

TITAN v6.30

User Manual Software

Index

1	Introduction.....	5
2	FAQ - Basic Concepts	6
3	Step-by-step configuration	11
4	Configuration	12
4.1	Mobile.....	12
4.1.1	Mobile– Status.....	12
4.1.2	Mobile– Basic Settings	13
4.1.3	Mobile– Keep Online	15
4.1.4	Mobile– Basic Settings– FailOver	16
4.1.5	Mobile– Basic Settings– Operators.....	17
4.1.6	Mobile – Basic Settings– SIM Swapping.....	18
4.2	Ethernet.....	19
4.2.1	Ethernet – Basic Settings	19
4.2.2	Ethernet– DHCP Server	22
4.3	WiFi	23
4.3.1	WiFi– Basic Settings	23
4.3.2	WiFi– DHCP Server	25
4.4	Firewall	26
4.4.1	>> Firewall NAT	26
4.4.2	– Firewall Authorised IPs	28
4.4.3	– Firewall MAC Filter	30
4.4.4	– Firewall Routes	31
4.5	– Serial Settings Serial portX.....	32
4.5.1	>> Serial Settings Serial portX.....	32
4.5.2	Serial Settings– SSL Certificates	36
4.6	External Devices.....	39
4.6.1	External Devices>> Logger configuration	39
4.6.2	External Devices– Temperature Sensor	45
4.6.3	External Devices– Modbus Devices	48
4.6.4	External Devices- GPS Receiver	57
4.6.5	External Devices– Generic Serial Device	59
4.6.6	External Devices– IEC102 Meter.....	61
4.7	Other.....	66
4.7.1	Other– AT Command	66
4.7.2	Other – DynDns.....	68
4.7.3	Other– Private DynDns	69
4.7.4	Other– SMS Control	72
4.7.5	Other– Periodic autoreset.....	74
4.7.6	Other– Time Servers (NTP).....	75
4.7.7	Other– Remote console (TCP Server)	77
4.7.8	Other – SNMP.....	78
4.7.9	Other – TACACS+.....	81
4.7.10	Other – MQTT	82
4.7.11	Other – HTTP/HTTPS	84
4.7.12	Other– User Permissions.....	85
4.7.13	Other – Passwords	87
4.7.14	Other– CA Certificates	88
4.7.15	–Other – Email Configuration.....	89
4.7.16	Other– Modbus Slave	91
4.7.17	Other –TITAN Scripts	97
4.7.18	Other Connectivity Tools.....	130
4.7.20	Other– Digital I/O	133
4.7.21	Other– Custom Skin.....	136
4.7.22	Other– Custom LED	137
4.7.23	Other- SYSLOG.....	138

4.7.24	Other– Backup/ Factory	140
4.7.25	Other- Reboot.....	141
4.7.26	Other– Firmware Upgrade	142
4.7.27	Other – LDAP	143
4.8	VPN	144
4.8.1	VPN– OpenVPN Server	144
4.8.2	VPN– OpenVPN Client	146
4.8.3	VPN– IPSec	147
4.8.5	- VPN	150
5	AT Commands	152
6	New features in firmware versions.....	169
7	Sales Contact	171

General notes

The recipient considers the product to be accepted and provided without interface for the recipient's products. The documentation and/or product are provided for testing, evaluation, integration, and information purposes. The documentation and/or products are provided "as is" and may contain deficiencies. The documentation and/or products are provided without warranty of any kind, either express or implied. To the fullest extent permitted by applicable law, Webdyn further disclaims all warranties, including, but not limited to, any implied warranties of merchantability, completeness, fitness for a particular purpose, and non-infringement of third-party rights. All risk arising out of the use or performance of the product and documentation remains with the recipient. This product is not designed for use in life-supporting devices or systems where product malfunction could reasonably be expected to cause personal injury. Applications incorporating the described product must be designed in accordance with the technical specifications provided in these guidelines. Failure to follow any of the required procedures may result in malfunction or serious discrepancies in results.

In addition, all safety instructions related to the use of mobile technical systems, including GSM products, which also apply to mobile phones, must be followed. Webdyn or its suppliers, regardless of any legal theory on which the claim is based, shall not be liable for any consequential, incidental, direct, indirect, punitive or other damages (including, but not limited to, damages for loss of business profits, business interruption, loss of business information or data, or other pecuniary loss) arising out of the use or inability to use the documentation and/or the product, even if Webdyn has been advised of the possibility of such damages. The above limitations of liability shall not apply in the case of mandatory liability, for example, under the Spanish Product Liability Act, in the case of intent, gross negligence, injury to life, limb or health, or breach of a condition that is fundamental to the contract. However, claims for damages arising from the breach of a condition that is fundamental to the contract shall be limited to foreseeable damage that is intrinsic to the contract, unless caused by intentional or gross negligence or based on liability for injury to life, limb or health. The foregoing provision does not imply a change in the burden of proof to the detriment of the recipient. Subject to change without notice. The interpretation of this general note shall be governed by and construed in accordance with Spanish law without reference to any other substantive law.

Important information

This technical description contains important information for starting up and using TITAN-based equipment. Please read it carefully before you start working with the equipment. The warranty will be void if damage occurs due to failure to follow these instructions for use. We cannot accept any liability for consequential losses.

Revisions

VERSION 6.30

1 Introduction

TITAN firmware enables 2G/3G/4G devices with advanced gateway capabilities.

In addition to classic router capabilities, TITAN-based equipment allows for 4G/3G/2G gateways - RS232/485 gateways, execute AT commands via SMS (to check coverage, switch relays, read digital inputs, etc.), accept GSM data calls (CSD) to access devices such as electricity meters, autonomously read Modbus RTU or Modbus TCP devices, send temperature and distance data to the cloud, and much more.

For a better understanding of the possibilities of this equipment, we recommend reading Chapter 2 of this manual, the FAQ. After reading it, you will have a more concrete idea of what it can actually do. Once you have read the FAQ, we recommend taking a look at the application notes (PDF documents external to this document). These examples will give you a much clearer idea of the possibilities of this equipment.

We provide all users of our modems and routers with free, fast and effective technical support whenever they need it. Therefore, if you still have questions after reading this manual, please do not hesitate to contact us atiotsupport@mtxm2m.com. The same applies if you need a feature that is not included in our equipment, or if you require customisation. Let us know and we will look into it in detail.

2 FAQ – Basic concepts

■ What can I do with TITAN-based equipment?

Basically, you can do the same things you can do with a standard 4G/3G/2G router, but with the addition of a number of advanced gateway features. For example, you can provide Internet connectivity to devices connected to your Ethernet port, as well as perform NAT to connect devices connected to that port from the Internet.

■ Is it possible to provide Internet connectivity to Wi-Fi devices?

Indeed, TITAN-based devices with Wi-Fi connectivity can function as Wi-Fi access points, providing Internet connectivity to Wi-Fi client devices that connect to them. They can also be configured as Wi-Fi clients, meaning that TITAN-based devices can connect to an existing Wi-Fi access point in your infrastructure and use that connection to access the Internet.

■ Do you have serial gateways?

Yes, you can have as many 4G/3G/2G-Serial gateways as the device you are using has serial ports. These gateways can be TCP Client or TCP Server type and can all work simultaneously.

■ RS232 or RS485 serial gateways?

All devices have at least one RS232 serial port and one RS485 port, or both. Depending on the type of serial port, the gateways will be of one type, another type, or both. TITAN-based devices allow IP-RS232/485 gateways to be established, but could they be secure SSL/TLS gateways?

Yes, TITAN-based equipment allows you to configure Server and Client type IP - RS232/485 gateways with SSL/TLS security with mutual authentication by digital certificate (client/server).

