



MC252 Documentation

<https://www.overvis.com/docs/en/mc252/>

2026-03-30

Table of Contents

1. Overvis MC252
2. Quick Start Guide
3. Overvis MC252 Modbus Gateway with 4G/Ethernet/RS-485 Operating Manual
4. Connections and Network Setup
5. User Web Interface
6. Modbus Interface Reference
7. Memory Card Data Saving
8. HTTP API
9. Authentication
10. System Information
11. Settings
12. System Commands
13. Updates
14. GSM
15. Modbus
16. SD Card
17. HTTP Status Codes
18. Operations Logic Programming Reference
19. Firmware Update
20. Firmware Downloads
21. Additional Software

Overvis MC252



The Overvis MC252 is a programmable RS-485 to Ethernet / GSM / LTE gateway and controller designed for industrial automation and remote monitoring applications. It bridges Modbus RTU/ASCII protocols to modern TCP/IP networks, enabling seamless integration of legacy industrial equipment with SCADA systems, cloud platforms, and IoT infrastructures.

Complete Documentation Available

View all documentation on a single page – Perfect for printing or offline reading

Key Capabilities

- **Protocol Bridge:** Convert RS-485 Modbus RTU/ASCII to Modbus TCP for seamless equipment integration
- **Dual Connectivity:** Ethernet (10/100 Mbps) and 4G/LTE with GSM fallback, configurable priority and automatic failover
- **Secure Communication:** Built-in WireGuard VPN client for encrypted tunnels to servers and cloud platforms
- **Flexible Operation:** Function as Modbus master, slave, or transparent bridge; address remapping option
- **Web Interface:** Complete browser-based configuration and monitoring interface with multi-level access control
- **HTTP API:** RESTful API for programmatic access to device settings, functions and real-time data
- **Programmable Logic:** Execute custom automation tasks, event triggers, and data logging via scripting
- **Data Storage:** MicroSD card support (up to 32TB) for firmware updates, configuration backups, scripting, and data logging

- **On-Device Display:** Built-in OLED screen shows real-time connection status and diagnostic information, and speeds-up quick start
- **Industrial Grade:** DIN-rail mountable, IP20 rated, operating temperature –35°C to +55°C

Common Use Cases

- Remote monitoring and control of RS-485 equipment over cellular networks
- Bridging legacy Modbus RTU devices to modern SCADA systems via Ethernet
- Network extension and consolidation across multiple locations
- Data logging and archival with local storage
- SMS-based alerting and remote diagnostics
- Interconnection of monitoring and control equipment
- Providing timer or schedule functionality for control equipment
- Integration with cloud platforms and IoT systems

Documentation Structure

Getting Started

- **Quick Start Guide** – Get your MC252 up and running in 15 minutes with step-by-step setup instructions
- **Operating Manual** – Complete safety information, technical specifications, and operating procedures
- **User Interface** – Web interface overview and navigation guide

Configuration

- **Connections & Network Setup** – Ethernet, GSM/LTE, VPN, and server link recommendations
- **Modbus Interface** – Modbus RTU/ASCII and TCP configuration, parameter reference
- **Memory Card** – Using microSD cards for storage, automations, logging, and firmware updates

Advanced Features

- **Logic Programming** – Create custom automation tasks, triggers, and event handlers
- **HTTP API Reference** – Complete REST API documentation for programmatic device control
- **Firmware & Updates** – Firmware versions, update procedures, and release notes
- **Software Tools** – Download utilities for testing and configuration

Resources

- MC252 Product Page – Full product information and specifications
- One-Page Documentation – Complete documentation on a single page for printing
- MC252 Full Manual (PDF) – Complete operating manual

Support

- **Knowledge Base:** Browse this documentation for detailed guides and references
- **Support Center:** Visit our Support Center for FAQs and troubleshooting
- **Report an Issue:** Submit a support ticket for technical assistance
- **Contact Sales:** Questions about purchasing or licensing? Contact our sales team

Ready to get started? Head to the Quick Start Guide to set up your MC252 in minutes.

Quick Start Guide

This guide walks you through setting up your Overvis MC252 gateway from wiring to successful connection in about 15 minutes.

The Overvis MC252 is a programmable gateway that bridges RS-485 Modbus devices to modern networks, letting you monitor and control industrial equipment remotely via Ethernet or cellular connections.

Read Safety Instructions First

Before setting up or operating the MC252 device, you must thoroughly read the safety information in the operating manual. This includes electrical safety requirements, installation precautions, and proper operating procedures.

Operating the device without following these instructions may result in equipment damage, injury, or loss of warranty coverage.

What You'll Need

From the delivery box: MC252 device, Ethernet cable, GSM antenna, 2GB microSD card.

You'll also need:

- 12V DC power supply (9-30V range supported)
- Device with web browser for initial setup (computer, phone, or tablet)
- RS-485 Modbus devices to connect
- Twisted-pair cable for RS-485 (Category 1+, shielded recommended)
- Stranded wire ($\geq 1 \text{ mm}^2$ cross-section), ferrules, and tools (screwdriver, wire stripper)

Safety First

Always disconnect power before making connections. Never open the device or operate it with damaged housing.

Keep water away from terminals.

Step 1: Physical Setup

Mount your MC252 on a standard 35mm DIN rail in a well-ventilated location. The device operates from -35°C to $+55^{\circ}\text{C}$, but avoid areas with excessive vibration, humidity, or corrosive atmospheres.

Install memory card: Insert the provided 2GB microSD card into the SD slot. The memory card is required for firmware updates, data logging, and automation logic programming features.

Install SIM card (if using GSM/LTE): If you plan to use cellular connectivity, insert your SIM card (with data service enabled) into the SIM slot now. Attach GSM antenna to the ANT connector.

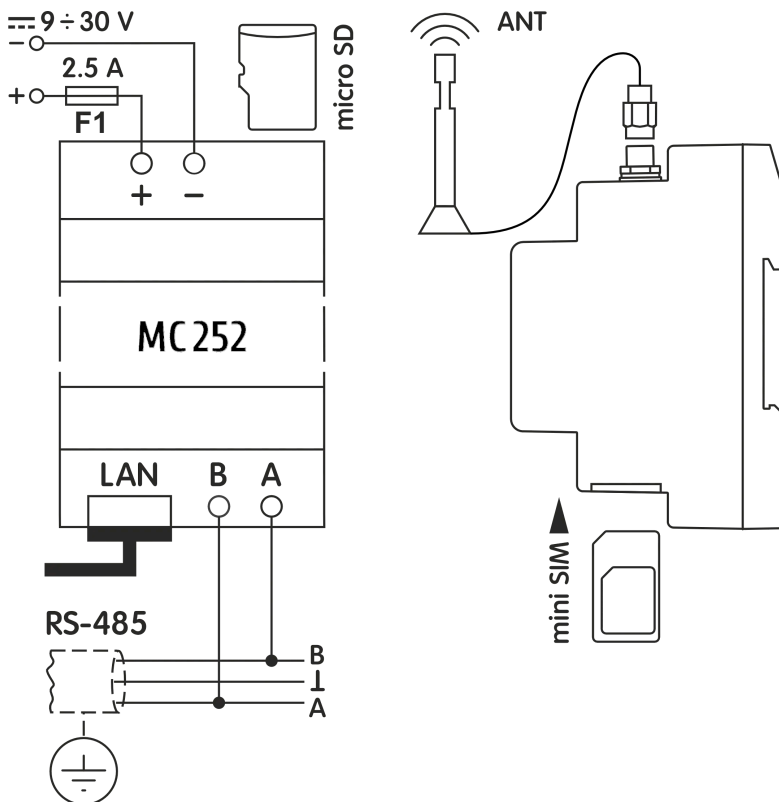
Network connection: Connect the Ethernet cable from MC252's LAN port to your router or directly to your computer.

Before making any electrical connections, ensure all devices are powered off.

Tighten terminal screws to 0.4 N·m — enough for solid contact without damage.

RS-485 wiring: Connect twisted-pair cable to MC252's RS-485 terminals: terminal A for non-inverted signal, terminal B for inverted signal (your devices might label these as A/D+/+ and B/D-/ - respectively). Run the cable to your Modbus devices and connect accordingly. Use shielded twisted-pair cable (Category 1 or better) for reliable communication over distance.

Power: Connect 12V DC power to the 9÷30V terminals. For safety, install a 2.5A fuse in the power circuit. Use stranded wire with ferrules, stripped 5mm. Wire cross-section should be 0.5–3 mm².



Step 2: Power On

When you apply power, all LEDs light up briefly during the 2-second initialization, then the device spends 10–15 seconds establishing network connections.

The PWR LED stays on to confirm power. The LAN LED turns on when Ethernet connects (blinking during data transfers). The GSM LED blinks slowly (every 1.5s) when registered to cellular network, or rapidly (3 times/second) after TCP/IP link start.

The display shows IP addresses (prefixed with (E) for Ethernet or (G) for GSM), plus data rates and signal strength.

Step 3: Access Web Interface and Configure

Note the IP address from the display (press R to wake it). If connected via Ethernet with DHCP, you'll see the assigned address.

› [Connecting directly to a PC?](#)

Open a web browser and enter the IP address shown on the display (e.g., `http://192.168.0.111`).

Login to dashboard: You'll see a login page. Press the R button on MC252 for temporary password-free access.

Configure network (LAN/Internet tab): Enable/disable DHCP, set static IP if needed, adjust subnet mask and gateway. For GSM, enter your mobile operator's APN (Access Point Name) and SIM card PIN (if present).

Configure Modbus RTU/ASCII (Modbus tab): Consult the manual of your Modbus devices for their exact baud rate, parity and stopbits settings, then match these exactly in MC252. If no manual is available, start with common values: 9600 or 19200 baud, Modbus RTU protocol, and AUTO-STOP parity (which automatically detects the correct stopbits setting).

MC252 has Modbus TCP server enabled by default on port 502. You can change the port or set Modbus passwords here if needed.

For detailed information about all web interface options, see the [Web Interface Guide](#).

Note

Click "Save & Reset" after changes. The device restarts in about 15 seconds with new settings.

Step 4: Test Modbus TCP Connection

Use a Modbus client application to test the gateway. Connect to MC252's IP address (from the display) on port 502.

- 1 Test MC252 itself first:** Read a holding register from MC252's own Modbus address (unit ID 111 by default). Try reading register 1 (firmware version) or register 175 (supply voltage). If this works, MC252's Modbus TCP server is correctly available.
- 2 Test connected devices:** Send a read request to one of your RS-485 devices using its actual Modbus address (check your device's manual for its address, commonly 1). Try reading a known register from that device.
- 3 Modbus passwords:** By default, Modbus read access is unrestricted (empty password). If you've set a password in the web interface, you'll need to send the password first. See the [Modbus Interface guide](#) for password authentication details.

Common issues:

- **No response from MC252 itself** → Check IP address, port 502, firewall settings
- **MC252 responds but RS-485 devices don't** → Verify baud rate, parity, A/B wiring, device addresses, Modbus protocol selection and power supply
- **Wrong data returned** → Check register address matches your device's documentation

Step 5: Connect to Overvis Cloud (Optional)

Overvis Cloud provides remote monitoring and control through a web dashboard. The device label includes a QR code and PIN for quick setup.

- 1 **Enable Cloud connection:** Open MC252's web interface and go to the Cloud tab. Enable the cloud connection and click "Save & Reset." Wait for the device to restart and establish connection.
- 2 **Access Overvis server:** Scan the QR code on the device label or manually enter the link from the label (format: `https://c.overvis.com/ABCD1234` where the last part is your PIN). The link redirects to Overvis server's login page. Optionally follow the Overvis link on the Cloud tab.
- 3 **Login or create account:** Enter your credentials if you have an account. New users should create an account first.
- 4 **Create Network:** After login, Overvis shows the "Create Network" page. If you came from the QR code/link, the PIN is pre-filled. Otherwise, enter the PIN from the device label and click "Check connection."
- 5 **Verify connection status:** Overvis displays the connection status — either "Connected" or "Device is not connected to the server." If not connected, check that MC252 has Internet access (Ethernet or GSM) and verify the Cloud tab settings in the web interface.
- 6 **Configure your network:** Give your network a descriptive name (a "network" represents your MC252 plus all connected Modbus devices). MC252 itself (unit ID 111) is added automatically.
- 7 **Add connected devices:** Select your RS-485 device models from the dropdown menus and enter their Modbus addresses. Overvis creates device instances from templates matching your selections.
- 8 **Test communication:** Open a device page in Overvis and read its parameters to confirm real-time communication is working.

Troubleshooting cloud connection:

- **"Device is not connected"** → Check Internet connectivity (verify IP addresses on display), review Cloud tab in web interface for connection settings, status and errors
- **PIN not accepted** → Verify you entered the exact PIN from this device label
- **Can't create account** → Ensure email address entered is valid, check email for verification link

Troubleshooting

Display shows (E) 0.0.0.0: Normal during first 20-60 seconds while DHCP negotiates. If it persists, check cable connection and router DHCP. Try direct PC connection with static addressing settings.

Display shows (E) 192.168.0.111: DHCP failed or is unavailable. MC252 has switched to its factory default static IP address. Either configure your client device to the same subnet (192.168.0.x) or enable DHCP on your router.

Can't access web interface: Verify IP address entered in browser matches what's on the display. Ensure Ethernet address is used (the one marked with (E) in the second line). Check PC and MC252 are on the same subnet. Temporarily disable firewall. Press R for password-free access. Clear browser cache.

GSM won't connect: Verify SIM card is fully inserted (until you hear a click), check the antenna is attached, check the GSM settings. Check that data service is active. Check signal strength on display (should be >0%). Try relocating antenna to better position. Try to disable PIN. Try manually configured APN settings.

RS-485 devices don't respond: Most common issue is mismatched settings. Verify baud rate matches all the devices. Check A/B wiring polarity (swapping these is very common - this doesn't cause equipment damage, but prevents communication). Try increasing delays in MC252 settings for RS-485. Ensure protocol (Modbus RTU/ASCII) and parity are the same for all the devices. Start with 9600 baud, RTU, AUTO-STOP parity.

Service Button Quick Reference

The R button functions depend on press duration:

Quick press: Wake display, show status, grant temporary web access

Hold 2-8 seconds: Prepare for safe memory card removal and restart device

Hold 8+ seconds: Factory reset (erases all settings)

Caution

Protect the R button from unauthorized access with tamper-evident tape or by installing MC252 in a locked enclosure.

What's Next?

- MC252 Operating Manual — Complete technical documentation
- Web Interface Guide — Configuration options explained
- Connections Guide — Network setup, VPN, and security
- Modbus Interface — Protocol details and register reference

Need Help?

For technical support and assistance:

- Email: support@overvis.com
- Support portal: www.overvis.com/support

Overvis MC252 Modbus Gateway with 4G/Ethernet/RS-485 Operating Manual



This Operating Manual explains the design, safety requirements, operating rules, and maintenance procedures for the Overvis MC252 Modbus Gateway with 4G/Ethernet/RS-485.

Safety information

ATTENTION

ALL REQUIREMENTS OF THIS OPERATING MANUAL ARE MANDATORY.

TO ENSURE SAFE OPERATION, IT IS STRICTLY FORBIDDEN TO:

- PERFORM INSTALLATION OR MAINTENANCE WITHOUT DISCONNECTING MC252 FROM THE MAINS
- OPEN OR REPAIR MC252 YOURSELF
- OPERATE MC252 IF THE HOUSING IS MECHANICALLY DAMAGED

PREVENT WATER FROM REACHING THE TERMINALS OR INTERNAL PARTS.

During operation and maintenance, observe the requirements of applicable regulatory documents, including:

- Regulations for Operation of Consumer Electrical Installations;
- Safety Rules for Operation of Consumer Electrical Installations;

- Occupational Safety Rules for Operation of Electrical Installations.

Only qualified personnel who have studied this Operating Manual should perform installation, adjustment, and maintenance.

When used in accordance with this manual, the Overvis MC252 is safe in operation.

The Overvis MC252 meets the requirements of the following standards: EN 60947-1; EN 60947-6-2; EN 55011; EN 61000-4-2.

The device contains no hazardous substances in excess of maximum permissible limits.

The development and production quality management system complies with ISO 9001:2015.

General description

The Overvis MC252 is a programmable RS-485 to Ethernet / GSM / LTE gateway and controller. It:

- collects data from connected Modbus devices;
- transfers data to a server or cloud service;
- provides access to data via Modbus TCP or SMS text messages;
- tracks events and executes actions (for example, sending SMS notifications, writing values to Modbus devices, or logging values to a memory card).

The Overvis MC252 provides the following capabilities:

- **Flexible connection options:** Ethernet (wired) or wireless GPRS / FDD-LTE, automatic selection of the connection method to the cloud server, automatic or manual selection of GSM options and communication parameters, automatic or manual Ethernet settings
- **Access security:** no default password, passwords for Setup Mode and connections to the Modbus network, optional encrypted VPN connection, connecting only to a selected cloud server, passwords for SMS-based device control
- **Flexible RS-485 communication:** Modbus RTU or ASCII, even/odd/no parity, wide range of baud rates, adjustable delays
- **Programmable logic** for data collection, event tracking, and actions in response to events (see Operations logic programming)
- **Service functions:** real-time clock, astronomical timer, diagnostic logs, automatic or manual firmware update

Front panel and indicators

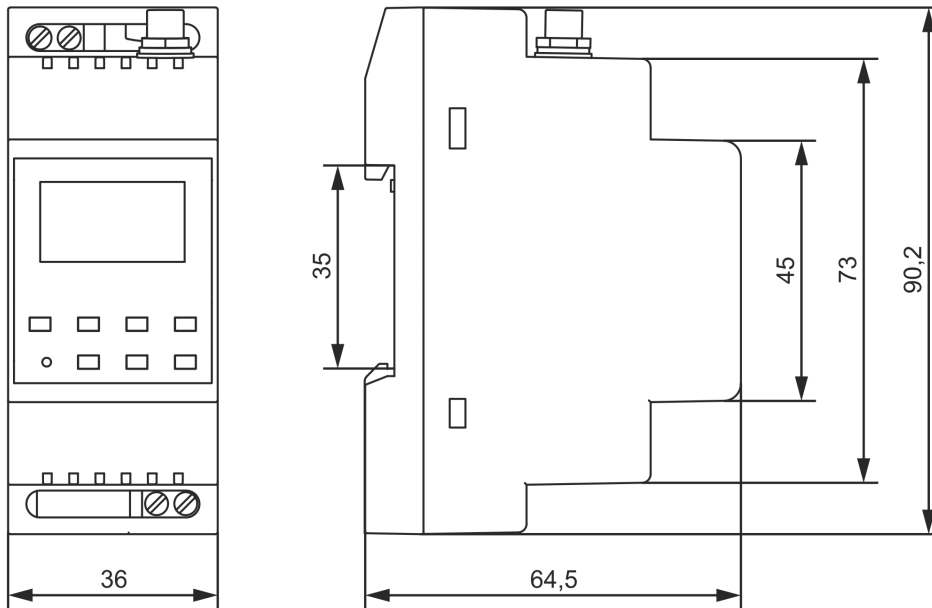


Fig.1 – Overall and mounting dimensions of MC252

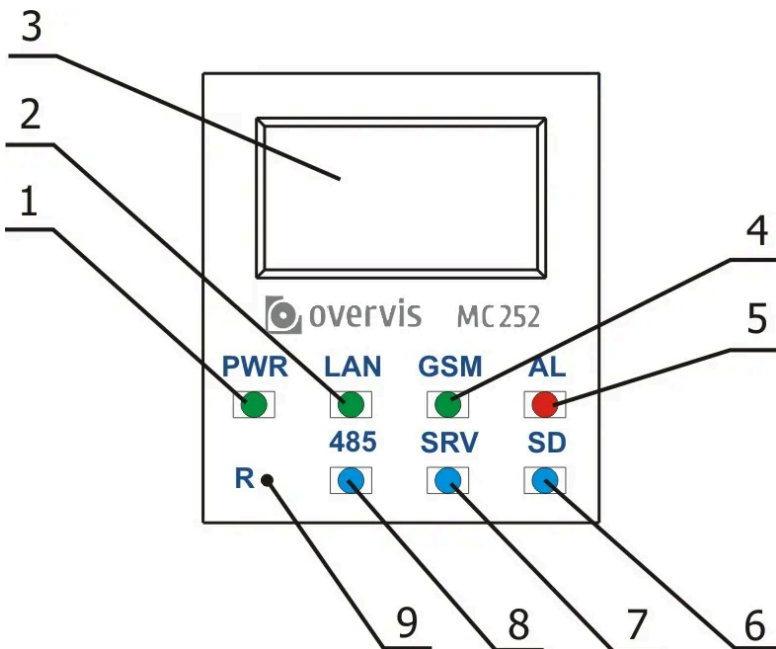


Fig.2 – Controls of the Overvis MC252

1. PWR LED is on when there is supply voltage;
2. LAN LED is on when connected to Ethernet network, and is blinking during the data exchange on the network;
3. The display shows the state of the device, active connections, device addresses, and the load of communication interfaces;
4. GSM LED flashes every 1.5 s when GSM (2G) communication is available; flashes 3 times per second when LTE (4G) communication is available, or when GPRS or FDD-LTE data is being exchanged;
5. AL LED warns about the error as a result of analysis of the received data;

6. SD is on when there is a memory card in its slot, and is blinking during the data exchange with the memory card;
7. SRV is on when the connection is present with the cloud server (or VPN), and is blinking during the data exchange via this connection;
8. 485 is on when waiting for data from a device on RS-485 bus, and is blinking during the data exchange on this bus;
9. R service button (accessible through a hole in the front panel, press with a thin non-conducting object) grants quick access to MC252 or resets the controller.

Operating conditions

The Overvis MC252 is designed for operation under the following environmental conditions:

- Ambient temperature: -35 ... +55 °C
- Atmospheric pressure: 84 ... 106.7 kPa
- Relative humidity (at +25 °C): 30 ... 80%

Do not operate MC252:

- In conditions of significant vibration or shock
- At high humidity levels (condensation)
- In aggressive environments (air containing acids, alkalis, etc.) or in severe contamination (grease, oil, dust, etc.)

Delivery set

Table 1 – MC252 delivery set

Item	Quantity
MC252	1
Cable for Ethernet connection	1
GSM antenna	1
Memory card (microSD 2 GB)	1
Operation Manual	1
Packing	1

Technical specifications

Table 2 – MC252 technical specifications

Parameter	Value
DC rated supply voltage	12 V
Data exchange interfaces via wired network	10Base-T/100Base-T Ethernet, RS-485

Parameter	Value
Supported Ethernet protocols	UDP, ARP, TCP
Data exchange interface via wireless network	GSM (900/1800), LTE (B1/B3/B5/B7/B8/B20)
Supported standards of wireless network	SMS, GPRS, FDD-LTE Cat.1
Integrated TCP/UDP clients	Modbus TCP, HTTP, WIREGUARD, NTP, DNS
Integrated TCP servers	Modbus TCP, HTTP
Maximum number of incoming TCP connections	4
Supported Modbus protocols via RS-485	Modbus RTU, Modbus ASCII
Transmission speed via RS-485	75 ... 230400 bps
Maximal output voltage of driver RS-485	3.3 V
Short circuit output current of driver RS-485 (maximum)	250 mA
Resistance of built-in terminator	1000 Ohm
The recommended number of connected devices in RS-485 network:	
– when the input current of receivers on RS-485 bus is less than 0.125 mA;	≤ 256
– when the input current of receivers on RS-485 bus is less than 1 mA	≤ 32
Readiness time when power is applied	≤ 15 s*
The supply voltage at which the operability is maintained	9 ... 30 V
Power consumption (under load)	≤ 6 W
Device service	Switchgear and controlgear
Rated operating condition	Continuous
Protection class rating	IP20
Electric shock protection class	III
Climatic design version	NF 3.1
Overvoltage category	II
Rated voltage of insulation	450 V

Parameter	Value
Rated impulse withstand voltage	2.5 kV
Conductor cross-section for connecting to terminals	0.5 ... 3 mm ²
Tightening torque of the terminal screws	0.4 N·m
Weight	≤ 0.400 kg
Overall dimensions (Fig. 1), H×W×D	64.5 × 90.2 × 36 mm
Installation (mounting) of the device	On standard 35 mm DIN-rail
Device orientation	Operable in any position in space
Housing material	Self-extinguishing plastic

* Establishing connections in Ethernet / GSM / Internet networks can take more time.

Device architecture

MC252 provides control over Modbus devices in an RS-485 network via Ethernet interfaces, GSM / LTE, or SMS. It also supports reading Modbus data.

The processor supports connection to a cloud server:

- via Ethernet network using an integrated Ethernet interface chip;
- via GPRS / FDD-LTE using a built-in modem (used when Ethernet connection is not available).

In addition, MC252 can be connected via the Modbus TCP protocol to exchange data with Modbus devices or with another MC252 controller.

MC252 receives and processes SMS messages that contain a password and a read/write command for Modbus devices.

When you insert a memory card, MC252 loads the operation logic program into internal memory. This program defines data collection and event tracking. It runs in the background.

You can write collected data to the memory card in tabular or binary format. For registered events, you can configure actions such as sending SMS messages or writing corrected Modbus values.

MC252 stores network settings, security parameters, and operation logic in its built-in memory.

Installation and wiring

Before you start:

- Unpack MC252 (we recommend keeping the original packing for the entire warranty period).
- Check MC252 for damage after transportation. If you find any damage, contact the supplier or manufacturer.
- Read this Operating Manual carefully (pay special attention to the power supply in the connection diagram).
- If you have any questions regarding installation, contact the manufacturer.

Wiring requirements

If the temperature of MC252 after transportation or storage differs from the ambient operating temperature, keep MC252 under operating conditions for at least two hours before connecting it to the power supply. This prevents condensation on internal components.

Installation safety

PERFORM ALL CONNECTIONS WHEN MC252 IS DE-ENERGIZED.

Installation errors can damage MC252 and connected equipment.

To ensure reliable electrical connections, use flexible (stranded) wires. Strip the insulation from the wire ends by 5 ± 0.5 mm and crimp with suitable ferrules. It is recommended to use wire with a cross-section of at least 1 mm².

- When connecting to the RS-485 bus, use twisted-pair cable of category 1 or higher. A shielded cable is recommended; in that case, ground the shield according to "ANSI/TIA/EIA-485-A-1998".
- When connecting to Ethernet, use the supplied cable or a category 5e twisted-pair cable with an 8P8C (RJ-45) plug.

Route and fasten wires so as to avoid mechanical damage, twisting, or abrasion of the insulation.

Electrical safety

DO NOT LEAVE EXPOSED WIRE PORTIONS PROTRUDING BEYOND THE TERMINAL BLOCK.

For a reliable contact, tighten the terminal screws with the force indicated in Technical specifications.

Tightening torque

Too little tightening torque can cause heating at the junction point. Eventually the terminal block may melt and the wire may burn.

Too much tightening torque can damage terminal block screws or over-compress and damage the connected wire.

To improve safety and reliability, it is recommended to install the fuse F1 (or its equivalent) in the MC252 supply circuit, rated for a current of no more than 2.5 A.

Electrical connection

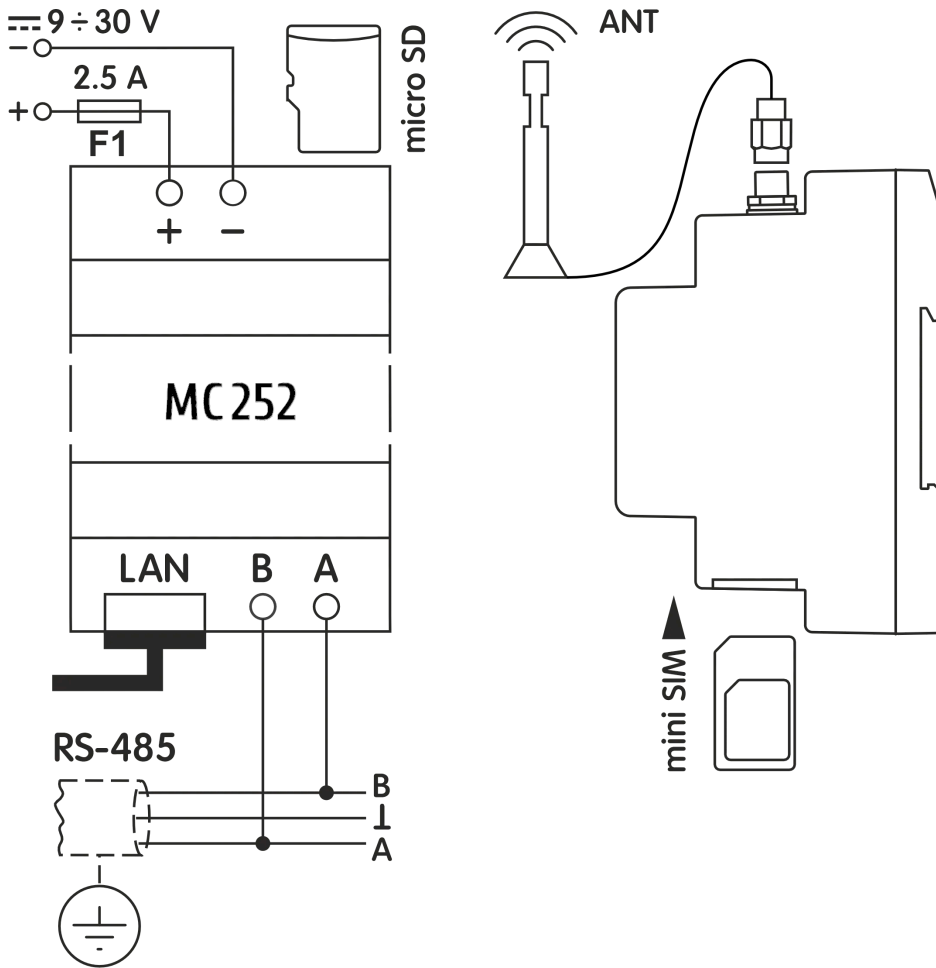


Fig.3 – Connection diagram of the device

- F1 – fuse (fuse element) rated for 2.5 A;
- Contact A – transmission of non-inverted signal;
- Contact B – transmission of inverted signal.

Follow these steps to connect the MC252:

- Connect the RS-485 bus cable to the RS-485 terminals (A and B) and to the RS-485 bus (or directly to the device with an RS-485 interface).
- If the MC252 should connect to the Internet via wired communication, to a local network, or directly to a computer, connect the Ethernet cable to the LAN connector and to the Ethernet network (or directly to a computer). Connection details, depending on the network type, are provided in Connections.
- Connect the appropriate DC power supply to the 9÷30V power connector.
- Insert the memory card (microSD) into the MicroSD slot if you plan to use data logging or operation logic programming features.
- If the MC252 is to connect to the Internet via wireless communication (or needs SMS exchange), insert the SIM card of the GSM operator into the SIM slot and connect the GSM antenna to the ANT connector (SMA F connector).

Power-up and normal operation

After power is turned on, all indicators except LAN and GSM light up, and MC252 performs initialization. After approximately 2 seconds, the indicators (except for the power indicator) go out, and the device proceeds to start the communication interfaces with networks. The display shows general information about the device (Fig. 4). Startup can take up to 15 seconds.

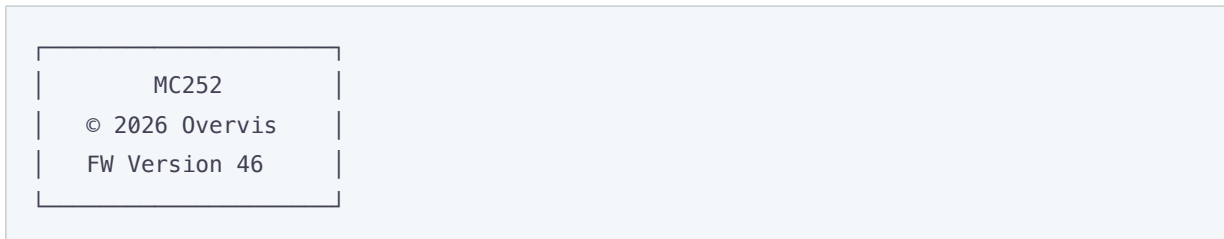


Fig.4 – Displaying of general information about the device on the display

After this, if enabled in settings, MC252 establishes a connection with the cloud server (or VPN) and starts executing the operation logic (if a program is loaded from the memory card). By default, cloud connections are disabled.

Malfunction indicator

IF THE AL INDICATOR IS CONSTANTLY LIT RED OR FAST BLINKING RED, PLEASE CONTACT THE MANUFACTURER OR THE PLACE OF PURCHASE OF THE DEVICE.

If configured, MC252 establishes TCP connections (via Ethernet and GSM/LTE networks) and waits for incoming TCP connections.

If a SIM card is installed, the GSM indicator shows the cellular connection status: blinking once every 1.5 seconds indicates successful network registration (but no data transfer yet); blinking 3 times per second indicates active TCP/IP data transfer via GPRS or FDD-LTE.

The display shows the interfaces load, the GSM signal strength and the IP addresses used, as shown in Figure 5.

Note

The display may reduce brightness over time and eventually enter energy-saving mode. In that case, wake it up with a short press of the R service button (see below).

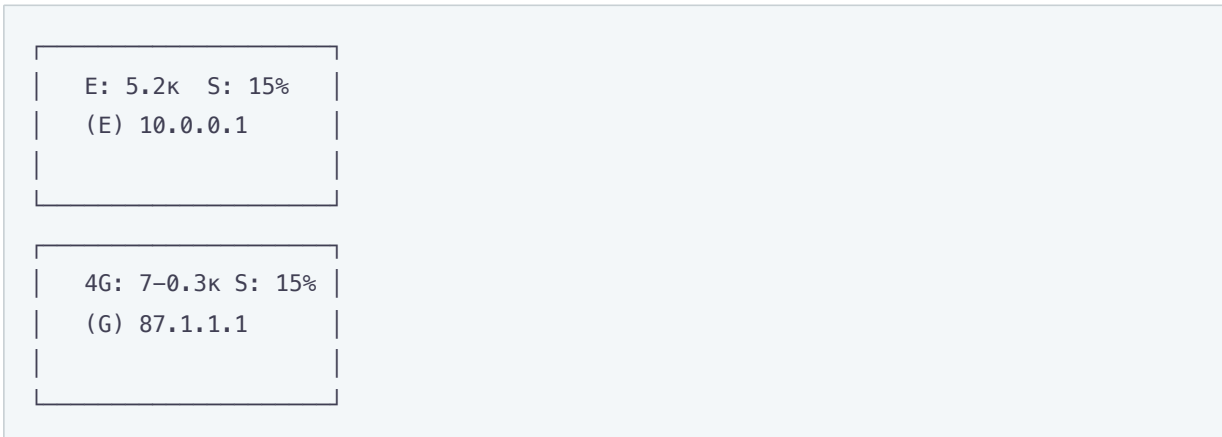


Fig.5 – Displaying the status of connections on the display

- E: 5.2k – the transmission speed via Ethernet is 5.2 kB/s;
- 4G: 7-0.3k – the LTE signal level is 70 % and the FDD-LTE transmission rate is 0.3 kB/s;
- S: 15% – the load of RS-485 is 15 %.
- (E): 10.0.0.1 – connection to the local network with the address 10.0.0.1;
- (G): 87.1.1.1 – wireless Internet access with the address 87.1.1.1.

Communication modes

The Overvis MC252 is a protocol converter that bridges RS-485 Modbus RTU/ASCII networks with Modbus TCP networks. It operates in multiple communication modes simultaneously, providing flexible connectivity through Ethernet, GSM/LTE networks, and RS-485 interfaces. Each mode serves a distinct purpose and can be used independently or in combination to meet your application requirements.

RS-485 Modbus RTU/ASCII master mode

In master mode, MC252 forwards Modbus requests from TCP clients to devices on the RS-485 bus and returns their responses.

- MC252 receives Modbus requests from TCP connections (incoming or outgoing)
- Translates Modbus TCP requests to Modbus RTU/ASCII format
- Forwards requests to target devices on the RS-485 network
- Returns responses back to the TCP client
- Supports both Modbus RTU and Modbus ASCII protocols

When an operation logic program is loaded, MC252 can also autonomously poll devices at configured intervals for data logging or event tracking purposes.

Request processing is described in detail in Modbus interface.

Use case: Data collection and event tracking for serial port equipment.

Integrated Modbus TCP server (slave mode)

MC252 acts as a Modbus TCP server, accepting incoming connections and processing Modbus requests.

- Listens for incoming TCP connections from SCADA systems, HMI panels, or other Modbus TCP clients
- Receives Modbus TCP requests from connected clients
- Responds with its own register values or translates requests to Modbus RTU/ASCII and forwards them to RS-485 devices (in master mode)
- Returns responses back to the TCP client
- Supports up to 4 simultaneous incoming TCP connections
- MC252's own registers can also be accessed directly (current time, supply voltage, logic calculation results, etc.)

Use case: Access to the RS-485 serial port equipment for the TCP devices, HMI panels, SCADA and other Modbus TCP software.

Connection to remote Modbus TCP servers (master mode)

MC252 can establish outgoing connections to remote Modbus TCP servers, enabling it to forward requests to remote devices.

- Initiates and maintains TCP connections to specified remote Modbus TCP servers via Ethernet, GSM/LTE or VPN
- Forwards Modbus requests received from other sources (RS-485 master, incoming TCP connections, or operation logic) to these remote servers
- Receives responses from remote servers and returns them to the requester
- Can connect to multiple remote servers simultaneously (up to 3 connections can be configured)
- Enables bridging between local RS-485 network and remote TCP-based Modbus devices

Use case: Data collection and event tracking for Modbus TCP equipment.

