Comunication with ZCS devices

Zucchetti Centro Sistemi S.p.A. - Green Innovation Division
Palazzo dell'Innovazione - Via Lungarno, 167-201
52028 Terranuova Bracciolini - Arezzo, Italy
tel. +39 055 91971 - fax. +39 055 9197515
innovation@zcscompany.com - zcs@pec.it - **zcsazzurro.com**

Zucchetti Centro Sistemi S.p.A. - Green Innovation Division
Palazzo dell'Innovazione - Via Lungarno, 167-201
52028 Terranuova Bracciolini - Arezzo, Italy
tel. +39 055 91971 - fax. +39 055 9197515
innovation@zcscompany.com - zcs@pec.it - **zcsazzurro.com**

1. Intro

Rev.	*Release date*	*Test owner*	*Note*
00	03/07/2025	LA	First release

The purpose of this document is to encapsulate all methods of communication with ZCS devices. For this reason, the document will be divided by devices and not by inverters, focusing mainly, as far as Stick Loggers are concerned, on the version:

- ZSM-ETH-USB
- ZSM-WIFI-USB

The second chapter will cover the whole datalogger part, ZVM-Connext and Azure HUB.

Next we will consider communication via api, both from the portal. Below is the documentation for the third-party API.

New Web Portal www.zcsazzurrosystemsweb.com: API Azzurro System.pdf
Old Web Portal www.zcsazzurroportal.com: API Azzurro Portal.pdf

We will conclude with some specific applications of these communication protocols and with the new products.

NB: all the modbus map must be required by open a ticket on the website www.zcsazzurro.com . The same for the Azzurro System API **token** or the Azzurro Portal API **Client Code**.

NB: all the application will be followed by a Python pseudocode.

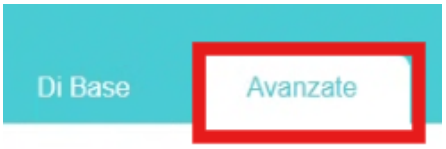
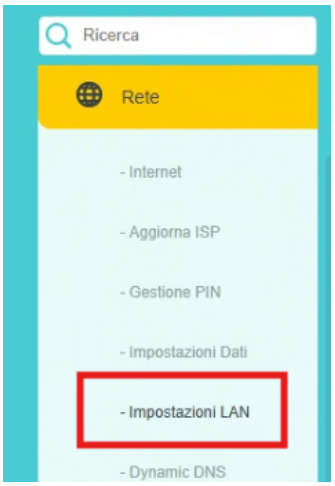
2. ZSM-ETH-USB

Communication with the inverters via ZSM-ETH-USB is done via Modbus TCP/IP over RTU. The structure of this communication therefore depends directly on the IP generated by the router to which the ETH stick is connected. All communication via ZSM-ETH-USB is done via port 8899.

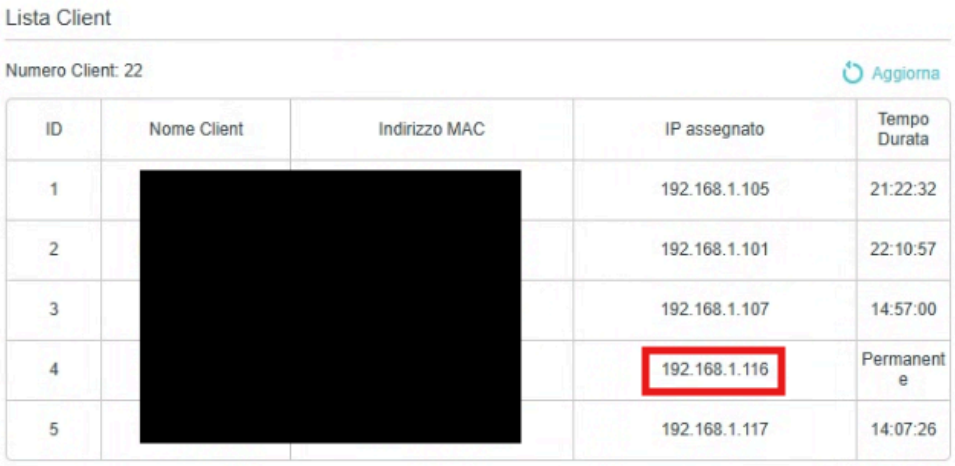
2.1 Example: connection to home router

For home use with a tp-link router, you can access your home router by typing 192.168.1.1 on the search bar of any browser. This search is fine for any type of router, only the subsequent screens may be slightly different.