■ Does the TITAN firmware also allow CSD call management? In other words, can I have a 4G/3G/2G-Serial gateway that also accepts a GSM/CSD data call for reading an electricity meter?

Yes, but with conditions. If the TITAN-based equipment is operating in 2G mode, there will be no problem. The equipment will accept the CSD call when the call is received (to perform a CSD-Serial gateway) and at the end of the call it will resume the 2G-Serial gateway. If the equipment is also configured to work in 3G or 4G mode, acceptance of the CSD/GSM call will depend on your telephone provider, as not all operators allow this. For example, if you use Movistar or Orange and have the device operating in "auto 4G/2G" mode, there will be no problem with accepting the CSD call regardless of whether the TITAN-based device is operating in 2G or 4G mode (in 4G it will failover to 2G), but with the Vodafone operator it will not do so as it is not supported.

■ I want a 4G/3G/2G router, but I need to be able to send AT commands directly to the TITAN-based device to send SMS messages, check coverage, etc. Do TITAN-based devices allow this?

Yes, all models and in several ways. It is possible to send AT commands REMOTELY from a "Telnet-type" connection, or via SSH, via Modbus TCP, via SNMP, via HTTP/HTTPS, MQTT/MQTTs, by SMS and even also

Commands embedded in the IP-Series gateways themselves. It is also possible to send AT commands LOCALLY through the Ethernet port via Telnet, SSH, Modbus TCP, SNMP, HTTP/HTTPs and through the RS232/RS485 serial port via embedded AT commands or Modbus RTU.

- Do TITAN-based devices support a Modbus TCP to Modbus RTU gateway?

Yes, it supports conversion between these protocols, so if you need to remotely access Modbus RTU devices, TITAN-based equipment will allow you to do so by acting as a Modbus TCP to Modbus RTU gateway.

- Can TITAN-based equipment also operate autonomously, reading Modbus RTU or Modbus TCP devices and sending the readings to the cloud?

Yes. TITAN-based devices can read Modbus RTU and/or Modbus TCP devices, even in combination, store the readings internally, and send them to web platforms using a JSON object via HTTP/HTTPs, MQTT/MQTTs or FTP.

- Does the JSON object sent to a web platform have a fixed format, or can it be modified to adapt to the format required by each web platform?

The data sent by TITAN devices in JSON format can be easily customised using the JSON Transformer Function Script feature. This means that data can be sent in the format required by any platform, without necessarily being in JSON format.

- What if the TITAN-based equipment is reading Modbus RTU/TCP devices autonomously? Does this prevent me from connecting sporadically and remotely to Modbus RTU/TCP devices from my office to take specific readings, change the firmware, etc.?

No. You can have the TITAN-based equipment configured to read Modbus devices autonomously and set up a sporadic direct gateway to access Modbus RTU devices from your premises.

- Do TITAN-based devices have data logger capabilities and can they store data?

Yes, they do. As mentioned above, TITAN-based devices can store many types of data for subsequent transmission to web platforms, MQTT, FTP. These include, for example, readings from Modbus RTU/TCP devices, but also temperature readings, generic RS232/RS485 serial data, IEC-102 meter readings, digital inputs and outputs, etc.

- And when and how is the data stored in the data logger sent to the web platforms?

By default, data is automatically sent to a web platform as it is obtained and stored in the internal data logger. The default sending process uses a FIFO method, but it is possible to configure it as a LIFO method, which is necessary for some applications. Stored records can be sent one at a time (default method) or configured to be sent by grouping the data into an array. It is also possible to configure (using a simple script) a schedule for sending the data.

- How are TITAN-based devices configured?

There are several ways to configure TITAN-based devices, but generally all device configuration can be done via web configuration, i.e. via the devices' internal web server.

- Is it possible to read/load a complete configuration onto the devices? This facilitates the production process when there are a significant number of devices to configure.

Yes, it is possible to perform complete configuration backups/restores (to copy the configuration from one TITAN-based device to another TITAN-based device).

- Is it possible to order TITAN-based computers preloaded with a configuration from the factory?

Yes, it is possible. However, this is only for large volumes (>500 units). Please consult your sales representative for more information.

- Do you support the DynDNS service?

Of course, it is also compatible with NO-IP. In addition, it also has the additional option of "private DNS" to send your current IP address whenever it changes or periodically to a private server, for example, your company's server. With this additional option, in addition to sending the current IP address, other data is also attached, such as coverage, technology, IMEI, status of digital inputs and outputs, etc.

- So, does TITAN-based equipment allow for the management of digital inputs and outputs?

Yes. If your TITAN-based equipment model has digital inputs or outputs, it will manage them. It is very easy to configure the internal data logger to store and send data to a web platform (via HTTP/HTTPS, MQTT/MQTTS or FTP) every time there is a change in a digital input, or to configure a time period for reading and sending their status. It is also possible to check the status of a digital input at any time using an AT command (which can be sent via Telnet, SSH, HTTP/S, SMS, MQTT/S, Modbus, SNMP, RS232, RS485, etc.). The same applies to digital outputs.

In addition, the digital inputs/outputs can be accessed from the "TITAN Scripts" section of the device, where a small script (in JavaScript) can be programmed so that the device uses these digital inputs/outputs at the user's convenience.

- Do TITAN-based devices have a clock?

Yes, they have an internal clock. This is synchronised by NTP via the Internet.

- Can TITAN-based devices be configured by SMS?

Yes, TITAN-based devices can be configured using AT commands. And AT commands (in addition to Telnet, SSH, HTTP/S, MQTT/S, Modbus, SNMP, RS232, RS485) can be sent to the device via SMS. Commands can be sent to the device to restart it, change its configuration, find out its IP address, check coverage, etc.

- Is it possible to customise the web configuration environment with company logos and images?

Yes. Any user is free to customise the web configuration environment with their logos, footers, page titles, etc. It is even possible to choose which menus your end customer can view or modify.

- Do TITAN-based devices have any indicator LEDs?

Yes, the number of LEDs depends on the model. You can choose their functionality to indicate coverage, whether the device has IP, SIM problems, etc. It is also possible to use the LEDs from TITAN Scripts, where you can program a small user JavaScript script to turn a device LED on or off depending on specific needs (for example, when a digital input is activated).

■ Is it possible to use TITAN-based equipment as a serial data logger?

Yes. TITAN-based equipment can be configured to read everything that comes through the serial port, store it in its internal data logger, and send it via HTTP/S, MQTT/S, or FTP to a platform over the Internet.

■ Can TITAN-based equipment be used to read IEC 870-5-102 meters?

Yes. With TITAN-based equipment, you can do the following regarding reading electricity meters with the IEC 870-5-102 protocol:

- Read the meter via a 4G/3G/2G-RS232/485 gateway
- Read the meter via a CSD data call
- Configure the TITAN-based equipment to read the meter autonomously every configured time period, sending the instantaneous values to a web platform (via HTTP/S, MQTT/S, or FTP) every time period and also, if desired, sending information on pricing once a day.
- In the "TITAN scripts" section, program a small JavaScript script to read the electricity meter at will (for example, continuously monitor a certain variable, obtain more accurate average values than hourly values, etc.).

■ Is it possible to configure a VPN with TITAN-based equipment?

Yes. TITAN-based equipment allows you to configure an IPSEC or OpenVPN type VPN. It is also possible to use the Zerotier VPN service.

■ What other relevant features does TITAN-based equipment have?

Throughout this manual, you will find detailed information on the various features, but in addition to what has already been mentioned in the previous paragraphs, TITAN-based equipment allows you to:

- Autoreset configuration (periodic autoreset, at a specific time, autoreset in case of IP connectivity issues, etc.)
- TACACS+ and LDAP authentication and authorisation service for HTTP, Telnet and SSH
- Secure configuration via HTTPS
- Configuration of user permissions to access the various features of the TITAN-based equipment based on the roles "admin", "user" and "guest".
- SMTP service configuration for sending emails
- Configuration as MODBUS SLAVE. This allows TITAN-based equipment to be configured and/or read by Modbus. For example, it can be used to read the status of its digital inputs via Modbus, or

obtain the time from the equipment, change digital output, etc., all from a Modbus Master device such as a PLC.