Reverse control translator mode (RS-485 slave, TCP master)

MC252 can operate as an RS-485 slave while acting as a Modbus TCP master, enabling reverse control scenarios.

- Receives Modbus RTU/ASCII requests from an RS-485 master device
- Translates these requests to Modbus TCP format
- Forwards them to remote TCP servers or local TCP-based devices
- Receives responses from TCP devices
- Returns responses back to the RS-485 master in RTU/ASCII format

Use case: Access to the Modbus TCP equipment for the RS-485 serial port master device.

Tunnel mode (transparent data forwarding)

In tunnel mode, MC252 accepts data "as is" (without protocol verification) and forwards it to all other directions that are configured for this mode.

This allows transmission of data in formats different from the Modbus protocol. For example, arbitrary data received via RS-485 can be redirected to a remote TCP server, and vice versa.

Tunnel mode can be configured individually for:

- Each connection to a remote TCP server
- The RS-485 interface

- Incoming connections to the Ethernet TCP port
- Incoming connections to the GSM/LTE TCP port

First, a data packet from one direction is fully received (for Ethernet or GSM/LTE, this is one TCP packet; for RS-485, the packet length is determined by the Modbus RTU maximum-pause rules). Then it is sequentially forwarded to the other tunnel directions (if there are more than two directions).

The maximum data packet length in tunnel mode is 254 bytes.

Use case: Connecting equipment with non-Modbus compatible protocols, or extending serial communication over IP networks.

Connection to cloud server (reverse connection)

MC252 can establish and maintain a persistent outbound connection to a cloud server (using Modbus TCP protocol or via VPN).

- MC252 initiates the outbound connection to the server, bypassing firewall issues
- After connection is established, MC252 sends its MAC address for identification as a first Modbus TCP packet (response for function 3)
- After that it operates in slave mode, waiting for and processing server requests
- The server sends Modbus requests through this connection
- MC252 forwards these requests to RS-485 devices or responds with its own register values
- Returns responses back to the server
- If the SRV LED is on, the connection to the server or VPN has been successfully established
- If the SRV LED blinks, data is being exchanged via this connection

Use case: Centralized monitoring and control of distributed equipment through cloud platforms like Overvis Cloud, enabling remote access from anywhere without configuring firewall port forwarding or static IP addresses.

Network extension (long-range RS-485 bridging)

Pair two MC252 units to extend RS-485 networks beyond physical distance limitations by converting to/from TCP.

- One MC252 operates in master mode on its RS-485 interface, while the other is in slave mode
- First MC252 receives RS-485 signals and converts them to TCP packets
- Data is transmitted over any distance via 4G/LTE or Ethernet networks
- Second MC252 receives TCP packets and converts them back to RS-485 signals
- Creates a transparent bridge between two RS-485 networks or segments
- Works in both Modbus protocol mode (with address translation) and tunnel mode (fully transparent)
- Each MC252 can operate in master or slave mode on its RS-485 interface as needed

Use case: Connecting RS-485 networks in separate buildings without running long cable runs, extending RS-485 beyond the 1200m distance limit, or accessing remote sites via cellular networks.

Network consolidation (address space remapping)

Combine multiple separate Modbus networks into one unified network by remapping device address spaces.

- Connect multiple isolated RS-485 networks, each with its own set of device addresses and/or baudrate and parity settings
- Configure address mapping to avoid conflicts (e.g., map first network's UIDs 1-10 to 1-10, second network's UIDs 1-10 to 11-20)
- Access all devices from a single TCP or RS-485 interface as if they were on one network
- Can combine local RS-485 devices with remote TCP devices seamlessly

Use case: Integrating multiple legacy systems with overlapping device addresses into a single SCADA system without physically re-addressing devices.

Access to Modbus network using SMS

If an active SIM card is installed, MC252 receives and processes incoming SMS messages.

- MC252 receives SMS messages sent to the SIM card's phone number
- If the SMS contains a correctly formatted Modbus request, MC252 processes it:
 - Forwards the request to the target device on RS-485 or TCP
 - Receives the response
 - Sends a reply SMS with the response data
- If the SMS is not a Modbus request, it is stored in the incoming SMS list for processing by operation logic programs

SMS Modbus request format is described in Modbus interface. Custom SMS processing is described in Operations logic programming.

Use case: Remote diagnostics and emergency control of equipment in locations without reliable Internet, such as pump stations or remote substations.

Data collection and event tracking

When an operation logic program is loaded into internal memory, MC252 reads specified parameters at a configured rate. These parameters may include registers of connected Modbus devices, MC252's own registers, or MC252 memory.

The device then performs configured calculations and evaluates the received data. As a result, the following actions can be executed:

- writing the read values to a log on the memory card;
- sending SMS notifications on events;
- writing new values to parameters.

The program is loaded into internal memory from the memory card. The procedure for preparing and loading the program into the device is described in Operations logic programming.

Use case: Autonomous data logging to memory card for later analysis, or automated responses like sending alarm notifications or regulation commands when temperature exceeds thresholds.

Configuration

MC252 configuration can be performed in two ways:

- via the web interface using a browser (see Web interfaces);

- via the Modbus protocol using any Modbus client software that works with MC252's own registers (see Modbus interface).

Configuration warning

When changing MC252 parameters, you may set values that interfere with or block subsequent connections. In this case, reset the parameters to factory defaults.

Use the R button to grant quick access to MC252, restart the controller, or reset factory settings. Access the button through a hole on the front panel and press with a thin non-conductive object.

To reset the device to factory settings and clear the internal operation logic memory:

- Press and hold the service button R for at least 8 seconds. After 2 seconds, the AL indicator will light up. After 8 seconds, the settings are reset, internal memory is cleared, and the device restarts; the indicators blink once.
- Release the R button.

To safely remove the memory card and restart the device (while preserving user settings):

- Press and hold the service button R for 2 to 8 seconds.
- When the AL indicator lights up, release the R button. If needed, remove the memory card after the SD indicator turns off.

To wake up the display, show connection information, and grant quick access to the device:

- Briefly press and release the service button R.
- The display lights up and shows connection information. For several minutes, configuration access (without a password) is granted for incoming connections.

Security recommendation

For security reasons, protect the R button from unauthorized access (for example, by using a tape seal or installing MC252 in a locked enclosure).

Maintenance

MAINTENANCE SAFETY

TERMINALS AND INTERNAL ELEMENTS MAY CARRY POTENTIALLY LETHAL VOLTAGE. DURING MAINTENANCE, DISCONNECT MC252 AND ALL CONNECTED EQUIPMENT FROM THE POWER SUPPLY.

MAINTENANCE SAFETY

DO NOT OPEN THE DEVICE. THERE ARE NO SERVICEABLE PARTS INSIDE.

- Only qualified personnel should perform maintenance.
- Recommended maintenance interval is every six months.

1. Check the reliability of wire connections; if necessary, retighten terminals with the torque specified in Technical Specifications.
2. Visually inspect the housing. If you detect cracks or other damage, take MC252 out of service and send it for repair.
3. If necessary, wipe the front panel and housing with a soft cloth.

 **Caution**

Do not use abrasives or solvents for cleaning.

Service life and warranty

1. The service life of MC252 is 10 years. After the service life expires, contact the manufacturer.
2. Shelf life is 3 years.
3. The warranty period is 5 years from the date of sale. During the warranty period, in the event of failure, the manufacturer provides free repair.

 **Warranty conditions**

IF YOU OPERATE MC252 IN VIOLATION OF THE REQUIREMENTS OF THIS OPERATING MANUAL, YOU LOSE THE RIGHT TO WARRANTY SERVICE.

4. The place of purchase or the manufacturer performs warranty service.
5. The manufacturer performs post-warranty service at current rates.
6. Before sending MC252 for repair, pack it in the original or other packaging that protects it from mechanical damage.

When returning MC252 for warranty or post-warranty service, please provide a detailed reason for the return in the claims data field.

Transportation and storage

You may transport and store MC252 in the original package at temperatures from minus 45 to +60 °C and relative humidity of no more than 80%. When transporting MC252, protect it against mechanical damage.

See also

- Versions and modifications
- Connections
- Operations logic programming
- Memory card data saving
- Web interfaces
- Modbus interface
- Firmware update

Connections and Network Setup

Important

Connecting an incorrectly configured device to a data transmission network can disrupt communication between other devices. Ensure all connected devices have compatible settings. Network connections involving more than 2 devices should be performed by qualified network personnel.

Quick Start

For a streamlined connection guide, see the Quick Start Guide.

IP Addressing

The Overvis MC252 uses standard TCP/IPv4 addressing for Ethernet communication. DHCP is enabled by default, allowing the device to automatically obtain network settings from your router. Without DHCP the factory default IP address is 192.168.0.111.

> IP Addressing Fundamentals

Table 1 - Factory addressing settings for MC252

Parameter	Value
Addressing using DHCP	Yes
IP Address	192.168.0.111
Subnet Mask	255.255.255.0
Gateway	192.168.0.1

Ethernet Connection Methods

With factory default settings, MC252 supports two connection methods via Ethernet.

Method 1: Network with DHCP Server

If your network includes a router or DHCP server that assigns IP addresses to new devices:

1. Connect MC252 to the network
2. Wait for the device to obtain an IP address (appears on the display with (E) prefix for Ethernet)
3. If the display shows 0.0.0.0, the address has not been received yet
4. If 192.168.0.111 appears after 20-60 seconds, DHCP acquisition failed and the device switched to its static address

Method 2: Direct Connection or Network Without DHCP

If DHCP is unavailable, or MC252 is connected directly to a computer:

1. MC252 will switch to static addressing after 20-60 seconds
2. Configure the client device with:
 - Subnet mask: 255.255.255.0
 - IP address: 192.168.0.X (where X = 1-254, excluding 111)
 - Ensure the chosen address doesn't conflict with other devices on the subnet

If your network uses different addressing (different mask or IP range from Table 1), or if 192.168.0.111 is already taken:

1. Temporarily disconnect the conflicting device from the network
2. Establish direct communication between your client device and MC252
3. Configure both devices for proper network communication
4. Reconnect to the network

Configuring Ethernet Connection on Windows PC

In some cases you may need to connect MC252 directly to your computer to access its web interface and change settings. Connect the device with an Ethernet cable to the LAN port on your PC, then configure your network adapter as described below.

The following example shows how to configure a Windows PC (Windows 7/8/10/11) to communicate directly with MC252 using factory settings. For other operating systems, configure your client device's addressing according to its documentation.

› [How to access network adapter settings on Windows 7 / 8 / 10](#)

› [How to access network adapter settings on Windows 11](#)

Configuring the Network Adapter

1. In the Connections window, select the desired network adapter. If multiple connections appear, choose the correct one by adapter name or consult your system administrator.
2. Right-click the connection icon and select **Properties**. The Properties window opens (Fig. 1).

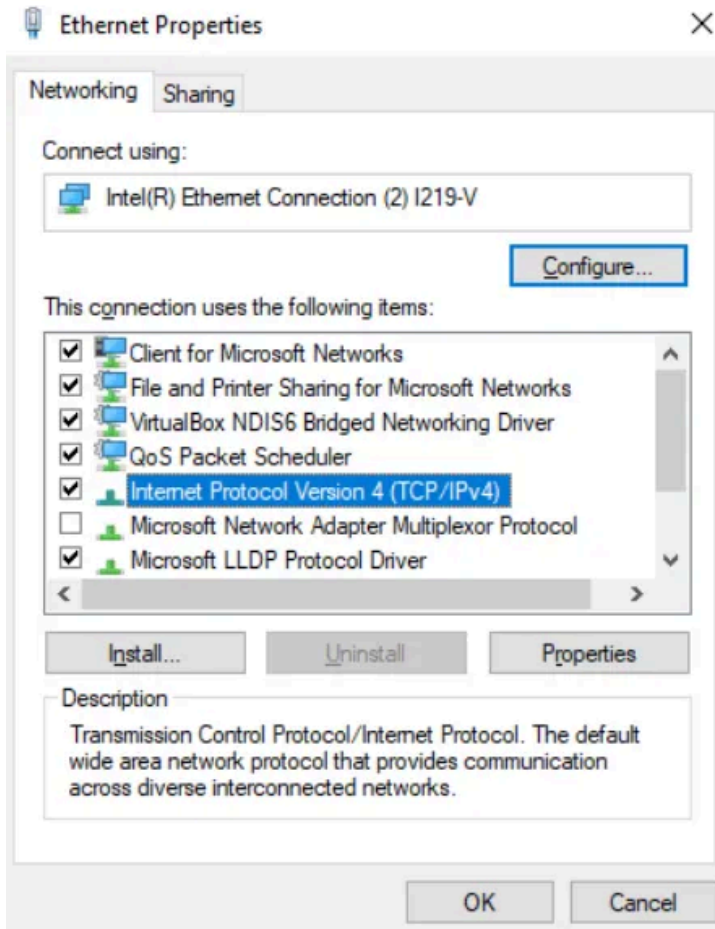


Fig. 1 – Connection properties window in Windows

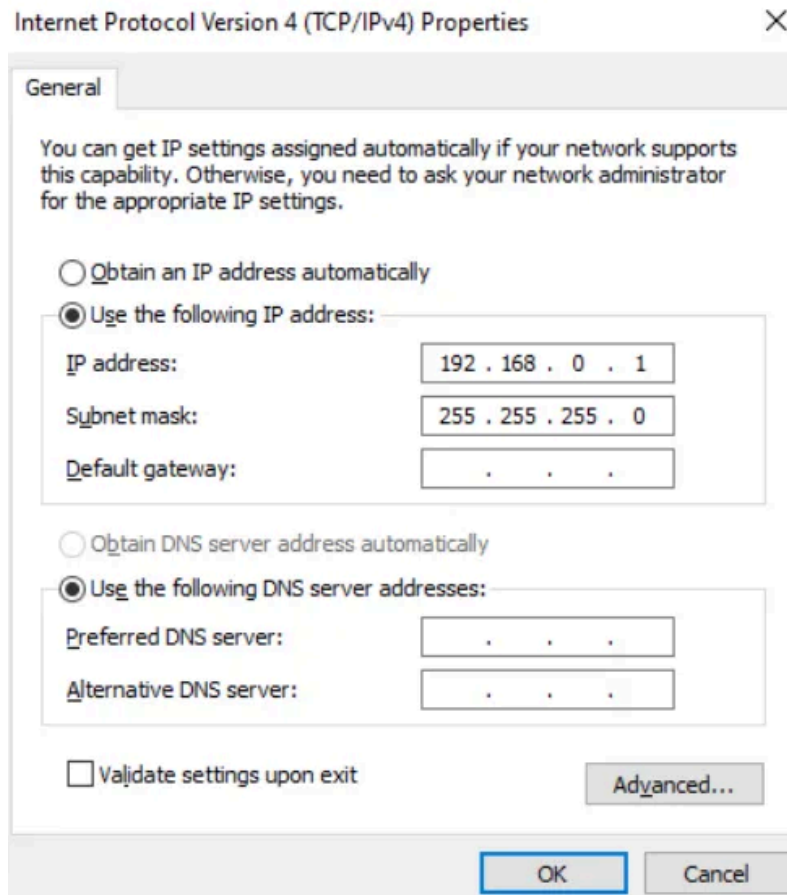


Fig. 2 – TCP/IPv4 properties window in Windows

3. Select Internet Protocol Version 4 (TCP/IPv4) from the component list. Ensure it's enabled (checkbox is checked). Click Properties. The TCP/IPv4 properties window opens (Fig. 2).
4. Select Use the following IP address
5. In the IP address field, enter an address in the range 192.168.0.1 to 192.168.0.254 (excluding 192.168.0.111, which MC252 uses)
6. In the Subnet mask field, enter 255.255.255.0
7. Leave Default gateway, Preferred DNS server, and Alternate DNS server fields blank or unchanged
8. Click OK to close the Protocol Settings window
9. Click OK to close the Connection Settings window
10. If prompted to restart the PC, select Yes

Internet Connection via Ethernet

 **STRONGLY RECOMMENDED**

Connect the device to the Internet under the supervision of your LAN system administrator and/or Internet service provider representative.

Basic Internet Setup

To connect MC252 to the Internet via Ethernet:

1. Connect MC252 to a local network router with DHCP addressing enabled
2. Ensure the router is connected to your Internet service provider (ISP)
3. **Do not** connect MC252 directly to the ISP's cable

This setup enables outgoing connections (e.g., active-mode connection to cloud servers, connections to other servers using hostnames or static IP addresses).

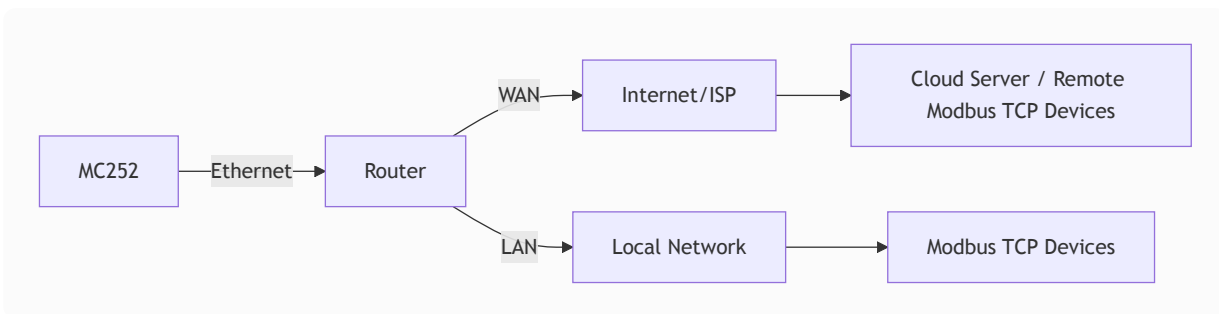


Fig. 3 – MC252 Internet connection topology via Ethernet with DHCP

Incoming Connections Setup

To access the device from the Internet via incoming connections (direct TCP connection or web interface):

1. **Acquire a Static IP:** Get a dedicated line with a static IP address from your ISP
 - Connect the ISP cable to the router's uplink port (usually color-coded, see router documentation)
 - Use a straight-through Ethernet cable to connect MC252 to the router
2. **Configure Router:** Following your ISP's recommendations, configure the router for Internet access using the router documentation.
3. **Set Up Port Forwarding:** Configure the router to redirect queries from your static public IP address to MC252's local IP address (factory default: 192.168.0.111)
 - For web interface access: redirect to port **80** on MC252's local address
 - For Modbus TCP or tunnel access: redirect to the incoming Ethernet connections port (register 450, factory default: **502**). See Modbus Interface for register configuration details.

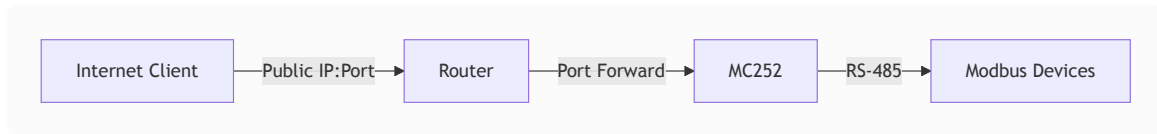


Fig. 4 – MC252 incoming connections via port forwarding

4. **Configure DHCP Reservation:** Either:

- Configure the router to always assign the same IP address to MC252 via DHCP, OR
- Disable DHCP in MC252 settings and use a static IP

This ensures port forwarding works correctly.

5. **Verify Security:** Confirm that MC252's Internet connection is protected by standard security measures (see Connection Security). Specifically, set the write access password to some unique values. It is also recommended to set the read access password.

6. **Access the Device:** When accessing MC252 from the Internet, use the static IP address provided by your ISP and the public port specified in your port forwarding configuration

GSM Connection

STRONGLY RECOMMENDED

If you are using corporate IoT APN, connect the device to the Internet under the supervision of your mobile service operator representative.

Ensure your tariff plan includes GPRS or LTE service (for Internet connectivity) and/or SMS messaging (for SMS-based control).

Basic GSM Setup

To connect MC252 via GSM:

1. **Obtain SIM Card:** Get a SIM card from your GSM operator
2. **Install Hardware:** Insert the SIM card into MC252 and connect the appropriate antenna to ensure proper signal strength at the MC252 location
3. **Verify SIM Recognition:** After starting MC252, confirm the SIM card is correctly identified
 - After communication initialization, the GSM indicator should blink continuously
 - If the indicator remains off for more than 4 seconds, check the SIM card, antenna, and GSM signal level on the MC252 display
4. **Verify Internet Connection** (if using GPRS/LTE): Ensure the operator settings are correct
 - When connected to the Internet, the GSM indicator flashes 3 times per second
 - If flashing less frequently, check the APN settings
5. **Configure APN:** If necessary, manually configure the APN according to your operator's recommendations (see APN Configuration Examples below)

This setup enables outgoing connections (connecting to cloud servers in active mode, or to other servers with hostnames or static IP addresses on the Internet).

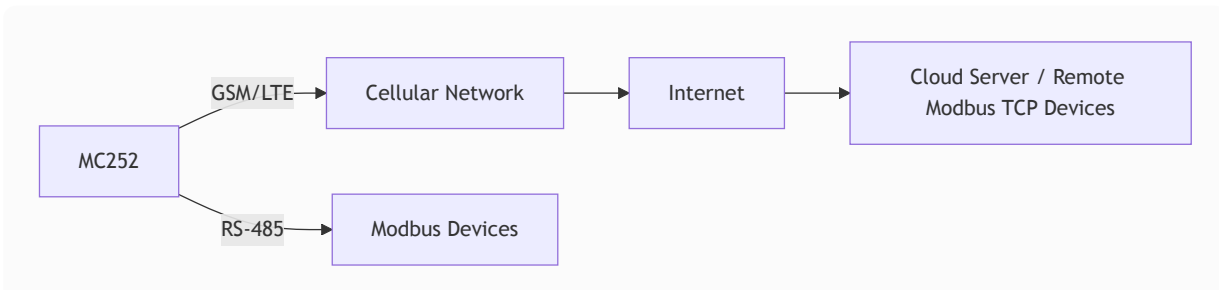


Fig. 5 – MC252 Internet connection topology via GSM/LTE

› **APN Configuration Examples**

Incoming Connections via GSM

1. **Obtain Static IP Service:** Get a static IP SIM card or service from your operator
2. **Install Hardware:** Insert the SIM card and connect the antenna to ensure proper signal strength at the MC252 location
3. **Configure Device:**
 - Set the GSM incoming connections port in settings. See Web interface description for details.
 - Configure APN settings according to your operator's recommendations
 - Restart MC252
4. **Verify Connection:** Confirm the SIM card is identified correctly
 - After interface initialization, the GSM indicator should continue to flash
 - If the indicator remains off for more than 4 seconds, check the SIM card, antenna, and GSM signal level on the device display

Note

Your operator may provide alternative access methods, such as private IP addresses instead of public IP addresses. Private IPs are used to interlink devices within the operator's network without providing full Internet access.

Connection Security

MC252 provides several security features to protect against unauthorized access:

Built-in Security Features

MC252 has basic protection against unauthorized access via network. Access for writing and/or reading via Modbus or SMS can be deactivated in settings.

Device settings can be changed remotely by entering a password (minimum 5 characters). Access passwords can be set for restriction of writing and/or reading via Modbus or SMS.

When entering the password, all settings are only available to the specific client using the specific protocol. In case of no requests from the client over a long period, the access returns to locked mode.

SECURITY WARNING

For Modbus connections, passwords are transmitted in unencrypted form. Even with correct password authentication, the HTTP and Modbus connections remain unsecured at all segments except VPN and GSM.

Security Limitations and Recommendations:

- **Attack Protection:** The device protection system is not designed to counter malicious network attacks, especially denial-of-service attacks aimed at blocking MC252 rather than gaining access
- **Network Segmentation:** For complex and large networks (especially when providing Internet access to MC252), it is strongly recommended to:
 - Separate MC252 from unprotected networks using standard security equipment
 - Use a properly configured router with traffic filtering
 - Deploy a firewall or similar protective measures

Server Connection

MC252 supports constant communication with data collection and management servers, for example, Overvis Cloud.

About Overvis

Overvis (www.overvis.com) is a system for monitoring and remote control of technological processes that enables you to read data from and control devices including MC252, store data in a database, review the data in various formats, and receive alarm notifications via SMS or email.

MC252 factory settings are preconfigured for Overvis connection, but the connection is disabled by default and must be enabled manually. The Overvis system uses MAC address-based authentication — the device's unique MAC address is sent to the server at each connection session.

Connection Methods Overview

There are three ways to connect MC252 to Overvis Cloud:

1. **Using PIN/QR Code from Label** (Recommended for first-time setup) — The fastest method using reverse connection where MC252 connects to Overvis. Simply scan the QR code on your device label. Detailed in Quick Start Guide.
2. **Using Activation Code** (Requires physical access) — Uses reverse connection with an 8-digit code displayed on the MC252 screen and web interface. This method proves physical device access, making it suitable for taking over previously used devices, verifying ownership, or when the label is unavailable.
3. **Direct Connection from Overvis** (For advanced users) — Overvis connects directly to MC252 through router port forwarding, requiring a static public IP address. Used rarely due to complexity and security considerations.

Method 1: Using PIN/QR Code from Device Label

This is the recommended method for first-time setup, using the registration information from your MC252 label.

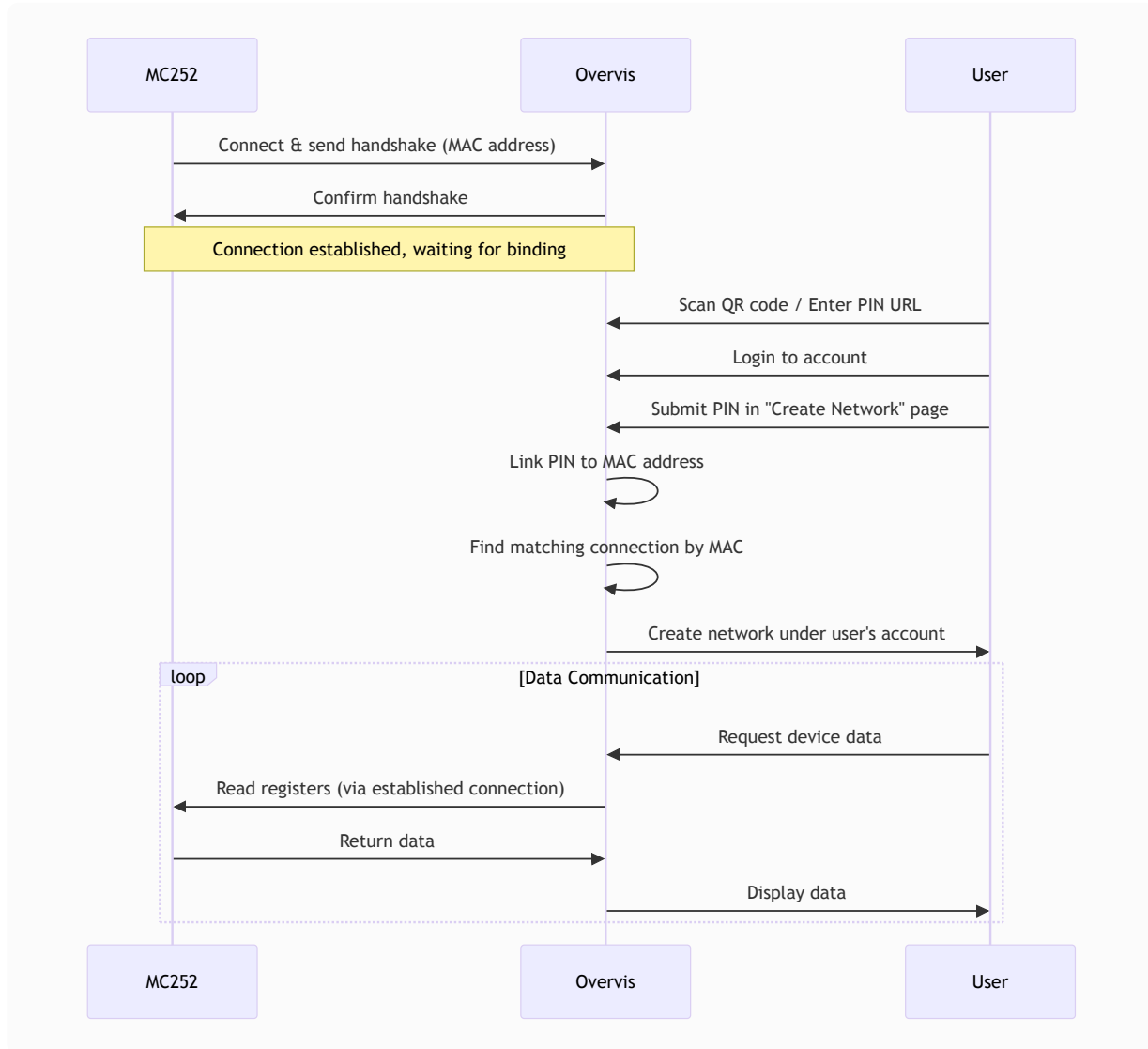


Fig. 6 – Connection sequence for PIN/QR code method (reverse connection)

Connection Flow: MC252 initiates outbound connection to Overvis and maintains it. User registers by linking the PIN to this existing connection. No port forwarding required.

Prerequisites:

- MC252 connected to Internet (via Ethernet or GSM)
- QR code label on device (or PIN code from label)
- Overvis Cloud account (or create during setup)

Connection Steps:

- 1 **Verify Internet Connection**

Ensure MC252 is connected to the Internet. Check the device display for a valid IP address—the prefix (E) indicates Ethernet connection, (G) indicates GSM. Address should not be 0.0.0.0.

2 Access Overvis Server

Scan the QR code on the device label with your phone or tablet, OR manually enter the URL from the label (format: <https://c.overvis.com/ABCD1234>). The link automatically redirects to Overvis server login page with the PIN embedded in the URL.

3 Login or Create Account

If you have an account, enter your credentials. New users should register for a free account first.

4 Create Network

After login, Overvis displays the “Create Network” page. If you came from the QR code/link, the PIN is pre-filled automatically. Otherwise, manually enter the PIN from the device label. Click “Check connection” to verify MC252 is online.

5 Configure Network

Give your network a descriptive name (a “network” represents your MC252 plus all connected Modbus devices). MC252 itself (Modbus unit ID 111) is added automatically. Add your RS-485 devices by selecting models and entering their Modbus addresses.

6 Verify Operation

Open a device page in Overvis and read parameters to confirm real-time communication.

After successful registration, the MC252 display will show either active or no code:

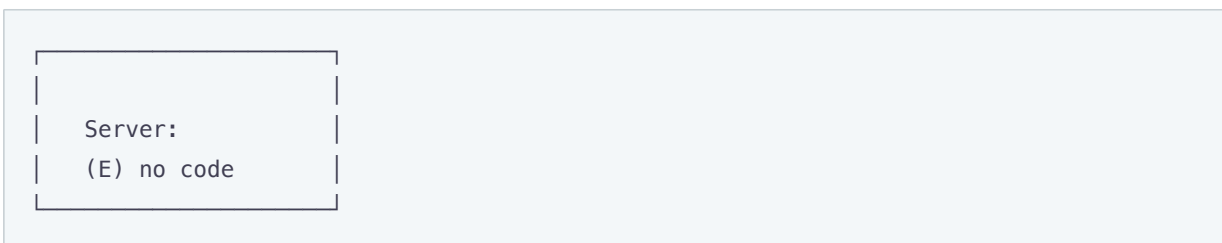


Fig. 7 – Status displayed after PIN/QR code activation

Note

no code indicates the device was registered using PIN/QR code method. This is normal and expected.

Method 2: Using Activation Code

This method requires viewing the MC252 device screen or web interface, proving you have physical access to the device. It's the primary method for taking over previously used devices and verifying ownership. The activation code also serves as a security measure—only someone with physical access can see and use it.

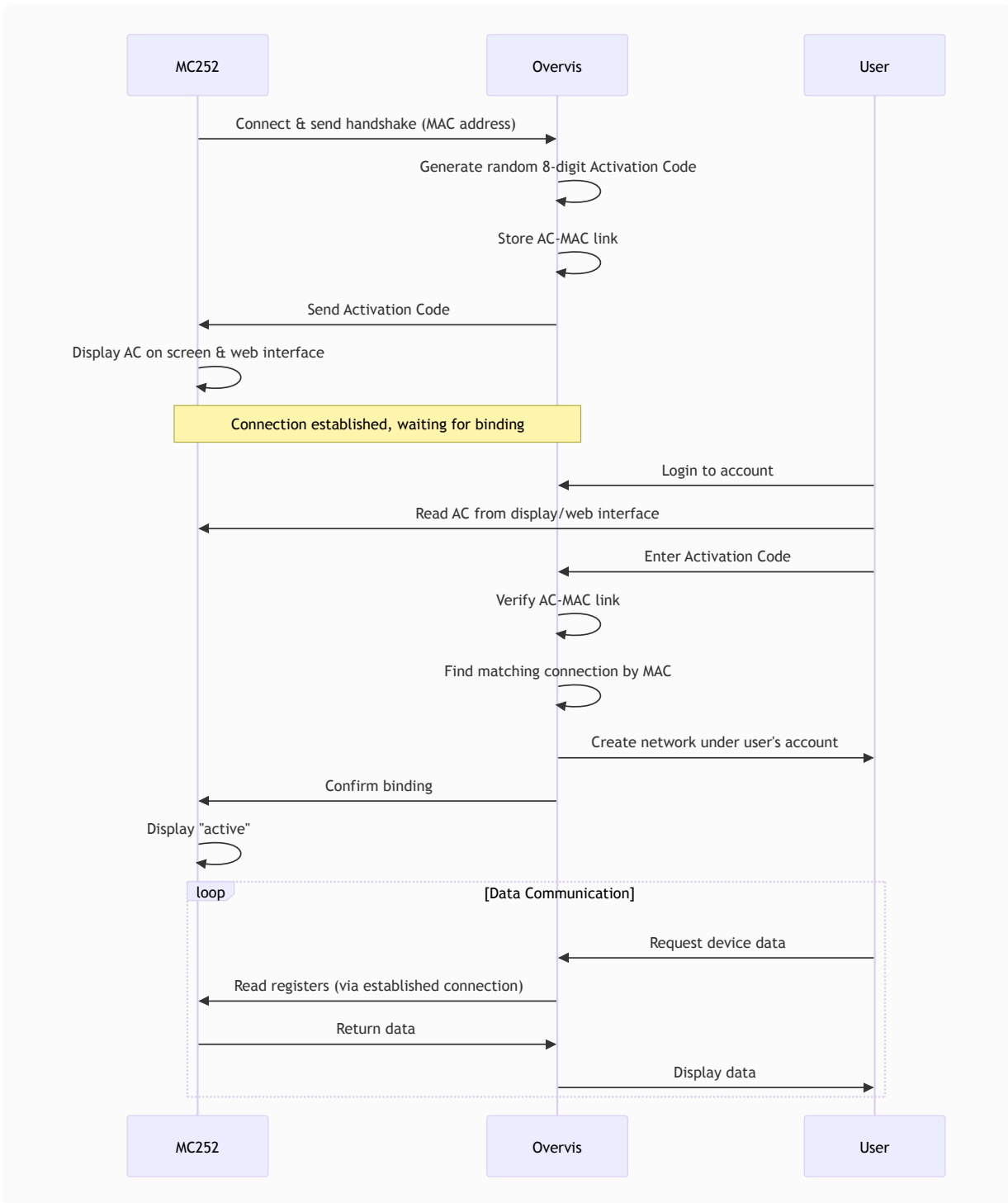


Fig. 8 – Connection sequence for Activation Code method (reverse connection)

Connection Flow: MC252 initiates outbound connection to Overvis and receives a unique 8-digit Activation Code. User enters this code to bind the device to their account. No port forwarding required.

Prerequisites:

- MC252 connected to Internet (via Ethernet or GSM)

- Overvis Cloud account

Connection Steps:

1 Connect to Internet

Connect MC252 to the Internet using one of the methods described in Internet Connection via Ethernet or GSM Connection.

2 Verify Connection and Get Code

Confirm that the server connection is established and obtain the 8-digit activation code by checking the device display OR the MC252 web interface status page (Cloud tab).

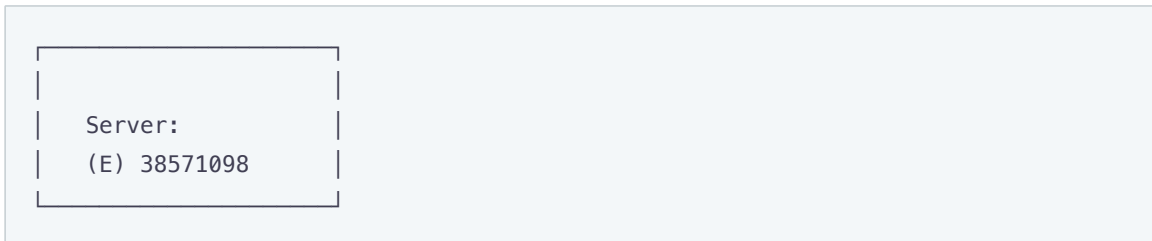


Fig. 9 – Activation code displayed on device screen

The prefix indicates connection method: (E) for Ethernet, (G) for GSM

 **Previously Used Devices**

If you acquired a used MC252 that was previously connected to another Overvis account, you must unbind it before registering it to your account. To unbind the device:

1. Access the MC252 web interface
2. Navigate to the **Cloud** settings page
3. Click the **Restart Activation** button

This removes the device from the previous owner’s account and generates a new activation code for your use. The device will display the new 8-digit code after unbinding.

