write	Low	None	n	n	n	y	n	1000	1	4	1
14	y	1.2.4	1/4/14	write	Low	None	n	n	n	y	n	1000	1	5	1
15	y	1.2.4	1/4/15	write	Low	None	n	n	n	y	n	1000	1	6	1
16	y	1.2.4	1/4/16	write	Low	None	n	n	n	y	n	1000	1	7	1
17	y	1.2.4	1/4/33	write	Low	None	n	n	n	y	n	1000	2	No	2
18	y	1.2.4	1/4/34	write	Low	None	n	n	n	y	n	1000	4	No	2
19	y	1.2.4	1/4/35	write	Low	None	n	n	n	y	n	1000	6	No	2
20	y	1.2.4	1/4/36	write	Low	None	n	n	n	y	n	1000	8	No	2
21	y	1.2.4	1/4/37	write	Low	None	n	n	n	y	n	1000	10	No	2
22	y	1.2.4	1/4/38	write	Low	None	n	n	n	y	n	1000	12	No	2
23	y	1.2.4	1/4/39	write	Low	None	n	n	n	y	n	1000	14	No	2
24	y	1.2.4	1/4/40	write	Low	None	n	n	n	y	n	1000	16	No	2
25	y	1.2.4	1/4/41	write	Low	None	n	n	n	y	n	1000	18	No	2
26	y	1.2.4	1/4/42	write	Low	None	n	n	n	y	n	1000	20	No	2
27	y	1.2.4	1/4/43	write	Low	None	n	n	n	y	n	1000	22	No	2
28	y	1.2.4	1/4/44	write	Low	None	n	n	n	y	n	1000	24	No	2
29	y	1.2.4	1/4/45	write	Low	None	n	n	n	y	n	1000	26	No	2
30	y	1.2.4	1/4/46	write	Low	None	n	n	n	y	n	1000	28	No	2
31	y	1.2.4	1/4/47	write	Low	None	n	n	n	y	n	1000	30	No	2
32	y	1.2.4	1/4/48	write	Low	None	n	n	n	y	n	1000	32	No	2
33	y	0.0.0	1/4/17	read	Low	None	n	n	n	n	y	5000	0	0	1
34	y	0.0.0	1/4/18	read	Low	None	n	n	n	n	y	5000	0	1	1
35	y	0.0.0	1/4/19	read	Low	None	n	n	n	n	y	5000	0	2	1
36	y	0.0.0	1/4/20	read	Low	None	n	n	n	n	y	5000	0	3	1
37	y	0.0.0	1/4/21	read	Low	None	n	n	n	n	y	5000	0	4	1
38	y	0.0.0	1/4/22	read	Low	None	n	n	n	n	y	5000	0	5	1
39	y	0.0.0	1/4/23	read	Low	None	n	n	n	n	y	5000	0	6	1
40	y	0.0.0	1/4/24	read	Low	None	n	n	n	n	y	5000	0	7	1
41	y	0.0.0	1/4/25	read	Low	None	n	n	n	n	y	5000	1	0	1
42	y	0.0.0	1/4/26	read	Low	None	n	n	n	n	y	5000	1	1	1
43	y	0.0.0	1/4/27	read	Low	None	n	n	n	n	y	5000	1	2	1
44	y	0.0.0	1/4/28	read	Low	None	n	n	n	n	y	5000	1	3	1
45	y	0.0.0	1/4/29	read	Low	None	n	n	n	n	y	5000	1	4	1
46	y	0.0.0	1/4/30	read	Low	None	n	n	n	n	y	5000	1	5	1
47	y	0.0.0	1/4/31	read	Low	None	n	n	n	n	y	5000	1	6	1
48	y	0.0.0	1/4/32	read	Low	None	n	n	n	n	y	5000	1	7	1
49	y	0.0.0	1/4/49	read	Low	None	n	n	n	n	y	5000	2	No	2
50	y	0.0.0	1/4/50	read	Low	None	n	n	n	n	y	5000	4	No	2
51	y	0.0.0	1/4/51	read	Low	None	n	n	n	n	y	5000	6	No	2
52	y	0.0.0	1/4/52	read	Low	None	n	n	n	n	y	5000	8	No	2
53	y	0.0.0	1/4/53	read	Low	None	n	n	n	n	y	5000	10	No	2
54	y	0.0.0	1/4/54	read	Low	None	n	n	n	n	y	5000	12	No	2
55	y	0.0.0	1/4/55	read	Low	None	n	n	n	n	y	5000	14	No	2
56	y	0.0.0	1/4/56	read	Low	None	n	n	n	n	y	5000	16	No	2
57	y	0.0.0	1/4/57	read	Low	None	n	n	n	n	y	5000	18	No	2
58	y	0.0.0	1/4/58	read	Low	None	n	n	n	n	y	5000	20	No	2
59	y	0.0.0	1/4/59	read	Low	None	n	n	n	n	y	5000	22	No	2
60	y	0.0.0	1/4/60	read	Low	None	n	n	n	n	y	5000	24	No	2
61	y	0.0.0	1/4/61	read	Low	None	n	n	n	n	y	5000	26	No	2
62	y	0.0.0	1/4/62	read	Low	None	n	n	n	n	y	5000	28	No	2
63	y	0.0.0	1/4/63	read	Low	None	n	n	n	n	y	5000	30	No	2
64	y	0.0.0	1/4/64	read	Low	None	n	n	n	n	y	5000	32	No	2

© EKINEX S.p.A. – All rights reserved

6 Warning

- Installation, electrical connection, configuration and commissioning of the device can only be carried out by qualified personnel.
- Opening the housing of the device causes the immediate end of the warranty period.
- ekinex® KNX defective devices must be returned to the manufacturer at the following address:

EKINEX S.p.A. Via Novara 37, I-28010 Vaprio d'Agogna (NO) Italy.

7 Other information

- This application manual is aimed at installers, system integrators and planners
- For further information on the product, please contact the ekinex® technical support at the e-mail address: support@ekinex.com or visit the website www.ekinex.com
- KNX® and ETS® are registered trademarks of KNX Association cvba, Brussels

© EKINEX S.p.A. The company reserves the right to make changes to this documentation without notice.