- TITAN scripts. This section allows you to write a small user script in JavaScript to, for example: read serial devices or send them customised data, read Modbus devices, read or change a digital output, send SMS, emails, SNMP traps, read an IEC 870-5-102 meter, send MQTT messages, perform PINGs, change the status of the device's LEDs, write to and read from files, send or receive files via FTP, etc.

3 Step-by-step configuration

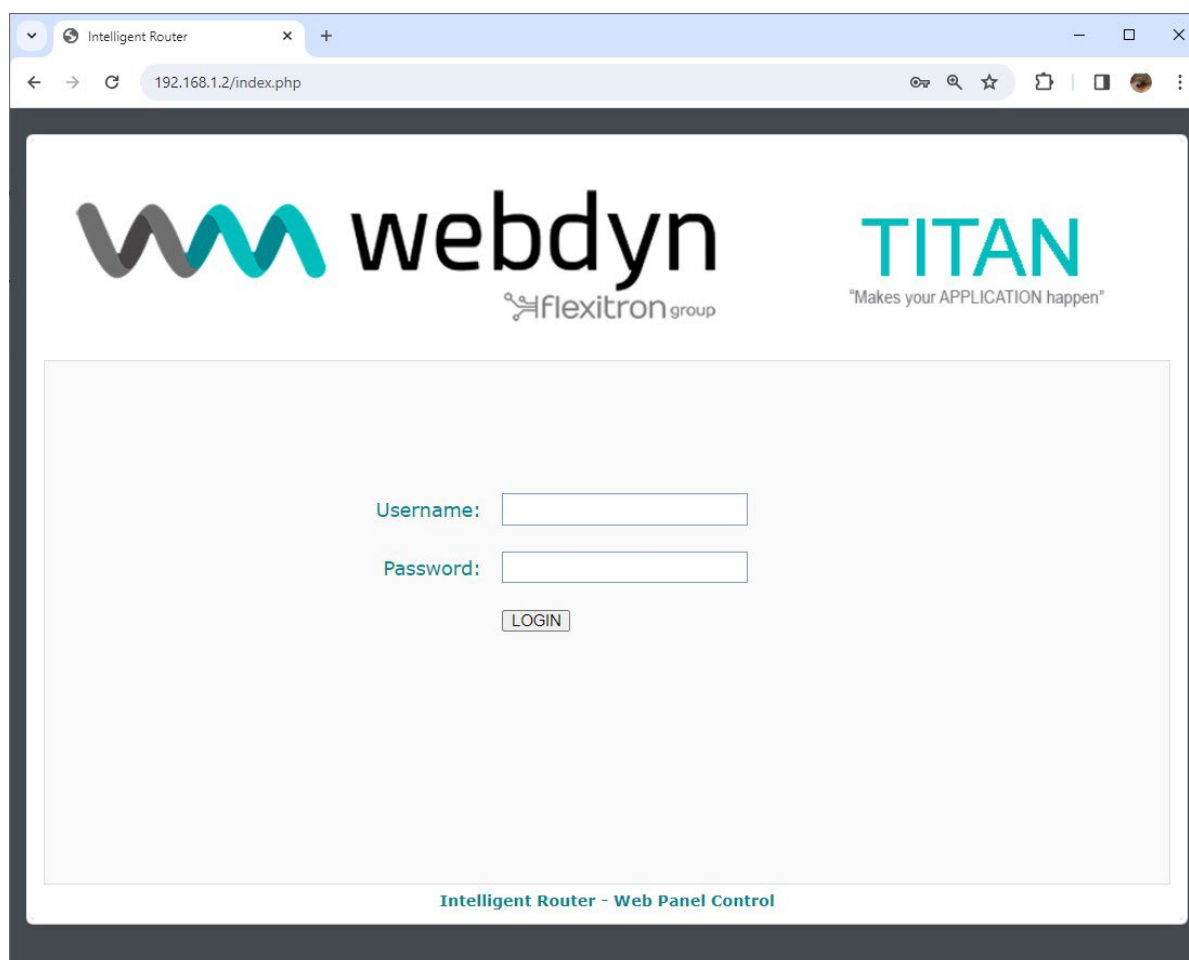
As mentioned in previous sections, TITAN-based equipment is mainly configured through its web environment.

What is required?

- A PC with a web browser (Chrome, Brave, Firefox, etc.) and an Ethernet port.
- An Ethernet cable connects the PC to the device.

Steps to access the configuration environment.

- Connect an Ethernet cable between the PC and the device.
- The PC must have a fixed IP address within the range 192.168.1.X configured on its Ethernet interface, as the default IP address of the Titan device is 192.168.1.2.
- Open a browser with the address <http://192.168.1.2>. A window like the following should appear:



4 Configuration

4.1 Mobile

The Mobile section refers to everything related to the device's 4G/3G/2G settings, from connection status and network configuration parameters to connection monitoring.

4.1.1 Mobile – Status

This screen displays the general status of the device.

- **Firmware Version:** Device FW version
- **WAN Mobile IP:** WAN IP address (IP address assigned to the 4G/3G/2G connection) if available.
- **GSM Module:** Indicates the manufacturer and model of the device's internal GSM module.
- **Network (2G/3G/4G):** Indicates whether the current WAN connection is using the 2G (GPRS), 3G or 4G network.
- **Signal Strength:** Indicates the signal strength. 0=none, 31=maximum.
- **Extra signal info:** For 3G and 4G connectivity, indicates the RSCP/ECN0 and RSRP/RSRQ.
- **Internal Temperature:** Shows the internal temperature of the processor.

The screenshot displays the 'webdyn' interface, powered by TITAN and part of the Flexitron group. The main content area is titled 'Mobile Status' and lists the following information:

- Firmware version:** 5.2.6.09 (Webdyn EasyModem)
- WAN Mobile IP:** 88.31.40.147 (WAN IP (2G/3G/4G) Network)
- GSM Module:** EC21, Revision: EC21EFAR06A05M4G
- IMEI:** 869101054287764 (Device identification)
- SIM:** SIM-1 (SIM READY) (Used SIM and status)
- Network (2G/3G/4G):** 4G (MOVISTAR) (Used network at this moment)
- Signal Strength:** 21 (-71dbm) (Signal Strength (0 ... 31))
- Extra signal info:** Rsrp: -99dBm | Rsrq: -9dB (For 3G & 4G Network)
- Temperature:** 48.0 (Temperature of internal processor (°C))

A signal strength bar chart is shown below the signal strength value, indicating a moderate signal level. The left sidebar contains a navigation menu with categories: Mobile (Status, Basic Settings, Keep Online), Ethernet (Basic Settings), Firewall (Authorized IPs), Serial Settings (Serial Port1-RS232, Serial Port2-RS485, SSL Certificates), External Devices (Logger configuration, ModBus Devices, Generic Serial Device, Temperature Sensor, IEC102 Meter, W-MBus), and Other (AT Command, DynDns, Private DynDns, Sms control).

4.1.2 Mobile – Basic Settings

This section allows you to configure the parameters for the Mobile WAN 4G/3G/2G connection. You will need to know some information about your SIM card, such as the APN, login and password. Your provider should provide you with this information.