3 Create network in Overvis and provide Activation Code

Log in to your Overvis Cloud account at www.overvis.com. On the “Create Network” page, enter the 8-digit activation code to bind the device to your account.

4 Configure Network

Give your network a descriptive name (a “network” represents your MC252 plus all connected Modbus devices). MC252 itself (Modbus unit ID 111) is added automatically. Add your RS-485 devices by selecting models and entering their Modbus addresses.

5 Verify Activation

After successful activation, the MC252 display will show active:

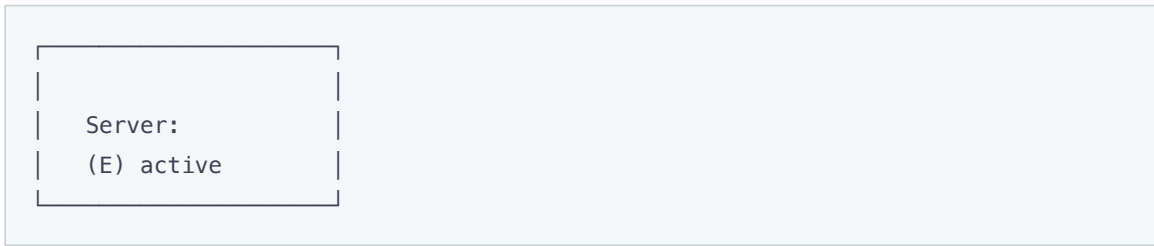


Fig. 10 – Activation status displayed on device screen

Method 3: Direct Connection from Overvis

This method allows Overvis to connect directly to your MC252 using its public IP address. It requires more advanced network configuration.

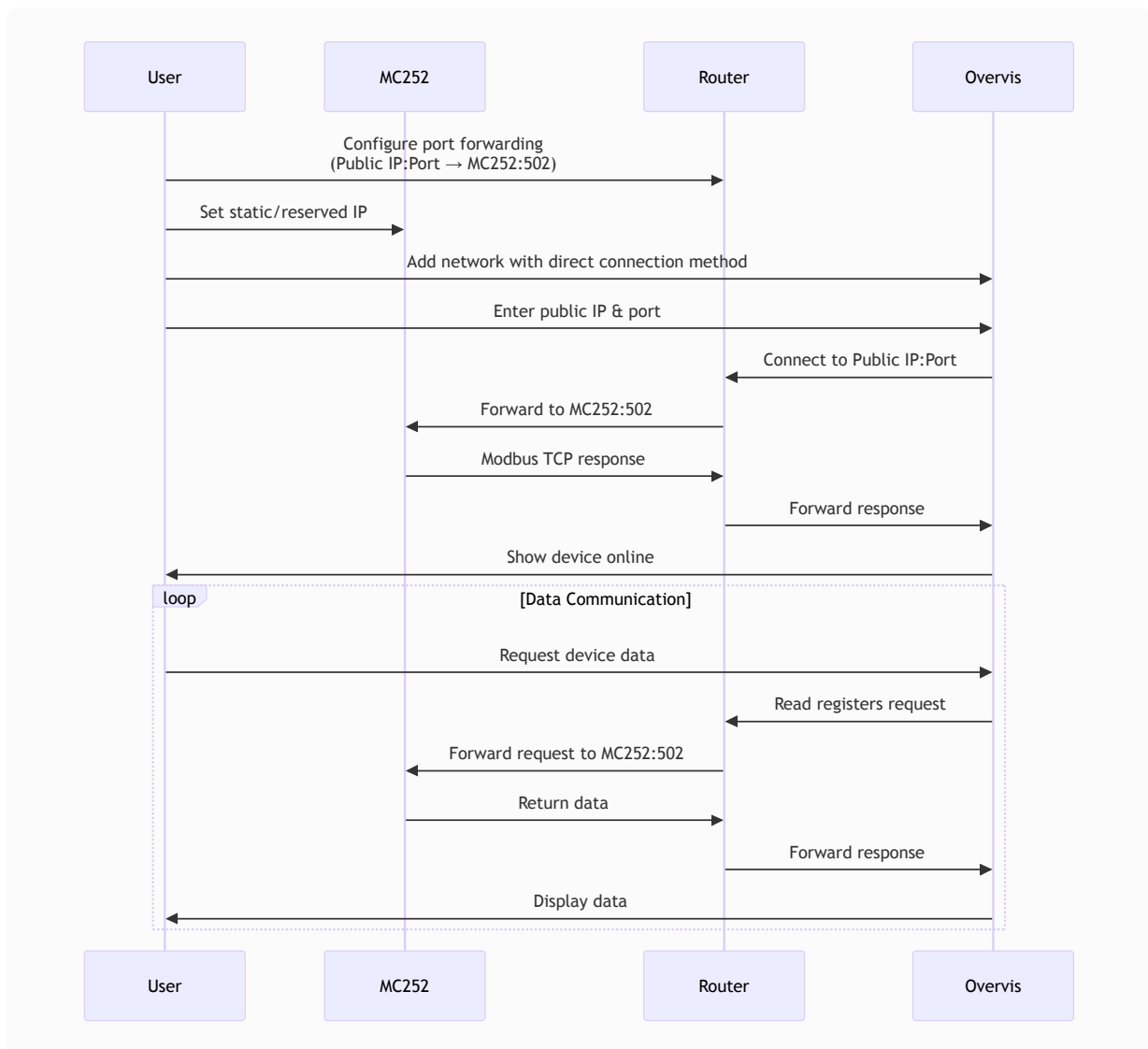


Fig. 11 – Connection sequence for direct connection method (incoming connection)

Connection Flow: Overvis initiates inbound connection to MC252 through router port forwarding. Requires static public IP.

Prerequisites:

- Static public IP address from your ISP
- Router with port forwarding capability
- MC252 connected to your LAN
- Overvis Cloud account
- Basic understanding of router configuration and network security

 **Advanced Configuration**

This method requires configuring your router's port forwarding and exposing MC252 to the Internet. Ensure you follow security best practices and consult your network administrator if operating on a corporate network.

Connection Steps:

1 Obtain Static Public IP

Contact your Internet Service Provider to obtain a static public IP address for your location. Dynamic IP addresses are not recommended as they may change. Some ISPs offer static IP as a paid add-on service.

2 Configure Router Port Forwarding

Access your router's configuration interface and set up port forwarding to redirect external connections to MC252. Forward to MC252's **Modbus TCP port** (default: **502**). External port can be the same (502) or different for security. Forward to MC252's internal IP address (e.g., 192.168.0.111).

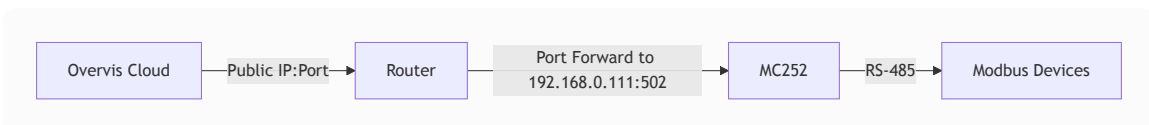


Fig. 12 – Direct connection port forwarding configuration

3 Configure DHCP Reservation or Static IP

Ensure MC252 always has the same internal IP address:

- **Option A:** Configure your router to assign MC252 a reserved IP via DHCP (recommended)
- **Option B:** Disable DHCP on MC252 and manually configure a static IP address in the web interface

4 Verify MC252 Accessibility

Test the connection from outside your network. Use a Modbus TCP client from a different network (e.g., mobile hotspot), connect to your public IP address on the forwarded port, and try reading a register from MC252 (unit ID 111, register 1 for firmware version).

5 **Configure Security** (Strongly Recommended)

Set strong Modbus TCP passwords in MC252 web interface. Consider changing the external port to a non-standard number. Enable Modbus TCP access logging in MC252 settings. Configure your router's firewall to restrict access by IP range if possible. See Connection Security for additional recommendations.

6 **Create Network in Overvis**

Log in to your Overvis Cloud account at www.overvis.com. On the "Create Network" page, select the direct connection method. Enter your static public IP address and the external port number (as configured in router). Enter MC252's Modbus address (default: 111) and Modbus password if configured.

7 **Configure Network**

Give your network a descriptive name (a "network" represents your MC252 plus all connected Modbus devices). MC252 itself (Modbus unit ID 111) is added automatically. Add your RS-485 devices by selecting models and entering their Modbus addresses.

8 **Verify Operation**

Open a device page in Overvis and read parameters to confirm real-time communication. Monitor connection stability over 24-48 hours.

Connecting to Other Servers

The only method of connection MC252 to other cloud servers and SCADA systems is the direct connection method via Modbus TCP.

Configure MC252 in the monitoring system as a remote Modbus TCP device. This requires setting up port forwarding on your router to redirect external connections to MC252's local IP address and Modbus TCP port (default: 502). Your server will initiate connections to MC252 through your router, similar to Method 3 described above. You'll need a static public IP address from your ISP, or alternatively, you can use a secure WireGuard VPN tunnel to bypass the need for public IP exposure.

Since MC252 uses the standard Modbus TCP protocol for direct connections, no special server software is required—any Modbus TCP client can communicate with the device. See the Modbus Interface documentation for complete register mappings and communication protocols.

Troubleshooting

Problem: MC252 display shows 0.0.0.0

The DHCP server is not responding. Wait up to 60 seconds for MC252 to switch to its static IP 192.168.0.111. If the issue persists, verify the network cable is properly connected and check your router's DHCP settings.

Problem: Cannot access MC252 web interface

Verify your client device is on the same subnet as MC252. Check firewall settings on your client device and confirm the MC252 IP address shown on the display. You can press the R button briefly to grant temporary password-free access.

Problem: LAN LED is off

Check the Ethernet cable connection and verify it's not damaged. Try using a different cable to rule out cable faults.

Problem: GSM LED stays off

The SIM card may not be inserted correctly — remove and reinsert it. Check that the SIM card PIN is disabled (disable it using a phone before inserting). Verify the antenna is connected properly. Poor signal strength may require relocating MC252 or using an external antenna.

Problem: GSM LED blinks slowly (every 1.5s) but no Internet

This indicates incorrect APN settings. Verify the APN configuration with your operator. Check that your SIM card has an active data plan and review the APN configuration in MC252 settings.

Problem: Cannot receive SMS commands

Verify the SIM card phone number is correct. Check the SMS format according to the Modbus Interface documentation. Ensure SMS service is enabled on the SIM card and verify any SMS passwords are configured correctly.

Problem: SRV LED stays off

MC252 is not connected to the Internet — check your Ethernet or GSM connection. Verify the server connection is enabled in the web interface. Check that the server address and port are correct in settings. Your network firewall may be blocking outgoing connections.

Problem: Activation code not displayed

The server connection has not been established yet. Verify Internet connectivity and wait up to 2 minutes after the Internet connection is established. Check the cloud server settings in the web interface.

Problem: "Device already registered" error

The device is bound to another Overvis account. Use the **Restart Activation** button in the Cloud settings page to unbind it, or contact the previous owner to remove the device from their account.

Problem: Intermittent connection drops

Check that the power supply voltage is within the required range (9-30V DC) and can provide sufficient current (up to 500mA). Look for electrical noise or interference sources near the device. Update MC252 firmware to the latest version.

Problem: RS-485 devices not responding

Verify the RS-485 wiring is correct (A and B terminals). Check that the RS-485 bus termination is properly configured. Verify Modbus RTU/ASCII settings match the connected devices. See the Modbus Interface documentation for detailed communication settings.

Further Reading

- **Web Interface** — Complete guide to MC252's web-based configuration interface

- **Modbus Interface** — Register mappings and communication protocols for Modbus TCP/RTU/ASCII

Need Help?

If you're experiencing issues not covered in this guide, we're here to help:

- **Technical Support:** Contact the manufacturer's support team
- **Documentation:** Visit overvis.com/support for additional resources
- **Community:** Check the user forums for solutions from other MC252 users

For warranty service or hardware issues, please contact your authorized distributor or the manufacturer directly.

User Web Interface

The Overvis MC252 features a built-in web interface for configuration, status monitoring, and management. You can access this interface using any standard web browser connected to the same network as the device.

Accessing the Interface

1. Ensure the MC252 is powered on and connected to your network via Ethernet (uses standard HTTP port 80).
2. Find the device's IP address. You can display the current IP address on the MC252 screen by shortly pressing the **R** (Service) button.
3. Enter this IP address into your web browser's address bar (e.g., `http://192.168.1.100`).

Note: If your network uses a proxy server, you may need to add the MC252's IP address to your browser's exception list to access it locally.

Authentication

Upon accessing the device, you will be presented with a login page.

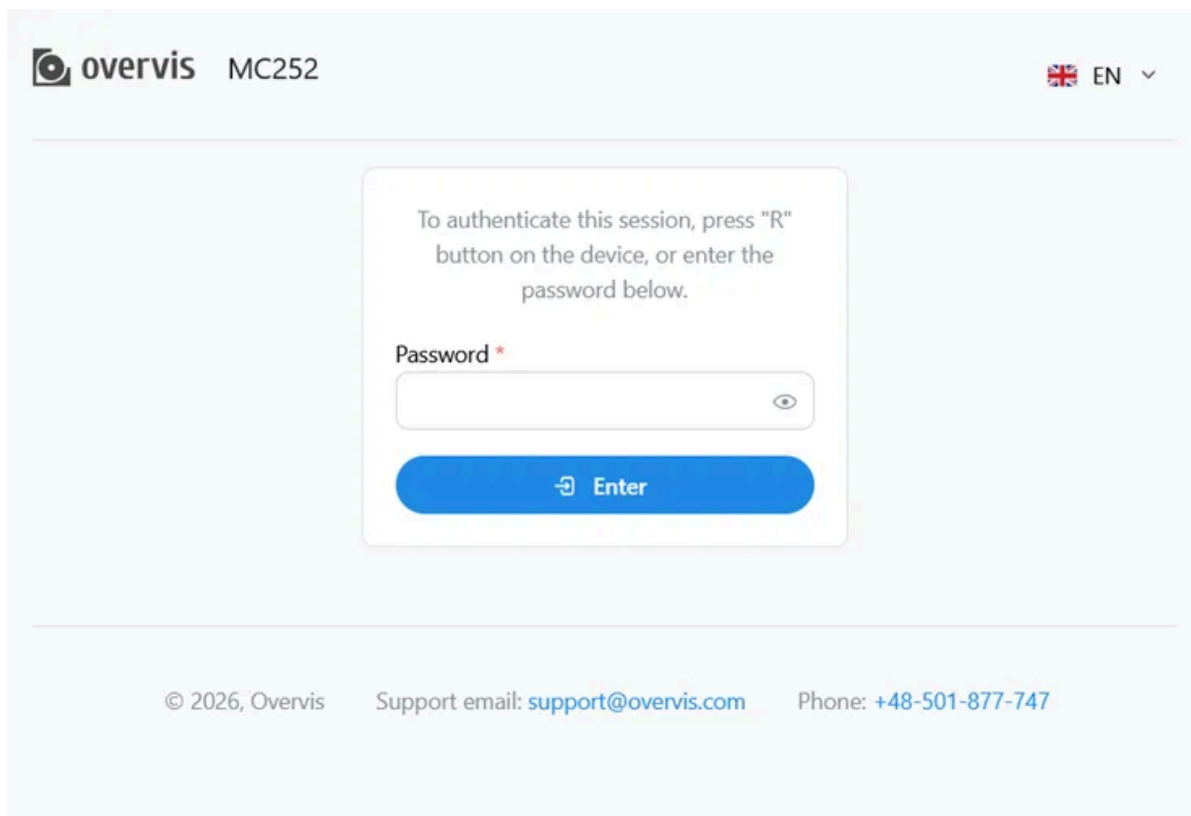


Fig. 1 - The login page

There are two ways to authorize access:

- **Password:** Enter the password printed on the device label.
- **Physical Access:** Shortly press the **R** (Service) button on the device to grant temporary access without a password (see Operating Manual).

Dashboard

After logging in, the Dashboard provides a real-time overview of the device status.

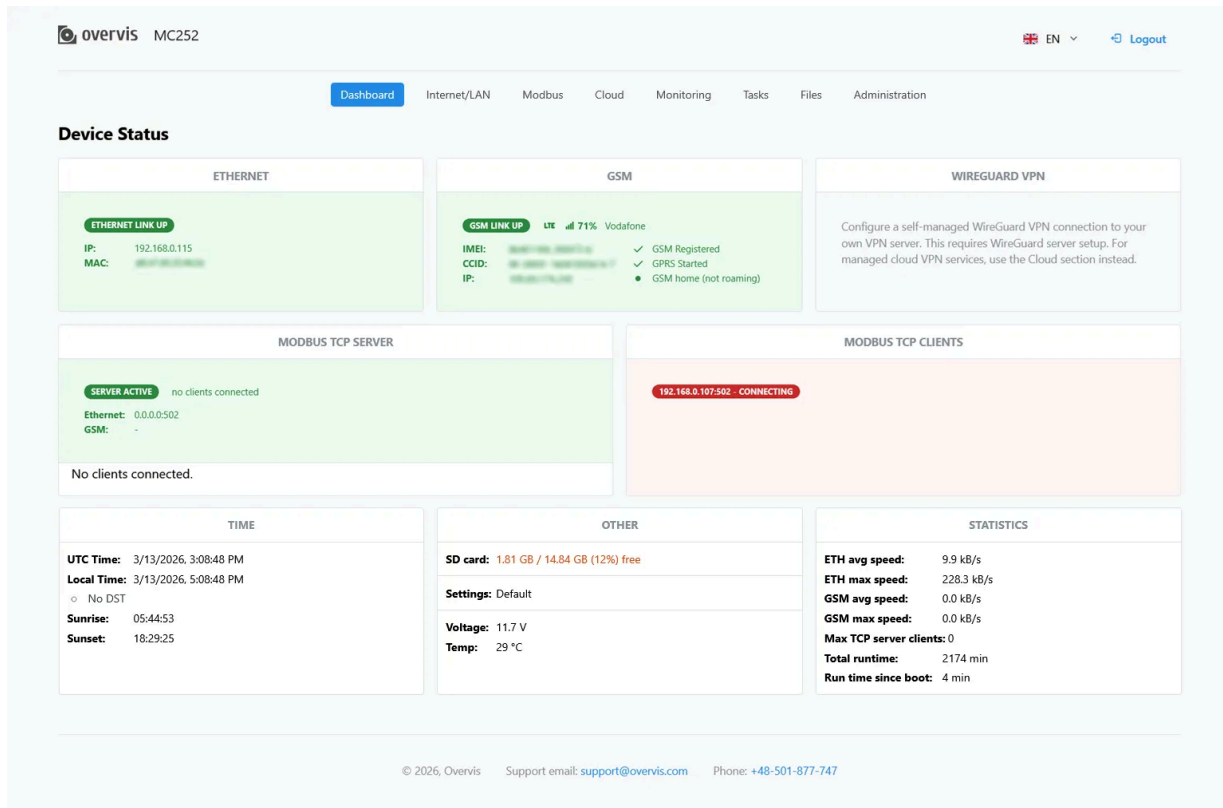


Fig. 2 - The dashboard page

The dashboard displays:

- TCP interface status
- Modbus connection states
- Memory card usage
- Data transfer statistics
- Current system time

Configuration

The web interface is divided into several tabs for different configuration aspects.

Tip: Most configuration pages include an “Open the documentation” link for detailed help on specific options. When you click **Save & reset**, settings are saved to memory and the device will restart to apply changes.

LAN/Internet

Configure network interfaces and connectivity settings.

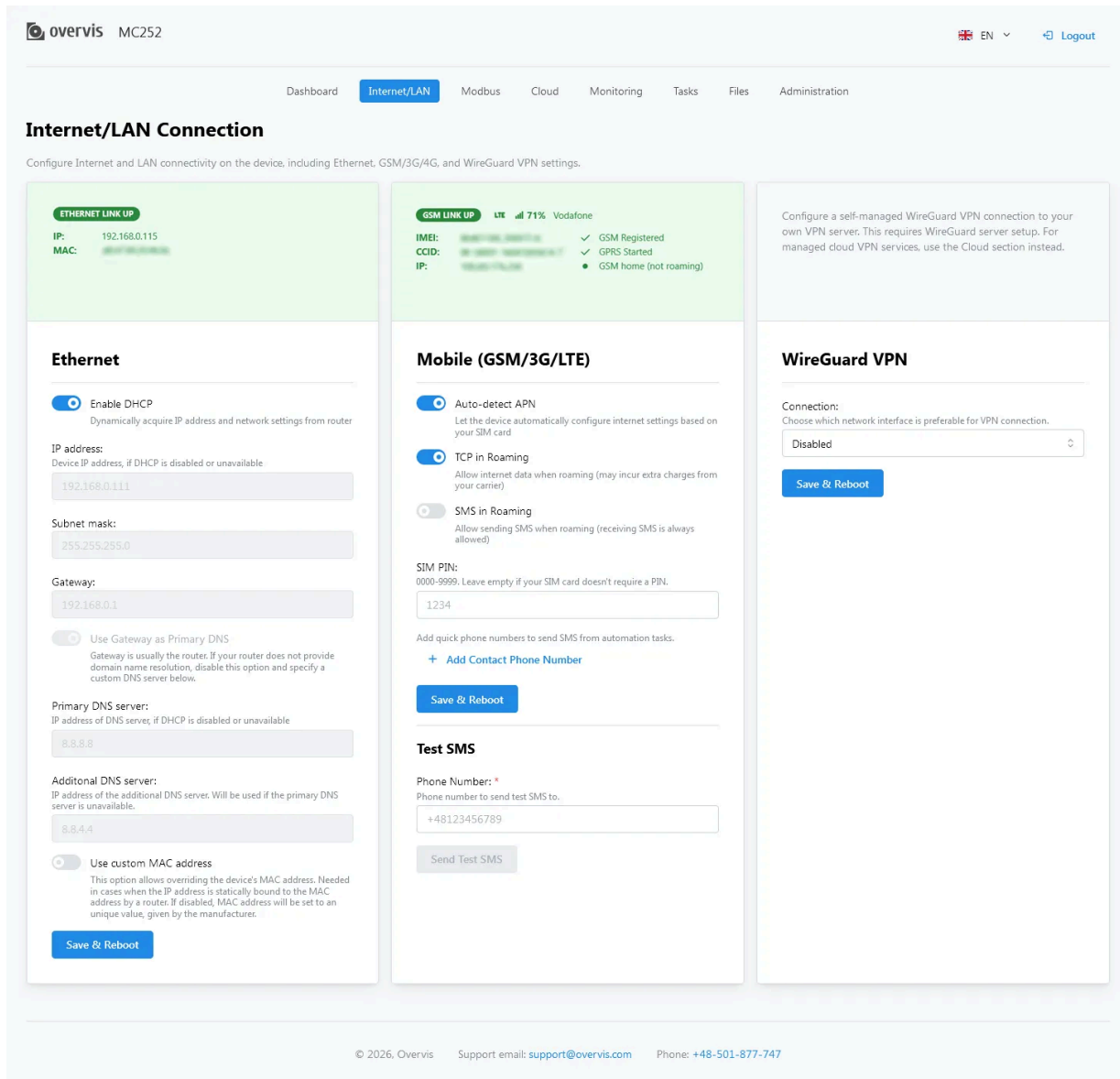


Fig. 3 - The LAN/Internet configuration page

- **Ethernet:** Configure IP addresses, subnets, and gateways.
- **GSM:** Configure APN settings, roaming options, PIN code, and quick phone numbers.
- **VPN:** Set up a WireGuard VPN client. VPN IP addresses can be used in place of Ethernet addresses for routing.

Modbus

Manage Modbus protocol settings for industrial communication.

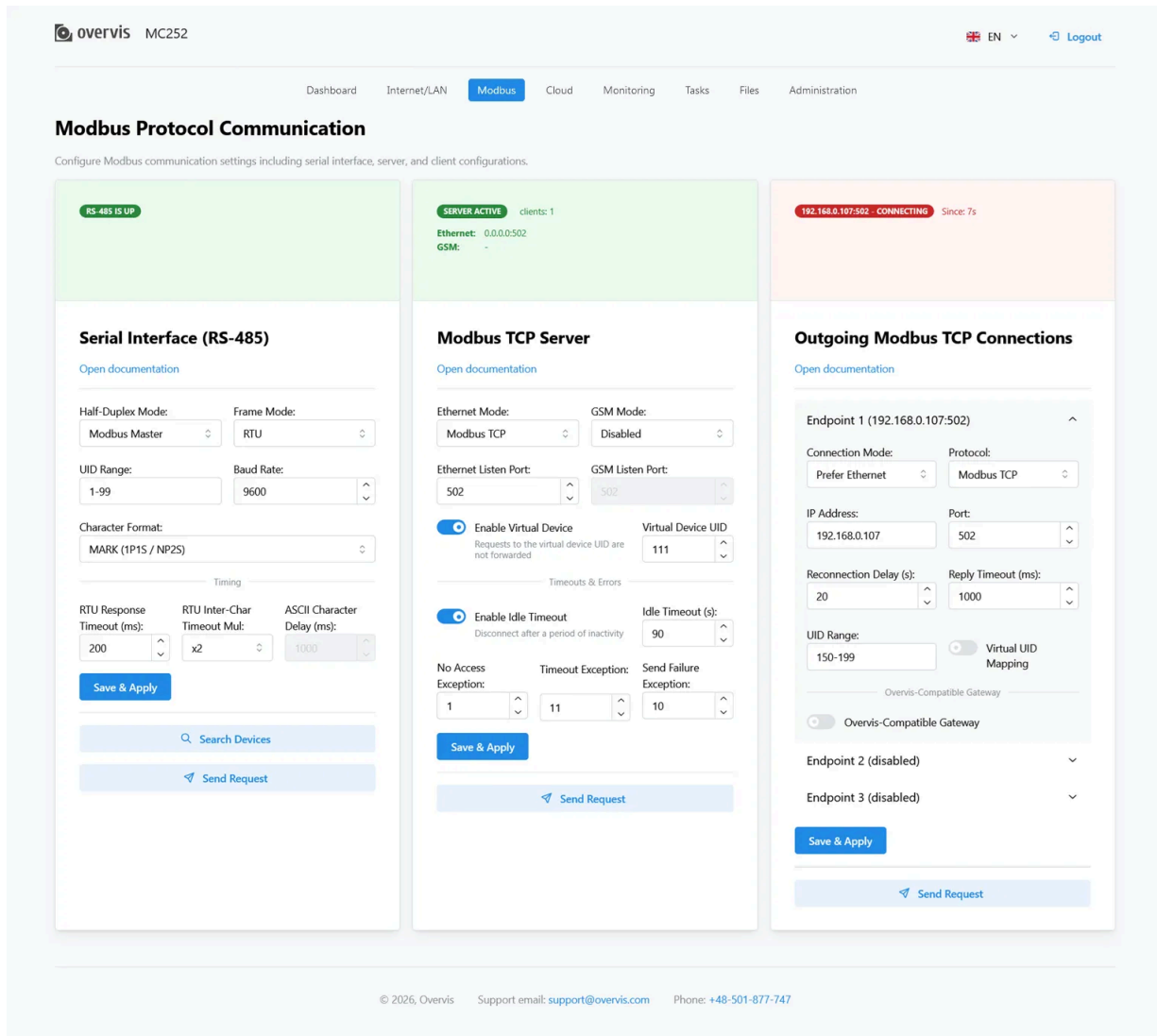


Fig. 4 - The Modbus configuration page

- **RS-485:** Configure serial interface parameters (mode, baud rate, parity, etc.).
- **Modbus TCP Server:** Allow incoming connections from Modbus clients (masters).
- **Modbus TCP Client:** Configure connections to remote Modbus servers (slaves).
- **Tunnel Mode:** Enable transparent data tunneling for specific directions to bypass protocol conversion, or for non-Modbus transfers.

Note: Modbus settings can often be applied immediately without a full device restart, though the connections will briefly reset.

RS-485 Modbus Devices Search

The **Search Devices** button allows to list the slave devices connected via serial RS-485 interface.

Search Modbus Devices
✕

Baud Rate

All speeds
⌵

Character Format

NO PARITY (NP1S)
⌵

Frame Mode

All modes
⌵

Fast search

Stop searching other settings once a device is found

Cancel

🔍 Start Search

Fig. 5 - The Search Devices tool options

The search covers all the Modbus unit addresses. The search can also cover RTU or ASCII framing and commonly used configurations (baudrate, parity and stopbits), or some of the parameters can be set fixed (to filter and speedup the search).

The search can be optionally simplified to stop matching the configuration once the first device answers on the RS-485 bus. If there is a set of devices with the same serial settings (or a single slave device on the bus), the **Fast search** option can be flagged. If the number and configuration of the devices are unknown, it is recommended to run a thorough search once (can take from half an hour to an hour).

Search Modbus Devices
✕

Reading registers 0..1: UID 216, 9600 bps, MARK (1P1S / NP2S), RTU 215 / 255

Found Devices: 1

UID	Baud Rate	Format	Mode	Data
1	9600	MARK (1P1S / NP2S)	RTU	Exception: 2

Stop Search

Fig. 6 - The devices search progress

After the search has been started, its progress and the current action are displayed, along with the list of the found devices. For each device, there are shown:

- Modbus unit address (UID);
- the MC252 configuration which has matched;
- the data or the exception code returned by the device.

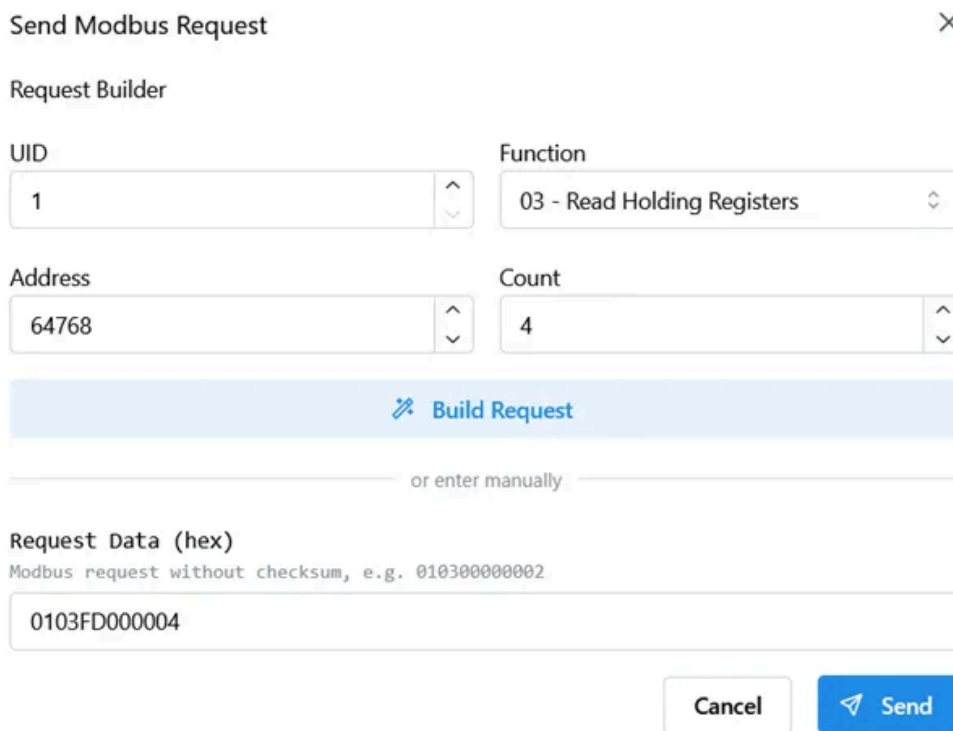
The exception code during the search does not mean a certain device problem (in most cases the device just didn't have the Modbus resources requested). The exception still means the device has received the MC252 request and has answered.

The search can be stopped by pressing the stop button or by closing the search tool.

The MC252 configuration is left unchanged by the search. If the needed device configuration differs, either it has to be configured to match the MC252, or the MC252 serial settings have to be manually matched with this device.

Modbus Communication Check

The **Send Request** button allows to check the MC252 configuration or to check the available Modbus devices (both at RS-485 and at remote TCP servers).



Send Modbus Request [X]

Request Builder

UID: 1 Function: 03 - Read Holding Registers

Address: 64768 Count: 4

[Build Request](#)

or enter manually

Request Data (hex)
Modbus request without checksum, e.g. 010300000002

0103FD000004

Cancel Send

Fig. 7 - The Send Request tool options

The MODBUS request (with device address byte, but without any other headers or checksums) can then be entered as a hexadecimal string of bytes.

The request can also be built by specifying a device Modbus address (UID), function to call and function parameters. These include address (e.g. holding register address) and count of items to read (or a single value to write).

The **Build request** button would convert the specified parts into a hexadecimal string below. This should be used before sending the request.

Send Modbus Request



Request Data: 0103FD000004

Transaction ID: 848

Status: **RESPONSE RECEIVED**

Registers: 5005, 0, 1, 2

Response Data: 010308138d000000010002

New Request

Close

Fig. 8 - The Modbus request results

After the request has been sent, the answer wait message and then the results are displayed. These could be one of the following:

- read data values;
- write success confirmation;
- an exception or an error.

Cloud

Setup the connection to the Overvis cloud platform.

The screenshot displays the 'Overvis Cloud Connection' configuration page. At the top, there's a navigation bar with 'Cloud' selected. The main heading is 'Overvis Cloud Connection' with a sub-heading 'Configure Overvis cloud server connection.' Below this, a message states 'Cloud Connection is Disabled' and encourages connecting to Overvis Cloud. A list of benefits is provided, including real-time monitoring, alerts, remote control, analytics, and secure connections. A 'Learn more about Overvis Cloud' button is present. On the right, a 'Cloud Settings' panel is open, showing 'Connection Mode' set to 'Disabled' and a 'Save & Reboot' button. The footer contains copyright information and contact details.

Fig. 9 - The cloud connection configuration page

- **Connection:** Select the connection method and server address.
- **Activation:** Link the device to your cloud account.

Monitoring

Setup the parameters logging task.

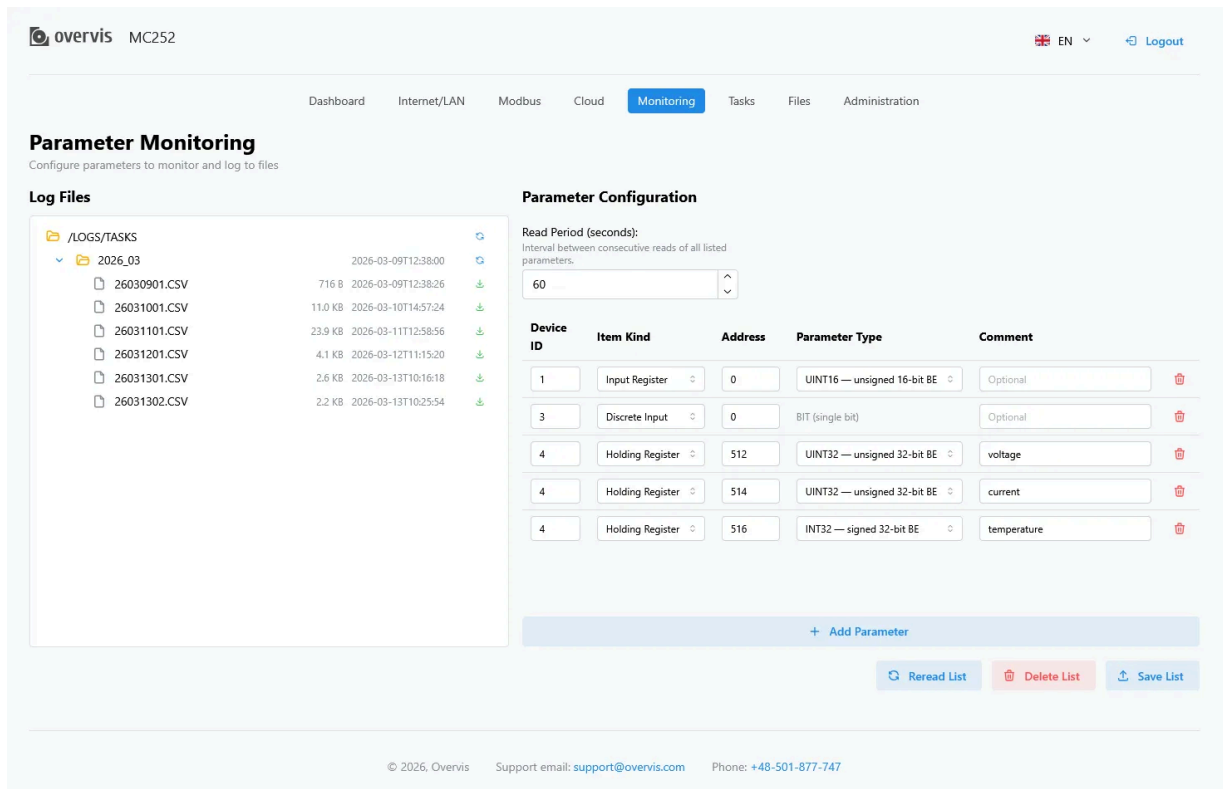


Fig. 10 - The monitoring setup page

The factory default configuration includes no monitoring task.

If the memory card is present, the monitored parameters can be added to the list on the right. Each parameter has an address on a Modbus device and the value type (the way it would be converted from the device registers). The optional comment can be added.

The **Save list** button allows to generate the automation task /TASKS/PARAMLOG.TXT and then reload the tasks.

The saved parameters log files would be displayed on the left of the page. A folder would be created for each month. Selecting the folder allows to expand it and show the files. The needed file can then be downloaded by pressing the corresponding button to the right of it.