Once you have entered the router wizard with your router password, you can search for LAN-connected devices by clicking on

Step 1	Step 2
	

Next, on the page that appears, you will find the various connected devices including, as in this example, a ZSM-ETH-USB antenna with the IP address 192.168.1.116



2.2 Example: communication via pseudocode

Once the IP of the ETH antenna has been set, it is possible to communicate with it, both read and write. Below is a possible Python function for TCP communication with the inverter.

ETH.py

In this code, you will find the entire communication part, both read and write. In writing, an array must be used in the value field, while in reading, the value field is the scaling of the read itself.

The new generation ZCS inverters have two Modbus functions, one for writing, and one for reading:

- Function code 0x03 for reading
- Function code 0x10 for writing

For the correct values to be entered when reading or writing, please refer to the modbus map of the respective inverter.

3. ZSM-WIFI-USB

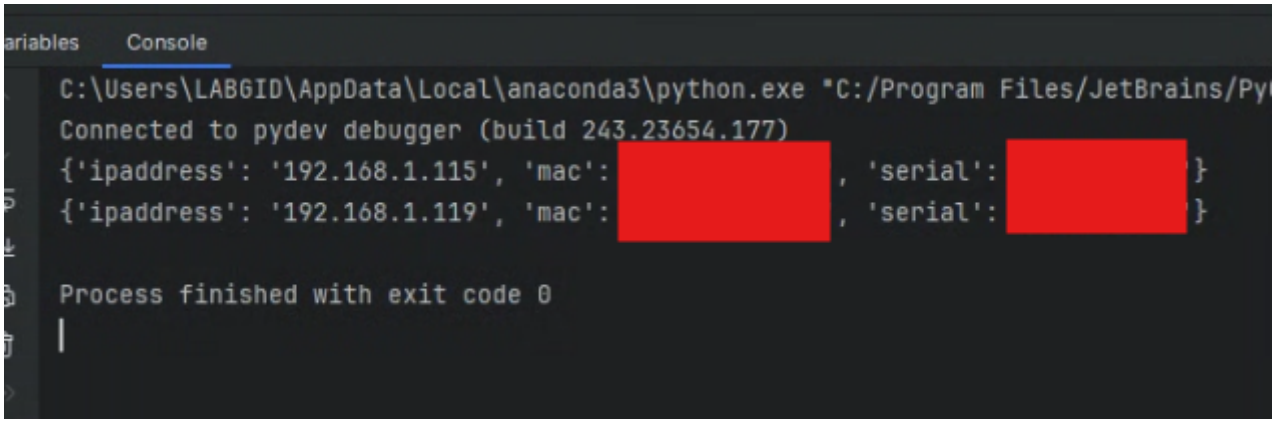
Communication with the inverters via ZSM-WIFI-USB is done via Modbus TCP/IP over RTU. The structure of this communication therefore depends directly on the IP generated by the router to which the WIFI stick is connected. All communication via ZSM-ETH-WIFI is done in LAN, so the various scripts below will only work if connected to the same local network as the WiFi device.

You can find the complete documentation of the PysolarmanV5 library at the following link: [pysolarmanv5 documentation]

3.1 Example: communication via pseudocode

To find the IP of the WIFI antenna, you must first scan your local network to see if ZSM-WIFI-USB antennas are found. The code below is an example for doing this scan:

WIFI - scanner.py



Once I have found the IP of the individual WIFI antennas, we can send write and read commands. Below is a possible Python function for TCP communication with the inverter.

WIFI - logger.py

In this code, you will find the entire communication part, both read and write.

Normally, our new generation inverters have two modbus functions, one for writing, and one for reading:

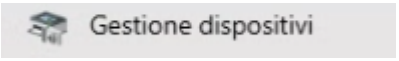
- Function code 0x03 for reading
- Function code 0x10 for writing

For the correct values to be entered when reading or writing, please refer to the modbus map of the respective inverter.