- **Mobile WAN:** If you require IP connectivity (4G/3G/2G), configure this option as "Enabled". If you are not going to use a SIM card, configure it as "Disabled".
- **SIM Mode:** depending on the device model, there is either 1 SIM slot or 2 SIM slots available. If there are 2 SIM slots: "SIM1" the router will use SIM1. "SIM2" the router will use SIM2. "SIM1 + SIM2 (backup)", the router will use SIM1 and, in case of problems, it will use SIM2.
- **SIM1 APN:** APN of the SIM1 operator. You should check with your GSM provider.
- **SIM1 Username:** Username of the SIM1 operator. You must check this with your GSM provider.
- **SIM1 Password:** Password for the SIM 1 operator. You must check this with your GSM provider
- **SIM1 PIN:** If your SIM1 card has a PIN, you must specify it here.
- **SIM1 Auth:** Authentication method: "NONE, PAP", CHAP, AUTO (PAP or CHAP)".
- **SIM2 APN:** APN of the SIM2 operator. You must check this with your GSM provider.
- **SIM2 Username:** Username of the SIM2 operator. You must check this with your GSM provider.
- **SIM2 Password:** Password for the SIM 2 operator. You must check this with your GSM provider
- **SIM2 PIN:** If your SIM2 card has a PIN, you must specify it here.
- **SIM2 Auth:** Authentication method: "NONE, PAP, CHAP, AUTO (PAP or CHAP)".
- **Network selection:**

"Auto (4G/3G/2G)": the device will use 4G if coverage is available, or 3G or 2G if not. "Auto (4G/2G)": the device will use 4G if coverage is available, or 2G if not. "4G": the device will always use the 4G network. If there is no 4G coverage, the device will not switch to 2G or 3G. "3G": the device will use the 3G network in all cases. If there is no 3G coverage, the device will not switch to 2G. "2G": the device will use the 2G network in all cases.
- **DNS selection:**

"Get DNS from Operator" makes the DNS used by the computer those assigned by the telephone operator.

"Selected DNS servers" makes the DNS servers used be those specified in the DNS1 and DNS2 parameters discussed below.
- **DNS1 and DNS2:** DNS servers for domain name resolution. We recommend using Google's 8.8.8.8 and 8.8.4.4 or those specified by your provider. These DNS servers will be used if the FailOver feature is selected when data output occurs via the 4G/3G/2G WAN interface.
- **Remote webserver management:** If you tick the box, you will be able to access the device's web configuration remotely via its public IP address (the one indicated in Mobile→ Status). You can change the port from the Other→ Http / Https menu.



webdyn
flexitron group

powered by TITAN

★ Mobile

- Status
- Basic Settings
- Keep Online

★ Ethernet

- Basic Settings

★ Firewall

- Authorized IPs

★ Serial Settings

- Serial Port1-RS232
- Serial Port2-RS485
- SSL Certificates

★ External Devices

- Logger configuration
- Modbus Devices
- Generic Serial Device
- Temperature Sensor
- IEC102 Meter
- W-MBus

★ Other

- AT Command
- DynDns
- Private DynDns
- Sms control
- Periodic Autoreset
- Time Servers
- Remote Console
- Snmp
- Tacacs+
- Mqtt
- Http / Https
- User Permissions
- Passwords Web UI
- CA Certificates
- Email Config

▶ Mobile ▶ Basic Settings

Mobile WAN

Enabled (IP active)

Enable Wireless WAN interface

Sim Mode

SIM1 + SIM2 (backup)

Sim selection

SIM1 APN:

movistar.es

APN of SIM card 1

SIM1 Username:

MOVISTAR

Username of SIM card 1

SIM1 Password:

Password of SIM card 1

SIM1 Pin:PIN of SIM card 1

SIM2 APN:

movistar.es

APN of SIM card 2

SIM2 Username:

MOVISTAR

Username of SIM card 2

SIM2 Password:

Password of SIM card 2

SIM2 Pin:PIN of SIM card 2

Authentication:

Auto

Authentication method

Network selection:

Auto (4G/3G/2G)

Network selection

DNS selection:

Get DNS from Operator

DNS1:

8.8.8.8

Preferred DNS1

DNS2:

8.8.4.4

Preferred DNS2

Remote webserver management:

☒

Enable remote webserver management

SAVE CONFIG

Additional notes.

- Once you have finished configuring, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the device for the new changes to take effect.

webdyn | 14

contact@webdyn.com | webdyn.com
V02.08 subject to change | Webdyn © by Flexitron

4.1.3 Mobile – Keep Online

From this screen, you can configure a PING to ensure the device's connectivity. If the PING fails on the configured occasions, the 4G/3G/2G connection will restart.

- **Enabled:** tick the box to allow the TITAN-based device to send a periodic PING to check connectivity.
- **Ping Server:** specify the IP or DNS address of the server to ping.
- **Period:** indicate how many minutes it takes between PING checks.
- **Timeout:** indicate the number of seconds the device should wait for a response to the PING before considering the PING to have failed.
- **Retries:** indicate the number of PING retries that should be performed before considering connectivity problems and restarting the 4G/3G/2G session.
- **Retry period:** indicates the number of minutes that must elapse between retries.



The screenshot shows the webdyn configuration interface. At the top, the webdyn logo is displayed with the text 'powered by TITAN' and 'flexitron group'. On the left, a sidebar menu lists various settings categories: Mobile (Status, Basic Settings, Keep Online), Ethernet (Basic Settings, DHCP Server), Firewall (NAT, Authorized IPs), External Devices (Logger configuration, ModBus Devices), and VPN. The main content area is titled 'Mobile > Keep Online'. It contains several configuration fields: 'Enabled' with a checked checkbox and a description 'Enable PING method for keep online Wan Session'; 'Ping Server' with a text box containing '8.8.8.8' and a label 'IP or DNS address'; 'Period' with a text box containing '5' and a label 'Minutes between pings (1 ... 1440)'; 'Timeout' with a text box containing '10' and a label 'Timeout in seconds (5 ... 20)'; 'Retries' with a text box containing '1' and a label 'Number of retries (0 ... 9)'; and 'Retry period' with a text box containing '1' and a label 'Minutes between retries (1 ... 1440)'. At the bottom left of the main area is a 'SAVE CONFIG' button.

Additional notes.

- Once you have finished configuring the settings, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the device for the new changes to take effect.

4.1.4 Mobile – Basic Settings – FailOver

This configuration screen can be accessed from the button located at the bottom of the Mobile→ Basic Settings configuration section.

Allows you to configure the device's failover feature. For example, to configure the data output of the TITAN-based device via its Wi-Fi or Ethernet interface and, in the event of a failure, activate the 4G/3G/2G interface so that data output occurs there until the Wi-Fi or Ethernet interface regains connectivity.

- **Enabled:** tick the box to activate the failover functionality.
- **IP1:** to check the connectivity of the failover interface used (Ethernet or Wi-Fi), the TITAN-based device periodically performs a PING on this IP. In case of failure, the ping is retried on IP2.
- **IP2:** to check the connectivity of the failover interface used (Ethernet or Wi-Fi), the TITAN-based device performs a PING on this IP2 if IP1 fails. In the event of a failure, the WAN interface (4G/3G/2G) is activated for use while the connectivity failure lasts.
- **Interface:** communications interface (Ethernet or Wi-Fi) used as the primary data output interface.
- **4G/3G/2G interface:** behavior of the 4G/3G/2G backup interface. You can choose between "On when failover" or "Always on" mode. In "On when failover" mode, the 4G/3G/2G WAN interface is off and only when the primary interface fails (Wi-Fi or Ethernet interface) is it activated and then used. In "Always on" mode, the 4G/3G/2G WAN interface is always active and is used when the primary interface (Wi-Fi or Ethernet) fails. As the 4G/3G/2G WAN interface is already active, the switchover is faster.