The **Delete list** button allows to remove the automation task and then restart the tasks for the changes to take effect.

The **Reread list** button allows to cancel any unsaved changes and read the list from the automation task /TASKS/PARAMLOG.TXT.

Tasks

Manage programmed logic and automation tasks.

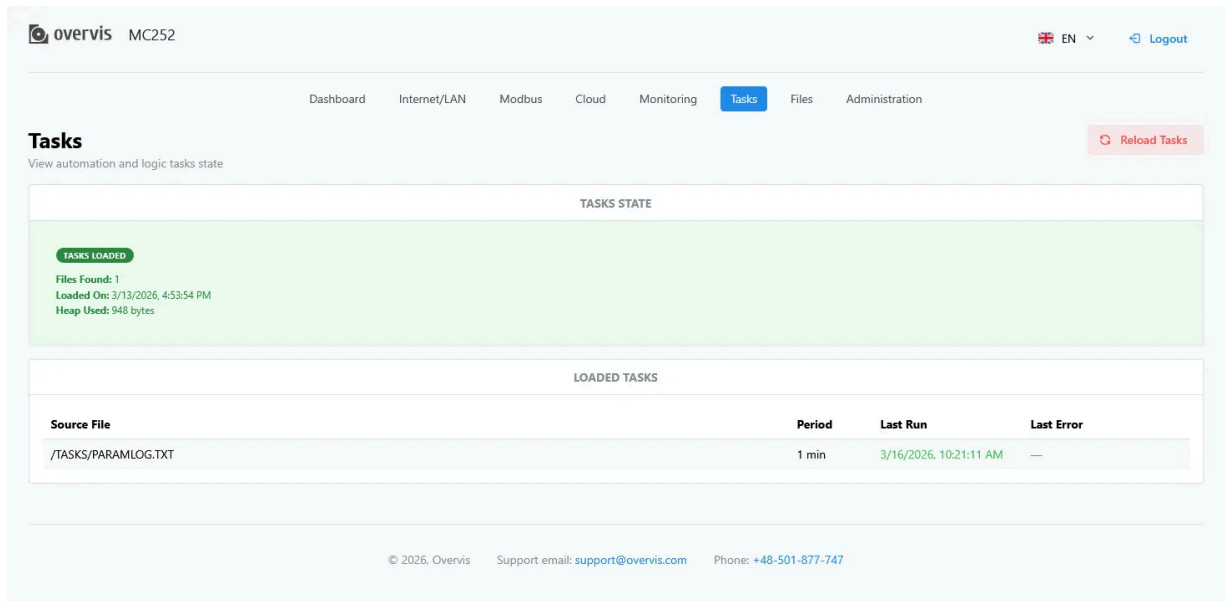


Fig. 11 - The tasks management page

The factory default configuration includes no logic tasks.

If the memory card with taskfiles in the TASKS folder is present, the tasks would be read into the internal memory (see Logic Programming). This page indicates the result of reading the folder TASKS, including the number of discovered files.

If during reading and verifying the program errors were detected, then it indicates the type of error, file and line number of the file error. If the program consisted of several files in the folder TASKS, then the internal memory will read all files except those in which errors are detected. Therefore, during error correction you should reread tasks, to clear again the internal memory.

The successfully read files are displayed below in the logic tasks table. The original taskfile name, the task programmed period of reruns, the task last run time, and the last run error, if any, are displayed for each of the tasks. If the task was recently run, the last run time is displayed in green. If the task hangs or otherwise misses its scheduled runs, the last run time is displayed in red.

The **Reload Tasks** button allows to clear the internal tasks memory and then reload the tasks (if the memory card is present).

Files

Manage the contents of the inserted memory card.

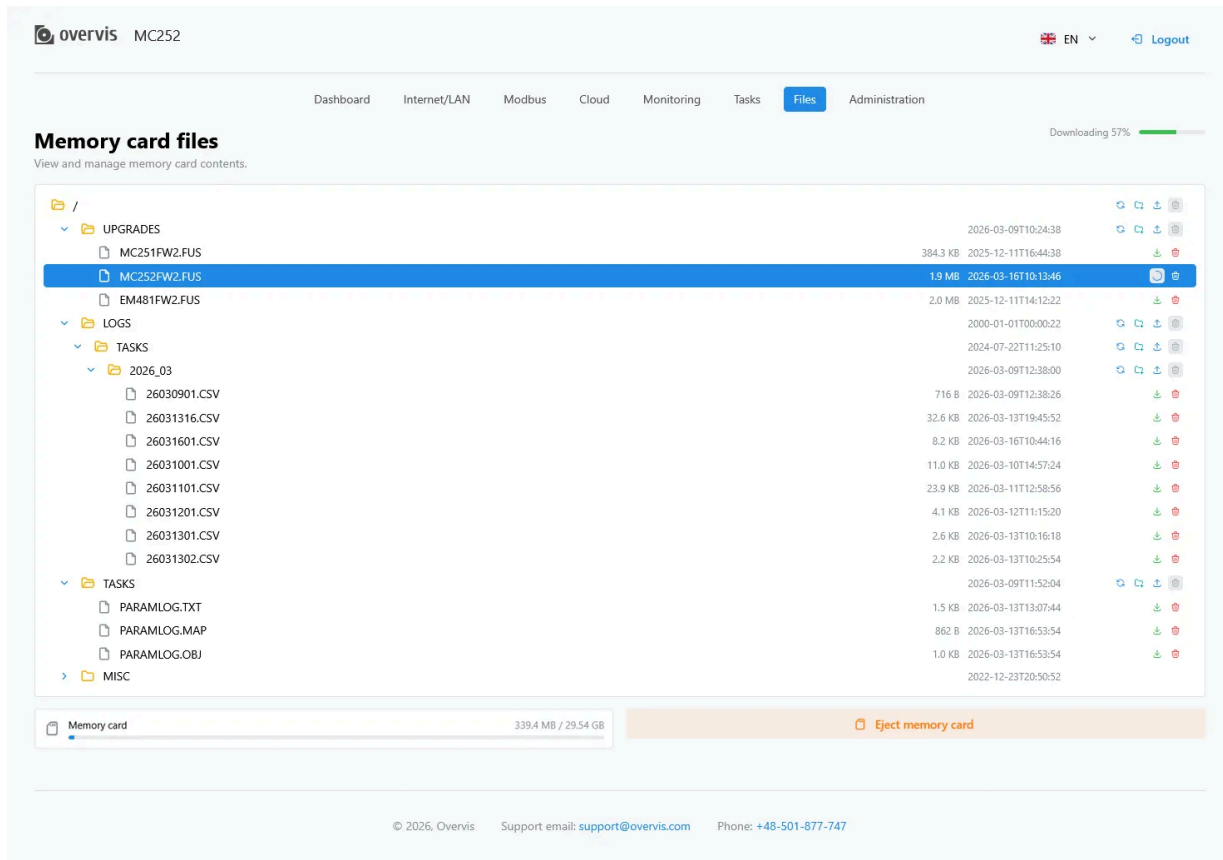


Fig. 12 - The memory card page

Folders can be opened and items can be selected in the tree view of the card contents.

There are options for a folder to re-read its contents, create a new subfolder, upload a file from the local device, remove a folder (provided it is empty). Files have options for download or removal.

The **Eject Memory Card** button allows to safely remove the card.

If the card is not removed within 30 seconds, it would be automatically re-mounted.

Administration

General system maintenance and device settings.

overvis MC252
EN ▼
Logout

Dashboard
Internet/LAN
Modbus
Cloud
Monitoring
Tasks
Files
Administration

Administration

Configure general system settings including time, security, device management, and astronomical settings.

Firmware

[Open documentation](#)

Installed version: 45
Available version: 46
<http://static.overvis.com/assets/firmware/MC252-6-1/MC252-6-1-current.fus>

Version on SD card: 46
Update is ready for installation.

The device will restart to install the update. The process may take up to 5 minutes, and the device will show the update status on its display.

Install Update

Automatic Firmware Update Settings ▼

Device Management

Texts Language:
Used for display texts and Modbus parameter logs.

English ▾

Automatic Reboot

Automatic reboot is an optional feature that restarts the device after a specified time of inactivity or a defined interval since startup. This can help resolve connectivity problems that may require restarting the LTE or Ethernet stack.

Autoreboot Mode:

Disabled ▾

Reboot After (minutes):

120 ▴ ▾

HTTP Server

Server Port:
TCP port for HTTP connections for the web interface and API.

80 ▴ ▾

Button Access Duration (s):
Time of free unauthenticated access after "R" button press.

600 ▴ ▾

Power & Saving

Brown-Out Voltage (mV):
Below this voltage the SD card will be safely unmounted.

9000 ▴ ▾

Display Saving Delay (minutes):
Inactivity time before the display saving mode is entered.

10 ▴ ▾

Display Saving Brightness (%):
Brightness level during the display saving mode.

5 ▴ ▾

Save & Reboot

Controls

Restart the device to apply pending changes or resolve connectivity issues.

Reboot Device

Reset all settings to factory defaults. This action cannot be undone.

Factory Reset

Time Settings

Timezone GMT Offset:
15-minute intervals (-96 to +96).
Eg. GMT+2 is 8.

8 ▴ ▾

DST Mode:
Preset or manual Daylight Saving Time settings.

Ukraine ▾

NTP Synchronization

NTP Server Connection:

Prefer Ethernet ▾

Primary NTP Server:

pool.ntp.org

Additional NTP Server:

europe.pool.ntp.org

Sync Period (hours):
Resync interval (0 = disable periodic sync).

24 ▴ ▾

Sync Threshold (sec):
Only resync if time difference exceeds this value.

2 ▴ ▾

Save & Reboot

Resync NTP Now

Security Settings

The following toggles control global device access and take precedence over the password settings below. If you require communication but want to secure it, consider setting passwords instead of fully disabling access.

Allow Read via SMS
Enable or disable the ability to read Modbus registers using SMS.

Allow Write via SMS
Enable or disable the ability to write Modbus registers using SMS.

Allow Modbus Read
Enable or disable reading from connected devices via Modbus.

Allow Modbus Write
Enable or disable writing to connected devices via Modbus.

Cloud Admin Access
Automatically grant admin access from the Overvis cloud.

Wireguard Admin Access
Automatically grant admin access from a Wireguard VPN.

Passwords

Admin Password: *
5-10 ASCII characters, no spaces. Required to access the web interface and can be provided via Modbus for full register access.

•••••• ▾

Modbus passwords are entered by writing to the password registers on the virtual device. They control access to connected devices and MC252 registers. Leave blank to disable password protection and allow unrestricted access.

Modbus Read Password:
0-10 ASCII characters. Grants read access to connected devices via Modbus.

•••••• ▾

Modbus Write Password:
0-10 ASCII characters. Grants write access to connected devices via Modbus.

•••••• ▾

MC252 supports reading and writing registers via SMS commands. Refer to the documentation for details. The passwords below are always required as part of SMS commands.

SMS Read Password: *
3-10 ASCII characters, no spaces. Required for SMS read commands.

•••••• ▾

SMS Write Password: *
3-10 ASCII characters, no spaces. Required for SMS write commands.

•••••• ▾

Save & Reboot

Logging

Enable or disable various system logging for debugging purposes. May affect performance. Logs will be stored on SD card in the folder "/LOGS/".

ATM Logging
Enable GSM AT modem logging.

System Logging
Enable system event logging.

LwIP Logging
Enable Ethernet IP stack system logging.

Forwarding Logging
Enable packet forwarding system logging.

WireGuard Logging
Enable WireGuard VPN system logging.

Tasks Logging
Enable system logging of tasks messages.

Tasks Data Logging

Configure logging of task data from logic files. Logs will be stored on SD card in the folder "/LOGS/TASKS/".

Log File Format:

.CSV (semicolon-separated) ▾

Storage Interval (days):
Period before old log files can be overwritten. Leave empty to disable.

184 ▴ ▾

File Size Limit (KB):
Maximum size of a single log file in kilobytes.

1024 ▴ ▾

Save & Reboot

Set Time

Time is shown and set according to the device's timezone. If NTP synchronization is enabled, it will automatically update the time and override manual changes.

Current Date:

03 / 16 / 2026 📅

Current Time:

10:23:59 AM

Set Time

Astronomical Settings

Those settings are used by user tasks to correctly calculate sunrise/sunset times.

Geographic quadrant:
Latitude/longitude sign

North Lat/East Lon ▾

Sun Zenith Type:
Type of twilight calculation

Civil ▾

Latitude

Degrees:

52 ▴ ▾

Minutes:

13 ▴ ▾

Seconds:

0 ▴ ▾

Longitude

Degrees:

21 ▴ ▾

Minutes:

32 ▴ ▾

Seconds:

0 ▴ ▾

Save & Reboot

© 2026, Overvis
Support email: support@overvis.com
Phone: +48-501-877-747

Fig. 13 - The administration page

- **Firmware:** Perform firmware upgrades.
- **Security:** Change passwords and access settings.
- **Logging:** Configure system log options and Modbus parameters log format.
- **Service controls:** Device soft or hard reset.
- **System Time:** Configure the internal clock.
- **Device management:** Configure general miscellaneous settings.

Need Help?

For technical support and assistance:

- Email: support@overvis.com
- Support portal: www.overvis.com/support

Modbus Interface Reference

The Overvis MC252 operates as a Modbus gateway, listening for Modbus TCP connections on port 502 (configurable). It supports connections from standard Modbus TCP client applications. Windows client software for basic testing is available for download [here](#).

Upon receiving a connection request, the MC252 checks its list of active clients. If the maximum list size (as per technical specifications) has not been reached, the new client is accepted.

Once connected, the MC252 processes Modbus requests from the client. In **RS-485 Slave Mode**, it also accepts requests from a Modbus Master on the RS-485 bus.

Request Processing

The device analyzes each request based on the requested function and the client's access rights (determined by passwords entered).

- **Blocked Requests:** If a request is blocked due to insufficient rights, the MC252 sends back a Modbus exception (default code 1).
- **Internal Requests:** If the request addresses the MC252 itself, it is processed internally, and a reply is sent back.
- **Redirection:**
 - **RS-485 Master Mode:** Requests for other devices are converted (Modbus TCP/RTU/ASCII) and redirected to the RS-485 bus. The RS-485 indicator lights up while waiting for a response.
 - **Remote Server:** If configured, requests can be redirected to one or several remote Modbus TCP servers via Ethernet or GSM/LTE.

Caution

Ensure there are no duplicate Modbus addresses (identifiers) across the RS-485 bus and the remote Modbus TCP server network. The response is accepted from the first responding device.

Caution

Ensure there are no redirection loops. E.g. the remote server address is not the MC252 own address, etc. Redirection loops cause the responses to take longer, tend to cause exceptions or connection losses.

If a valid response is received, the MC252 forwards it back to the client.

Troubleshooting

- **Exception Illegal Function (Code 1) is returned for a request:** Sent either by MC252 (if insufficient rights for the request) or by the target device (if the request unsupported). Check the MC252 settings, Administration tab for protection settings. Send the correct password to the MC252 before using this request. Try allowing the requests.
- **Exception Gateway Path Unavailable (Code 10 or 0x0A) is returned for a request:** Sent if the request cannot be redirected. Check the connection to the remote is configured and established (if the target should be requested)

using Modbus TCP). Check the Modbus Master mode is configured (if the target should be requested using Modbus RTU/ASCII). Check the Modbus address ranges on MC252 Modbus settings tab.

- **Exception Target Device Failed to Respond (Code 11 or 0x0B) is returned:** Sent if no response is received from the target device. Check the device settings match the MC252 Modbus tab configuration. Check the target device is powered on.
- **Some exception (not 1, 10 or 11) is returned:** Sent by the target device if it could not perform the request or produce the response data. Check the device manual. Try some other request.
- **The data has been acquired, but the values seem wrong:** Check the device manual. Ensure the Modbus device address and Modbus register address in the request are correct. Check the data formatting in the Modbus client software (which is sending this request). Try dividing the acquired value by 10, 100 or 1000.

SMS Modbus Access

With an active SIM card, the MC252 can process Modbus requests via SMS.

SMS Format

Incoming SMS messages are checked for a valid Modbus request format.

- **Modbus Requests:** Must begin with a configured password. If the password matches, the request is processed.
- **Other Messages:** Stored in the incoming SMS list and can be processed by a task file (see Logic Programming).

Request Syntax: [Password] [Access] [DeviceID] [Resource] [Address] [Value]

- **Password:** As configured in settings.
- **Access:** R (Read, FC 1-4) or W (Write, FC 5-6).
- **DeviceID:** Modbus address of the target device.
- **Resource:**
 - H: Holding Registers
 - I: Input Registers
 - D: Discrete Inputs
 - C: Flags (Coils)
- **Address:** Register address.
- **Value:** (Write requests only) The value to write.

Examples:

- Read register 100 of device 1: abc r1h100 (assuming password is "abc")
- Write 5000 to register 174 of device 2: stanc12 w2h174 5000 (assuming password is "stanc12")

SMS Responses

The MC252 sends a response SMS upon processing the Modbus request command.

- **Success:** Returns the command (without password) and the register value.
 - Example: r1h100 2200 (Read value 2200 from register 100 of device 1).
- **Error:** Returns the command, EXC., the exception code, and a description.
 - Example: r3h873 EXC.2 ILLEGAL DATA ADDRESS (Register 873 of device 3 is not readable).

Table 1 - Standard Modbus Exception Codes

Code	Exception	Description
1	ILLEGAL FUNCTION	The received function code cannot be processed
2	ILLEGAL DATA ADDRESS	The data address specified in the request is not available
3	ILLEGAL DATA VALUE	The value contained in the request data field is invalid value
4	DEVICE FAILURE	Unrecoverable error has occurred when the addressee has tried to perform the requested action
5	ACKNOWLEDGE	The addressee accepted the request and processed it, but it takes a long time
6	DEVICE BUSY	The addressee is busy processing the command. The client can retry the request later
8	MEMORY PARITY ERROR	The parity error was detected when the addressee has tried to read the extended memory
10	GATEWAY PATHS NOT AVAILABLE	The gateway cannot redirect the request, since there is no path (connection) to the addressee
11	TARGET DEVICE FAILED TO RESPOND TO GATEWAY	The gateway did not receive a response to the forwarded request, because the addressee did not respond in time

Configuration via Modbus

The MC252 can be configured using any Modbus TCP client.

- 1 **Connect:** Use the MC252's IP address (press R button on device to view) and Modbus ID (default 111).
- 2 **Enter Password:** Write the password (default on device label) into the password registers (see **Current Mode Parameters**).
- 3 **Verify Mode:** If the password is correct, the Mode register will read 1 (Setup Mode).

Managing Settings

In Setup Mode, you can modify the **Changeable Settings** registers.

- **Save Changes:** Write 2 to the Command Register. Verify by comparing changeable parameters with saved parameters.
- **Save & Apply:** Write 4 to the Command Register. Applies Modbus parameters immediately without restart.
- **Cancel Changes:** Write 9 to the Command Register. Reverts changeable parameters to saved values.
- **Factory Reset:** Write 444 to the Command Register. Resets all settings to defaults.
- **Restart Device:** Write 1 to the Command Register. Required for some settings to take effect.
- **Exit Setup Mode:** Write 0 to the first password register (100). This clears the password and command registers.

MC252 Parameters

Table 2 - Parameter Data Formats

Parameter	Range of values	Description	Number of occupied registers
Number	0 – 65535	Integer number (16 bit) in standard range of Modbus register values	1
Number	-32768 – +32767	Integer number (16 bit) in complementary arithmetic	1
Number	0 – 4294967295 in two registers, MSB part – first	Integer number, which value can exceed the limit for Modbus register (65535)	2
Number	-2147483648 – +2147483647 in two registers, the MSB part is the first	Integer number (32 bit) in complementary arithmetic	2
Character string	In every register – number of 0 to 255 - ASCII character code or 0 (the end of string)	A set of values, each of which is equal to the code of one character in the ASCII encoding. If the string is shorter than the maximum length, the code 0 is placed after the last character.	Max. length of string for this parameter
IP-address (IP-mask)	In every register – one byte (0 – 255)	Set of four byte of address IPv4, from left to right	4
MAC-address	In every register – one byte (0 – 255)	Set of six byte of address MAC-48, from left to right	6

Table 3 - Parameter Groups

Group	Description	Access	Address
Device description	Device and firmware identification	Any mode, read only	0 – 3
Current mode	Access and general commands controls	Password entry is available in any mode, command entry - only in Setup Mode (after password entry)	100 – 120
Current status	Interfaces and software modules operation, current time and statistics	Any mode, read only	121 – 295, 900 – 944, 2000–2023
Changeable settings	Interfaces and features selection and configuration	Only in Setup Mode, reading or writing	300 – 899,

Group	Description	Access	Address
Active settings	The configuration being used by the device at the moment	In any mode, read only	2300 – 2899,
Saved settings	This set is saved regardless the power of the device and is used at restart	Only in mode of setting, read only	3300 – 3899,
Clock setting	See Table 5.8.	Only in clock setting mode, for reading or writing	34817 – 34825

Device Description Parameters

Table 4 - Device Description Parameters

Parameter	Description	Address
Device type	The code that defines the Modbus device for the manufacturer (47 – MC252)	0
Firmware version	Firmware version of embedded software	1
Check code	CRC32 of firmware of embedded software	2 – 3

Current Mode Parameters

Table 5 - Current Mode Parameters

Parameter	Range of values	Initial value	Description	Address
Entered password	String of characters	0	When entering a valid password, the client is given the appropriate permission (see registers 510 - 569) When you enter an empty string, the client rights are reset to the rights level at the time of connection	100 – 119
Control command	0 – 65397, writing in the configuration mode	0	See Table 5.1 for the list of commands.	120

Table 5.1 - Control Commands (Register 120)

Value	Command	Description
0	No activity	No action performed
1	Restart	Restart the MC252 device

Value	Command	Description
2	Save	Save settings changes via Modbus
3	Apply	Apply settings without restarting (Modbus parameters only)
4	Save and apply	Execute "Save" then "Apply" commands
6	Export	Save settings to a file on the memory card (see Memory card)
7	Import	Read settings from a file on the memory card and save them
9	Cancel	Revert to saved settings
81	Synchronize clock	Synchronize clock with NTP server
161	Test SMS	Send a test SMS to the main subscriber number
444	Factory Reset	Reset settings to factory defaults
35381	Start clock setting	Allow access to clock setting registers
35431	Cancel clock setting	Close access to clock setting registers without changes
40959	Update task memory	Erase the logic program (re-read if memory card is inserted)
64893	Download Updates	Download latest firmware from cloud to MC252FW2.FUS
65397	Update Firmware	Program firmware from file MC252FW2.FUS

Current Status Parameters

Table 6 - Current Status Parameters

Parameter	Description	Address
Mode (for details see reg. 122)	0: User's mode;	121
	1: Setting mode	
Tabs of access	See Table 6.1 for the list of access bits.	122
Time, min	Number of minutes since the moment of start-up	123 – 124
Number of TCP clients	Number of occupied connections of TCP	125
Limit of TCP clients	Number of prospective clients of TCP	126
Load of RS-485, query/s	Total number of query/s via RS-485	127
Effective load of RS-485, query/s	Number of responds without errors via RS-485 per second	128

Parameter	Description	Address
Load of RS-485 per second, %	Load of RS-485 for the last second considering the set rate of RS-485 and time of inactivity	129
Load of RS-485 per minute, %	Load of RS-485 for the last minute	130
Load of RS-485 for 5 minutes, %	Load of RS-485 for the last 5 minutes	131
Load of Modbus TCP, query/s	The number of queries received from clients via Modbus TCP per second	132
Effective load of Modbus TCP, query/s	Number of responds without errors being sent to the client via Modbus TCP per second	133
Load of GSM, kB/s	Load of wireless channel with GSM being switched on	134
Load of Ethernet, 100 kB/s	Load of wire channel with Ethernet being switched on	135
Max. number of clients of TCP	Maximal number of simultaneously connected clients via TCP – from the moment of start up	136
Max. load of Modbus TCP, query/s	Maximal number of queries received per second from the clients via Modbus TCP – from the moment of start up	137
Max. load of RS-485, %	Maximal load of RS-485 for 5 minutes – from the moment of start up	138
Max. load of GSM, kB/s	Maximal load of GPRS/LTE – from the moment of start up	139
Current IP-address of Ethernet	IP-address, by which MC252 device is accessible in Ethernet network ¹	140 – 143
Current MAC-address of Ethernet	MAC-address, by which MC252 is detected in Ethernet network	144 – 149
Unused parameter	The parameter is reserved for compatibility	150 – 164
Time to connect to the data collection server	0 – connection to the data collection server is set;	165
	1 – connection to the data collection server is performed;	
	2 – 65534: the number of seconds before reconnecting;	
	65535: connection to the server is not used	
Number of programmed restarts	Number of restarts in accordance to the user setting – for total operational time	166

Parameter	Description	Address
Number of critical errors	Number of noted errors (failures) causing the restart of the device – for total operational time	167
Total operation time, min	The number of minutes of operating time - for the total operational time	168 – 169
Current time	Number of seconds since 1st of January of specified year (see reg. 172)	170 – 171
Year of countdown	Year, since 1st of January which is taken for time counting	172
Time zone, min	Time zone, for the time count, number of minutes with sign as to UTC+00	173
Temperature, °C	Temperature inside MC252	174
Power voltage, mV	Bus voltage of 12 V	175
Time to connect to the first remote TCP server, s	0 – connection to the remote server is set;	
	1 – connecting to the remote server;	
	2 – 65534: the number of seconds before reconnecting;	
	65535: connection to the remote TCP server is not used	176
Interface for connecting to the first remote TCP server	0 – the connection is not set;	177
	1 – Ethernet connection is set;	
	2 – GSM connection is set	
DST increment, min	Effective day saving time increment	178
Unused parameter	The parameter is reserved for compatibility	179 – 209
IP-address of client 1	IP-address of client, 0.0.0.0 – not connected	180 – 183
Port of client 1	Port of client, 0 – not connected	184
IP-address of client 2	IP-address of client, 0.0.0.0 – not connected	185 – 188
Port of client 2	Port of client, 0 – not connected	189
IP-address of client 3	IP-address of client, 0.0.0.0 – not connected	190 – 193
Port of client 3	Port of client, 0 – not connected	194

Parameter	Description	Address
IP-address of client 4	IP-address of client, 0.0.0.0 – not connected	195 – 198
Port of client 4	Port of client, 0 – not connected	199
Unused parameter	The parameter is reserved for compatibility and is equal to 0	200 – 209
Year (current time)	Current year	210
Month (current time)	Current month	211
Day of the month (current time)	Current day of the month	212
Hour (current time)	Current hour	213
Minute (current time)	Current minute	214
Second (current time)	Current second	215
Day of week (current time)	Current day of week (1 – Monday)	216
Month (winter time)	Current month, excluding summer time	217
Day of the month (winter time)	Current day of the month, excluding summer time	218
Hour (winter time)	Current hour, excluding summer time	219
Unused parameter	The parameter is reserved for compatibility	220 – 229
Time of day, s	Number of seconds from midnight of the current day	230 – 231
Time of sunrise, s	0 – 86399: Number of seconds from midnight to sunrise; 86400: Sunrise is not observed on this day	232 – 233
Sunset time, s	-1: Sunset is not observed on this day; 0 – 86399: Number of seconds from midnight to sunset;	234 – 235
Time of day, ms	Number of milliseconds from midnight of the current day	236 – 237
Memory card occupancy, 0.01%	10000 – memory card is missing or full	238
Estimated time spent on 1% of the memory card, days	0 – less than one day; 1 – 9999 – days for 1% of the memory card capacity; 10,000 – more than 10,000 days;	239

Parameter	Description	Address
	65535 – unknown	
Time before connecting to the second remote TCP server, s	Similar to register 176	240
Connection interface to the second remote TCP server	Similar to register 177	241
Time before connection to the third remote TCP server, s	Similar to register 176	242
Connection interface to the third remote TCP server	Similar to register 177	243
Current IP-address of GSM ²	IP-address obtained from GPRS ¹ provider	900 – 903
Unused parameter	The parameter is reserved for compatibility	904 – 914
GSM signal level, % ²	Signal level and quality of radio communication with GSM provider	915
Unused parameter	The parameter is reserved for compatibility	916 – 940
Memory card capacity, kB	0 – card missing or unformatted in FAT / FAT32	941 – 942
The amount of data that has not been written to the memory card since the start, bytes	0 – there were no data record losses	943 - 944
	4294967295 – more than 4 GB losses	
Firmware Download Status Bits	See Table 6.2 for the list of status bits.	2004
Firmware file download progress (x 0.01%)	Proportion of downloaded data size to total file size.	2005
	10000 – File fully downloaded	
Loaded programming logic tasks number	The number of the tasks correctly loaded into the internal memory	2020
Code of the first illegal logic task error	The code of the first error met in the first taskfile that could not be loaded	2021
Line number of the first illegal logic task error	The line number where the first error was met in the first taskfile that could not be loaded	2022
Loaded programming logic tasks memory usage, bytes	The amount of RAM reserved for the programming logic tasks execution	2023

Parameter	Description	Address
Firmware File Header	The string identifier of the version, e.g., "MC252, ver.45".	2030 – 2061
A blank string indicates the file is either unverified or invalid		

Table 6.1 - Tabs of Access Bits (Register 122)

Bit	Description	Value 0	Value 1
0	Permission to obtain RS-485 reading rights (via password)	Cannot be obtained	Can be obtained
1	Current RS-485 reading permission	No permission	Permission granted
2	Permission to obtain RS-485 write/control rights (via password)	Cannot be obtained	Can be obtained
3	Current RS-485 write/control permission	No permission	Permission granted
4	Permission to obtain MC252 register access (via password)	Cannot be obtained	Can be obtained
5	Current MC252 register access permission	No permission	Permission granted
6	Reserved	-	Always 1
7	Permission to configure MC252	No permission	Permission granted
8	Reserved	Always 0	-
9	Connection right	-	Client has right (Always 1)
12	Permission to set the clock	No permission	Permission granted

Table 6.2 - Firmware Download Status Bits (Register 2004)

Bit	Status	Value 0	Value 1
1	Busy Status	Waiting for command	File download in progress
2	Update Download Error	No error	Download error occurred
3	Server Connection	No connection	Connected to server
4	File Data Reception	No data received	File data received
6	File Download Completion	File not downloaded	File fully downloaded

Bit	Status	Value 0	Value 1
7	File Validity	Not confirmed	File is valid

Settings Parameters

Table 7 - Settings Parameters

Note

The internal structure of all sets of settings is similar to the structure of the set described, except for the initial address.

Parameter	Range of values	Factory setting	Description	Address
Ethernet network				
Static IP-address	IP-address	192.168.0.111	If the dynamic addressing is switched off or not available, IP-address of the device in Ethernet network is equal to this value	300 – 303
Subnetwork mask	IP-mask	255.255.255.0	It is used only with static IP-address	304 – 307
Gateway	IP-address	192.168.0.1	It is used only together with static IP-address for communication with other networks, or as an address of DNS/DHCP servers	308 – 311
Switch on the dynamic addressing with a help of DHCP	0 – 1	1	0 – for addressing in Ethernet, the specified values of the IP address, mask and gateway are used; 1 – If DHCP server is available in the network, then IP address, mask and gateway are received from the server	312
Unused parameter	0	0	Should be 0 for compatibility	313
Switch on the use of server gateway DNS	0 – 1	1	It is used if DHCP is not available (switched off): 0 – DNS of gateway is not used; 1 – The gateway DNS is used to determine the IP addresses of other	314

Parameter	Range of values	Factory setting	Description	Address
			servers, if they are specified by host names	
IP-address of DNS server	IP-address	8.8.8.8	It is used if DHCP is not available (switched off);	315 – 318
			When the server of DNS gateway is used, it sets IP-address of additional DNS server	
IP-address of additional DNS server	IP-address	0.0.0.0	It is used if DHCP is not available (switched off);	319 – 322
			0.0.0.0 – it is not used	
Unused parameter	0	0	Should be 0 for compatibility	323 – 329
GSM network				
PIN-code of SIM-card	0 – 65535	65535	0 – 9999: this code is used for the SIM card if it demands the PIN code;	330
			Other values: the code isn't used; SIM card and GSM are unavailable if the card demand a code	
Enable automatic detection of GSM operator's APN	0 – 1	1	0 – connection is established by manually specified APN parameters;	331
			1 – APN is automatically determined for the operator by SIM card ICCID code	
Enable roaming exchange	0 – 1	1	0: GPRS/LTE is blocked in roaming;	332
			1: GPRS/LTE may be used in roaming	
Activate SMS in roaming	0 – 1	0	0 – SMS can be only received in roaming;	333
			1 – SMS can be received and sent in roaming	
Connection port via GSM	0 – 65535	0	It is used for external connection to the device via GSM with static IP, for	334

Parameter	Range of values	Factory setting	Description	Address
			communication using the Modbus TCP protocol or in tunnel mode.	
			0 – it is disabled	
Parameter is not used	0	0	It is not used; it should be equal to 0 for compatibility	335 – 351
APN log-in of GPRS/LTE service	Character string		Provided by the GSM service provider; up to 40 characters	352 – 391
APN password of GPRS/LTE service	Character string		Provided by the GSM service provider; up to 24 characters	392 – 415
APN address of host	Character string		Provided by the GSM service provider; up to 34 characters; there cannot be spaces in a string	416 – 449
TCP Server				
Connection port via Ethernet	1 – 65535	502	It is used for external connection to MC252 via Ethernet for exchange via Modbus TCP protocol or in tunnel mode	450
Disconnect inactive clients	0 – 1	1	0 – incoming TCP connection is kept regardless of the time between requests from the client; 1 – disconnect clients that did not send requests for longer than a specified time	451
Max. request waiting time, s	0 – 600 000	90	Used if disconnection of inactive clients is selected	452 – 453
Unused parameter	0	0	Should be 0 for compatibility	454 – 456
Own Modbus-identifier of MC252	0 – 247	111	0 – all queries are sent via Modbus to the Modbus network, the device registers are unavailable by Modbus; 1 – 247 – the device responds Modbus queries with this Modbus identifier without forwarding them on	457
RS-485 network				

Parameter	Range of values	Factory setting	Description	Address
Bit rate via RS-485, bit/sec	75 – 230 400	9 600	It is used in case of data exchange between the devices via RS-485, the same value for the devices on the same RS-485 bus cable	458 – 459
Quotient for Modbus RTU silence time between frames	0 – 5	1	It is used for transmissions via RS-485 in Modbus RTU mode. During response reception, if the pause between the bytes is longer than the silence time, the frame is considered complete. 0 - standard silence time (depends on the transmission speed and is equal to the transmission time of 3.5 bytes, or 1.75 ms for speeds above 19200 bps) 1 - 5: N quotient for a prolonged silence time multiplied by 2 ^N	460
Byte format when transmitting via RS-485	0 – 5	5	It is used in case of data exchange between the devices via RS-485. See Table 7.1 for formats.	461
Waiting time for starting the Modbus RTU response, ms	0 – 60 000	200	It is used for transmissions via RS-485 in Modbus RTU mode. After transmission of query, if the first byte of the response was not received within this time interval, the waiting for the response is terminated. The response is always waited for at least the silence time between frames	462
Enable ASCII exchange mode in Modbus network	0 – 1	0	Exchange mode via RS-485, the same value for all units on the same RS-485 bus cable. 0 – RTU exchange mode (format: 1 start bit, 8 data bits, 2 stop-bits, parity bit, and stop bit or only 1 stop bit – total from 10 to 11 bits); 1 – ASCII exchange mode (format: 1 start bit, 7 data bits, 2 stop-bits or parity bit and stop bit - total is 10 bits). The non-standard byte formats (register 461, values 4 and 5) are not	463

Parameter	Range of values	Factory setting	Description	Address
			available in this case, format 3 (2 stop bits) is used instead	
Response time for subsequent Modbus ASCII character, ms	0 – 60 000	1 000	It is used in case of data transfer via RS-485 in Modbus ASCII mode. If you receive a response, if the next byte of the response was not received within this time interval, then the response waiting is stopped. Waiting is always not less than the transmission time of one character (depends on the transmission speed)	464
Connection to the cloud server				
Mode of connection to the cloud server	0 – 8	0	See Table 7.2 for connection modes.	465
Cloud server connection port	0 – 65535	20502	The port to which the party is addressed, making connection between MC252 and the server (see reg. 465)	466
Time of waiting for response from the cloud server, s	0 – 3 600	120	0 – the server silence time is not limited; 1–3600 – max. time of server silence after which the connection will be stopped and must be remade again	467
Delay time before reconnecting to the cloud server, s	0 – 30 000	15	It is used when connecting to the server. After losing connection to the server, the reconnection will be performed after the specified waiting time	468
Unused parameter	0	0	Should be 0 for compatibility	469 - 473
Cloud server address	Character string	modbus.overvis.com	It is used when connecting to the server. Address of the remote server with which the connection is maintained. The string of up to 36 characters can be indicated as address. This string	474 - 509