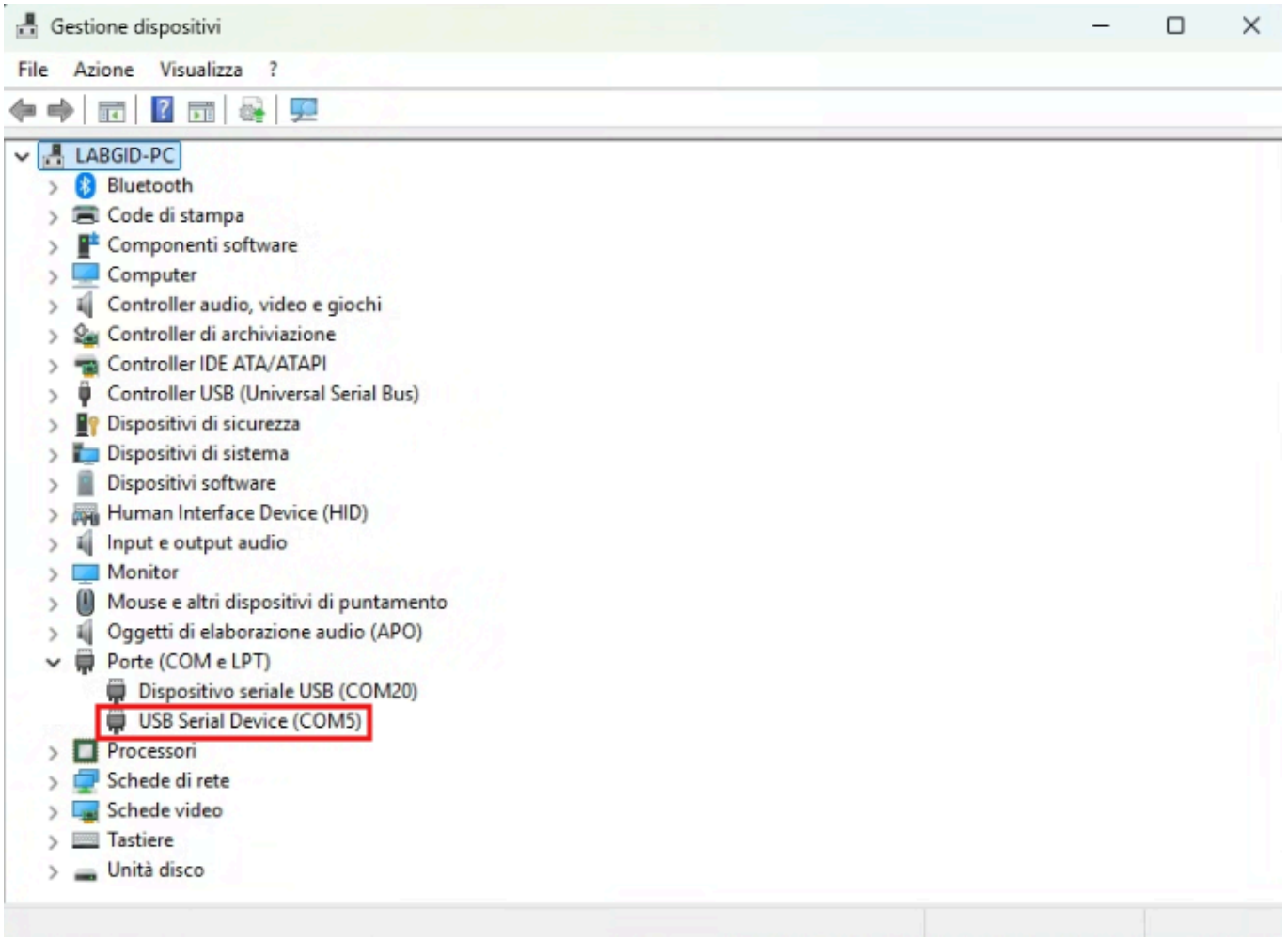
4. ZSM-CONNEXT

Using the USB-C power supply inside the ZVM-Connext it is possible to send read and write commands to the ZVM-Connext itself:

⚠ **Attenzione**	**The ZVM-Connext is already powered by USB-C and there is no need to power the system further. If powered in both ways, the system risks damage.**

To communicate directly with the ZVM-Connext, you need to know the COM port on which the device is connected, and to do this you need to go to Device Manager 

and search the COM connections for the ZVM-Connext:



Once the device has been found, you can interact with the ZVM-Connext by following a simple COM communication using the NI-Visa libraries (NI-Visa download) and use terminal commands directly on the ZVM-Connext itself.

4.1 Example: communication via pseudocode

Below is a library developed with the main functions available for the ZVM-Connext:

AzzurroConnext.py

5. AZZURRO HUB

This guide describes the functionality and use of the passthrough mode implemented via three Modbus TCP servers, each associated with a dedicated RS485 line. This function allows direct communication with connected Modbus RTU devices via TCP connections.
Supported Modbus functions: Only Modbus functions 0x03 (read multiple address) and 0x10 (write multiple address) are supported.
The separate Modbus TCP servers are:

TCP Port	RS485 Line
55400	RS485 dedicated to inverters
55402	RS485 dedicated to EVCH
55401	RS485 dedicated to meters

Only one active connection is allowed on each port at a time. When a new connection is established, any existing active connection is immediately disconnected to give priority to the new connection.