The screenshot shows the webdyn configuration interface. At the top, the webdyn logo is displayed with the text "powered by TITAN" and "flexitron group". On the left, a sidebar menu lists various configuration sections: Mobile (Status, Basic Settings, Keep Online), Ethernet (Basic Settings, DHCP Server), Firewall (NAT, Authorized IPs), External Devices (Logger configuration, ModBus Devices), and VPN. The main content area is titled "Mobile Basic Settings WAN Failover". It contains several configuration fields: "Enabled" (checkbox), "IP1:" (text input with value 8.8.8.8), "IP2 (backup):" (text input with value 8.8.4.4), "Interface:" (dropdown menu showing "Ethernet (eth0)"), "Period:" (dropdown menu showing "5 seconds"), and "4G/3G/2G interface:" (dropdown menu showing "On when failover (slow mode)"). To the right of these fields, explanatory text is provided for each: "Enable failover.", "IP address for check Ethernet/Wifi connection. **Mandatory.** Example: 8.8.8.8", "IP or DNS address for check Ethernet/Wifi connection. Example: 8.8.4.4", "Interface for failover", "Connectivity checking period", and "Behaviour of 4G/3G/2G interface". At the bottom left of the main area is a "SAVE CONFIG" button.

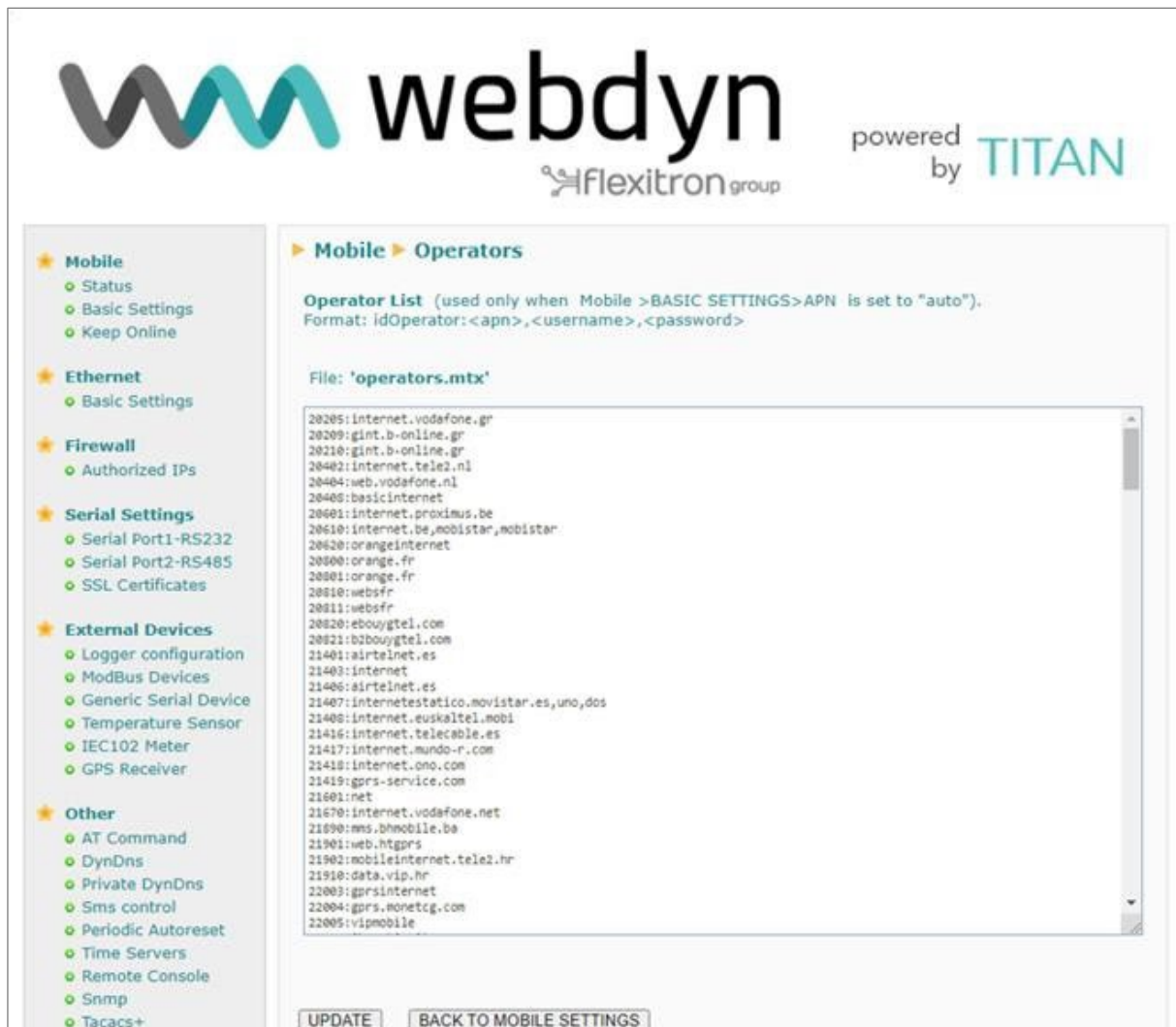
4.1.5 Mobile – Basic Settings – Operators

This configuration screen can be accessed via the button located at the bottom of the Mobile configuration section. ☐ Basic Settings section.

It allows you to configure the APN, user and password to be used in case of automatic APN. You can use the APN in automatic mode by specifying the text "auto" (without quotation marks) in the APN fields of the Mobile Basic Settings section.

The "operators.mtx" file must have the following format:

<ID_Operator>:<apn>,<username>,<password> (where <username>,<password> are optional)



Additional notes.

- Once you have finished configuring the settings, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the device for the new changes to take effect.

4.1.6 Mobile – Basic Settings – SIM Swapping

This configuration screen can be accessed from the button located at the bottom of the Mobile→ Basic Settings section (by clicking on the button labelled "SIM swapping config"). This configuration will only be available on devices that have DUAL SIM.

The screenshot shows the webdyn TITAN configuration interface. The top header includes the webdyn logo and the TITAN logo with the tagline "Makes your APPLICATION happen". The left sidebar contains a menu with categories: Mobile (Status, Basic Settings, Keep Online), Ethernet (Basic Settings, DHCP Server), Wifi (Basic Settings, DHCP Server), and Firewall (NAT, Authorized IPs, MAC Filter, Routes). The main content area is titled "Mobile > Basic Settings > SIM Swapping". It contains the following settings:

- Main SIM:** A dropdown menu currently set to "SIM1". To its right is the text "Select main SIM after reboot".
- Swap by network:** A text input field containing "30". To its right is the text "(Seconds) Registration fails. (0, 30 ... 1800). Default 30 seconds. 0 = disabled".
- Swap by IP:** A text input field containing "90". To its right is the text "(Seconds) No IP (0, 90 ... 1800). Default 90 seconds. 0 = disabled".
- Swap by coverage:** A text input field containing "0". To its right is the text "(Seconds) Low coverage (0, 600 ... 86400). Default 0. 0 = disabled".
- Swap by keepalive:** An unchecked checkbox. To its right is the text "Swap SIM if keep alive ping fails".

At the bottom left of the main content area is a button labeled "SAVE CONFIG".

- **Main SIM:** If you have selected a Dual SIM configuration in the "Mobile→ Basic settings" menu, i.e. a "SIM1 + SIM2" configuration, this menu allows you to select the main SIM, i.e. the SIM that will start operating first after a reset of the Titan router.
- **Swap by network:** allows you to select a time in seconds during which, if the Titan router is unable to register on the network within the specified seconds, the SIM swap process will begin. A value of "0" indicates that this feature will not be used. The minimum value (default value) is "30" seconds and the maximum value is "1800" seconds.
- **Swap by IP:** allows you to select a time in seconds during which, if the Titan router is unable to obtain an IP address from the network within the specified seconds, the SIM swap process will begin. A value of "0" indicates that this feature will not be used. The minimum value (default value) is "90" seconds and the maximum value is "1800" seconds.
- **Swap by coverage:** allows you to select a time in seconds at which, if the Titan router is continuously registered on the network but with critical coverage, the SIM card swap process is initiated. The value is "600" seconds and the maximum value is "86400" seconds. The default value is "0", disabled.
- **Swap by keep alive:** if this option is selected, if the Titan router restarts the IP connection due to a ping failure of the keep online function (which can be activated from the "Mobile→ Keep online" menu), the SIM card will be changed. This option is disabled by default.