Parameter	Range of values	Factory setting	Description	Address
			should not have any spaces	
Protection				
Password for Setup Mode access	Character string	specified on the device label	It is used to access the configuration mode. The string of 5 to 10 characters in length can be indicated as password. This string should not have any spaces	510 – 519
Parameter is not used	0	0	It is not used; it should be equal to 0 for compatibility	520 – 529
Password for writing permission using incoming SMS	Character string	specified on the device label	It is used to verify the authenticity of incoming SMS with request for record or with acknowledgment of the fault. The string of 3 to 10 characters in length can be specified as password. This string should not have any spaces	530 – 539
Password for reading permission using incoming SMS	Character string	specified on the device label	It is used to verify the authenticity of incoming SMS with request for reading or with acknowledgment of the fault. The string of 3 to 10 characters in length can be specified as password. This string should not have any spaces	540 – 549
Password for writing permission via Modbus to the other devices	Character string		It is used to access devices connected to the MC252, to request write or control functions that can change the status of these devices. The string up to 10 characters in length can be specified as password. This string should not have any spaces	550 – 559
Password for reading permission via Modbus	Character string		It is used to access devices connected to the MC252, to request read functions, or to access the MC252 registers, except for registers of version, password, mode and tabs. The string up to 10 characters in length can be indicated as password. This string should not have any spaces	560 – 569

Parameter	Range of values	Factory setting	Description	Address
Enable the protection mode against writing via SMS	0 – 1	0	0 – Protection against recording is regulated with help of other parameters (password);	570
			1 – Blocking of queries via SMS for function of writing	
Enable the protection mode against reading via SMS	0 – 1	0	0 – Protection against reading is regulated with help of other parameters (password);	571
			1 – Blocking of queries via SMS for function of reading	
Enable the protection mode against writing via Modbus	0 – 1	0	0 – Protection against recording is regulated with help of other parameters (password) or deactivated;	572
			1 – Blocking of any queries for functions, excepting functions of Modbus 1, 2, 3, 4, 7, 17, 20	
Enable the protection mode against reading via Modbus	0 – 1	0	0 – Protection against reading is regulated with help of other parameters (password) or deactivated;	573
			1 – Blocking of queries for functions of Modbus 1, 2, 3, 4, 7, 17, 20, excepting reading using function 3 of registers of version, mode and tabs	
Parameter is not used	0	0	It is not used; it should be equal to 0 for compatibility	574
Miscellaneous				
Parameter is not used	0	0	It is not used; it should be equal to 0 for compatibility	575 – 630
Restart time, min	5 – 7 200	120	Used when automatic restart is enabled.	631

Parameter	Range of values	Factory setting	Description	Address
Automatic restart mode	0 – 2	0	It is used when automatic restart is enabled:	632
			0 – the automatic restart is disabled;	
			1 – the device is restarted after a specified period of time since the start;	
2 – the device is restarted after a specified period of time since the last transmission via Ethernet or GSM networks.				
Modbus exception code generated when access is denied	0 – 255	1	0 – if the access to Modbus registers is denied, the response to the client is not returned;	633
			1 – 255 – if you deny access to the client who sent the request, this exception code is returned	
Modbus exception code generated when there is no response	0 – 255	11	0 – if there is no response from the addressee (Gateway Timeout), the response to the client is not returned;	634
			1 – 255 – if there is no response from the request recipient, this exception code is returned to the client	
Parameter is not used	0	0	It is not used; it should be equal to 0 for compatibility	635
Modbus exception code generated if there is no connection to query addressee	0 – 255	10	0 – If there is no connection to the query addressee (Gateway Path Unavailable), response is not returned to the client;	636
			1 – 255 – if there is no connection to the query addressee, this exception code is returned to the client	
RS-485 transmission mode	0 – 2	0	0 – Master mode (Modbus Master): RS-485 is used to send queries;	637

Parameter	Range of values	Factory setting	Description	Address
			1 – Slave mode (Modbus Slave): RS-485 is used to receive queries from additional client;	
			2 – tunnel mode, used to transmit data "as is", without protocol verification	
First Modbus-identifier of RS-485	1 – 255	1	Two parameters define a range of Modbus identifiers used for RS-485.	638
			In the master mode the queries with addresses in this range (and also the broadcast ones with address 0) are sent via RS-485.	
			In the slave mode the queries with addresses in this range	
			(and also the broadcast ones and the queries to MC252 address) are received via RS-485	
Last Modbus-identifier of RS-485	1 – 255	255	same as above	639
Connection to the first remote TCP server				
IP-address of the remote server	IP-address	192.168.0.112	It is used with enabled redirection of queries to TCP remote server. IP-address of the remote server wherewith connection is maintained	640 – 643
Port of the remote server connection	0 – 65535	502	Port of the remote server to which the TCP connection will be established	644
Time to wait for response from remote server, ms	0 – 60 000	1 000	It is used during redirection of queries to the remote server. After the query transfer, if the correct response failed to be received within this time interval, response waiting is stopped	645
Standby time to repeated connection to the remote server, s	0 – 240	20	It is used during redirection of queries to the remote server. After connection with the server is lost, the repeated	646

Parameter	Range of values	Factory setting	Description	Address
			connection will be performed after preset standby time	
Remote server connection mode	0 – 12	0	See Table 7.3 for connection modes.	647
First Modbus-identifier of the remote server	1 – 255	1	It is used during redirection of queries to the remote server.	648
			Two parameters define the range of Modbus identifiers used on the remote server.	
			Queries with addresses in this range (and also the broadcast ones with address 0) are sent to the remote Modbus TCP server	
Last Modbus-identifier of the remote server	1 – 255	255	same as above	649
Parameter is not used	0	0	It is not used; it should be equal to 0 for compatibility	650 – 699
Daylight saving time				
Daylight saving time transition mode	0 – 200	12	See Table 7.4 for transition modes.	700
Preset month for transition to daylight saving time	1 – 12	3	It is used if you selected the automatic transition to daylight saving time on the specified days. The month when the clock will be set one hour ahead	701
Preset week of the month for transition to daylight saving time	1 – 10	10	It is used if you selected the automatic transition to daylight saving time on the specified days. Week of the month when the clock will be set one hour ahead.	702
			1 – 5 – week of the month, including the part weeks;	

Parameter	Range of values	Factory setting	Description	Address
			other values – the last week of the month	
Preset day of the week for transition to daylight saving time	1 – 7	7	It is used if you selected the automatic transition to daylight saving time on the specified days. The day of the week when the clock will be set one hour ahead	703
Preset hour for transition to daylight saving time	0 – 22	2	It is used if you selected the automatic transition to daylight saving time on the specified days. The hour of the day at which the clock will be set one hour ahead	704
Preset month to revert to standard time	1 – 12	10	It is used if you selected the automatic transition to daylight saving time on the specified days. The month when the clock will be set one hour back	705
Preset week of the month to revert to standard time	1 – 10	10	It is used if you selected the automatic transition to daylight saving time on the specified days. Week of the month when the clock will be set one hour back.	706
			1 – 5 – week of the month, including the part weeks;	
			other values – the last week of the month	
Preset day of the week to revert to standard time	1 – 7	7	It is used if you selected the automatic transition to daylight saving time on the specified days. The day of the week when the clock will be set one hour back	707
Preset hour to revert to standard time	1 – 23	3	It is used if you selected the automatic transition to daylight saving time on the specified days. The hour of the day at which the clock will be set one hour back	708

Sunrises and sunsets calculation

Parameter	Range of values	Factory setting	Description	Address
Sun day	0 – 3	1	0 – official;	709
			1 – civil;	
			2 – nautical;	
			3 – astronomical;	
Latitude, degree	0 – 89	46	The absolute value of the latitude	710
Latitude, minute	0 – 59	29		711
Latitude, second	0 – 59	10		712
Longitude, degree	0 – 179	30	The absolute value of the longitude	713
Longitude, minute	0 – 59	43		714
Longitude, second	0 – 59	40		715
Quadrant	0 – 3	0	0 – N latitude, E longitude;	716
			1 – N latitude, W longitude;	
			2 – S latitude, E longitude;	
			3 – S latitude, W longitude	
Connection to the service servers				
NTP server connection mode	0 – 4	0	0 – clock synchronization with the server is not used;	717
			1 – to connect to the servers using Ethernet or GSM, preferably via Ethernet;	
			2 – to connect to the servers using Ethernet or GSM, preferably via GSM;	
			3 – to connect to the servers only via Ethernet;	
			4 – to connect to the servers only via GSM	
Time period of connection to NTP	1 – 240	24	It is used if you have enabled synchronization of clocks with the	718

Parameter	Range of values	Factory setting	Description	Address
servers, h			server clock. The time interval over which the server time is received	
Minimum shift of clock for synchronization, s	1 – 180	2	It is used if you have enabled synchronization of clocks with the server clock. The synchronization is performed after receiving the server time, if the difference between the clocks is no less than this value	719
Firmware update server connection mode	0 – 4	0	0 – firmware download is not used;	720
			1 – to connect to the servers using Ethernet or GSM, preferably via Ethernet;	
			2 – to connect to the servers using Ethernet or GSM, preferably via GSM;	
			3 – to connect to the servers only via Ethernet;	
			4 – to connect to the servers only via GSM	
Parameter is not used	0	0	Not used, must be equal 0 for compatibility	721 – 723
Parameter logging				
Minimum supply voltage for safe removal of memory card, mV	0 – 24 000	9 000	If the supply voltage is below the specified value, the memory card will be removed safely. The card can be used again after the supply voltage exceeds the minimum plus 0.5 V.	724
			0 – do not remove the memory card, including in a case if the supply voltage is unknown	
Format for logging parameters in task files	0 – 7	2	Used if there is a memory card and logging actions in task files.	725
			0 – not used	

Parameter	Range of values	Factory setting	Description	Address
			1 – compact file of data bytes;	
			2 – CSV table with text separator “;”;	
			3 – similar 2 with a separator “”;	
			4 – similar to 2 with delimiter - tab character	
			5 – CSV table without headers, with text separator “;”;	
			6 – similar 5 with a separator “”;	
			7 – similar to 5 with delimiter - tab character	
Maximum limited size of log files, kB	0 – 65535	1023	Used if there is a memory card present, logging actions in task files, and logging is enabled. The size of the generated files is limited to the specified size plus 1 kB	726
Minimum stored period in recorder mode, days	0 – 184	184	Used if there is a memory card present, logging actions in task files, and logging is enabled. 0-183 – the oldest files (older than the specified number of days ago) can be deleted to write new data; Other values – old files are saved, new data recording is suspended when the memory card is full	727
Parameter is not used	0	0	Not used, must be equal to 0 for compatibility	728 – 739
Abonents				
Main abonent phone number	Character string		Can be used to send SMS. Up to 20 characters. There can be no spaces in the line	740 – 759
Connection to the second remote TCP server				

Parameter	Range of values	Factory setting	Description	Address
Remote server IP address	IP address	192.168.0.113	Similar to 640–643	760 – 763
Remote server connection port	0 – 65535	502	Similar to 644	764
Waiting time for a response from a remote server, ms	0 – 60 000	1 000	Similar to 645	765
Waiting time before reconnect-ting to a remote server, ms	0 – 240	20	Similar to 646	766
Remote server connection mode	0 – 8	0	Similar to 647	767
First Modbus identifier of the remote server	1 – 255	1	Similar to 648	768
Last Modbus ID of the remote server	1 – 255	255	Similar to 649	769
Connection to the third remote TCP server				
Remote server IP address	IP address	192.168.0.113	Similar to 640–643	770 – 773
Remote server connection port	0 – 65535	502	Similar to 644	774
Waiting time for a response from a remote server, ms	0 – 60 000	1 000	Similar to 645	775
Waiting time before reconnect-ting to a remote server, ms	0 – 240	20	Similar to 646	776
Remote server connection mode	0 – 8	0	Similar to 647	777

Parameter	Range of values	Factory setting	Description	Address
First Modbus identifier of the remote server	1 – 255	1	Similar to 648	778
Last Modbus ID of the remote server	1 – 255	255	Similar to 649	779
Parameter not used	0	0	Not used, must be 0 for compatibility	780 – 799
Preset password for access to the first remote Modbus TCP server	Character string		Used only if connection to a remote server is selected and its ID is set (reg. 730). If a password is set, it will be entered immediately after connecting to the server. Obtained access rights depend on the settings of the remote server. A string up to 10 characters long can be specified as a password. No spaces can be in the string	800 – 809
Preset password for access to the second remote Modbus TCP server	Character string		Similar to 800 – 809	810 – 819
Preset password for access to the third remote Modbus TCP server	Character string		Similar to 800 – 809	820 – 829
Parameter not used	0	0	Not used, must be 0 for compatibility	830 – 849
Protocol for incoming Ethernet connections	0 – 1	0	0 – Modbus TCP protocol; 1 – tunnel mode, used for data transmission "as is", without protocol verification	850
Protocol for incoming GSM connections	0 – 1	0	0 – Modbus TCP protocol; 1 – tunnel mode, used for data transmission "as is", without protocol	851

Parameter	Range of values	Factory setting	Description	Address
			verification	
Clock settings ³				
Day-light saving time offset, min	-1440 – +1440		Current gain. It is set during manual transition to the daylight saving time, when selecting the automatic mode it will be adjusted within 5 minutes. The value must be a multiple of 15	34817
Time zone offset, min	-1440 – +1440	120	It is used during synchronization of the clock with the server clock. The value must be a multiple of 15	34818
Second	0 – 59		The time is to be set at the clock	34819
Minute	0 – 59			34820
Hour	0 – 23			34821
Day	1 – 31			34822
Month	1 – 12			34823
Year	0 – 65534			34824
Set the clock	0 – 65535	0	It is used to set the clock. When recording to this register with any value, the new clock settings in registers 34817 - 34824 will be set	34825

Table 7.1 - RS-485 Byte Formats (Register 461)

Value	Format	Description
0	EVEN	1 parity bit, 1 stop bit
1	ODD	1 parity bit, 1 stop bit
2	0 (SPACE)	1 zero bit, 1 stop bit
3	1 (MARK)	1 unit bit, 1 stop bit (similar to 2 stop bits)
4	ABSENT	No parity bit, 1 stop bit
5	AUTO-STOP	No parity bit, 2 stop bits (tx), 1 stop bit (rx)

Table 7.2 - Cloud Server Connection Modes (Register 465)

Value	Description
0	Connection to server is not used
1	Connect via any interface (prefer Ethernet)
2	Connect via any interface (prefer GSM)
3	Connect only via Ethernet
4	Connect only via GSM
5	Connect to VPN server via any interface (prefer Ethernet)
6	Connect to VPN server via any interface (prefer GSM)
7	Connect to VPN server only via Ethernet
8	Connect to VPN server only via GSM

Table 7.3 - Remote Server Connection Modes (Register 647)

Value	Description
0	TCP remote server is not used
1	Modbus TCP connection via Ethernet or GSM (prefer Ethernet)
2	Modbus TCP connection via Ethernet or GSM (prefer GSM)
3	Modbus TCP connection only via Ethernet
4	Modbus TCP connection only via GSM
5	Similar to 1 with virtual identifiers ⁴
6	Similar to 2 with virtual identifiers ⁴
7	Similar to 3 with virtual identifiers ⁴
8	Similar to 4 with virtual identifiers ⁴
9	Similar to 1 in tunnel mode
10	Similar to 2 in tunnel mode
11	Similar to 3 in tunnel mode
12	Similar to 4 in tunnel mode

Table 7.4 - Daylight Saving Time Transition Modes (Register 700)

Value	Country/Mode
0	Automatic transition not used (manual setting)
1	Brazil
2	Great Britain
3	Germany
4	Greece
5	Jordan
6	Italy
7	Namibia
8	Poland
9	Portugal
10	USA
11	Turkey
12	Ukraine
13	Finland
14	France
15	According to preset days

FAQ

Q: What is the default Modbus TCP port and device address?

A: MC252 listens on TCP port **502** (configurable in register 450) and uses Modbus address **111** by default (configurable in register 457). You can view the current IP address by pressing the R button on the device.

Q: How do I enter Setup Mode to change configuration?

A: Write the password (found on the device label) to registers 100-119. If correct, register 121 will read 1 (Setup Mode). You can then modify settings in the Changeable Settings registers (300-899).

Q: How do I save my configuration changes?

A: Write command 2 to register 120 to save (then restart MC252 for the changes to take effect), or 4 to save and apply immediately (only for Modbus settings). Write 9 to cancel unsaved changes.

Q: I have saved the new settings, how do I restart MC252 remotely?

A: Write command 1 to register 120 to restart. MC252 will reboot and usually be available again in half a minute. You may need to reconnect then.

Q: How do I configure RS-485 communication parameters?

A: Key registers are:

- **458-459**: Baud rate (default 9600)
- **461**: Byte format/parity (see Table 7.1)
- **463**: Protocol mode (0=RTU, 1=ASCII)
- **637**: Transmission mode (0=Master, 1=Slave, 2=Tunnel)

Q: What is the difference between Master and Slave RS-485 modes?

A: In **Modbus Master mode** (default), MC252 sends queries to RS-485 devices. In **Modbus Slave mode**, MC252 receives queries from an external Modbus Master on RS-485. Use Slave mode when MC252 should act as a slave device on an existing RS-485 network.

****Q:** I have configured RS-485 and set MC252 modbus device ID parameter to match my RS-485 device address. But I still get errors like `Illegal address`.

A: This can happen because MC252 (configured this way) reads its own virtual Modbus device registers instead of redirecting the requests to your RS-485 device. Do not set virtual Modbus ID in the Modbus server parameters to any of your devices addresses. However, reading registers of any device with ID other than this virtual ID (111 by default) should work.

Q: How can I connect the RS-485 device if its protocol differs from Modbus?

A: Select **Tunnel mode** for RS-485. You may also need to select the tunnel for either the MC252 server or one of the remote servers connections.

Q: How do I send Modbus commands via SMS?

A: Format: [Password] [R/W] [DeviceID] [H/I/D/C] [Address] [Value]. Example: `abc r1h100` reads holding register 100 from device 1 (password "abc"). The response SMS contains the request command without the password, and result data or error code.

Q: How do I set the device clock via Modbus?

A: First, write command 35381 to register 120 to enter Clock Setting Mode. Then write time values to registers 34819-34824 (second, minute, hour, day, month, year). Finally, write any value to register 34825 to apply. Write command 35431 to register 120 to cancel.

Q: How can I connect MC252 to multiple remote Modbus TCP servers?

A: MC252 supports up to 3 remote TCP servers. Configure each in registers 640-649 (first), 760-769 (second), and 770-779 (third). The main parameters are the connection mode and the server IP address. Also, set Modbus ID range for each server to avoid extra traffic and delays.

Q: What does the "virtual identifiers" option do for remote servers?

A: When enabled (register 647 for the first remote server, values 5-8), MC252 remaps Modbus addresses before forwarding. The range is renumbered starting from 1. For example, if range is 15-17, address 16 becomes 2 on the remote server. This helps to avoid address conflicts, when multiple gateways share address space, or when multiple Modbus TCP devices of the same model are connected.

Q: How do I reset the device to factory defaults?

A: Enter Setup Mode by writing the password, then write command 444 to register 120. All settings will be reset to factory defaults, including passwords. Restart MC252 for the changes to take effect.

Q: Why can't I read certain status registers?

A: Some registers (like GSM IP address, register 900-903) are only available after the authentication. Enter the password first. Also, the clock setting registers (34817-34825) require Clock Setting Mode.


Q: How do I check the current firmware version?



A: Read register 1 for the firmware version number.


Need Help?


For technical support and assistance:

- Email: support@overvis.com
- Support portal: www.overvis.com/support

1. The IP address of the device in the GSM network is commonly a dynamic one. To access the device via GSM using its IP address, please contact the GSM operator.   ²

2. The content of these registers is available only in Setting Mode.   ²

3. Registers for clock setting are available only in Clock Setting Mode (see registers 120, 122). 

4. In virtual identifier mode, before forwarding the request to the remote server, the virtual identifier of the addressee in the request is replaced by the real one so that numbering in the server range starts from 1. For example, for the range 15-17, virtual identifier 16 will be replaced with real 2. Broadcast identifier 0 is processed without changes. 

 ²  ³  ⁴

Memory Card Data Saving

The Overvis MC252 supports microSD compatible memory cards formatted as FAT/FAT32. Only the first volume of the card is used (the maximum usable capacity on the card is 32 GB). The card can be inserted before powering the device or while the device is in operation.

Data Loss Warning

Removing a memory card while the device is in operation may result in partial or complete loss of information on it.

Tip

The card can be removed safely when unmounted after the reset (while the device title is displayed) .

MC252 uses a memory card for the following actions:

- taskfiles operations;
- logging of collected data;
- diagnostic logging;
- export and import of settings;
- firmware updates;
- navigation on the memory card, downloading and uploading files through the WEB-interface.

When the device is started or when a memory card is inserted, its parameters are checked (it may take up to 3 seconds). After that, the card can be used for other actions.

When the device is restarted, or when the supply voltage drops below the value specified in the settings (see Modbus Register Map, register 724), MC252 safely ejects the memory card. Before unmounting, temporary data is saved and files are closed.

Taskfiles operations

MC252 performs several operations with taskfiles in the TASKS folder:

```
▼ ■ TASKS/
  ├── .txt    (Source taskfiles)
  ├── .MAP    (Token lists created at compilation)
  ├── .OBJ    (Compiled bytecode)
  └── .CNF    (Taskfiles parameters)
```

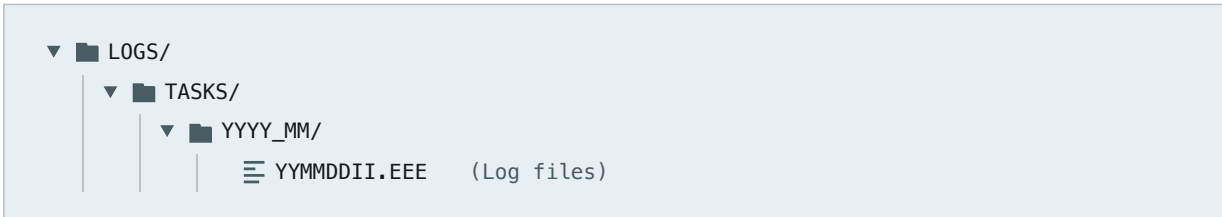
- **Reading:** MC252 reads text taskfiles from the TASKS folder (and its subfolders).
- **Compilation:**
 - Stores taskfiles found token lists into MAP text files.
 - Stores taskfiles compiled bytecode into .OBJ binary files.

- **Execution:** Reads and writes taskfiles parameters into .CNF text files.

See Logic Programming for details on files in the TASKS folder.

Logging of collected data

MC252 saves the collected data to the log in the LOGS\TASKS folder on the memory card. The order of data collection and conditions for logging are specified in the taskfiles (see Logic Programming).



If the folder is missing, it will be created. For each month, a subfolder is created with a name in the format YYYY_MM, where:

- YYYY – year;
- MM – month.

In this subfolder, for each day of the month, a file is created with the name in the format YYMMDDII.EEE, where:

- YY – the last two digits of the year;
- MM – month;
- DD – day of the month;
- II – index;
- EEE – file extension CSV or DAT (depending on the settings, see Modbus Register Map, register 725).

The data is appended to the end of the current file. A new file with the next index is created when one or more of the following conditions are met:

- the file is filled up to the maximum size (specified in the settings, from 1 KB to 64 MB, see Modbus Register Map, register 726);
- an error occurs while writing to the log file,
- the card is unmounted or removed, or the device is restarted.

File Indexing

Before creating the file, the presence of files in the subfolder with names for the given day of the month is checked, and the maximum occupied index is found. Files are numbered starting from 01.

1. Numeric indices: 01 ... 99
2. Alphanumeric indices: A0 ... A9, AA ... AZ, B0 ... ZZ

In total, up to 1035 indexes can be used for one day of the month. After that, recording is suspended until the date changes.

 **Note**

In the event of write errors, data remains in the write queue in temporary memory. Write attempts continue for up to 10 minutes. After that, data is removed from the queue, and the number of lost bytes is summed for later reporting.

Log Formats

Binary Data (.DAT) Text Table (.CSV)

When the binary log format is selected, the MC252 saves the collected data in a compact form into files with the DAT extension. Records of a fixed size of 24 bytes are appended to the files. Each record can contain the value of one parameter or a service message.

Table 1 - Format of the service record in the log data bytes file

Bytes	Field	Range of values	Description
0 – 3	Timestamp	0 – 4294967295	Time in Epoch format: number of seconds since midnight 1.01.1970 UTC+00
4 – 7	Service record code	4294967295	Indicator for distinguishing from other types of records
8 – 15	Message type	0	0 – losses due to repetitive errors
16 – 23	Message	0 – 4294967295	For loss reporting – the number of bytes

Table 2 - Format of parameter recording in the log data bytes file

Bytes	Field	Range of values	Bits	Description
0 – 3	Timestamp	0 – 4294967295		Time in Epoch format: number of seconds since midnight 1.01.1970 UTC+00
4	Parameter type	0 - 49, 128 – 177	0 – 6	Parameter type index: 6 – parameter is the bit (Modbus coil or digital input) other values – parameter is in the registers (Modbus holding or input) (see Parameter Types)
			7	0 – parameter is from read/write table (Modbus coil or holding register) 1 – parameter is from the read-only table (Modbus discrete input or input register)
5	Device ID	1 – 255		address of the Modbus device
6 – 7	Parameter address	0 – 65535		starting address of the parameter on the Modbus device
8 – 15	Parameter value	-9223372036854775808 – +9223372036854775807		The value converted to a signed 64-bit integer

Bytes	Field	Range of values	Bits	Description
16 – 23		0 - 18446744073709551615		Data read from the device before being converted to a parameter value

Filling the memory card

The time it takes for an empty memory card to be filled, can be calculated using the formula:

$$T_{full} \approx (V_{free} * T_{upd}) / (N_{par} * L_{siz})$$

, where: T_{full} – the time till the card is full; V_{free} – free space on the memory card; T_{upd} – task run (parameters update) period; N_{par} – the number of parameters to be logged (if the log record is made with each task run); L_{siz} – the size of the log entry (depends on its format).

Examples:

1. **Binary data format:** Writing 7 parameters every 20 seconds. A 2 GB card will be filled in ~383,479,222 s (or more than 12 years).
2. **Text table format:** Writing 3 parameters 10 times per second (without comments). A 32 GB card can be filled in ~16,361,780 s (or more than 6 months).

Tip

When the **recorder mode** is used (see Modbus Register Map, register 727), after the free space on the memory card is exhausted, the oldest files can be deleted before writing new data. When the **recorder mode is off**, new data logging will be paused until space becomes available.

Diagnostic logging

MC252 can be configured to store diagnostic logs into the LOGS folder. See Web Interface for configuring the logged events.

```

▼ ■ LOGS/
  |
  |  ATM.LOG   (GSM AT-modem exchange log)
  |  SYS.LOG   (System events log)
  |
  
```

Export and import settings

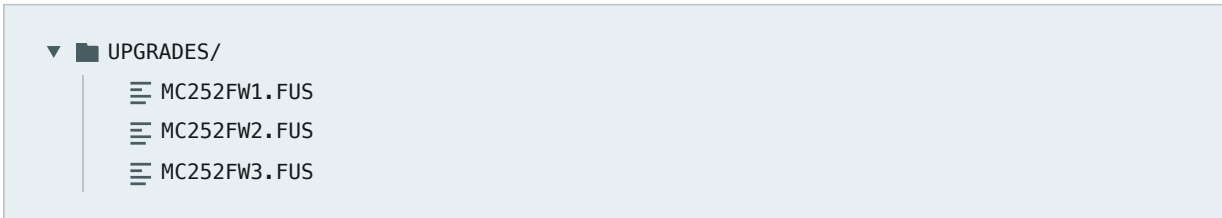
MC252 can export the saved settings from the internal memory to the SETTINGS\MC252SET.DAT file, or import the settings from this file and save them to the internal memory.

```

▼ ■ SETTINGS/
  |
  |  MC252SET.DAT  (Settings file, up to 16 kB)
  |
  
```

Firmware updates

MC252 can update the firmware (see Firmware Update) with one of three files:



The file size can be up to 10 MB each in size.

FAQ

Q: What type of memory card does MC252 support?

A: MC252 supports microSD compatible memory cards formatted as FAT or FAT32. Only the first volume is used, with a maximum usable capacity of 32 GB.

Q: Can I insert or remove the memory card while the device is running?

A: You can insert the card while the device is running. However, removing the card during operation may result in data loss. To safely remove the card, wait until after a device reset while the device title is displayed on screen.

Q: What happens if the power supply drops while the card is in use?

A: MC252 monitors supply voltage and safely unmounts the memory card when voltage drops below a configurable threshold (register 724). Before unmounting, temporary data is saved and files are closed to prevent corruption. A high-capacitive power supply is recommended to give enough time for this feature.

Q: Which log format should I use — binary (.DAT) or text (.CSV)?

A:

- **Binary (.DAT):** More compact (24 bytes per record), better for long-term logging with limited card space, but requires parsing tools to read.
- **Text (.CSV):** Human-readable, can be opened directly in spreadsheet software, but takes more space (up to 200 bytes per record with comments).

Q: How long will it take to fill my memory card with log data?

A: Use the formula: $T_{full} \approx (V_{free} * T_{upd}) / (N_{par} * L_{siz})$. For example, logging 7 parameters every 20 seconds in binary format on a 2 GB card takes over 12 years to fill. Logging 3 parameters 10 times per second in CSV format on a 32 GB card takes about 6 months.

Q: What happens when the memory card is full?

A: It depends on the **recorder mode** setting (register 727). When enabled, the oldest files are automatically deleted to make room for new data. When disabled, logging pauses until you manually free up space.

Q: How much data can be logged per day?

A: The file size limit is configurable (up to 64Mb). And up to 1035 files per day can be created in the day folder. After all file indices are used, recording is suspended until the date changes.

Q: What happens if a write error occurs during logging?

A: Data remains in a temporary write queue and the device retries for up to 10 minutes. If writing still fails, the data is removed from the queue and the number of lost bytes is recorded for reporting in subsequent log entries.

Q: Can I use the same memory card for multiple MC252 devices?

A: Yes, but be careful with taskfiles — each device will try to load and execute taskfiles from the TASKS folder. Settings export/import files are also shared. Consider using separate cards or carefully managing folder contents if moving cards between devices.

Q: Where are taskfile compilation results stored?

A: In the same TASKS folder alongside your source files:

- .MAP files contain token lists from compilation
- .OBJ files contain compiled bytecode
- .CNF files store taskfile runtime parameters

Q: How do I back up my MC252 settings to the memory card?

A: Export settings to the SETTINGS\MC252SET.DAT file using Modbus commands. This file (up to 16 KB) can be imported later to restore settings or transfer them to another device.

Q: Why does the SD card take up to 3 seconds to become available after insertion?

A: MC252 performs parameter checks on the card after insertion or device startup. This verification ensures the card is properly formatted and usable.

Need Help?




For technical support and assistance:

- Email: support@overvis.com
- Support portal: www.overvis.com/support

HTTP API

This HTTP API provides an interface for interacting with the Overvis MC252 device. All requests are made to paths starting with `/api`. The API supports GET and POST methods, some of which require authorization.

General Principles


- All data is transferred in JSON format.
- Successful responses use the `200 OK` status.
- Errors are handled with appropriate HTTP statuses (`400 Bad Request`, `404 Not Found`, `503 Service Unavailable`, etc.).
- API request errors additionally contain a JSON object: `{ "error": "error code", "message": "error text" }`.
- HTTP keep-alive is supported.
- Access to the API uses **3** access levels:
 -  **0** - Guest
 -  **1** - Power User
 -  **2** - Administrator

Authorization


Authorization via API

To perform protected requests, you must complete the authorization procedure:

1. Get a salt at `/api/login/salt/`.
2. Log in via `/api/login/` by sending a SHA1 password hash: `login + password + salt`.
3. All subsequent requests must include the session token `Bearer xxxxxxxxxx` in the `Authorized` header.

Logout is performed via `/api/cmd/logout/`. After successful authorization, the user receives access level  **1** (Power User).

Authorization via Front Panel Button


Press the button on the front panel of the MC252 device once. After this, a user with **Guest** rights temporarily receives access level  **1** (Power User) **for 10 minutes**.









Authorization via VPN Cloud

All requests coming from the **VPN cloud** automatically receive access level  **2** — Administrator.

















Endpoints Summary

GET Requests

Path	Purpose	Access Level
<code>/api/version/</code>	Device firmware version	 0

Path	Purpose	Access Level
/api/login/salt/	Get salt for password hashing	 0
/api/identify/	Physical device identification	 1
/api/state/get/?key&...	Get current device state by key (or all)	 1
/api/settings/active/get/?key&...	Get active settings by key (or all)	 1
/api/settings/saved/get/?key&...	Get saved settings by key (or all)	 1
/api/modbus/resp/?tid=...	Get Modbus response by tid	 1
/api/modbus/find/result/?list	Get Modbus device search result	 1
/api/modbus/class-find/result/	Get Modbus template search result	 1

POST Requests

Path	Purpose	Access Level
/api/login/	User authentication	 0
/api/cmd/logout/	End session	 1
/api/cmd/memory/card_eject/	Safe SD card ejection	 1
/api/cmd/time/sync/	Time synchronization	 1
/api/cmd/time/set/	Set time	 1
/api/cmd/reboot/	Reboot device	 1
/api/cmd/callback/reconnect/	Reconnect to cloud server via Modbus	 1
/api/cmd/updates/download/	Download available updates	 1
/api/cmd/updates/program/	Apply update	 1
/api/cmd/settings/reset/	Reset settings to factory defaults	 1
/api/cmd/settings/import/	Import configuration	 1
/api/cmd/settings/export/	Export current settings	 1
/api/cmd/gsm/sms-test/	Send test SMS	 1
/api/activation/	License or device activation	 1
/api/card/file/info/	Get file info on SD card	 1
/api/card/file/read/	Read file from SD card	 1

Path	Purpose	Access Level
/api/card/file/write/	Write file to SD card	■ 1
/api/card/remove/	Remove file from SD card	■ 1
/api/card/dir/	Get list of files and directories	■ 1
/api/settings/saved/set/	Change configuration parameters	■ 1
/api/modbus/req/	Send Modbus request	■ 1
/api/modbus/find/start/	Start Modbus device search on RS-485	■ 1
/api/modbus/find/stop/	Stop/cancel Modbus device search on RS-485	■ 1
/api/modbus/class-find/search-meta-info/	Get meta info from search template	■ 1
/api/modbus/class-find/start/	Start template Modbus device search	■ 1
/api/modbus/class-find/stop/	Stop/cancel template Modbus device search	■ 1

Authentication

Get Salt for Password Hashing

Provides a unique salt string used by the client to hash the password before authentication.

GET /api/login/salt/

Response:

Field	Type	Description
salt	string	Unique salt string for password hashing.

Example Response:

```
{
  "salt": "sFtqKto6hnaURPGQLJOCT1QMx7myDEGLBsCz3a2L"
}
```

User Authentication

Allows a user to authenticate and receive an access token.

POST /api/login/

Request Parameters:

Field	Type	Description
username	string	Username.
password	string	SHA1 hash of the password in hex format.

Note: SHA1 password hash is calculated as $SHA1(\text{username} + \text{password} + \text{salt})$.

Example Request:

```
{
  "username": "admin",
  "password": "90d54ed4126a0924528810aa5673a6d616d5f274"
}
```

Response:

Upon successful authentication, a token and its expiration time are returned.

Field	Type	Description
authType	string	Authentication type (Bearer).
token	string	Access token.
ttlSec	integer	Token time-to-live in seconds.

```
{
  "authType": "Bearer",
  "token": "ej5k2pVg4Pd8yBdUqFPq8bdaStpeAZhyelpmkht0ivdK8r7E",
  "ttlSec": 60
}
```

Logout

Ends the current user session and invalidates the token.

POST /api/cmd/logout/

Example Request:

```
{}
```

Response:

Empty JSON upon successful command execution.

```
{}
```

System Information

Physical Device Identification

Returns unique information about the current device, including model, firmware version, MAC address, and public key.

GET /api/identify/

Access Level: ■ 1

Response:

Field	Type	Description
manufacturer	string	Device manufacturer.
device	string	Device model name.
deviceId	integer	Device identifier.
firmwareType	integer	Firmware type (e.g., basic, extended, etc.).
firmwareVer	integer	Firmware version.
releaseDate	string	Release date and time in ISO 8601 format.
mac	string	Ethernet interface MAC address.
uniqueMac	boolean	Flag indicating if the device MAC address is unique.
pubKey	string	Device public key for cryptographic authentication.

Example Response:

```
{
  "manufacturer": "Novatek-Electro Ltd.",
  "device": "MC-251",
  "deviceId": 46,
  "firmwareType": 1,
  "firmwareVer": 46,
  "releaseDate": "2025-05-26T00:00:00",
  "mac": "d8:80:39:69:0e:6d",
  "uniqueMac": true,
  "pubKey": "WB2dxH7Jk+tmW2TDw0mWxtBKjkKf4siJi42bDyCLLXQ="
}
```

Firmware Version

GET /api/version/

Access Level: ■ 0

Response:

Field	Type	Description
device	string	Manufacturer's device model.
deviceId	integer	Manufacturer's device identifier.
firmwareType	integer	Firmware modification.
firmwareVer	integer	Firmware version.
releaseDate	string	Release date and time in ISO 8601 format.

Example Response:

```
{
  "device": "MC-251",
  "deviceId": 46,
  "firmwareType": 1,
  "firmwareVer": 46,
  "releaseDate": "2025-05-14T00:00:00"
}
```

Get Device State

Retrieves the current state of the device by key (or all states).

GET /api/state/get/?key&...

Access Level: ■ 1

Available Keys:

- ethernet: Ethernet interface state
- gsm: GSM interface state
- vpn: VPN interface state
- mbTcpCallback: Modbus TCP cloud connection state
- activation: Device activation state
- fwUpdate: Update state
- mbTcpServer: TCP server state
- mbTcpClients: TCP client state
- inputs: Inputs state
- outputs: Outputs state

- `statistics`: General statistics
- `memTaskfiles`: Task files state
- `memCard`: SD card state
- `settings`: Settings state
- `time`: Device time
- `misc`: Miscellaneous states

ethernet

Returns information about the current Ethernet connection state.