Attenzione	**Before connecting, make sure there are no operations in progress, otherwise you risk interrupting existing sessions.**

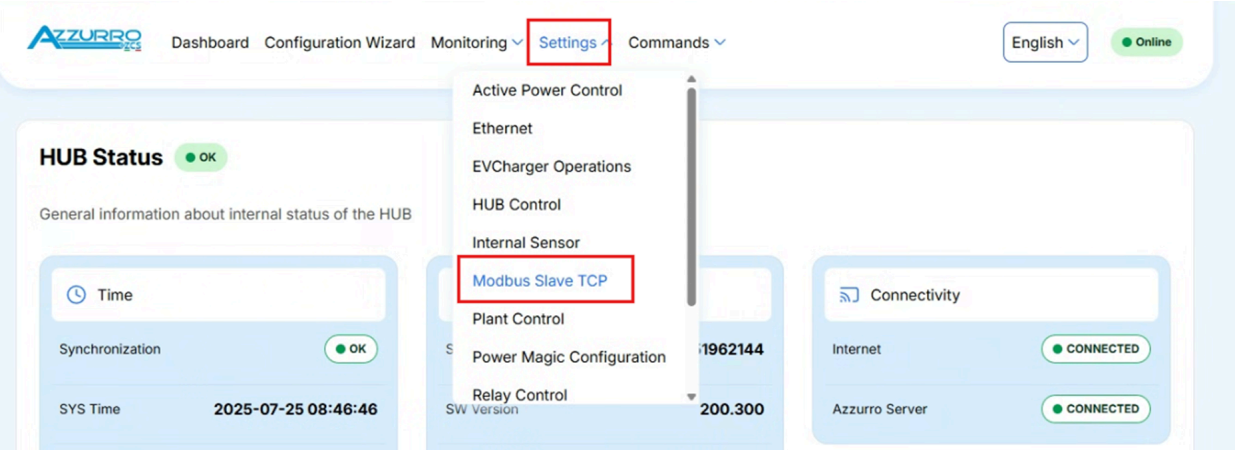
If it is necessary to set a static IP address to connect to the HUB:

- You can disable DHCP for Ethernet in Settings → Ethernet → DHCP
- You can deactivate DHCP for WiFi in Settings → WiFi Station → DHCP
- You can set a static MAC address for WiFi in Settings → WiFi Station → Local MAC

This allows you to manually configure network settings.

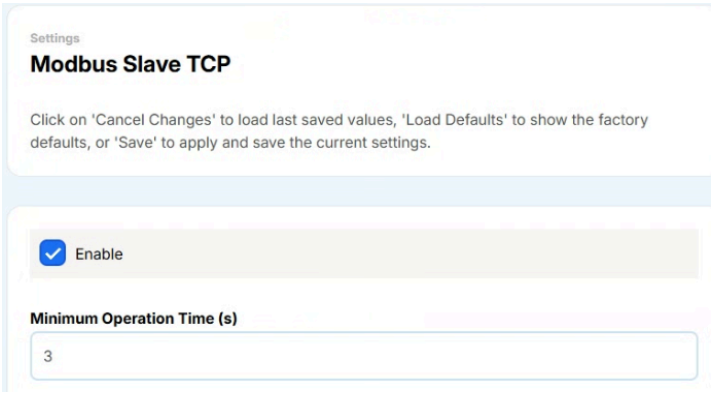
For activate the passthrough mode:

- Accessing the internal portal
- Go to Settings → Modbus Slave TCP



Zucchetti Centro Sistemi S.p.A. - Green Innovation Division
Palazzo dell'Innovazione - Via Lungarno, 167-201
52028 Terranuova Bracciolini - Arezzo, Italy
tel. +39 055 91971 - fax. +39 055 9197515
innovation@zcscompany.com - zcs@pec.it - **zcsazzurro.com**

- From this section, the service can be activated or deactivated.



On the same page, it is also possible to set a minimum delay between Modbus requests, which is useful to avoid overloading or line saturation.