4.2 Ethernet

The "Ethernet" configuration section refers to the configuration of the Ethernet ports on TITAN-based equipment.

4.2.1 Ethernet – Basic Settings

This section allows you to configure the basic network parameters for the Ethernet connection. This menu varies slightly depending on whether the TITAN-based device has one or more Ethernet ports.

TITAN-based devices with 1 Ethernet port:



The screenshot displays the webdyn TITAN configuration interface. At the top, the webdyn logo (a stylized 'w' with a blue and green wave) and the TITAN logo (in blue) are visible, along with the Flexitron group logo. The breadcrumb navigation path is "Ethernet > Basic Settings > LAN Port". On the left, a sidebar menu lists categories: Mobile (Status, Basic Settings, Keep Online), Ethernet (Basic Settings, DHCP Server), Firewall (NAT, Authorized IPs, MAC Filter, Routes), and Serial Settings. The main content area contains configuration fields for the LAN Port: IP Address (192.168.1.2), IP Subnet Mask (255.255.255.0), IP Gateway (blank), DNS 1 (blank), and DNS 2 (blank). Each field has a corresponding label to its right: Local IP LAN, Local Mask, IP Gateway (Leave blank if not used), DNS Server 1, and DNS Server 2. At the bottom of the form are two buttons: "SAVE CONFIG" and "REFRESH".

- **IP Address:** static local IP address of the Ethernet interface (default 192.168.1.2)
- **IP Subnet Mask:** subnet mask (default 255.255.255.0)
- **IP Gateway:** IP address of the outgoing gateway. Leave blank if no gateway is to be used.
- **DNS1:** primary DNS server
- **DNS2:** secondary DNS server

TITAN-based devices with multiple Ethernet ports:

The screenshot displays the webdyn TITAN configuration interface. At the top, the webdyn logo (a stylized 'w' with a blue and green wave) and the TITAN logo (in blue) are visible, along with the Flexitron group logo and the tagline "Makes your APPLICATION happen".

The left sidebar contains a navigation menu with the following sections:

- ★ Mobile
 - Status
 - Basic Settings
 - Keep Online
- ★ Ethernet
 - Basic Settings
 - DHCP Server
- ★ Firewall
 - NAT
 - Authorized IPs
 - MAC Filter
 - Routes
- ★ Serial Settings
 - Serial Port1-RS232
 - Serial Port2-RS232
 - Serial Port3-RS485
 - Serial Port4-TTL
 - SSL Certificates
- ★ External Devices
 - Logger configuration
 - ModBus Devices
 - Generic Serial Device
 - Temperature Sensor
 - RS485 Master

The main content area is divided into two sections:

WAN Port Configuration: This section is titled "Ethernet > Basic Settings > WAN Port". It includes the following fields:

- IP mode: A dropdown menu set to "static".
- Static or dynamic IP (dhcp): A label for the IP mode.
- IP Address: A text box containing "192.168.10.2".
- Local IP LAN: A label for the IP address.
- IP Subnet Mask: A text box containing "255.255.255.0".
- Local Mask: A label for the subnet mask.
- IP Gateway: An empty text box.
- IP WAN for Ethernet: A label for the gateway.
- DNS 1: An empty text box.
- DNS Server 1: A label for the primary DNS server.
- DNS 2: An empty text box.
- DNS Server 2: A label for the secondary DNS server.
- Remote webserver management: A checkbox that is currently unchecked.
- Enable remote webserver management: A label for the checkbox.

LAN Ports Configuration: This section is titled "Ethernet > Basic Settings > LAN Ports". It includes the following fields:

- IP Address: A text box containing "192.168.1.2".
- Local IP LAN: A label for the IP address.
- IP Subnet Mask: A text box containing "255.255.255.0".
- Local Mask: A label for the subnet mask.

At the bottom of the LAN Ports section, there are two buttons: "SAVE CONFIG" and "REFRESH".

On devices with more than one Ethernet port, there will always be one WAN port and the rest will be LAN ports.

WAN port configuration:

- **IP Mode:** Allows you to use a static or dynamic (DHCP) IP configuration for the WAN port.
- **IP Address:** Static local IP address of the WAN Ethernet interface (default 192.168.1.2)
- **IP Subnet Mask:** Subnet mask of the WAN Ethernet interface (default 255.255.255.0)
- **DNS1:** Primary DNS server
- **DNS2:** Secondary DNS server
- **Gateway IP:** Outbound Gateway IP address

LAN ports configuration:

- **IP Address:** static local IP address of the Ethernet LAN interface (default 192.168.10.2)
- **IP Subnet Mask:** subnet mask of the Ethernet LAN interface (default 255.255.255.0)

Additional notes.

- Once you have finished configuring, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the device for the new changes to take effect.

4.2.2 Ethernet – DHCP Server

This section allows you to enable and configure the DHCP server assigned to the Ethernet interface of the TITAN-based device.

- **Enabled:** if you tick the box, the DHCP server on the Ethernet interface will be activated.
- **Starting IP Address:** indicates the first IP address that the DHCP server will assign
- **Ending IP Address:** indicates the last IP address that the DHCP server will assign
- **MAC Address / IP Address:** these two parameters allow the DHCP server to always assign the same IP to a specific MAC address.

webdyn powered by **TITAN**
flexitron group

Ethernet > DHCP Server

Enabled: ☒ DHCP Server enabled / disabled

Starting IP Address: First IP address for DHCP (ex 192.168.1.100)

Ending IP Address: Last IP address for DHCP (ex 192.168.1.110)

SAVE CONFIG

MAC Address	IP Address	
00:06:01:09:24:2C	192.168.1.100	Delete

MAC Address: Set a MAC address (ex 54:42:49:0A:E9:2C)

IP Address: Set assigned IP address (ex 192.168.1.120). **Important note:** This IP must be **outside** the previous range of automatic assignment

SAVE RULE

Additional notes

- Once you have finished configuring, press the "SAVE CONFIG" button to save the changes. Press the "SAVE RULE" button for each MAC/IP pair you want to create. You can create up to 10.
- Remember that you must restart the device for the new changes to take effect.

4.3 WiFi

The "WiFi" configuration section refers to the configuration of the WiFi interface. By configuring WiFi in "Access Point" mode, you can, for example, give WiFi client devices access to the Internet or configure the device itself. By configuring it in "WiFi Client" mode, you can make the TITAN-based device use another WiFi Access Point on your premises as an Internet gateway.

4.3.1 WiFi – Basic Settings

This section allows you to configure the basic network parameters of the Wi-Fi interface.