Response Fields:

Field	Type	Description
<code>ip</code>	string / null	Current IP address assigned to the device, or null if not assigned.
<code>mac</code>	string	Ethernet interface MAC address.
<code>linkUp</code>	boolean	Indicates if there is an active link.

gsm

Returns information about the current GSM module state.

Response Fields:

Field	Type	Description
<code>signalLevel</code>	integer / null	GSM signal level in %, or null if GSM is not connected.
<code>ip</code>	string / null	IP address assigned to the GSM interface, or null if not assigned.
<code>imei</code>	string / null	Modem IMEI, or null if undefined.
<code>simState</code>	integer	SIM card state: 0 — not inserted, 1 — inserted and active.
<code>pinState</code>	integer	PIN code state:
		0 - Unknown state;
		1 - PIN not required;
		2 - PIN required, attempts unknown;
		3 - PIN required, 0 attempts left;
		4 - PIN required, 1 attempt left;
		5 - PIN required, 2 attempts left;

Field	Type	Description
		6 - PIN required, 3 attempts left;
		7 - PIN required, >3 attempts left;
		8 - Other auth required (PUK, etc.).
ccid	string / null	SIM card CCID, or null if undefined.
provider	string / null	SIM card operator name, or null if undefined.
radioBand	string / null	GSM standard: 2g, 3g, 4g, or null if undefined.
isTcpIpBusy	boolean	Indicates if the GSM interface TCP/IP stack is busy.
isGsmRegistered	boolean	Indicates registration in the operator's network.
isGsmRoaming	boolean	Indicates roaming status.
isGprsStarted	boolean	Indicates active GPRS connection.
trafficTotal	integer	Total data transferred via GSM interface (in bytes).

vpn

Returns information about the current VPN (WireGuard) connection state.

Response Fields:

Field	Type	Description
linkState	string	Connection state:
		disabled - VPN disabled in settings;
		netifCreated - Network interface created;
		peerConfigured - Peer connection configured;
		peerConnected - Connection to peer established;
		peerUp - VPN tunnel is up and running.
configServerReached	boolean / null	Configuration server availability.
lastPeerUp	string / null	Time of last peer connection (ISO 8601), or null.
netifIp	string / null	VPN interface IP address, or null.
netifMask	string / null	VPN interface subnet mask, or null.
peerIp	string / null	Remote peer IP address, or null.

Field	Type	Description
peerPort	integer	Remote peer port.
interface	string / null	Connection interface: eth, gsm, or null.
configState	string	Configuration state:
		errorKey - Error: invalid key;
		errorMem - Error: insufficient memory;
		disabled - VPN disabled in settings;
		pause - Delay between configuration attempts;
		serverHostnameResolving - Resolving config server hostname;
		serverConnecting - Connecting to config server;
		serverConnected - Connected, receiving peer config;
		peerHostnameResolving - Resolving peer hostname;
		ready - Configuration received and ready.
ownPubKey	string	Device's own public key.

mbTcpCallback

Modbus TCP cloud connection state.

Response Fields:

Field	Type	Description
state	string	Current connection state (disabled, waiting, connecting, connected).
waitingTimeSec	integer / null	Time waiting for connection, in seconds.
interface	string / null	Connection interface: eth, gsm, wg, or null.
remoteIp	string / null	Server IP address, or null.
remotePort	integer	Server remote port.
connectionTimeSec	integer	Time since connection established, in seconds.
idleTimeSec	integer	Connection idle time, in seconds.
activationWithCode	string / null	Activation state:
		unused - Device has not received activation code;

Field	Type	Description
		activated - Device is activated and bound to network;
		set - Device received code and awaits binding;
		null - No connection or error.
activationCode	string	8-digit device activation code.

activation

Returns information about the current device activation state.

Response Fields:

Field	Type	Description
infrastructureName	string	Infrastructure name.
infrastructureShortUrl	string	Short URL to infrastructure site.
domain	string	Server address used for activation.
endpointId	string	Unique device identifier.
endpointName	string	Device name.
propertyId	string	Organization identifier.
propertyName	string	Organization name.
pin	string	8-digit device PIN code.
hasBonusesEur	string	Bonus amount on organization account.
activeTill	string / null	Activation expiration date (ISO 8601), or null if expired.
supportEmail	string	Support service email.

fwUpdate

Firmware update process state.

Response Fields:

Field	Type	Description
updateState	string	Update state:
		SAVED_FILE_UNKNOWN - Update file not found;

Field	Type	Description
		CARD_ABSENT - SD card not installed;
		CARD_NOT_READY - SD card not ready;
		CARD_FILES_CHECKING - Checking update files on SD card;
		CARD_FILE_UPDATED_READY - Update file on SD card ready;
		FILE_VERSION_MATCHES_ACTIVE - File version matches current.
downloadState	string	Download state:
		SERVER_FILE_UNKNOWN - File on server not determined;
		DOWNLOADER_BUSY - Downloader busy;
		CARD_ABSENT - SD card not installed;
		CARD_NOT_READY - SD card not ready;
		CARD_FILES_CHECKING - Checking files on SD card;
		SERVER_FILE_UPDATED_READY - File on server ready for download;
		SERVER_FILE_DOWNLOADING - Downloading file;
		DOWNLOADED_FILE_CHECKING_INTEGRITY - Checking integrity;
		DOWNLOADED_FILE_CHECKING_APPLICABILITY - Checking compatibility;
		DOWNLOADED_FILE_SAVING - Saving file to SD card;
		SERVER_FILE_VERSION_MATCHES_SAVED - Server version matches saved.
url	string / null	Firmware file URL or null.
progress	string / null	Download progress percentage (0.00 – 100.00) or null.
errorCode	string / null	Error code if any, or null. Common codes:
		ERROR_FR_DISK_ERR - Low-level disk error;
		ERROR_FR_INT_ERR - Internal file system error;
		ERROR_FR_NOT_READY - Media not ready;
		ERROR_FR_NO_FILE - File not found;
		ERROR_FR_DENIED - Access denied;
		ERROR_FR_EXIST - File already exists;

Field	Type	Description
		ERROR_FR_WRITE_PROTECTED - Write protected;
		ERROR_FR_NOT_ENOUGH_CORE - Not enough memory;
		ERROR_HTTP_RESPONSE - General HTTP error;
		ERROR_CONNECT - TCP connection error;
		ERROR_RESPONSE_TIMEOUT - Server timeout.
availableVersion	integer / null	Firmware version available for download, or null.
savedVersion	integer / null	Firmware version saved on SD card, or null.
currentVersion	integer	Currently installed firmware version.

mbTcpServer

List of active Modbus TCP server connections.

Response Fields:

Field	Type	Description
isActive	boolean	Indicates if the server is running.
listenPortEth	integer	Port for Ethernet clients.
listenPortGsm	integer	Port for GSM clients.
clients	array	Array of client objects (interface, remoteIp, remotePort, etc.).

mbTcpClients

State of Modbus TCP clients.

Response Fields:

Field	Type	Description
state	string	Current state (waiting, connected, connecting).
waitingTimeSec	integer / null	Time waiting for connection, in seconds.
interface	string / null	Connection interface.
remoteIp	string	Client IP address.
remotePort	integer	Client remote port.

Field	Type	Description
connectionTimeSec	integer	Time since connection established, in seconds.
idleTimeSec	integer	Connection idle time, in seconds.

inputs

Example Request:

GET /api/state/get/?inputs

Example Response:

```
{
  "inputs": {
    // TODO: Not implemented
  }
}
```

outputs

Example Request:

GET /api/state/get/?outputs

Example Response:

```
{
  "outputs": {
    // TODO: Not implemented
  }
}
```

statistics

Device operation statistics.

Response Fields:

Field	Type	Description
serialAvgPerSecReq	integer	Average RS-485 request rate (req/sec).
serialAvgPerSecResp	integer	Average RS-485 response rate (resp/sec).
serialAvgPerSecLoad	integer	Average RS-485 load (%/sec).
mbAvgPerSecReq	integer	Average Modbus request rate (req/sec).

Field	Type	Description
ethAvgPerSecKb	string	Average Ethernet data rate (KB/sec).
gsmAvgPerSecKb	string	Average GSM data rate (KB/sec).
mbTcpServerMaxClients	integer	Maximum number of clients on Modbus TCP server.
runTimeMin	integer	Uptime since last reboot (minutes).
totalRunTimeMin	integer	Total device uptime (minutes).

memTaskfiles

Task files and programmable logic state information.

Example Request:

```
GET /api/state/get/?memTaskfiles
```

Response Fields:

Field	Type	Description
filesFound	integer / null	Number of task files found on the SD card, or null if undetermined.
loadedOn	string / null	Date and time when task files were loaded (ISO 8601), or null.
heapUsedB	integer / null	Memory used for task execution in bytes, or null.
errorFile	string / null	File where the first error was found, or null if there are none.
errorLine	integer / null	Line number of the first error, or null if there are none.
errorCode	string / null	Task file compilation error code, or null if there are none.
tasks	array	Array of loaded task objects.

Compilation Error Codes:

Code	Description
ERROR_IO	I/O error while processing a task file.
ERROR_OUT_OF_MEMORY	Not enough memory to process a task file.
ERROR_NO_PATH	Cannot open the task files directory.
ERROR_NO_FILE	Cannot open a task file.
ERROR_LINE_UNRECOGNIZED	Expected an explicit line type such as DEF, PUT, IF, or DO.

Code	Description
ERROR_ITEM_NAME	Name is too long or invalid.
ERROR_UNKNOWN_TYPE	Too many elements or the line type cannot be determined.
ERROR_EXPECTED_ASTERISK	Unterminated string escape sequence.
ERROR_ITEM_REF	Invalid item reference.
ERROR_ILLEGAL_ARGUMENT	Immediate value is outside the allowed range.
ERROR_EXPECTED_SEPARATOR	More elements expected, or unexpected item name found.
ERROR_VALUE_BITSIZE_UNSUPPORTED	This value bit size is not supported.
ERROR_EXPECTED_DOUBLE_QUOTES	Unterminated string literal.
ERROR_EXPECTED_EOL	Task file line is too long.
ERROR_VERSION_UNSUPPORTED	Task file version is not supported.
ERROR_CRC	Task data corruption detected (CRC mismatch).
ERROR_UID_BROADCAST	Broadcast UID 0 cannot be used for reading.
ERROR_FILE_FORMAT_UNSUPPORTED	Unsupported file format. Try UTF-8 or ASCII.
ERROR_NAME_RESERVED	Reserved word used as an item name.
ERROR_NAME_ALREADY_DEFINED	Duplicate item names.
ERROR_UNSATISFIED_FORWARD_DECLARATION	Reference to a missing item.
ERROR_STRING_FUNCTION_ARGUMENT_TOO_SHORT	String function argument is too short.
ERROR_VALUE_FORMAT	Invalid immediate value format.
ERROR_ILLEGAL_ARRAY_INDEX	Invalid array index.
ERROR_ARRAY_INDIRECT_INDEX_NOT_ALLOWED_HERE	Indirect array index is not allowed here.
ERROR_ITEM_UNDECLARED	Item is not declared.
ERROR_OUT_OF_PARAMETER_STORAGE_MEMORY	Not enough memory for parameter storage.
ERROR_ILLEGAL_ARRAY_SIZE	Invalid array size.
ERROR_UNEXPECTED_VARIABLE_REF	Unexpected variable reference.
ERROR_UNEXPECTED_CONDITION_REF	Unexpected condition reference.
ERROR_INDEX_BITSIZE_UNSUPPORTED	This index bit size is not supported.

Code	Description
ERROR_EXPECTED_CLOSING_BRACKET	Closing bracket expected.
ERROR_UNKNOWN_STRING_FUNCTION	Unknown string function.
ERROR_LOGGING_UNMAPPED_PARAMETER	Logging an unmapped parameter.
ERROR_MAPPED_PARAMETER_INTERSECT_PARTIALLY	Mapped parameters partially overlap.
ERROR_MULTIPLE_WRITERS_FOR_MAPPED_PARAMETER	Multiple writers for the same mapped parameter.
ERROR_OBSOLETE_FILE_LAYOUT	Obsolete task file layout.

Task Object Fields:

Field	Type	Description
sourceFile	string	File from which the task was loaded.
periodMs	integer	Task cycle period in milliseconds.
state	string	Current task state: idle or running.
lastRunOn	string / null	Time of the last task cycle run (ISO 8601), or null.
lastErrorCode	string / null	Task execution error code, or null if there is none.

Task Execution Error Codes:

Code	Description
EXC_MB_ILLEGAL_FUNCTION	Modbus: illegal function.
EXC_MB_ILLEGAL_DATA_ADDRESS	Modbus: illegal data address.
EXC_MB_ILLEGAL_DATA_VALUE	Modbus: illegal data value.
EXC_MB_SLAVE_DEVICE_FAILURE	Modbus: slave device failure.
EXC_MB_ACKNOWLEDGE	Modbus: request acknowledged.
EXC_MB_SLAVE_BUSY	Modbus: slave device busy.
EXC_MB_MEMORY_PARITY_ERROR	Modbus: memory parity error.
EXC_PARAM_PATH_UNAVAILABLE	Device unavailable; request not sent.
EXC_PARAM_TARGET_FAILED	Device did not respond.
EXC_TYPE_UNKNOWN	Unknown parameter type.

Code	Description
EXC_ACTION_UNKNOWN	Unknown action type.
EXC_I0	External resource error.
EXC_OUT_OF_MEMORY	Not enough memory.
EXC_STACK_OVERFLOW	Stack overflow (too many CALL).
EXC_STACK_UNDERFLOW	Stack underflow (RETURN without CALL).
EXC_RESULT_INFINITY	Result is infinity (division by zero).
EXC_RESULT_COMPLEX	Result is complex (SQRT of negative).
EXC_OUT_OF_BOUNDS	Index out of bounds.
EXC_UNKNOWN	Unknown error.

Example Response:

```
{
  "memTaskfiles": {
    "filesFound": 2,
    "loadedOn": "2026-01-12T15:17:31Z",
    "heapUsedB": 814,
    "errorFile": "/TASKS/CLOCKS.TXT",
    "errorLine": 3,
    "errorCode": "ERROR_UNSATISFIED_FORWARD_DECLARATION",
    "tasks": [
      {
        "sourceFile": "/TASKS/INIT.TXT",
        "periodMs": 60000,
        "state": "idle",
        "lastRunOn": "2026-01-12T15:18:31Z",
        "lastErrorCode": null
      }
    ]
  }
}
```

memCard

SD card state information.

Response Fields:

Field	Type	Description
capacityKb	integer / null	Total SD card capacity in KB, or null.
freeKb	integer / null	Free space on SD card in KB, or null.
isMounted	boolean	Mount status (true — mounted).
isInserted	boolean	Insertion status (true — inserted).

settings

Settings state information.

Response Fields:

Field	Type	Description
modifyOn	string / null	Last settings modification time (ISO 8601), or null.
isUserModified	boolean	Indicates if settings are modified but not applied.
isDefault	boolean	Indicates if current settings are defaults.

time

Current device time information.

Response Fields:

Field	Type	Description
timeLocal	integer	Current local time (ISO 8601).
timeUtc	integer	Current UTC time (ISO 8601).
isDst	boolean	DST flag.
sunriseLocalSec	integer / null	Sunrise time in seconds from start of day, or null.
sunsetLocalSec	integer / null	Sunset time in seconds from start of day, or null.
polarDay	boolean	Indicates polar day.
polarNight	boolean	Indicates polar night.

misc

Additional diagnostic data.

Response Fields:

Field	Type	Description
voltage	string / null	Current supply voltage, or null.
temperature	string / null	Current temperature in Celsius, or null.

Settings

Get Settings

Read settings from active or saved area.

- Active area: Current settings the device is working with.
- Saved area: Settings saved in non-volatile memory, applied at next startup.

GET /api/settings/active/get/?key1&key2... **GET** /api/settings/saved/get/?key1&key2...

Access Level:  1

Example Response:

```
{
  "key1": "value1",
  "key2": "value2"
}
```

Set Settings

Write settings to the saved area.

POST /api/settings/saved/set/

Access Level:  1

Example Request:

```
{
  "key": "value"
}
```






















Response:

Empty JSON upon success.

```
{}
```

Available Settings Keys

The following keys can be used with GET and POST requests for settings.

Key	Description	Access Level
timeGmt	GMT time offset correction	 1
ethernet	Ethernet interface settings	 1
gsm	GSM module settings	 1
ownSrv	Connection to own server	 1
wgCloud	Cloud VPN connection settings	 1
wgManual	Custom WireGuard VPN parameters	 1
cbCloud	Cloud Modbus connection settings	 1
areset	Automatic reset settings	 1
protect	Device protection parameters	 1
download	File server settings	 1
firmware	Firmware management	 1
httpd	Built-in HTTP server parameters	 1
language	Interface language	 1
serial	Serial port settings	 1
astro	Astronomical settings (sunrise/sunset)	 1
ntp	NTP time synchronization settings	 1
tsMode	Daylight saving time mode	 1
tsManual	Manual daylight saving time settings	 1
logFlags	Logging flags	 1
rmSrv	Remote control server	 1
forwarding	Port forwarding table	 1
tasksLogging	Programmable logic data logging settings	 1
powerSaving	Power and resource saving settings	 1

timeGmt

Time zone management (GMT offset).

Request Parameters:

Parameter	Type	Description
timeGmt	integer	GMT offset in 15-minute intervals (from -96 to +96).

Example Request:

```
{
  "timeGmt": 8
}
```

Note: 8 means +2 hours offset (8 × 15 minutes).

ethernet

Ethernet interface settings.

Request Parameters:

Parameter	Type	Description
ethernet	object	
dhcpEnable	boolean	true — enable DHCP on Ethernet interface; false — disable.
macManualEnable	boolean	true — use custom MAC address.
dnssGateEnable	boolean	true — use gateway address as primary DNS.
ip	string	Ethernet interface IP address.
mask	string	Ethernet interface subnet mask.
gate	string	Ethernet interface gateway.
ipDnss	array	Array of DNS server IP addresses for Ethernet .
macManual	string	Custom MAC address for Ethernet interface.

Example Request:

```
{
  "ethernet": {
    "dhcpEnable": true,
    "macManualEnable": false,
    "dnssGateEnable": true,
    "ip": "192.168.0.115",
    "mask": "255.255.255.0",
    "gate": "192.168.0.1",
    "ipDnss": ["8.8.8.8", "8.8.4.4"],
    "macManual": "01:02:03:04:05:06"
  }
}
```

gsm

GSM module settings.

Request Parameters:

Parameter	Type	Description
gsm	object	
apnAutodetectEnable	boolean	true — automatically configure APN settings based on SIM card code.
tcpInRoamingEnable	boolean	true — allow TCP/UDP data transfer in roaming.
smsInRoamingEnable	boolean	true — allow SMS sending in roaming. (SMS reception is always allowed).
pinCode	integer / null	PIN code (0000-9999), or null if not set.
apnHost	string	APN host name for manual configuration (max 34 chars).
apnLogin	string	APN login for manual configuration (max 40 chars).
apnPassword	string	APN password for manual configuration (max 24 chars).
balanceUssdReq	string	USSD request for balance check (max 12 chars, e.g., *111#).
phones	array	Array of 4 phone number strings.

Example Request:

```
{
  "gsm": {
    "apnAutodetectEnable": true,
    "tcpInRoamingEnable": true,
    "smsInRoamingEnable": false,
    "pinCode": null,
    "apnHost": "",
    "apnLogin": "",
    "apnPassword": "",
    "balanceUssdReq": "*111#",
    "phones": ["+48123456789", "", "", ""]
  }
}
```

ownSrv

Connection to own server.

Request Parameters:

Parameter	Type	Description
ownSrv	object	
vUid	integer / null	Virtual Modbus device address (1-255), or null to disable.
mbeRo	integer / null	Modbus exception code for access denied (1-255), or null to disable response.
mbeTo	integer / null	Modbus exception code for timeout (1-255), or null to disable response.
mbePu	integer / null	Modbus exception code for send error (1-255), or null to disable response.
ethListenPort	integer	TCP server port on Ethernet interface (1-65535).
delayForKeepaliveSec	integer / null	TCP server keep-alive time (0-36000 sec), or null to disable.
atmListenPort	integer / null	TCP server port on GSM interface (1-65535), or null to disable.
ethClientsProtocol	integer	Data protocol for Ethernet TCP server: 1 - RAW, 2 - ModbusTCP.
atmClientsProtocol	integer	Data protocol for GSM TCP server: 1 - RAW, 2 - ModbusTCP.

Example Request:

```
{
  "ownSrv": {
    "vUId": 111,
    "mbeRo": 1,
    "mbeTo": 11,
    "mbePu": 10,
    "ethListenPort": 502,
    "delayForKeepaliveSec": 90,
    "atmListenPort": null,
    "ethClientsProtocol": 2,
    "atmClientsProtocol": 2
  }
}
```

wgCloud

Cloud VPN connection settings.

Request Parameters:

Parameter	Type	Description
wgCloud	object	
hostname	string	VPN server hostname (max 46 chars).
connPort	integer	VPN server TCP port (1-65535).
connMode	integer / null	Connection mode: null (disabled), 1 (Eth pref), 2 (GSM pref), 3 (Eth only), 4 (GSM only).
delayForKeepaliveSec	integer	VPN server keep-alive time (0-36000 sec).
delayForConnSec	integer	Delay between connection attempts (1-30000 sec).

Example Request:

```
{
  "wgCloud": {
    "hostname": "",
    "connPort": 34967,
    "connMode": 1,
    "delayForKeepaliveSec": 20,
    "delayForConnSec": 5
  }
}
```

wgManual

Custom WireGuard VPN settings.

Request Parameters:

Parameter	Type	Description
wgManual	object	
hostname	string	VPN server hostname (max 46 chars).
connPort	integer	VPN server TCP port (1-65535).
connMode	integer / null	Connection mode: null (disabled), 1 (Eth pref), 2 (GSM pref), 3 (Eth only), 4 (GSM only).
ip	string	Device IP address in VPN network.
mask	string	Device subnet mask.
deviceKey	string	Device private key (base64).
peerPublicKey	string	Remote peer public key (base64).

Example Request:

```
{
  "wgManual": {
    "hostname": "",
    "connPort": 34967,
    "connMode": 1,
    "delayForKeepaliveSec": 20,
    "delayForConnSec": 5,
    "deviceIp": "10.0.0.111",
    "deviceMask": "255.255.255.0",
    "deviceKey": "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA=",
    "peerPublicKey": "AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA="
  }
}
```

cbCloud

Cloud Modbus connection settings.

Request Parameters:

Parameter	Type	Description
cbCloud	object	
hostname	string	Server hostname (max 46 chars).
connPort	integer	Server TCP port (1-65535).
connMode	integer / null	Connection mode: null (disabled), 1 (Eth pref), 2 (GSM pref), 3 (Eth only), 4 (GSM only).
delayForKeepaliveSec	integer	Server keep-alive time (0-36000 sec).
delayForConnSec	integer	Delay between connection attempts (1-240 sec).

Example Request:

```
{
  "cbCloud": {
    "hostname": "",
    "connPort": 8502,
    "connMode": 1,
    "delayForKeepaliveSec": 120,
    "delayForConnSec": 5
  }
}
```

areset

Automatic reset settings.

Request Parameters:

Parameter	Type	Description
areset	object	
delayMin	integer	Time before automatic restart (5-7200 min).
mode	integer / null	null - Disabled; 1 - Reset after delayMin uptime; 2 - Reset after delayMin of no Modbus activity.

Example Request:



```
{
  "areset": {
    "delayMin": 120,
    "mode": 2
  }
}
```

protect

Device protection parameters.

Request Parameters:

Parameter	Type	Description
protect	object	
smsWriteEnable	boolean	false - disable SMS write.
smsReadEnable	boolean	false - disable SMS read.

Parameter	Type	Description
mbGateWriteEnable	boolean	false - disable write to connected Modbus devices.
mbGateReadEnable	boolean	false - disable Modbus read.
wgCloudAdminAuthEnable	boolean	true - grant Admin ( 2) access to all Cloud requests.
wgManualAdminAuthEnable	boolean	true - grant Admin ( 2) access to all WireGuard VPN requests.
passwordAdmin	string	Device administrator password (5-10 ASCII chars, no spaces).
passwordMbGateWrite	string	Password for writing to connected Modbus devices (0-10 chars). "" disables check.
passwordMbRead	string	Password for Modbus read (0-10 chars). "" disables check.
passwordSmsWrite	string	Password for SMS write (3-10 chars).
passwordSmsRead	string	Password for SMS read (3-10 chars).

Example Request:

```
{
  "protect": {
    "smsWriteEnable": true,
    "smsReadEnable": true,
    "mbGateWriteEnable": true,
    "mbGateReadEnable": true,
    "wgCloudAdminAuthEnable": true,
    "wgManualAdminAuthEnable": false,
    "passwordAdmin": "11111",
    "passwordMbGateWrite": "",
    "passwordMbRead": "",
    "passwordSmsWrite": "gap",
    "passwordSmsRead": "gap"
  }
}
```

download

File server settings.

Request Parameters:

Parameter	Type	Description
download	object	
connMode	integer / null	Connection mode: null (disabled), 1 (Eth pref), 2 (GSM pref), 3 (Eth only), 4 (GSM only).
loadRatio	integer	Download speed limit (1-50 % of max bandwidth).
delayForConnSec	integer	Delay between connection attempts (1-30000 sec).

Example Request:

```
{
  "download": {
    "connMode": 1,
    "loadRatio": 15,
    "delayForConnSec": 60
  }
}
```

firmware

Firmware management.

Request Parameters:

Parameter	Type	Description
firmware	object	
checkPeriodSec	integer / null	Update check period (1-1000000 sec), or null to disable.
downloadPeriodHr	integer / null	Automatic download period (0-1439 hours), or null to disable.
updateHour	integer / null	Hour of day for automatic update (0-23), or null to disable.
url	string	Update source URL (1-96 ASCII chars, no spaces).

Example Request:

```
{
  "firmware": {
    "checkPeriodSec": 900,
    "downloadPeriodHr": 6,
    "updateHour": 3,
    "url": "static.mcdownloads.com/firmware/mc252-6-1-current.fus"
  }
}
```

httpd

Built-in HTTP server parameters.

Request Parameters:

Parameter	Type	Description
httpd	object	
port	integer	Web interface port (1-65535).
freeAccessSec	integer	Open access time after button press (5-3600 sec).

Example Request:

```
{
  "httpd": {
    "port": 80,
    "freeAccessSec": 600
  }
}
```

language

Interface language code.

Note: If the display does not support the specified code, English will be used.

Request Parameters:

Parameter	Type	Description
language	string	Language code: en - English; ua - Ukrainian.

Example Request:

```
{
  "language": "en"
}
```

serial

Serial interface settings.

Request Parameters:

Parameter	Type	Description
serial	object	
uidRange	string	Modbus UID address range (e.g., 112–255).
hdxMode	integer	Half-duplex mode: 0 - slave; 1 - master; 2 - raw mode.
charFormat	integer	Byte format: 0-5 (see Modbus section).
baudRate	integer	Baud rate (75-230400 bps).
frameMode	integer	Frame format: 0 - RTU, 1 - ASCII.
delayForRtuQ	integer	RTU character delay multiplier (2 ⁿ): 0-5.
delayForRtuMs	integer	RTU response delay (0-60000 ms).
delayForAsciiMs	integer	ASCII character delay (0-60000 ms).

Example Request:

```
{
  "serial": {
    "baudRate": 38400,
    "charFormat": 3,
    "frameMode": 1,
    "delayForRtuMs": 200,
    "delayForAsciiMs": 1000,
    "hdxMode": 1,
    "uidRange": "112-255",
    "delayForRtuQ": 1
  }
}
```

astro

Astronomical settings (sunrise/sunset).

Request Parameters:

Parameter	Type	Description
astro	object	
longDeg	integer	Longitude degrees (0-179).
longMin	integer	Longitude minutes (0-59).
longSec	integer	Longitude seconds (0-59).
latDeg	integer	Latitude degrees (0-179).
latMin	integer	Latitude minutes (0-59).
latSec	integer	Latitude seconds (0-59).
quadrant	integer	Quadrant: 0 - N/E; 1 - N/W; 2 - S/E; 3 - S/W.
sunZenithType	integer	Zenith type: 0 - Official; 1 - Civil; 2 - Nautical; 3 - Astronomical.

Example Request:

```
{
  "astro": {
    "longDeg": 30,
    "longMin": 19,
    "longSec": 0,
    "latDeg": 59,
    "latMin": 57,
    "latSec": 0,
    "quadrant": 0,
    "sunZenithType": 2
  }
}
```

ntp

NTP time synchronization settings.

Request Parameters:

Parameter	Type	Description
ntp	object	
hostnames	array	Array of NTP server hostnames (max 20 chars each).

Parameter	Type	Description
connMode	integer / null	Connection mode: null (disabled), 1 (Eth pref), 2 (GSM pref), 3 (Eth only), 4 (GSM only).
syncPeriodHr	integer	Synchronization period (0-240 hours).
syncDiffSec	integer	Synchronization threshold (0-180 sec).

Example Request:

```
{
  "ntp": {
    "hostnames": ["ntp.time.in.ua", "ntp2.stratum1.ru"],
    "connMode": 1,
    "syncPeriodHr": 24,
    "syncDiffSec": 1
  }
}
```

tsMode

Daylight saving time mode.

Request Parameters:

Parameter	Type	Description
tsMode	integer / null	DST Mode: null (Disabled), 1 (Brazil), 2 (UK), 3 (Germany), 4 (Greece), 5 (Jordan), 6 (Italy), 7 (Namibia), 8 (Poland), 9 (Portugal), 10 (USA), 11 (Turkey), 12 (Ukraine), 13 (Finland), 14 (France), 15 (Manual - see tsManual).

Example Request:

```
{
  "tsMode": 12
}
```

tsManual

Manual daylight saving time settings.

Request Parameters:

Parameter	Type	Description
tsManual	object	
bwdDow	integer	Winter time transition day of week (1-7, Mon-Sun).
bwdHour	integer	Winter time transition hour (0-23).
bwdMonth	integer	Winter time transition month (1-12).
bwdWeek	integer	Winter time transition week (1-5, 6=last).
fwdDow	integer	Summer time transition day of week (1-7, Mon-Sun).
fwdHour	integer	Summer time transition hour (0-23).
fwdMonth	integer	Summer time transition month (1-12).
fwdWeek	integer	Summer time transition week (1-5, 6=last).

Example Request:

```

{
  "tsManual": {
    "bwdDow": 7,
    "bwdHour": 2,
    "bwdMonth": 10,
    "bwdWeek": 6,
    "fwdDow": 7,
    "fwdHour": 2,
    "fwdMonth": 3,
    "fwdWeek": 6
  }
}

```

logFlags

Logging settings.

Request Parameters:

Parameter	Type	Description
logFlags	object	
atm	boolean	GSM interface logging (true/false).
system	boolean	System event logging (true/false).

Parameter	Type	Description
lwip	boolean	LWIP protocol logging (true/false).
forwarding	boolean	Port forwarding logging (true/false).
wireguard	boolean	WireGuard logging (true/false).

Example Request:

```
{
  "logFlags": {
    "atm": true,
    "system": true,
    "lwip": false,
    "forwarding": false,
    "wireguard": true
  }
}
```

rmSrv

TCP client settings (Remote Servers).

Request Parameters:

Parameter	Type	Description
rmSrv	array	Array of 3 server objects.
ip	string	Server IP address.
connPort	integer	Connection port (1-65535).
delayForRcvMs	integer	Receive delay (1-60000 ms).
delayForConnSec	integer	Delay between connections (1-30000 sec).
connMode	integer / null	Connection mode: null (disabled), 1 (Eth pref), 2 (GSM pref), 3 (Eth only), 4 (GSM only).
gateUid	integer / null	MC-251 compatible gateway UID (1-255), or null if unknown.
uidRange	string	UID range (e.g., 1-110).
password	string	Password for MC-251 compatible gateway.

Parameter	Type	Description
protocol	integer	Protocol: 1 - RAW, 2 - ModbusTCP.
virtualUids	boolean	Virtual UIDs (true/false).

Example Request:

```
{
  "rmSrv": [
    {
      "ip": "192.168.0.102",
      "connPort": 502,
      "delayForRcvMs": 1000,
      "delayForConnSec": 20,
      "connMode": 3,
      "gateUid": 112,
      "uidRange": "1-110",
      "password": "11111",
      "protocol": 2,
      "virtualUids": false
    },
    {
      "ip": "192.168.0.113",
      "connPort": 502,
      "delayForRcvMs": 1000,
      "delayForConnSec": 20,
      "connMode": null,
      "gateUid": null,
      "uidRange": "1-255",
      "password": "",
      "protocol": 2,
      "virtualUids": false
    },
    {
      "ip": "192.168.0.114",
      "connPort": 502,
      "delayForRcvMs": 1000,
      "delayForConnSec": 20,
      "connMode": null,
      "gateUid": null,
      "uidRange": "1-255",
      "password": "",
      "protocol": 2,
      "virtualUids": false
    }
  ]
}
```

forwarding

Port forwarding settings (Ethernet to GSM only).

Request Parameters:

Parameter	Type	Description
fwd	array	Array of 4 forwarding objects.
listenPort	integer / null	Local listening port (null to disable, or 1-65535).
isUdp	boolean	true for UDP, false for TCP.
serverAddress	string	Remote server host address. (If listenPort: 53 & isUdp: true & serverAddr: "", uses DNS settings).
serverPort	integer	Remote server port (1-65535).

Example Request:

```

{
  "fwd": [
    {
      "serverAddr": "www.cameronsworld.net",
      "serverPort": 443,
      "listenPort": 443,
      "isUdp": false
    },
    {
      "serverAddr": "",
      "serverPort": 0,
      "listenPort": null,
      "isUdp": false
    },
    {
      "serverAddr": "",
      "serverPort": 0,
      "listenPort": 53,
      "isUdp": true
    },
    {
      "serverAddr": "time.google.com",
      "serverPort": 123,
      "listenPort": 123,
      "isUdp": true
    }
  ]
}

```

tasksLogging

Programmable logic data logging settings.

Note: Logging is only available when an SD card is installed.

Request Parameters:

Parameter	Type	Description
tasksLogging	object	
format	integer / null	Log file format:
		<ul style="list-style-type: none"> null - logging disabled; 1 - compact binary format .DAT;

Parameter	Type	Description
		2 - text table .CSV with semicolon delimiter;
		3 - text table .CSV with comma delimiter;
		4 - text table .CSV with tab delimiter;
		5 - text table .CSV without headers, semicolon delimiter;
		6 - text table .CSV without headers, comma delimiter;
		7 - text table .CSV without headers, tab delimiter.
storeIntervalDays	integer / null	Period during which log files are not overwritten (logger mode).
		null - disables old data deletion (disables logger mode).
fileSizeLimitKb	integer	File size limit (minus 1 kilobyte, 0 to 65535).

Example Request:

```
{
  "tasksLogging": {
    "format": 2,
    "storeIntervalDays": 184,
    "fileSizeLimitKb": 1023
  }
}
```

powerSaving

Power and resource saving settings.

Request Parameters:

Parameter	Type	Description
powerSaving	object	
bodVoltage	integer	Minimum supply voltage in mV, below which the SD card will be safely unmounted (from 0 to 24000)
delayForDisplaySav ingMin	integer	Time till the display saving mode is entered (from 5 to 30).
displaySavingBrightn ess	integer	Brightness in the display saving mode, in percents. (from 0 to 50).

Example Request:

```
{
  "powerSaving": {
    "bodVoltage": 9000,
    "delayForDisplaySavingMin": 10,
    "displaySavingBrightness": 5
  }
}
```

Export Settings

Saves settings to a file in the root folder / of the SD card.

POST /api/cmd/settings/export/

Request Parameters:

Field	Type	Description
filename	string	Filename in FAT-32 DOS (8.3) format.

Example Request:

```
{
  "filename": "sgs.bin"
}
```

Import Settings

Loads settings from a file in the root folder / of the SD card.

POST /api/cmd/settings/import/

Request Parameters:

Field	Type	Description
filename	string	Filename in FAT-32 DOS (8.3) format.

Example Request:

```
{
  "filename": "sgs.bin"
}
```

Reset Settings

Resets all device settings to default values.

POST /api/cmd/settings/reset/

Example Request:

```
{}
```

System Commands

Reboot Device

Initiates a delayed device reboot.

POST /api/cmd/reboot/

Request Parameters:

Field	Type	Description
rebootSec	integer	Delay before reboot, from 0 to 180 seconds.

Example Request:

```
{
  "rebootSec": 3
}
```

Set Time

Sets the device time immediately.

POST /api/cmd/time/set/

Request Parameters:

Field	Type	Description
time	string	Time in ISO 8601 format relative to UTC0.

Example Request:

```
{
  "time": "2025-05-15T16:50:27"
}
```

Sync Time

Immediately synchronizes device time with NTP servers specified in settings.

POST /api/cmd/time/sync/

Example Request:

```
{ }
```

Activate License

Sends an activation request via WireGuard interface. Supported **only** with active WireGuard connection.