Attenzione	**Reducing the minimum delay between operations may compromise the proper functioning of the system, causing communication errors, congestion of the RS485 line, or abnormal behaviour of connected devices.**

Attenzione	**Access to the configuration and monitoring pages requires the password '0715'.**

5.1 Example: communication via pseudocode

Once the IP of the **Azzurro Hub** has been set and the necessary settings have been made, the commands are enabled for both reading and writing. Below is a possible Python function for TCP communication with the inverter via the Azure Hub.

AzzurroHub.py

In this code, you will find the entire communication part, both read and write.

For the correct values to be entered when reading or writing, please refer to the modbus map of the respective inverter.

6. API

6.1 Azzurro Portal

As stated in the introduction, the API documentation is as follows:

API Azzurro Portal.pdf

The ClientCode must be required via ticket on zcsazzurro.com.

These APIs work via a POST call. Below is an example of both functions described in the document above:

API realtime - AzzurroPortal.py
API historic - AzzurroPortal.py

For this API to work, both the inverter and the chip making the POST request need to be connected online.

The maximum number of days that can be requested via this API is 1 day with a timeout of at least 3s.

6.1.1 Home Assistant integration

An interesting application of the Azzurro Portal API is the Home Assistant module developed by the user aturri and all documentation can be found in the following link:

GitHub - aturri/ha-zcsazzurro: A modern integration for ZCS Azzurro devices in Home Assistant

Basically, in your Home Assistant block (https://www.home-assistant.io/) you can find the readings of your inverter directly in the dashboard thanks to these APIs and their integration.

6.2 Azzurro System

6.2.1 Reading API

As stated in the introduction, the API documentation is as follows:
API Azzurro System.pdf
The token must be required via ticket on the website zcsazzurro.com.

These APIs work via a POST call. Below is an example of both functions described in the document above:

API realtime - AzzurroSystem.py
API historic - AzzurroSystem.py
API Alarm Realtime - AzzurroSystem.py
API Alarm Historic - AzzurroSystem.py

For this API to work, both the inverter and the chip making the POST request need to be connected online.

The maximum number of days that can be requested via the Historic API is approximately 6 days with a request timeout of at least 10s.

6.1.2. Writing API

As stated in the introduction, the API documentation is as follows:

API Azzurro System.pdf

The token must be required via ticket on the website zcsazzurro.com.

These APIs work via a POST call. Below is an example of writing the log:

API Write - AzzurroSystem.py
API Write Callback - AzzurroSystem.py

In order for this API to work, you need both the inverter and the chip making the POST request to be connected online and the inverter to be associated with an advanced system connected to your account of which you are the owner. In addition, you need to create a callback service on which the response json to the API command sent will be sent.

Attenzione	**The write API can only be used with ZSM-WIFI-USB and ZSM-ETH-USB monitoring systems.**

7. PSC x00

7.1. PSC 100

The PSC 100 is identifiable as a single panel that can be wall-mounted by means of special fixings and is internally composed of the following components:

1. Industrial Ethernet or Fibre Optic Ring Switches
2. Communication Manager
3. CCO module for PBUS communication
4. Breakers, fuses, arresters

The data is transmitted to the Central Plant Controller (CCI or Plant Control System) via the Fibre Optic or Ethernet ring connection, realising two-way communication between the inverter and the monitoring system.

The inverter models currently compatible with the PSC 100 are listed below:

- 250KTL-HV Z0
- 330KTL-HV Z0
- 350KTL-HV Z0

Communication can be done via fibre optics.

8. Power Magic

Communication with the Power Magic is done via Modbus TCP/IP. The structure of this communication therefore depends directly on the IP generated by the router to which Power Magic is connected or the IP set by the IT where the system is integrated. All communication with the Power Magic is done via port 502.

8.1 Example: communication via pseudocode

Once the Power Magic IP has been set up and the necessary settings have been made, the commands are enabled for both reading and writing. Below is a possible Python function for TCP communication with the inverter via Power Magic.

PowerMagic.py

In this code, you will find the entire communication part, both read and write. In writing, an array must be used in the value field, while in reading, the value field is the scaling of the read itself.

The new generation Power Magic have two modbus functions, one for writing, and one for reading:

- Function code 0x03 for reading
- Function code 0x06 for writing the single register
- Function code 0x10 for writing multiple registers

For the correct values to be entered when reading or writing, please refer to the modbus map of Power Magic to be requested by opening a ticket on the website zcsazzurro.com .

9. Heat Pump

ZCS heat pumps have an RS485 communication protocol. To know the correct values to enter when reading or writing, please refer to the modbus map of the heat pumps, which can be obtained by opening a ticket at zcsazzurro.com .