- **Enabled:** Checking this box activates the Wi-Fi interface of the TITAN-based device.
- **Wifi mode:** Allows you to select between "Wifi Access Point" or "Wifi Client" operating mode. Use "Wifi Access Point" mode if you want to use TITAN-based equipment to provide Internet access to Wifi devices using the 4G/3G/2G interface. Use "Wifi Client" if you want to use an existing Wifi infrastructure (e.g. a company's fibre router) as your Internet connection.
- **Wifi SSID:** Public SSID (in "Wifi Access Point" mode, this indicates the SSID that the TITAN-based device will create; in "Wifi Client" mode, it indicates the SSID to which the TITAN-based device will connect).
- **Security:** allows you to specify the Wi-Fi security mode. WPA2 is recommended.
- **KEY:** specifies the Wi-Fi password required for WPA2 mode
- **IP Mode:** in "Wifi Client" mode, specifies whether a static IP address will be used as the Local Wifi IP address or whether the IP must be obtained via DHCP.
- **IP Address:** IP address of the Wi-Fi interface. Default 192.168.2.1
- **Subnet Mask:** subnet mask of the Wi-Fi interface
- **DNS1:** primary DNS server
- **DNS2:** secondary DNS server
- **IP Gateway:** In "WiFi Client" mode, this allows you to specify the gateway when the local IP is static.
- **Internet Access:** enable to access the Internet from a Wi-Fi device
- **LAN Access:** enable to access devices connected to the device's Ethernet port from a Wi-Fi device.

★ Mobile

- Status
- Basic Settings
- Keep Online

★ Ethernet

- Basic Settings

★ Wifi

- Basic Settings
- DHCP Server

★ Firewall

- NAT
- Authorized IPs

★ Serial Settings

- Serial Port1-RS232
- Serial Port2-RS485
- SSL Certificates

★ External Devices

- Logger configuration
- ModBus Devices
- Generic Serial Device
- Temperature Sensor
- IEC102 Meter
- W-MBus
- GPS Receiver

★ Other

- AT Command

► Wifi ► Basic Settings

Enabled: Enable Wifi

Wifi mode: Select between Wifi Access Point or Wifi Client

Wifi SSID: Select the public name for the Wifi Network

Security: Select security mode

KEY: Password for WPA2-Personal security mode

Channel: Wifi Channel

IP Address: Local IP LAN

IP Subnet Mask: Local Mask

DNS 1: DNS Server 1

DNS 2: DNS Server 2

IP Gateway: Left blank if not used or using another WAN interface

Internet access: ☐ Check if wifi devices can access to Internet

LAN access: ☐ Check if wifi devices can connect with LAN devices

SAVE CONFIG

4.3.2 WiFi – DHCP Server

From this section, you can enable and configure the DHCP server assigned to the Wi-Fi interface of the TITAN-based device.

- **Enabled:** if you tick the box, the DHCP server on the Wi-Fi interface will be activated.
- **Starting IP Address:** indicates the first IP address that the DHCP server will assign
- **Ending IP Address:** indicates the last IP address that the DHCP server will assign.
- **MAC Address / IP Address:** these two parameters allow the DHCP server to always assign the same IP to a specific MAC address.

webdyn powered by TITAN flexitron group

Mobile

- Status
- Basic Settings
- Keep Online

Ethernet

- Basic Settings

Wifi

- Basic Settings
- DHCP Server

Firewall

- NAT
- Authorized IPs

Serial Settings

- Serial Port1-RS232
- Serial Port2-RS485
- SSL Certificates

External Devices

- Logger configuration

Wifi > DHCP Server

Enabled: ☐ DHCP Server enabled / disabled

Starting IP Address: First IP address for DHCP (ex 192.168.2.100)

Ending IP Address: Last IP address for DHCP (ex 192.168.2.110)

SAVE CONFIG

MAC Address	IP Address
MAC Address:	Set a MAC address (ex 54:42:49:0A:E9:2C)
IP Address:	Set assigned IP address (ex 192.168.2.120) Important note: This IP must be outside the previous range of automatic assignment

SAVE RULE

Additional notes.

- Press the "SAVE RULE" button for each MAC/IP pair you wish to create. You can create up to 10. Remember that you must restart the device for the new changes to take effect.
- Once the configuration is complete, press the "SAVE CONFIG" button to save the changes.

4.4 Firewall

Section for configuring the security aspects of the device.

4.4.1 Firewall >> NAT

Section from which ports can be mapped for external access to internal devices connected to the equipment. For example, if you have an IP camera connected to the Ethernet port of TITAN-based equipment and you want to access it from a remote computer, this section must be configured appropriately. You can create up to a total of 10 rules.

- **Service name:** descriptive name of a mapping rule
- **Protocol:** indicates the port mapping protocol. TCP, UDP, or both.
- **Input port:** indicates the input port to the TITAN-based device via the WAN interface
- **Output port:** indicates the redirection port to the internal device connected to the TITAN-based equipment.
- **Server IP Address:** IP address of the device to be controlled externally (e.g. the IP address of a camera).

webdyn powered by TITAN flexitron group

Mobile

- Status
- Basic Settings
- Keep Online

Ethernet

- Basic Settings

Wifi

- Basic Settings
- DHCP Server

Firewall

- NAT
- Authorized IPs

Serial Settings

- Serial Port1-RS232
- Serial Port2-RS485

Firewall > NAT

Service name	Protocol	Input Port	Output Port	Server IP Address
Service name:	Protocol:	Port:	Output Port:	Server IP Address:
	TCP+UDP			
Insert a name for the service	Select TCP/UDP protocol	Input port (0 ... 65535) - Router	Output port (0 ... 65535) - Destination server	Set the IP of the destination server

SAVE SERVICE

Additional notes.

- Once the configuration is complete, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the equipment for the new changes to take effect.
- Also remember that to perform NAT correctly:
 - 1.- The LAN IP address of the device to be controlled must be within the network range of the LAN IP address of the TITAN-based equipment.

2.- The Gateway IP address of the device to be controlled must be the LAN IP address of the TITAN-based equipment. See the examples in the Appendix for more information.

- It is possible to specify the interface for NAT. If you are unsure about the utility, use the "auto" option. Using this option can be useful for OpenVPN, for example, if you want to access devices connected to the Ethernet port via NAT (and thus have the same configuration for remote access to different points).

4.4.2 Firewall – Authorised IPs

This screen allows you to define up to 3 authorised IP addresses to accept connections on the WAN interface for the different services of the device. For example, if you specify an authorised IP address 90.166.108.200 (such as your office IP address), certain services of the TITAN-based device can only be accessed from that IP address.

- *Authorised IP1*: Authorised IP address number 1
- *Authorised IP2*: Authorised IP address number 2
- *Authorised IP3*: Authorised IP address number 3
- *Router configuration*: Specifies whether remote connections to the device's web configuration environment are accepted from any IP address or only from authorised IP addresses.
- *Serial Gateways*: Specifies whether remote connections to 4G/3G/2G-RS232/485 gateway services are accepted from any IP address or only from authorised IP addresses.
- *Remote console*: Specifies whether remote connections to the remote console service (telnet or SSH) are accepted from any IP address or only from authorised IP addresses.
- *NAT*: Specifies whether remote connections to the device's mapped ports are accepted from any IP address or only from authorised IP addresses.
- *ModBus TCP Slave*: Specifies whether remote connections to the Modbus TCP Slave service of the TITAN-based device are accepted from any IP address or only from authorised IP addresses.
- *SNMP*: Specifies whether remote connections to the SNMP service of the TITAN-based device are accepted from any IP address or only from authorised IP addresses.
- *OpenVPN*: Specifies whether remote connections to the TITAN-based computer's "OpenVPN Server" service are accepted from any IP address or only from authorised IP addresses.
- *PING*: Will only respond to PING requests made from authorised IP addresses.
- *Outgoing Connections*: Allows you to specify whether the TITAN-based device can provide Internet access to all IP addresses or only to authorised IP addresses. For example, imagine that you only want Ethernet or Wi-Fi devices connected to the TITAN to be able to send data to your server, preventing the device from being misused for other types of actions (Internet browsing, etc.).