POST /api/activation/

Request Parameters:

Field	Type	Description
activation	object	Activation data object.
infrastructureName	string	Infrastructure name.
infrastructureShortUrl	string	Short URL to infrastructure site.
domain	string	Server address used for activation.
endpointId	string	Unique device identifier.
endpointName	string	Device name.
propertyId	string	Organization identifier.
propertyName	string	Organization name.
pin	string	8-digit device PIN code.
hasBonusesEur	string	Bonus amount on organization account.
activeTill	string / null	Activation expiration date (ISO 8601, UTC), or null to reset activation.
supportEmail	string	Support service email.

Updates

Check for Updates

Initiates a check for available updates (firmware) on the server.

POST /api/cmd/updates/check/

Example Request:

```
{}
```

Download Updates

Initiates download of available updates (firmware). SD card must be installed.

POST /api/cmd/updates/download/

Example Request:

```
{}
```

Program Update

Initiates installation of a previously downloaded update (firmware) from the SD card.

POST /api/cmd/updates/program/

Example Request:

```
{}
```

GSM

Send Test SMS

Sends a test SMS message to the specified phone number via the GSM modem.

POST /api/cmd/gsm/sms-test/

Request Parameters:

Parameter	Type	Description
phone	string	Phone number in international format (+48...).

Example Request:

```
{  
  "phone": "+48..."  
}
```

Modbus

Find Modbus Devices

Starts searching for Modbus devices on the RS-485 bus.

POST /api/modbus/find/start/

Request Parameters:

Parameter	Type	Description
baudRate	integer	Baud rate (bps: 75 to 230400).
charFormat	integer	Byte format: 0 - E1S; 1 - O1S; 2 - SPACE (0P1S); 3 - MARK (1P1S / NP2S); 4 - NP1S; 5 - AUTO STOP (Rx: NP1S / Tx: NP2S).
frameMode	integer	Frame format: 0 - RTU, 1 - ASCII. Optional.
fastSearch	boolean	Enable fast search (true/false). Optional.

Example Request:

```
{
  "baudRate": 9600,
  "charFormat": 3,
  "frameMode": 0,
  "fastSearch": false
}
```

Stop Search

Stops the active Modbus device search.

POST /api/modbus/find/stop/

Response:

Parameter	Type	Description
status	string	Status: DONE - search stopped successfully; BUSY - unable to stop search (try again later).

```
{
  "status": "DONE"
}
```

Get Search Result

Returns the list of found devices.

GET /api/modbus/find/result/?list

Request Parameters:

Parameter	Description
list	If specified, the response includes the list of found devices foundDevices.

Response:

Parameter	Type	Description
foundDevices	array	List of found devices.
uid	integer	Device UID.
baudRate	integer	Baud rate (bps: 75 to 230400).
charFormat	integer	Byte format: 0 - E1S; 1 - O1S; 2 - SPACE (0P1S); 3 - MARK (1P1S / NP2S); 4 - NP1S; 5 - AUTO STOP (Rx: NP1S / Tx: NP2S).
frameMode	integer	Frame format: 0 - RTU, 1 - ASCII.

```
{
  "foundDevices": [
    {
      "uid": 1,
      "baudRate": 9600,
      "charFormat": 0,
      "frameMode": 0
    }
  ]
}
```

Template Search and Identification

Get META Info from Template

POST /api/modbus/class-find/search-meta-info/

Request Parameters:

Parameter	Type	Description
filePath	string	Path to template file on SD card.

Example Request:

```
{
  "filePath": "/templates/em-481.txt"
}
```

Response:

Parameter	Type	Description
status	string	Status: DONE (success), ERROR (failed).
errorCode	string / null	Error code if failed. See Error Codes.

Start Search by Template

POST /api/modbus/class-find/start/?search-meta-info

Request Parameters:

Parameter	Type	Description
filePath	string	Path to template file on SD card.

Parameter	Type	Description
baudRate	integer	Baud rate.
charFormat	integer	Byte format.
frameMode	integer	Frame format: 0 - RTU, 1 - ASCII.
uid	integer	Device UID (1-254).

Example Request:

```
{
  "filePath": "/templates/em-481.txt",
  "baudRate": 9600,
  "charFormat": 3,
  "frameMode": 0,
  "uid": 111
}
```

Response:

Parameter	Type	Description
status	string	Status: DONE (started), ERROR (failed).
errorCode	string / null	Error code if failed. See Error Codes.

Stop Template Search

POST /api/modbus/class-find/stop/

Response:

Parameter	Type	Description
status	string	Status: DONE (stopped), BUSY (cannot stop).

Get Search Result

GET /api/modbus/class-find/result/

Response:

Parameter	Type	Description
mbReqPassed	integer	Number of completed Modbus requests.
checksPassed	integer	Number of executed template commands.

Parameter	Type	Description
linesPassed	integer	Number of read template lines.
mbUid	integer	Device UID.
deviceClass	string / null	Identified device class, or null.
scriptTime	string / null	Template file creation time (ISO 8601), or null.
scriptVersion	string / null	Template file version, or null.
status	string	Status: DONE (success), BUSY (searching), ERROR (failed).
errorCode	string / null	Error code if failed. See Error Codes.

Template Search Error Codes

Code	Description
File System Errors	
ERROR_FR_DISK_ERR	Low-level disk error
ERROR_FR_INT_ERR	Internal file system error
ERROR_FR_NOT_READY	Storage device not ready
ERROR_FR_NO_FILE	File not found
ERROR_FR_NO_PATH	Path not found
ERROR_FR_INVALID_NAME	Invalid file or path name
ERROR_FR_DENIED	Access denied
ERROR_FR_EXIST	File already exists
ERROR_FR_INVALID_OBJECT	Invalid file object
ERROR_FR_WRITE_PROTECTED	Write protected
ERROR_FR_INVALID_DRIVE	Invalid drive
ERROR_FR_NOT_ENABLED	File system not mounted
ERROR_FR_NO_FILESYSTEM	No file system
ERROR_FR_MKFS_ABORTED	Formatting aborted
ERROR_FR_TIMEOUT	Timeout
ERROR_FR_LOCKED	File locked

Code	Description
ERROR_FR_NOT_ENOUGH_CORE	Not enough memory
ERROR_FR_TOO_MANY_OPEN_FILES	Too many open files
ERROR_FR_INVALID_PARAMETER	Invalid parameter

Search Engine Errors

ERROR_META	Template structure error
ERROR_LABEL	Invalid or missing label
ERROR_CONDITION	Condition expression error
ERROR_CHECK_TYPE	Unknown check type
ERROR_MODIFIER_PARAMETER	Modifier parameter error
ERROR_DATA_SOURCE	Data source error
ERROR_DATA_SOURCE_FUNCTION	Unknown data source function
ERROR_DATA_SOURCE_ADDRESS	Invalid data source address
ERROR_CHECK_PARAMETERS	Check parameters error
ERROR_ACTION	Action error
ERROR_ACTION_TYPE	Unknown action type
ERROR_UNSUPPORTED_UTF_FORMAT	Unsupported UTF format
ERROR_FILE_NOT_READ	Failed to read template file
ERROR_LINE_OUTOFBOUND	Line out of bound
ERROR_UID_HIDDEN_BY_VDEVICE	UID hidden by virtual device
ERROR_ARG	General argument error
ERROR_MEM	Memory allocation error
ERROR_BUSY	Resource busy
ERROR_SERIAL_IS_SLAVE	Device is in slave mode
ERROR_EOF	End of file reached

Modbus Request/Response

Send Modbus Request

POST /api/modbus/req/

Request Parameters:

Parameter	Type	Description
reqData	string	Hexadecimal string of Modbus request (without CRC).

Example Request:

```
{
  "reqData": "010300000002"
}
```

Response:

Parameter	Type	Description
tid	integer	Transaction ID assigned to the packet.

Get Modbus Response

GET /api/modbus/resp/?tid=...

Request Parameters:

Parameter	Type	Description
tid	integer	Transaction ID.

Response:

Parameter	Type	Description
respData	string	Hexadecimal string of Modbus response.
status	string	Status: DONE (received), BUSY (pending).

Reconnect to Cloud via Modbus

Disconnects and reconnects to the cloud server.

POST /api/cmd/callback/reconnect/

Request Parameters:

Field	Type	Description
requestNewCode	boolean	Request new activation code upon reconnection.

Example Request:

```
{  
  "requestNewCode": true  
}
```

SD Card

List Files and Directories

Returns a list of files and subdirectories.

POST /api/card/dir/

Request Parameters:

Field	Type	Required	Description
path	string	Yes	Absolute path to directory.
fromIdx	integer	No	Start index (default 0).
toIdx	integer	No	End index.

Example Request:

```
{
  "path": "/logs",
  "fromIdx": 0
}
```

Response:

Field	Type	Description
fromIdx	integer / null	Index of first returned element.
toIdx	integer / null	Index of last returned element.
total	integer	Total items in directory.
items	array	Array of file/directory objects.

Each item in items contains:

Field	Type	Description
name	string	File or directory name.
attrib	string	Attributes: d (dir), r (read), w (write).
size	integer	File size in bytes.
datetime	string	Last modification time (ISO 8601).

Example Response:

```
{
  "fromIdx": 0,
  "toIdx": 0,
  "total": 1,
  "items": [
    {
      "name": "ATM.LOG",
      "attrib": "-rw",
      "size": 5768906,
      "datetime": "2025-04-16T16:07:38"
    }
  ]
}
```

Get File Info

Returns metadata about a file.

POST /api/card/file/info/

Request Parameters:

Parameter	Type	Description
path	string	Absolute path to file.

Example Request:

```
{
  "path": "/sgs.bin"
}
```

Response:

Field	Type	Description
name	string	File name.
attrib	string	File attributes in POSIX format (drw).
size	integer	File size in bytes.
datetime	string	Last modification date (ISO 8601 format).

Example Response:

```
{
  "name": "SGS.BIN",
  "attrib": "-rw",
  "size": 860,
  "datetime": "2025-04-16T15:12:52"
}
```

Read File

Returns file data block in base64 encoding.

POST /api/card/file/read/

Request Parameters:

Field	Type	Description
path	string	Absolute path to file.
blockIdx	integer	Block number (starting from 0).

Example Request:

```
{
  "path": "/sgs.bin",
  "blockIdx": 0
}
```

Response:

Field	Type	Description
blockIdx	integer	Current block number.
blockCount	integer	Total number of blocks in the file.
data	string	Base64 encoded data of the current block (1024 bytes).

Example Response:

```
{
  "blockIdx": 0,
  "blockCount": 2,
  "data":
  "63sM/2R3ki8uAC4AAQBMA9AHCAQBAAEBAAEAAAEBAW8FAQELAw8DPFAAWAJuZf//9gGXiHgAyADoAxQABQAA
  AFoAAACAJQAAGIwo2MCoAHP//8AwKgAAQgICAgICAQE2IA5aQ5tZ2FwAAAAAAAAAAAAZ2FwAAAAAAAAAAAA
  AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
  AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
  AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAKAH/AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAB4TADs5AAAC
  bnRwLnRpbWUuaW4udWEAAAAAAAAAG50cDIuc3RyYXR1bTEucnUAAAAAAAAABGAEMAwOHAgOKBwIAABMAKjExM
  SMAAAAAAAAAAAAAAMCoAHD2AegDFAAAAAAAAF8AAAAAAAAAAAAADAqABx9gHoAxQAAAAAAAH/AAAAAAAA
  AAAAAAwKgAcvYB6AMUAAAAAAB/wAAAAAAAAAAAAAMCoAHP2AegDFAAAAAAAAF8AAAAAAAAAAAAADAqAB
  09gE="
}
```

Write File

Uploads file data to SD card.

POST /api/card/file/write/

Request Parameters:

Field	Type	Description
path	string	Absolute path to file (e.g., /file.bin).
blockIdx	integer	Block number to write (starting from 0).
blockCount	integer	Total number of blocks to write.
data	string	Base64 encoded data to write (1024 bytes per block).

Note: Data in blocks must be strictly 1024 bytes, except for the last block.

Example Request:

```
{
  "path": "/file.bin",
  "blockIdx": 0,
  "blockCount": 2,
  "data":
  "63sM/2R3ki8uAC4AAQBMA9AHCAQBAAEBAAEAAAEBAW8FAQELAw8DPFAAWAJuZf//9gGXiHgAyADoAxQABQAA
  AFoAAACAJQAAgIwo2MCoAHP///8AwKgAAQgICAgICAQE2IA5aQ5tZ2FwAAAAAAAAAAAAZ2FwAAAAAAAAAAAA
  AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
  AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
  AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
  AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
  AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
  AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
  bnRwLnRpbWUuaW4udWEAAAAAAAAAG50cDIuc3RyYXR1bTEucnUAAAAAAAAABGAEMAwHAgKBwIAABMAKjExM
  SMAAAAAAAAAAAAAAAAAAMCoAHD2AegDFAAAAAAAAf8AAAAAAAAAAAAAAAAADAqABx9gHoAxQAAAAAAAAH/AAAAAAAA
  AAAAAAwKgAcvYB6AMUAAAAAAAAAB/wAAAAAAAAAAAAAAAAAMCoAHP2AegDFAAAAAAAAf8AAAAAAAAAAAAAAAAADAqAB
  09gE="
}
```

Response:

Empty JSON on success.

```
{}
```

Remove File

Removes a file from the SD card.

POST /api/card/remove/

Request Parameters:

Field	Type	Description
path	string	Absolute path to file.

Example Request:

```
{
  "path": "/1.pdf"
}
```

Response:

Empty JSON on success.

```
{}
```

Eject SD Card

Prepares SD card for safe removal.

POST /api/cmd/memory/card_eject/

Example Request:

```
{}
```

Response:

Empty JSON on success.

```
{}
```

HTTP Status Codes

HTTP Status	Meaning
200	OK
204	No Content
400	Bad Request
401	Unauthorized
403	Forbidden
404	Not Found
405	Method Not Allowed
408	Request Timeout
413	Payload Too Large
423	Locked
431	Request Header Fields Too Large
500	Internal Server Error
503	Service Unavailable
507	Insufficient Storage

Operations Logic Programming Reference

After running, Overvis MC252 starts execution of the operations logic program, if it was placed earlier in the internal memory.

If there is no program in the internal memory, and a memory card is present (formatted as FAT or FAT32), MC252 may load the taskfiles automatically. Such a reading runs once after startup or after installing a new memory card, but only if the internal memory does not contain a program. MC252 searches for and checks the taskfiles placed in the folder TASKS on the memory card.

Files can have arbitrary names and extensions (except for .MAP, .OBJ and .CNF) and placed in subfolders of the folder TASKS.

The found correct files are read into the internal memory and they form the operations logic program.

The internal logic memory can be cleared along with the settings reset to factory defaults. To preserve the settings and clear the internal logic memory only, you should use either Modbus or Web-interface command.

1. Using Web-interface

1.1 access the device using the browser (see Web Interface); 1.2 open Tasks tab; 1.3 Press the button ReLoad Tasks and confirm clearing the memory.

2. Via MODBUS

2.1 connect to the device via MODBUS and enter the setup mode (see Modbus Register Map); 2.2 to clear the internal logic memory, write the value 40959 into the control command register 120.

Note

The program in the internal memory may be deleted when updating the embedded software. If the memory card with the task files remains installed in the MC252, the program will be automatically read again after the update.

The tab Tasks indicates the result of reading the folder TASKS, including the number of discovered files. The successfully read files are displayed below in the logic tasks table. If during reading and verifying the program errors were detected, then it indicates the type of error, file and line number of the file error. If the program consisted of several files in the folder TASKS, then the internal memory will read all files except those in which errors were detected. Therefore, during error correction you should reload tasks, to clear again the internal memory.

The Modbus registers (see Modbus Register Map, registers 2020–2023) indicate the result of reading the folder TASKS, including the number of read files and the amount of used memory. If during reading and verifying the program errors were detected, then the type of error and line number are indicated. If the program consisted of several files in the folder "TASKS", then the internal memory will read all files except those in which errors are detected. Therefore, during error correction you should check the number of read files. After corrections were made, the internal memory should be cleared so that the program would be re-read.

Taskfile

Taskfile describes set of actions repeated at a specified time interval. The actions can include gathering data, processing and comparing the values and special actions after fulfillment of the preset conditions (according to the processing results).

File consists of lines. Each line can be one of the following:

- empty;
- option;
- definition;
- action.

Lines can include comments preceded by `;` or `#`. The text after these symbols is ignored. Lines can start from labels, followed by `:`. Labels are used for action references (e.g. `CALL function1 ... function1: RETURN`). Line parts are separated by or `TAB` symbols.

Tip

Lines are processed case-insensitive, so `CALL`, `Call` and `cALL` are the same action name.

Taskfile options

Options configure taskfile compilation or execution.

Options start with `@` followed by option kind. Then comes the option value.

Compilation options

PROTOCOLVERSION

Checked at compile time. If version is out of compiler supported versions range, taskfile compilation error occurs.

Examples:

```
@PROTOCOLVERSION 11
```

VARBITS

Sets the variable type size. Greater values allow for more available parameter cast types, and handle correctly bigger values, but take more memory.

Note

The compiler may support only one option value. Controller firmware change may be required to change this option. The option can be used to specify the variable type size the taskfile is intended to be used with.

- 32 - use 32 bit (4 bytes) signed integer variables.
- 64 - use 64 bit (8 bytes) signed integer variables.

Examples:

```
...  
@VARBITS 64
```

Execution options

UPDATE

This option is a set interval between two consecutive runs of the task. Actual interval depends on the controller load and overall taskfiles complexity (some actions such as parameter read/write or SMS sends can prolong the run execution). If the next run time comes before the other run has been finished, the new one is delayed and will be executed as soon as possible.

Note

If more than one run was delayed, they would be skipped, which could affect some calculations (e.g. counters). In this case long operations should be revised, or 'UPDATE' interval increased.

Option value is seconds integer in range 0..2000000.

0 - no interval, immediate re-runs.

Default is 60.

Examples:

```
...  
@UPDATE 10 ; rerun task every 10 seconds
```

UPDATEDIVISOR

This options is a modifier for the set interval between two consecutive runs of the task. If set to non-zero value, set interval is managed with subsecond precision and equals 'UPDATE' / 'UPDATEDIVISOR'.

Option value is hertz integer in range 0..500.

0 - intervals calculation precision in seconds.

Default is 0.

Examples:

```
...  
@UPDATE 2  
@UPDATEDIVISOR 3 ; the interval between runs will be 2/3 or about 667 ms
```

PARAMLOADRATIO

This option defines forced wait delays between device parameter read/write operations (this does not affect memory parameters, see Parameter definition below). This allows more time for the other clients requests processing. The delay is proportional to the time taken by the previous parameter operation.

Option value is percents integer in range 0..100.

- 100 - no forced delays.
- 50 - delay time equals the previous operation time.
- 25 - delay time is 3 times greater than the operation time.

Default is 25.

Examples:

```
@PARAMLOADRATIO 50 ; parameters operations are separated with delays of equal length
```

PARAMTIMEOUT

Wait limit for device parameter read/write operations (this does not affect memory parameters, see Parameter definition below). After timeout expires, operation results in error.

Option value is milliseconds integer in range 0..5000.

Default is 5000.

Examples:

```
@PARAMTIMEOUT 1000 ; wait no more than 1 second for the parameter operation completion
```

RESETDATA

Determines whether task holding items (variables and conditions) are reset before each run.

- 0 - variables and conditions retain their values in RAM, and are reset at powerdowns or controller reset.
- 1 - variables and conditions are reset before each run.

Default is 0.

Examples:

```
@RESETDATA 1 ; holding items values left after the run are not used in the subsequent runs
```

Taskfile definitions

Definitions declare logical links, strings and Modbus register mapping, or holding items types.

Definition lines start with DEF followed by new defined name. Then comes the definition type which can be one of the following groups:

- string;
- Modbus device;
- parameter;
- variable(s);
- condition(s).

Caution

If the definition references the other defined items, these should be defined earlier in the taskfile.

String definition

String definition is marked by " pair, with the string inside. String can include special references, enclosed in * (the escape character * should be doubled to be included into the string itself).

Special references include:

- VAR(variable) - for a numeric value out of a variable. lasterror variable may be also used here;
- ERR(variable) - for a literal representation of an error code name out of a variable. lasterror variable may be also used here;
- PHONE(index) - for a number out of a list of GSM abonents in the device settings. Index is a number in range 0 to 4.

Examples:

```
DEF my_string1 "2 ** 2 = *VAR(var_mul_result)*" ; this string
template may produce a "2 * 2 = 4" string
DEF e_msg "Handled error #*VAR(lasterror)*: *ERR(lasterror)*" ; this string
template may produce a "Handled error #1: FUNCTION_ILLEGAL" string
```

Modbus device definition

Modbus device can be used as a parameter source or storage.

Modbus device is defined by one of the following types:

- MBWRDENIED - only reading functions may be used;
- MBWRSINGLE - single register/coil write functions can be used;
- MBWRMULTI - multiple register/coil write functions can be used;
- MBWRANY - both single/multiple register/coil write functions can be used.

The type is followed by 2 or 3 arguments:

1. Modbus unit address - one of the following: fixed ID in range 1..255, own Controller virtual device ID as *, or indirect ID as a variable plus integer value (e.g. `some_variable+10`);
2. read registers/coils per one Modbus request limit, in range 1..125;
3. the argument is present only for MBWRMULTI, MBWRANY types - write registers/coils per one Modbus request limit, in range 1..125.

Examples:

```
DEF mc_252 WRHANY * 125 125 ; own controller virtual device
```

Parameter definition

Parameters are used to access external data: get data to or from variables.

Parameter is defined by one of the data types.

Note

type is used to cast value when reading from or writing to the parameter, the value is always casted to/from the default type set in the taskfile options.

Table 1 - Parameter types available for any variable type size

No.	Type	Description
0	UINT16	unsigned (non-negative) 16-bit (2 bytes) integer, serialized as Big Endian (most significant byte first, e.g. 258 is stored as 0x01, 0x02);
1	INT16	signed 16-bit (2 bytes) integer, serialized as Big Endian;
2	INT16BLE	signed 16-bit (2 bytes) integer, serialized as Little Endian (least significant byte first, e.g. 258 is stored as 0x02, 0x01);
3	INT32	signed 32-bit (4 bytes) integer, serialized as Big Endian (most significant byte first, e.g. 66051 is stored as 0x00, 0x01, 0x02, 0x03);
4	INT32BLE	signed 32-bit (4 bytes) integer, with bytes serialized as Little Endian (least significant byte first, e.g. 66051 is stored as 0x03, 0x02, 0x01, 0x00);
5	INT32WLE	signed 32-bit (4 bytes) integer, with words serialized as Little Endian (least significant word first, e.g. 66051 is stored as 0x02, 0x03, 0x00, 0x01);
6	BIT	1-bit integer (used for example to access Modbus coils and discrete inputs);
7	INT32BE	same as INT32

No.	Type	Description
8	F32EP0R	IEEE 754 single precision (4 bytes) floating-point, serialized as Big Endian;
9	F32BLEE P0R	IEEE 754 single precision (4 bytes) floating-point, with bytes serialized as Little Endian;
10	F32WLEE P0R	IEEE 754 single precision (4 bytes) floating-point, with words serialized as Little Endian;
11	F32EP1R	single precision (4 bytes) floating-point, serialized as divided by 10 Big Endian;
12	F32BLEE P1R	same as F32BLEEP0R, but divided by 10 before serializing;
13	F32WLEE P1R	same as F32WLEEP0R, but divided by 10 before serializing;
14	F32EP2R	single precision (4 bytes) floating-point, serialized as divided by 100 Big Endian;
15	F32BLEE P2R	same as F32BLEEP0R, but divided by 100 before serializing;
16	F32WLEE P2R	same as F32WLEEP0R, but divided by 100 before serializing;
17	F32EP3R	single precision (4 bytes) floating-point, serialized as divided by 1000 Big Endian;
18	F32BLEE P3R	same as F32BLEEP0R, but divided by 1000 before serializing;
19	F32WLEE P3R	same as F32WLEEP0R, but divided by 1000 before serializing;
20	UINT16B LE	unsigned (non-negative) 16-bit (2 bytes) integer, serialized as Little Endian;
21	UINT8	unsigned (non-negative) 8-bit (1 byte) integer;
22	INT8	signed 8-bit (1 byte) integer;

Table 2 - Parameter types available for 64-bit variable type

No.	Type	Description
23	UINT32	unsigned (non-negative) 32-bit (4 bytes) integer, serialized as Big Endian (most significant byte first, e.g. 66051 is stored as 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x02, 0x03);
24	UINT32B LE	unsigned (non-negative) 32-bit (4 bytes) integer, with bytes serialized as Little Endian (least significant byte first, e.g. 66051 is stored as 0x03, 0x02, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00);

No.	Type	Description
25	UINT32W LE	unsigned (non-negative) 32-bit (4 bytes) integer, with words serialized as Little Endian (least significant word first, e.g. 66051 is stored as 0x02, 0x03, 0x00, 0x01, 0x00, 0x00, 0x00, 0x00);
26	INT64	signed 64-bit (8 bytes) integer, serialized as Big Endian (most significant byte first, e.g. 66051 is stored as 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x02, 0x03);
27	INT64BL E	signed 64-bit (8 bytes) integer, with bytes serialized as Little Endian (least significant byte first, e.g. 66051 is stored as 0x03, 0x02, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00);
28	INT64WL E	signed 64-bit (8 bytes) integer, with words serialized as Little Endian (least significant word first, e.g. 66051 is stored as 0x02, 0x03, 0x00, 0x01, 0x00, 0x00, 0x00, 0x00);
29	F64EP0R	IEEE 754 double precision (8 bytes) floating-point, serialized as Big Endian;
30	F64BLEE P0R	IEEE 754 double precision (8 bytes) floating-point, with bytes serialized as Little Endian;
31	F64WLEE P0R	IEEE 754 double precision (8 bytes) floating-point, with words serialized as Little Endian;
32	F64EP1R	double precision (8 bytes) floating-point, serialized as divided by 10 Big Endian;
33	F64BLEE P1R	same as F64BLEEP0R, but divided by 10 before serializing;
34	F64WLEE P1R	same as F64WLEEP0R, but divided by 10 before serializing;
35	F64EP2R	double precision (8 bytes) floating-point, serialized as divided by 100 Big Endian;
36	F64BLEE P2R	same as F64BLEEP0R, but divided by 100 before serializing;
37	F64WLEE P2R	same as F64WLEEP0R, but divided by 100 before serializing;
38	F64EP3R	double precision (8 bytes) floating-point, serialized as divided by 1000 Big Endian;
39	F64BLEE P3R	same as F64BLEEP0R, but divided by 1000 before serializing;
40	F64WLEE P3R	same as F64WLEEP0R, but divided by 1000 before serializing;
41	F64EP4R	double precision (8 bytes) floating-point, serialized as divided by 10000 Big Endian;
42	F64BLEE P4R	same as F64BLEEP0R, but divided by 10000 before serializing;

No.	Type	Description
43	F64WLEE P4R	same as F64WLEEP0R, but divided by 10000 before serializing;
44	F64EP5R	double precision (8 bytes) floating-point, serialized as divided by 100000 Big Endian;
45	F64BLEE P5R	same as F64BLEEP0R, but divided by 100000 before serializing;
46	F64WLEE P5R	same as F64WLEEP0R, but divided by 100000 before serializing;
47	F64EP6R	double precision (8 bytes) floating-point, serialized as divided by 1000000 Big Endian;
48	F64BLEE P6R	same as F64BLEEP0R, but divided by 1000000 before serializing;
49	F64WLEE P6R	same as F64WLEEP0R, but divided by 1000000 before serializing;

The type is followed by 1 or 3 arguments:

1. source - one of the following: MEMTEMP, MEMBAT, MEMFILE, MEMFLASH (for memory areas) or a Modbus device defined earlier.
2. For Modbus device source it is the Modbus data table: one of the following C, D, H, I for coils, discrete inputs, holding registers or input registers respectively. For memory sources it is optional - if specified, it maps the memory parameter to own controller virtual device addresses.
3. For Modbus device source it is the starting address (either fixed in range 0..65535, or indirect as a variable plus integer value, e.g. some_variable+0). For memory sources it is optional - if specified, it maps the memory parameter to own controller virtual device addresses. Parameter may occupy several addresses depending on its type and size.

Memory sources for the parameters allow to read presets or store settings and temporary values to be processed at consecutive runs.

- MEMTEMP - RAM space, data is lost at powerdown or controller reset, small area size but excellent write speed;
- MEMBAT - memory area powered by internal clock battery, very limited size but good write speed;
- MEMFLASH - controller settings area, data retains at reset, small size and usually read-only;
- MEMFILE - extendable file area, requires memory card with the taskfiles to be present, average speed but excellent area size.

Tip

If several memory sourced parameters of the same size (other than MEMFILE) are mapped to the same Modbus address, they would share the same memory area. This can be used for the parameters in different taskfiles, to transfer data between the tasks. In that case only one taskfile can write to this shared parameter (being the writer). The other taskfiles can only read this shared parameter (being the readers of the data written by the writer). MEMFILE parameters are not stored in the memory area and do not use this feature.

Examples:

```
DEF local_time UINT32 mc_251 H 170 ; own controller registers 170..171
DEF alg_mode UINT32 MEMTEMP H 5000 ; RAM value, mapped to own holding register 5000
```

Variable definition

Variables are used to operate integer values. All variables have the same type, which is set in taskfile options.

Variables can be mostly defined automatically when first mentioned. Some operations though (like 'ISKNOWN', copying, or array items assignment) require the variable to be defined earlier.

Variables are defined with VAR type. Variable arrays are defined with VARS type followed by array size in range 1..65535 (array of a single variable is effectively the same as a simple variable).

There is one pre-defined variable: lasterror. Error occurrences write the error code to it. It cannot be written like the other variables, but RAISE command effectively writes to it. This variable can be used in error handlers for error-specific actions.

Examples:

```
DEF variable_1 VAR
DEF array_with_5_items VARS 5
```

Condition definition

Conditions are used to operate boolean values.

Conditions can be mostly defined automatically when first mentioned. Some operations though (like 'ISKNOWN', copying or array items assignment) require the condition to be defined earlier.

Conditions are defined with COND type. Condition arrays are defined with CONDS type followed by array size in range 1..65535 (array of a single condition is effectively the same as a simple condition).

Examples:

```
DEF is_temperature_high COND
```

Taskfile actions

Actions form iterative logic algorithm. Actions can be one of the following:

- assertion;
- check;
- command.

Actions are executed consecutively with these exceptions:

- check actions would skip a single action, if their condition is not TRUE;
- command actions like GO, CALL, RETURN would jump to a label or return from the function call;
- command action RAISE or any error occurrence would either jump to the error handler or stop this execution run due to an unhandled error;
- command action EXIT stops this execution run.

There are 2 labels used as the main taskfile function definition:

- run - each time the taskfile run is executed, the action labelled run is performed first (if there is no such label, the first action in the taskfile is the first to be performed);
- onerror - if an error occurs with no other handlers specified, a handler labelled onerror is used (if there is no such label, the error would be unhandled, see Error handling below).

The action may in some cases reference the other items which were not defined earlier in the taskfile:

- variable arrays and condition arrays should be defined before they are referenced;
- evaluations which can take arguments of different types (like direct copying), require arguments to be defined earlier;
- other variables and conditions can be referenced without pre-definition;
- labels can be referenced by CALL, TRYCALL or GO before they are defined;
- strings can be referenced by SENDSMS, SMSRCVD, NOSMSRCVD, SYSLOG or PARAMLOG before they are defined;
- in all the other cases the referenced item should be defined earlier in the taskfile.

Assertion action

Assertions are actions producing some result data, which they store in one of the holding items (variable or condition).

Assertions start with a PUT followed by result target item name. Then comes the function with the proper result type for this target (see Variable evaluation, Condition evaluation below). If no function is specified, the argument is directly copied into target (this can also be used to get the items from the array indexed by the variable, or put them back).

Examples:

```
DEF numbers VARS 3
PUT numbers[0] 0
PUT numbers[1] 1
PUT numbers[2] 2 ; numbers contains [0, 1, 2]
PUT index 2
PUT x numbers[index] ; x contains 2
PUT var_mul_result MUL x 2 ; var_mul_result contains 4
```

Check actions

Checks are actions used for branching. They skip the next action if their function (with condition result, that is boolean) produces FALSE or an unknown value.

Checks start with IF followed by the function with the condition result type (see Condition evaluation below). If no function specified, the argument should be a condition name.

Examples:

```
PUT lesser 5
PUT greater 3
IF LE lesser greater ; LE 5 3 produces FALSE, for 5 <= 3 is not true
GO then_section ; this command will not be executed
; else section
PUT tmp lesser ; lesser and greater variables values exchange
PUT lesser greater
PUT greater tmp
GO endif_section
then_section:
; two variables already sorted, nothing to do
endif_section:
; two variables are sorted now, lesser <= greater
```

Command actions

Commands are actions producing no result data or discarding it.

Commands are one of the following:

- EXIT - stops taskfile execution. Has 1 argument: error name, OK (if no error) or lasterror.
- RAISE - forces an error occurrence. Has 1 argument: error name, or lasterror (for raising the same error higher).
- GO - next action will not be performed, execution will continue from the specified label. Has 1 argument: label name.
- CALL - starts some user function execution. Next action execution is postponed, execution will now continue from the specified label. Has 1 argument - function entry label name.

- TRYCALL - starts some user function with its own error handling. Next action execution is postponed, execution will now continue from the specified label. Has 2 arguments - function entry label label name and error handler label name.
- RETURN - ends user function execution. Next action will not be performed, execution will continue from the previously postponed action (by the previously met 'CALL').
- WRITE - outputs data to the parameter. Has 2 arguments: parameter name, then variable name or an integer value.
- PARAMLOG - writes the parameter to the data log with a comment. Has 2 arguments: parameter name, then string name. The parameter must be in Modbus registers or otherwise mapped to Modbus, as its address is used in logs.
- PARAMLOGNOCOMMENT - writes the parameter to the data log without a comment. Has 1 argument: parameter name. The parameter must be in Modbus registers or otherwise mapped to Modbus, as its address is used in logs.
- SYSLOG - writes the string to the system log. Has 1 argument: the string name.
- SENDSMS - sends the string as an SMS. Has 2 arguments: the addressee string, then the SMS text string name. If the addressee strings results in no other characters than 0, the SMS would not be sent.

Examples:

```
WRITE param_hysteresis 5
EXIT OK
```

Variable evaluation

Variable evaluation functions are used in assertion action (PUT) to calculate a new variable value.

They can produce numeric value or an unknown result (see Unknown values below).

Arithmetic evaluations

Most of the following functions take 2 arguments (e.g. a and b) and perform an operation.

- ADD - addition ($a + b$). The first argument is variable, the second can be either variable or an integer value.
- SUB - subtraction ($a - b$). The first argument is variable, the second can be either variable or an integer value.
- MUL - multiplication ($a * b$). The first argument is variable, the second can be either variable or an integer value.
- DIV - division (a / b , integer result). The first argument is variable, the second can be either variable or an integer value.
- MOD - remainder of division ($a - (a / b * b)$). The first argument is variable, the second can be either variable or an integer value.
- SQRT - square root (integer). Takes a single variable or an integer value argument.
- VAL - direct copy. Takes a single argument: variable or an integer value. As with the other direct copying evaluations, function name VAL can be omitted.

Bits evaluations

Most of the following functions take 2 arguments (e.g. a and b) and perform an operation.

- **BITSAND** - logical AND of every bit of a with the corresponding bit of b. The first argument is variable, the second can be either variable or an integer value.
- **BITSOR** - logical OR of every bit of a with the corresponding bit of b. The first argument is variable, the second can be either variable or an integer value.
- **SHR** - arithmetic shift to the right of the bits of a, b times right. The first argument is variable, the second can be either variable or an integer value.
- **SHL** - shift to the left of the bits of a, b times left. The first argument is variable, the second can be either variable or an integer value.
- **BITSBIT** - bit b copy out of the bits of a. Is equivalent to $\text{SHR } a \ b$ followed by $\text{BITSAND } a \ 1$. The first argument is variable, the second can be either variable or integer value.
- **BITSNOT** - bits inversion. Takes a single variable or an integer value argument.

Parameter input evaluation

- **READ** - takes parameter as an argument, and reads then casts its value to the variable default type.

Array items evaluations

- **MAX** - maximal item of the array. Takes a single argument: array of variables.
- **MAXIDX** - index of the maximal item of the array in range of 0 to array size minus 1. Takes a single argument: array of variables.
- **MIN** - minimal item of the array. Takes a single argument: array of variables.
- **MINIDX** - index of the minimal item of the array in range of 0 to array size minus 1. Takes a single argument: array of variables.
- **SUM** - sum of the array items. Takes a single argument: array of variables.
- **SELECTBY** - takes 2 arguments: array of variables and array of conditions of the same size, and selects the variable corresponding to the first condition which is TRUE.

Task configuration evaluations

- **GETUPDPERIODSEC** - Task update period in seconds. No arguments.
- **GETUPDPERIODMS** - Task update period in milliseconds. No arguments.
- **GETUPDNAMEIDX** - first integer in the taskfile name. This can be used as a base or a factor for parameters mapping, to create several similar tasks with the same contents. No arguments.

Real time clock evaluations

- **GETRTC** - Real time clock in seconds (since base year, usually 2000). Local, with DST. No arguments.
- **GETYEAR** - Clock year. No arguments.
- **GETMONTH** - Clock month, 1 to 12. No arguments.
- **GETDAY** - Clock day of month, 1 to 31. No arguments.
- **GETHOURS** - Clock hours. No arguments.

- GETMINUTES - Clock minutes, 0 to 59. No arguments.
- GETSECONDS - Clock seconds, 0 to 59. No arguments.
- GETDOW - Clock day of week, 0 to 6, with 0 for Monday, 6 for Sunday. No arguments.
- GETDAYTIME - Day seconds, 0 to 86399. No arguments.
- GETSUNRISE - Day sunrise second, -1 to 86400, with -1 for polar day, 86400 for polar night. No arguments.
- GETSUNSET - Day sunset second, -1 to 86400, with -1 for polar day, 86400 for polar night. No arguments.

Physical parameters evaluations

- GETVOLTAGE - Power supply voltage in millivolts. No arguments.
- GETTEMPERATURE - Temperature in degrees. No arguments.

Examples:

```
PUT square MUL x x
```

Condition evaluation

Condition evaluation functions are used in either assertions (PUT actions) or checks (IF actions) to determine a condition value.

They can produce TRUE, FALSE or an unknown result (if referring the other unknown item values, see Unknown values below).

All functions come in direct or inversed form, which differ only in result being inverted or not before use. The functions are listed in both forms, e.g. 'EQ' / 'NE' for equality and non-equality (inverted EQ) checks.

Comparison checks

- EQ / NE - equal / non-equal ($a = b$ / $a \neq b$). The first argument is variable, the second can be either variable or an integer value.
- GE / LS - greater-or-equal / lesser ($a \geq b$ / $a < b$). The first argument is variable, the second can be either variable or an integer value.
- GR / LE - greater / lesser-or-equal ($a > b$ / $a \leq b$). The first argument is variable, the second can be either variable or an integer value.

Logical checks

- AND / NAND - logical AND / inverted logical AND ($a \&\& b$ / $!(a \&\& b)$). Takes 2 condition arguments.
- OR / NOR - logical OR / inverted logical OR ($a \|\| b$ / $!(a \|\| b)$). Takes 2 condition arguments.
- VAL / NOT - direct copy / inverted copy ($a / !a$). Takes a single argument: condition or boolean value (TRUE or FALSE). As with the other direct copying evaluations, function name VAL can be omitted.

SMS checks

- SMSRCVD / N0SMSRCVD - checks whether a new SMS starting with a specified text came from a number starting with specified digits. The first argument is a phone number start, the second is an SMS text start. If the phone number is

an empty string, SMS from any abonents are checked.

Error checks

- **ISKNOWN / ISNOTKNOWN** - checks whether the holding item value is known. Takes a single argument: variable or condition (the values are unknown if they are read from an uninitialized parameters or evaluated from the other unknown items, see Unknown values below).

Examples:

```
PUT c_equal EQ x 5
IF c_equal
EXIT OK

; the same function can be used in check directly
IF EQ x 5
EXIT OK
```

Fractional values and fixed point calculations

Data processing is performed by variable evaluations which store values of integer type.

However, parameters read and write can cast data types from and to fractional values of single and double precision as defined in IEEE 754 standard. This casting can be done with a multiplier (depending on the parameter type used), to preserve a certain number of decimal digits of the value fractional part. Some connected devices may also have integer parameters representing a fractional value (pre-multiplied in a similar way as above).

This allows to operate fractional values as integers.

Note

Simple arithmetics like addition or subtraction do not affect fixed decimal point position.

A care should be taken using operations like multiplication, division or square root.

For example, `MUL 2 2 = 4`, but if the first argument represents `0.2` value multiplied by `10`, and the second - `0.02` value multiplied by `100` (fixed point values with 1 and 2 digit precision respectively), the result represents `0.004` multiplied by `1000` (`10*100`, as the multipliers were also multiplied).

A square root of the previous value (`4` standing for a value `0.004` multiplied by `1000`), `SQRT 4` produces `2`, which is a wrong result, because of the multiplier `1000`. For `SQRT 1000` is not an integer. To get a correct result, this value multiplier should be adjusted to a nearest multiplier with even number of zeroes, that is `10000`. So `4` should be pre-multiplied by `10` (producing `40` for `0.004` multiplied by `10000`). Then the `SQRT 40` produces `6` which is a correct result (standing for `0.06` multiplied by `100`).

Arrays

Holding items can be organized in arrays of fixed size. Arrays should be defined before referencing them in evaluations (e.g. `DEF some_array VARS 5` or `DEF the_other_array CONDS 2`).

There are 3 ways of using the arrays:

- array items with an integer value index can be used in most of the functions and assertions instead of a single item name as an argument (or an assertion result, e.g. `PUT some_array[3] MUL some_array[2] some_array[1]`).
- array items with variable index can be used: a) either in assertions as a result for a direct copying or a single-argument functions (e.g. `PUT some_array[ar_idx] SQRT distance`), b) or as an argument for a direct copying (including `NOT` function which is just an inverted copying of a condition, e.g. `IF NOT c_array[counter]`).
- certain array evaluation functions take arrays as arguments, performing searching or digesting and producing a single variable result.

Error handling

Errors can occur during actions execution.

Some commands like `CALL` or `RETURN` result in error due to misuse, which requires program correction. The other commands or functions result in error due to either wrong argument values (square root of a negative variable) or external reasons (e.g. parameter could not be read or written).

Error occurrence in the function causes execution to interrupt and jump to the function error handler. This can happen once per function call: if no handler has been provided, or another error occurs in the error handler itself, the execution continues at the caller function handler. For the main taskfile function, `onerror` label is used as a default taskfile error handler.

A pre-defined variable `lasterror` can be used to read error code and perform error-specific actions.

If no caller handler is available, the error is considered unhandled - the default handler is performed similar to this:

```
SYSLOG default_err_msg
EXIT lasterror
;
DEF default_err_msg "Unhandled error #*VAR(lasterror)*: *ERR(lasterror)*"
```

Unknown values

Error during the assertion usually causes the result target to have an unknown value. Unknown values can also appear without an actual error, e.g. when reading an uninitialized parameter or using the variable which has not been asserted.

Unknown values tend to spread, as most functions referring an unknown variable or condition value will result in an unknown value themselves. Some functions may still determine the result, notwithstanding of unknown items in their arguments. `SELECTBY` array function can get the result early (before reaching any of the unknown items later). `AND` logical check can result in `FALSE` if any of its arguments is `FALSE` (while the other one can have any value), `OR` can likewise result in `TRUE`, etc.

Critical commands may require additional checks (`ISKNOWN` / `ISNOTKNOWN`) and direct variable assertions, or parameter initializations, to guard against unknown states.

IF check action treat unknown function result as FALSE, and would skip the next command. This allows to choose between direct and inverted check function to guard the branch against executing in unknown states.

Taskfiles examples

Below there are examples of finished programs, each consists of a single task file. To run the sample on the MC252 it is necessary to:

- 1 Create a text file (e.g. having the .txt extension) with program text.
- 2 Put the file in the folder TASKS.
- 3 Put the prepared folder onto the microSD memory card, formatted as FAT or FAT32.
- 4 Put the memory card into the MC252.

Example 1

This example describes a program that, in the event of fault of the device, will send SMS with warning.

In the taskfile text:

- 3 is Modbus ID of the device 0M-310;
- 240 is the register address which is monitored for the fault.



```
# SMS sending when bit 0 of register 240 of device 3 is set

#11th version of Protocol
@PROTOCOLVERSION 11
#task rerun interval will be every 3 seconds
@UPDATE 3
#limit of response waiting for Modbus query is 1000 msec = 1 sec
@PARAMTIMEOUT 1000
#after each query the delay is added, equal to the response waiting time,
#so the other clients can perform their queries
@PARAMLOADRATIO 50

#MC252 can read and write no less than 120 registers per one query
#note the * character – it is Modbus ID of the MC252 own virtual Modbus device
DEF mc252 MBWRANY * 120 120
#OM-310 has Modbus ID equal to 3 and allows reading 4 registers per one query,
#but writing of only one register per one query
DEF om310 MBWRSINGLE 3 4

#during each update, it is required to read
#the holding register with address 240 from OM-310
#UINT16 – means that the 16-bit value is unsigned (it can't be less than 0)
DEF alarms UINT16 om310 H 240

#each run starts here
PUT alarm READ alarms

#copy zero bit of the register 240
PUT alarm BITSBIT alarm 0
#now the variable holds the parameter 240.0 value 0 or 1

#variable is compared with 1, the condition produces `TRUE`, if alarm = 1
PUT is_alarm EQ alarm 1
#condition is met, if the previous condition is not fulfilled, and vice versa
PUT no_alarm NOT is_alarm

#if the condition is_alarm is met (if 240.0 = 1), then send one SMS
IF is_alarm
SENDSMS technician_number alarm_msg

#program end
EXIT OK

#SMS text
```

```
DEF alarm_msg "0M-310 (3) - avaria"  
  
#the phone to receive the SMS can be specified below  
DEF technician_number "01234567"
```

Example 2

In this example, the program controls the hysteresis value on the second channel of the TR-101 device, depending on the temperature of the sensor on the first channel.

The program uses clock battery-powered memory as a storage of the temperature limits and corresponding hysteresis parameters. These parameters are mapped to the MC252 own registers to simplify program configuration.

In the taskfile text:

- 16 is Modbus ID of the device TR-101;
- 4 is the address of the channel 1 temperature register;
- 47 is the address of the channel 2 hysteresis register;
- 5500 is the address for the lower temperature limit;
- 5501 is the address for the upper temperature limit;
- 5502 is the address for the hysteresis at crossing the lower temperature limit;
- 5503 is the address for the hysteresis at crossing the upper temperature limit.



```
@PROTOCOLVERSION 11
@UPDATE 20      #program will run every 20 seconds
@PARAMLOADRATIO 50

DEF mc252 MBWRANY * 120 120
#TR-101 has Modbus ID 16 and can read no less than 100 registers per a query,
#but write one register per one query
DEF tr101 MBWRSINGLE 16 100

#INT16 – registers with signed integers, as the temperature can be less than 0
DEF t_lower INT16 MEMBAT H 5500
DEF t_upper INT16 MEMBAT H 5501
DEF t_chan1 INT16 tr101 H 4
#UINT16 – as the hysteresis is no less than 0
DEF h_at_lower UINT16 MEMBAT H 5502
DEF h_at_upper UINT16 MEMBAT H 5503
DEF h_chan2 UINT16 tr101 H 47

DEF temperatures VARS 3
DEF hysteresi VARS 3
DEF checks CONDS 3

#program start
run:

#lower and upper temperature limits, as well as its current value
PUT temperatures[0] READ t_lower
PUT temperatures[1] READ t_upper
PUT temperatures[2] READ t_chan1

# hysteresis for temperatures that are below (or above) than the limits
PUT hysteresi[0] READ h_at_lower
PUT hysteresi[1] READ h_at_upper
# current hysteresis value
PUT hysteresi[2] READ h_chan2

# was the temperature out of the limits?
PUT checks[0] LE temperatures[2] temperatures[0]
PUT checks[1] GE temperatures[2] temperatures[1]
# in other cases – do not change the hysteresis (current value)
PUT checks[2] TRUE

# the desired hysteresis is selected from hysteresi array
# according to the conditions of checks array
```

```
PUT hysteresis SELECTBY hysteresi checks

# is hysteresis not set to the desired value yet?
IF NE hysteresis hysteresi[2]
# write a new hysteresis to TR-101
WRITE h_chan2 hysteresis

EXIT OK
```

Example 3

In this example, a program is described that monitors the temperature measured by OB-215. When the temperature exceeds -15 degrees for more than 10 minutes, it sends an SMS and starts logging the temperature values.

In the taskfile text:

- 11 is Modbus ID of the OB-215 device;
- 6 is the address of the register from which the temperature is read.



```
@PROTOCOLVERSION 11
# the program will run every 15 seconds
@UPDATE 15
DEF controller MBWRANY * 120 120
DEF ob215 MBWRSINGLE 11 4
DEF temperature INT16 ob215 H 6
# alarm flag – temperature rise for more than 10 minutes
DEF alarm_temp_high UINT16 MEMTEMP H 5000
# counter for a delay of 10 minutes (600 seconds)
DEF counter UINT16 MEMTEMP H 5001

run:
PUT v_temp READ temperature
PUT v_alarm READ alarm_temp_high
# v_alarm is auto-defined above as a variable
# (by a PUT v_alarm READ construction)
IF ISNOTKNOWN v_alarm
    PUT v_alarm 0
PUT v_counter READ counter
IF ISNOTKNOWN v_counter
    PUT v_counter 0
PUT temp_limit -150

IF GR v_temp temp_limit
    GO noticed_temp_high
WRITE alarm_temp_high 0
WRITE counter 0
EXIT OK

noticed_temp_high:
PUT counter_limit 10
PUT counter_limit MUL counter_limit 10
IF GE v_counter counter_limit
    GO temp_high_too_long
PUT v_period GETUPDPERIODSEC
PUT v_counter ADD v_counter v_period
WRITE counter v_counter
EXIT OK

temp_high_too_long:
# SMS should be sent only once –
# (the flag v_alarm will become 1 at the next run)
IF EQ v_alarm 0
    SENDSMS phone1 txt
```

```
WRITE alarm_temp_high 1
PARAMLOG temperature txt
EXIT OK

DEF txt "0B-215 (11) – avaria, tmp *VAR(v_temp)* > *VAR(temp_limit)*"
DEF phone1 "01234567" ; phone number for the SMS can be specified here
```

FAQ

Q: Where should I place my taskfiles on the SD card?

A: Place your taskfiles in the TASKS folder in the root of the SD card. The SD card must be formatted as FAT or FAT32. Files can have any name and extension except .OBJ, .MAP, .CNF (e.g., task.txt), and can also be organized in subfolders.

Q: How do I know if my taskfile has syntax errors?

A: When MC252 loads taskfiles, it checks them for errors. If errors are detected, the file won't be loaded. You can check the loading status via Modbus registers (see Modbus Register Map, registers 2020-2023). The system will indicate the error type and line number where the error was found.

Q: Will my program survive a power cycle or firmware update?

A: The program stored in internal memory is retained at power downs or restarts. It may be deleted during firmware updates, due to data formats of the logic execution engine updates. However, if you keep the SD card with taskfiles installed, the program will be automatically reloaded after the update.

Q: How do I clear the program from internal memory without resetting all settings?

A: Enter Setup Mode via Modbus and write command 40959 to register 120. This clears only the internal logic memory while preserving your device configuration. If the SD card with the taskfiles is installed, the program will be automatically reloaded.

Q: What happens if my task takes longer than the UPDATE interval?

A: If the next run is scheduled before the current one finishes, it will be delayed and executed as soon as possible. If multiple runs are delayed, they will be skipped, which could affect the calculations (like counters). In this case, consider optimizing long operations, splitting the task in two or increasing the UPDATE interval.

Q: Can I share data between multiple taskfiles?

A: Yes. Use memory-sourced parameters (like MEMTEMP or MEMBAT) mapped to the same Modbus address in different taskfiles. Only one taskfile should write to the shared parameter (the "writer"), while others can read from it.

Q: How do I handle decimal/fractional values if variables are integers?

A: Use fixed-point arithmetic by multiplying fractional values by a power of 10. For example, to work with 2 decimal places, multiply values by 100. Parameter types like F32EP2R automatically handle this conversion when reading/writing. However, be careful with multiplication, division and root operations as they affect the decimal point position.

Q: What's the difference between MEMTEMP, MEMBAT, MEMFLASH, and MEMFILE?

A:

- MEMTEMP - RAM space, data is lost at powerdown or controller reset, small area size but excellent write speed;
- MEMBAT - memory area powered by internal clock battery, very limited size but good write speed;
- MEMFLASH - controller settings area, data retains at reset, small size and usually read-only;
- MEMFILE - extendable file area, requires memory card with the taskfiles to be present, average speed but excellent area size.

Q: Why does my IF check skip the command even when the condition seems true?

A: The IF check treats unknown values as FALSE. If your variable hasn't been initialized or a READ operation failed, the value may be unknown. Use ISKNOWN to verify the variable state before checking it.

Q: How do I debug my taskfile program?

A: Use SYSL0G commands to write diagnostic messages to the system log. You can also map intermediate values to Modbus registers using memory parameters (e.g., MEMTEMP H 5000) and read them via Modbus.

Q: Can I use the same Modbus address for different parameter types?

A: Yes, for the parameters of the same size with the same address. Memory-stored parameters in that case will share the same memory area. This is useful for transferring data between taskfiles. However, MEMFILE parameters (stored on SD card) don't support the sharing feature.

Q: What happens when an error occurs during program execution?

A: Execution jumps to the error handler. If TRYCALL was used, it jumps to the specified handler label. Otherwise, it looks for the nearest upper-level handler, and at last, for the onerror label in the taskfile. If no handler is available, the error is logged and the run exits with the error code.

Q: How do I send SMS only once when an alarm condition is triggered?

A: Use a flag variable stored in persistent memory (like MEMBAT) to track whether the SMS was already sent. Check the flag before sending. Set it after sending and clear the flag after the alarm condition ends. See Example 3 for a working implementation.

Q: How do I check the time interval which the event is active for?

A: Either use a counter (memory-stored parameter to increment at each update period while the event continues and to reset to 0 after the event stops) or a timestamp (variable to store a clock from the MC252 registers at the event start and to be cleared at the event end).

Q: How do I check if the task is run for the first time after reset or after the power-off?

A: A MEMTEMP flag, if checked (and then set to 1 in case it was not set), can tell whether the task run is the first after the power cycle. A MEMBAT or a MEMFILE flag can similarly tell if it is the very first run.

Q: How do I scan an array? How do I make a code part executed a set number of times?

A: Use a variable as a loop index. Check before or after the cycle code. Use 1 or more G0 operators to continue the cycle or to end/break it. Do not forget to increment or decrement the index variable inside.

Q: Can I make a repeated code part execution with a complex loop end condition?

A: Yes, using IF and G0. However, it is not recommended without explicit timeouts. The loop which is not limited by a constant can turn infinite or in other way exceed the task configured update time. It could cause this task update skips or other task lags.

Q: Can I call a function from a function? How many nesting levels are there?

A: Yes, you can CALL or TRYCALL a function, which in turn calls some other function. The nesting stack size equals the number of RETURN operators in the taskfile. Because of the limited stack size, the recursion is not recommended.

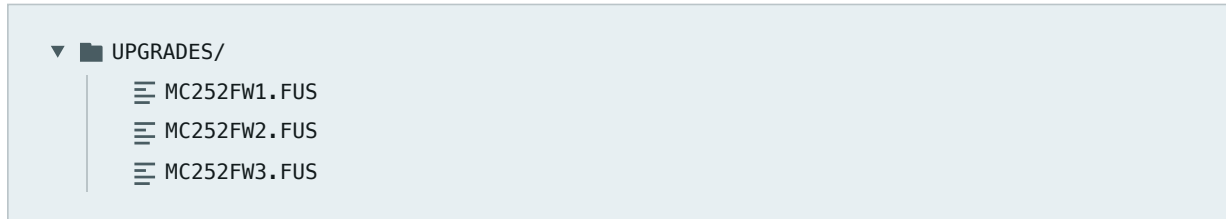
Need Help?

For technical support and assistance:

- Email: support@overvis.com
- Support portal: www.overvis.com/support

Firmware Update

To update the firmware, Overvis MC252 uses the MC252FW1.FUS, MC252FW2.FUS or MC252FW3.FUS file in the UPGRADES folder on the memory card formatted in FAT or FAT32 format.



Before you start

Data Safety

- Automatic firmware update **does not** erase your device configuration or Modbus settings.
- Manual update to a newer version in most cases retains the settings, unless specifically warned so in the firmware description.
- Automation tasks, settings, logs and historical data stored on the SD card **are not** erased by an update.

Power Supply

Ensure the device has a stable power supply during the update process. Do not power off the device while the update is in progress (usually 2-4 minutes).

Recoverability

The update process faults (e.g. caused by power losses or memory card failure) are recoverable, by repeating the update until it is complete. The update file passes multi-stage check before the update starts. In case of memory card failure and file damage, the update can be completed by replacing the card with the file manually.

Checking current version

Before updating, check your current firmware version to see if an update is necessary. You can find the version number:

- At the bottom of the **Web Interface** pages, or on the Firmware panel of the Administration tab.
- On the device display during the startup sequence.
- By reading Modbus register 1.

Updating from the file can be done in the firmware update mode.

Getting the update file into MC252

The update files can be written to the MC252 memory card in 5 ways:

1. Automatically via Internet

MC252 is by default configured to periodically check the updates server and download the newer firmware, when Internet connection is available.

Note: Ensure your device has a valid Internet Gateway and DNS server configured in the Ethernet settings, or has a GSM/LTE Internet access.

2. Download manually using WEB interface

2.1 Access the device using the browser (see Web Interface); 2.2 Open Administration page; 2.3 If the newer firmware is available on the server, and the download has not started automatically, the Download Update button will appear on the page; 2.4 Press the Download Update button to start the download process; 2.5 The download progress will be displayed on the page;

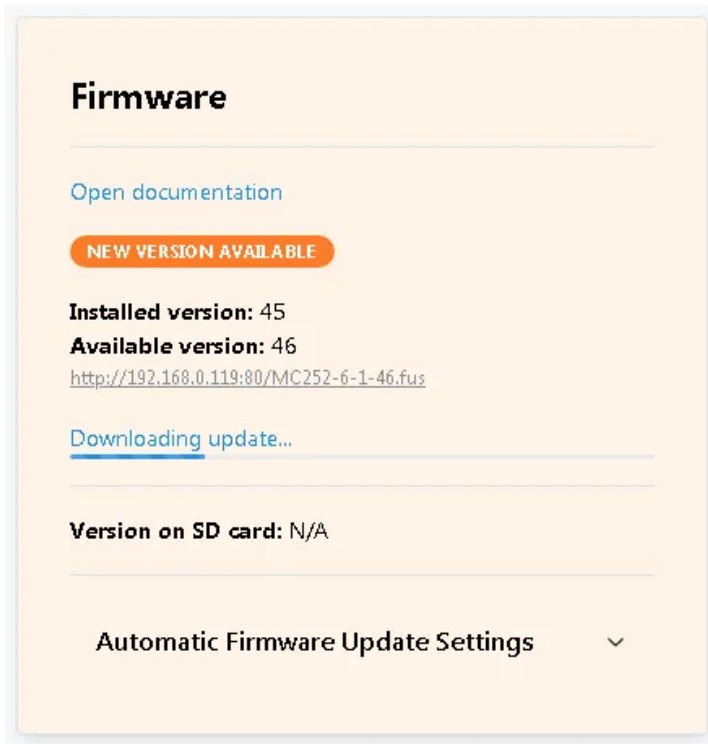


Fig 1 - The firmware update download process

2.6 After the download completion, the file will be checked. This may take a minute;

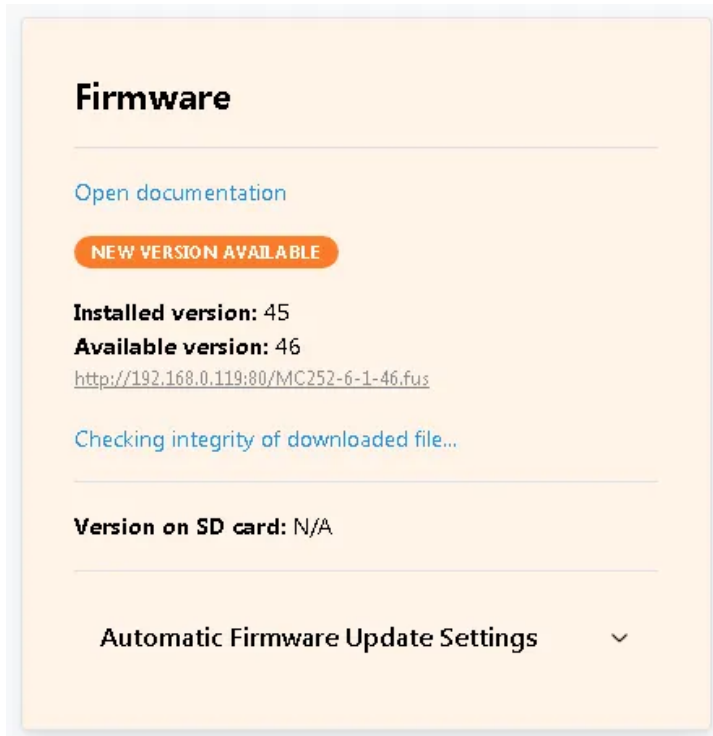


Fig 2 - The downloaded firmware update file check

2.7 After the file check, the update prompt with **Install Update** button will be displayed, confirming that the file suits.

3. Download manually using MODBUS commands

3.1 connect to the device via MODBUS and enter the setup mode (see Configuration via Modbus); 3.2 to verify the version of the downloaded update file, read the header line in registers 2030 - 2061; 3.3 check if the file download is in progress by reading register 2004; 3.4 to download the newer version from the cloud server, write the value 64 893 into the control command register 120; 3.5 control the file download by reading registers 2004 - 2005; 3.6 after download completion, double-check the version of the downloaded file in registers 2030 - 2061.

4. Upload from the client device using WEB interface

4.1 access the device using the browser (see Web Interface); 4.2 open Files page (if the files are not displayed there, make sure the memory card is properly installed); 4.3 select the UPGRADES folder (if the folder is absent, create it in the root of the files tree); 4.4 make sure the newer firmware file on the client device is named MC252FW2.FUS; 4.5 press the upload button (the blue upwards arrow button) and select the new firmware file at the client device; 4.6 wait until the file upload finishes; 4.7 open Administration page, the downloaded file check status (and the **Install Update** button, if the file fits) should be available there, as shown above.

5. **Write file to SD card on a personal computer or other external device and put the card into the MC252** (see below for a way to actually update the firmware from a file with a physical access to MC252).

Updating MC252 from the file

MC252 firmware can be updated from the file on the memory card in 4 ways:

1. Automatically

MC252 is by default configured to start update from the newer firmware file at a night hour (default is 03:00 AM).

2. Update manually using WEB interface

2.1 Access the device using the browser (see Web Interface); 2.2 Open Administration page; 2.3 If the newer firmware file is available on the memory card, the Install Update button will appear on the page;

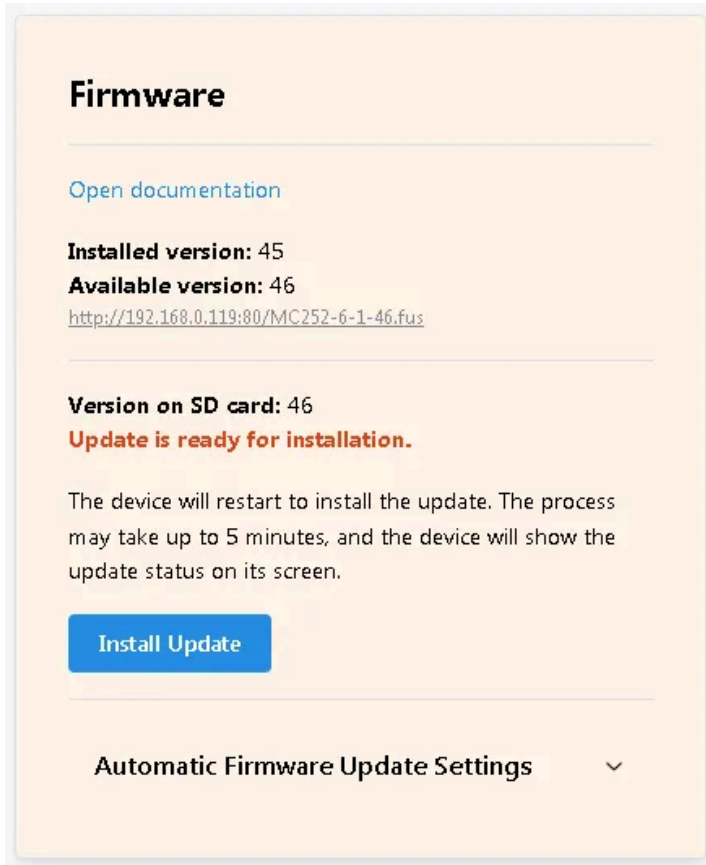


Fig 3 - The firmware update prompt

2.4 Press the Install Update button to start the firmware update process.

3. Update manually using MODBUS commands

3.1 connect to the device via MODBUS and enter the setup mode (see Configuration via Modbus); 3.2 to verify the version of the downloaded update file, read the header line in registers 2030 - 2061; 3.3 to start the update from the file, write the value 65397 into the control command register 120.

4. Manually

By entering the firmware update mode and selecting the firmware update (see below).

Firmware update process

At the update command the device will automatically restart and enter update mode. Wait for the firmware update, the process may take 2 to 4 minutes. The device remains offline while in the update mode.

Firmware update mode

MC252 can be put into firmware update mode during device power up or restart.

The firmware update mode may be entered automatically (e.g. by a web-interface command or in case of updating failure) or manually (or if the button R has been pressed during startup. The R button is the recessed button located on the front panel of the device).

Table 1 - Manually entering the firmware update mode

No.	Step	R button	Display	Time	Remark
1	Initialization	Pressed		1.5 s	To cancel entering the update mode, release R button
2	Notification about entering the update mode	Pressed	Entering upgrade mode	5 s	To cancel entering the update mode, release R button
3	Offer to enter the update mode	Pressed	To enter upgrade mode release button	2 s	Release the button at this step in order to enter the update mode
4	Protection against incidental pressing	Pressed			Holding the button pressed will cancel entering the update mode

After entering this mode manually, the update file has to be selected. To cancel the updating, cut off the power supply of MC252 or wait until the update mode is exited automatically due to file selection absence.

Table 2 – Update file selection

No.	Step	R button	Display	Time	Remark
1	Finding the available files		Searching for upgrade files...	(depends on the files found)	
2	Offering of the update file	Released	(name and version of the update file)	5 s	To select a file, press and release R button at this step
3	Offering all the update files	Released		(depends on the found files)	Step 2 is repeated for each update file
4	Repeating the offer	Released		(depends on time of step 3)	Steps 2 - 3 are repeated 3 times

No.	Step	R button	Display	Time	Remark
5	Protection against unintentional update mode entrance	Released			The absence of selection causes the update mode exit

At automatic update mode entrance (or after the manual selection of the file), the firmware update is started.

Table 3 – Firmware update

No.	Step	R button	Display	Time	Remark
1	Update start		Firmware upgrade	2 – 10 s	
2	Update process		(progress bar shows the process of updating)	(depends on the update file)	
3	Updating is successfully completed		Firmware upgrade success	3 s	
4	Startup of the firmware				

The errors and warnings during the updating process are shown on the display.

ATTENTION

IN CASE OF A CRITICAL ERROR DURING THE UPDATE, THE OPERATION OF THE MC252 DEVICE IS NOT POSSIBLE. REPLACEMENT OF MEMORY CARD AND/OR FILE AND MANUAL UPDATE PROCESS REPEAT ARE REQUIRED TO RECOVER THE DEVICE. IN CASE THE PROBLEM REPEATS, PLEASE CONTACT THE MANUFACTURER.

If the critical error is met, the indication of a critical error is made during an hour, after that the MC252 device automatically restarts. If an error is a result of an incidental event, the firmware will be restored from the file on the memory card.

Table 4 – Firmware update mode warning and error codes

Code	Warning	Automatic actions	Remark
2	Firmware cannot be started	Initialization of emergency update mode: autostart updating from the selected file or from the first available file (if any)	The warning is a result of other errors and is automatically corrected using the available update files
3	Error during the firmware update process	Similar to No.2, but the file with an error has less priority.	The error is automatically corrected using the available update files

Code	Warning	Automatic actions	Remark
4	The update files are not available	Escape from the update mode and start the available firmware	MC252 can continue operation, but the update file should be put into the UPGRADES folder on the memory card in order to update firmware
5	Emergency - firmware cannot be started	Waiting for manual restart of the device or restart automatically in 1 hour	The error is met after a triple emergency mode entrance as a result of other errors. The correct firmware update file should be put into the UPGR ADES folder on the memory card. If the error is repeated, contact the manufacturer
6	Emergency - error during the firmware update process	Waiting for manual restart of the device or restart automatically in 1 hour	The error is met after a triple emergency mode entrance as a result of other errors. The correct firmware update file should be put into the UPGR ADES folder on the memory card. If the error is repeated, contact the manufacturer
7	Emergency - no available update files, and the firmware cannot be started	Waiting for manual restart of the device or restart automatically in 1 hour	The error is met after a triple emergency mode entrance as a result of other errors. The correct firmware update file should be put into the UPGR ADES folder on the memory card. If the error is repeated, contact the manufacturer

Troubleshooting & FAQ

Q: Will I lose my configuration or data during an update?

A: No, the firmware update to a newer version process is designed to preserve your device configuration, Modbus settings, and historical data. However, downgrading the firmware may revert the settings to factory defaults. In either case, backing up critical data is always recommended.

Q: The "Download Update" button is not appearing in the Web Interface.

A: This can happen if:

- The device is already running the latest firmware.
- The device does not have Internet access. Check your Gateway and DNS settings.
- The update server is temporarily unreachable.
- The memory card is absent/full/faulty.

Q: The firmware does not update automatically.

A: This can happen if:

- The device is already running the latest firmware.
- The device does not have Internet access. Check your Gateway and DNS settings.
- The update server is temporarily unreachable.
- The memory card is absent/full/faulty.

- The device is not powered on at an update hour (at nighttime by default)

Q: The device does not see the manually written file on the memory card.

A: The file should be named MC252FW2.FUS and placed into the UPGRADES folder on the FAT32 microSD memory card. Check also the following:

- The file is the correct MC252 update file downloaded from the Firmware Downloads.
- The memory card is inserted.

Q: The update process seems to hang.

A: The update process typically takes 2-4 minutes. If it takes significantly longer:

- Do not power off the device immediately.
- Check the device display for any error codes (see Table 4).
- If the device is unresponsive for over 10 minutes, try a manual restart. If the internal firmware is corrupted, the device should enter the emergency update mode from the file automatically.

Q: Can I downgrade the firmware?

A: Yes, you can install an older firmware version by manually placing the older firmware file (renamed to MC252FW1.FUS) in the UPGRADES folder on the SD card and initiating the update. However, the device settings are not always retained and may be reverted to factory defaults after firmware downgrade.

Need Help?

For technical support and assistance:

- Email: support@overvis.com
- Support portal: www.overvis.com/support

Firmware Downloads

Latest Version

Version	Release Date	MD5	Download
46 (Latest)	2026-03-20	3897439700b4ab972a3f9fe813235eff	Download

See Firmware update instructions for the ways to update the firmware.

Changelog

Version 46

2026-03-20 Download

- **New Feature:** Monitoring and logging parameters taskfile generation.
- **New Feature:** Getting an index out of task filename.
- **Improvement:** Display saving configurable via WebUI.
- **Improvement:** Modbus parameters logging settings configurable via WebUI.
- **Improvement:** Files list sorting.
- **Bugfix:** Taskfile memory parameters without Modbus mapping.
- **Bugfix:** Taskfile commands with 2 operands compilation.

Version 45

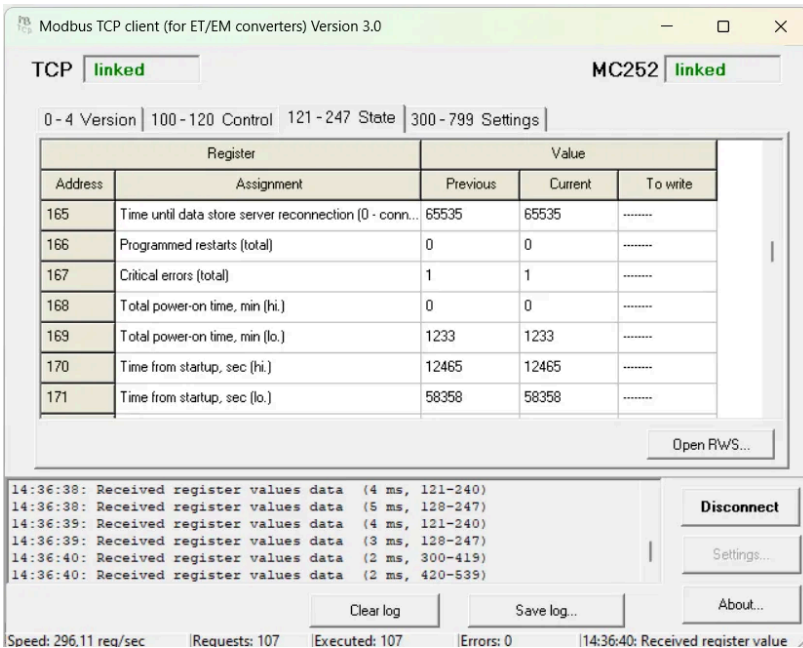
2025-10-31 Download

- **New Feature:** Initial demo release.

Additional Software

This page lists available software utilities for the Overvis MC252 RS-485 Modbus gateway. These tools are designed to help with device configuration, testing, and Modbus TCP communication.

Windows Modbus TCP Client Software (Novatek-Electro)



The Modbus TCP Client is a Windows application designed for basic communication and testing with network devices that support the Modbus TCP protocol. It enables reading and writing to device holding registers.

This application can also connect to other devices on the network, by using bridge and gateway devices (such as EM-482, EM-483, ET-485 converters, and MC251, MC252, EM-480, EM-481, EM-486 controllers).

Key features include:

- **Simple Interface:** Fast connection to devices for viewing registers and modifying values.
- **Configuration Management:** Save device configurations, including register mappings to a file.
- **Reusable Write Sets:** Save sets of register write operations for repeated use.
- **Event Logging:** Connection logs, data transfers, and errors, which can be exported to a file.

System Requirements

- **OS:** Windows 98/ME/2000/XP/Server 2003/2008/Vista/7/8/10/11
- **Network:** Network adapter
- **Storage:** 2 MB free disk space

Download: Setup_MBTCP_Client(ver3.0).exe