- ★ **Mobile**
 - Status
 - Basic Settings
 - Keep Online
- ★ **Ethernet**
 - Basic Settings
- ★ **Wifi**
 - Basic Settings
 - DHCP Server
- ★ **Firewall**
 - NAT
 - Authorized IPs
- ★ **Serial Settings**
 - Serial Port1-RS232
 - Serial Port2-RS485
 - SSL Certificates
- ★ **External Devices**
 - Logger configuration
 - ModBus Devices
 - Generic Serial Device
 - Temperature Sensor
 - IEC103 Meter

▶ Firewall ▶ Authorized IPs (for WAN interfaces)

Authorized IP1:		Remote connections from this IP are allowed
Authorized IP2:		Remote connections from this IP are allowed
Authorized IP3:		Remote connections from this IP are allowed

Router configuration:	ALLOW ANY IP ▼	Security for remote configuration connection
Serial gateways:	ONLY AUTHORIZED IP ▼	Security for remote serial connection
Remote console:	ALLOW ANY IP ▼	Security for remote console connection
ModBus TCP Slave:	ALLOW ANY IP ▼	Security for ModBus TCP Slave connections
SNMP:	ALLOW ANY IP ▼	Security for SNMP
TCP & UDP exceptions:	ALLOW ANY IP ▼	Security for TCP & UDP port exceptions
PING:	ALLOW ANY IP ▼	Security for incoming PING from WAN

Additional notes

- Once you have finished configuring, press the "SAVE CONFIG" button to save your changes. Remember that you must restart the equipment for the new changes to take effect.
- IP restrictions are only at the WAN connectivity level. Wi-Fi access is protected by the WPA2 network key itself. In other words, if you have Wi-Fi and need security, enable WPA2 in the Wi-Fi section ▶ Basic Settings.
- If you use "Outgoing connections" restrictions, remember that you need to specify the IP address of the DNS server to use if you use domain names instead of IP addresses.
- If you need more than three authorised IP addresses, you can specify more than one IP address in any box, separating them with commas.

4.4.3 Firewall – MAC Filter

From this section, you can configure the device to filter communications through the Ethernet (or Wi-Fi) interface based on MAC addresses.

- **Mode** – All MAC allowed except listed: All MAC addresses are allowed except those indicated in the list, which will be blocked. This is the default setting.
- **Mode** – Only listed MAC allowed: All MAC addresses are blocked except for those listed, which will be allowed.
- **MAC List**: List of permitted or blocked addresses, depending on the operating mode.

The screenshot shows the webdyn TITAN configuration interface. The top header features the webdyn logo (a stylized 'w' with a blue and green wave) and the text 'webdyn flexitron group'. To the right is the 'TITAN' logo with the tagline 'Makes your APPLICATION happen'.

The left sidebar contains a navigation menu with the following items:

- ★ **Mobile**
 - Status
 - Basic Settings
 - Keep Online
- ★ **Ethernet**
 - Basic Settings
- ★ **Firewall**
 - Authorized IPs
 - MAC Filter
- ★ **Serial Settings**
 - Serial Port1-RS232
 - Serial Port2-RS485
 - SSL Certificates
- ★ **External Devices**
 - Logger configuration
 - ModBus Devices
 - Generic Serial Device
 - Temperature Sensor

The main content area is titled 'Firewall ► MAC Filter'. It contains the following configuration options:

- Mode:** A dropdown menu set to 'Only listed MAC allowed'. To its right is the text 'MAC Filter mode'.
- MAC list:** A text area containing two MAC addresses: '01:02:03:04:05:06' and '07:08:09:00:0a:0b'. To the right of this area is an 'Example:' section showing the same two MAC addresses and a note '(File: 'macs.mtx')'.
- SAVE CONFIG**: A button at the bottom left of the configuration area.

4.4.4 Firewall – Routes

From this section, you can configure the device to add up to 5 routing paths per interface.

- **Network** – Indicates the network for the route to be configured.
- **Mask** – Indicates the network mask for the route to be configured.
- **Gateway** – IP address of the gateway to be used for the route to be configured.
- **Interface** – Physical interface of the route to be configured.

The screenshot shows the webdyn TITAN configuration interface. The top header includes the webdyn logo (a stylized 'w' with a blue wave) and the text 'webdyn flexitron group'. To the right is the 'TITAN' logo with the tagline 'Makes your APPLICATION happen®'. On the left is a sidebar menu with categories: Mobile (Status, Basic Settings, Keep Online), Ethernet (Basic Settings, DHCP Server), Wifi (Basic Settings, DHCP Server), Firewall (NAT, Authorized IPs, MAC Filter, Routes), and Serial Settings (Serial Port1-RS232). The main content area is titled 'Firewall > Routes'. It features a table with four columns: Network, Mask, Gateway, and Interface. Below the table are input fields for each column: 'Network:' with a text box and hint 'Network for route (Ex: 10.10.20.0)'; 'Mask:' with a text box and hint 'Mask for route (Ex: 255.255.255.0)'; 'Gateway:' with a text box and hint 'Blank: default gateway'; and 'Interface:' with a dropdown menu showing 'ETH' and hint 'Interface for route'. An 'ADD ROUTE' button is located below the input fields.

Network	Mask	Gateway	Interface
Network:			
Mask:			
Gateway:			
Interface:	ETH		

ADD ROUTE

4.5 Serial Settings – Serial portX

From the Serial Settings section, it is possible to set the configuration of each serial port on the TITAN-based device (speed, parity, etc.) as well as the special function that each one will have (IP-serial gateways, Modbus TCP-RTU gateways, etc.).

4.5.1 Serial Settings >> Serial portX

TITAN-based devices have several serial ports. These can be used to create 4G/3G/2G-Serial and CSD-Serial gateways to remotely control RS232 or RS485 devices. The number of RS232 or RS485 serial ports will depend on the TITAN-based device model used.

- **Baud rate:** specifies the speed of the serial port (115200, ... , 300)
- **Data bits:** specifies the number of data bits (7, 8)
- **Parity:** specifies the parity (none, even [even], odd [odd])
- **Stop bits:** number of stop bits (1, 2)
- **Flow Control:** specifies flow control (none, hardware)
- **Timeout ms:** indicates the number of milliseconds to wait without receiving data through the serial port before sending the data via IP. If you specify "0" (default value), the data is sent via IP as it arrives through the serial port. A number such as 10 specifies that no data should be sent unless there has been a minimum period of 10 ms without receiving data through the serial port. This allows the data to reach its destination less fragmented.
- **Allow local embedded AT Commands:** selecting this box allows the sending of LOCAL AT commands embedded in a Server or Client type 4G/3G/2G-Series gateway. AT commands must be sent via the RS232 or RS485 serial port of the TITAN-based device but encapsulated between the <MTXTUNNEL> and </MTXTUNNEL> tags. For example, if you want to obtain coverage, you can send the command <MTXTUNNEL>AT+CSQ</MTXTUNNEL>. Or if you want to reset a device remotely, you can send the command AT^MTXTUNNEL=REBOOT
- **Allow remote embedded AT Commands:** selecting this box allows REMOTE AT commands embedded in a Server or Client type 4G/3G/2G-Series gateway to be sent. AT commands must be sent via the established gateway connection but encapsulated between the <MTXTUNNELR> and </MTXTUNNELR> tags. For example, if you want to obtain coverage, you can send the command <MTXTUNNELR>AT+CSQ</MTXTUNNELR>. Or if you want to reset a device remotely, you can send the command <MTXTUNNELR>AT^MTXTUNNEL=REBOOT </MTXTUNNELR>

- **Allow incoming GSM call (CSD Data Call):** selecting this box indicates that a CSD type call is accepted by the serial port. This is only valid when the port is configured as a TCP Server or TCP Client gateway and in some cases (such as reading IEC-102 meters) when it is configured as "Nothing or Used by External Devices". When a CSD call is received and accepted, IP connections are suspended until the CSD call ends.
- **Function: Nothing or Used by External Device:** select this function option if you do not wish to use a specific serial port as an IP/serial gateway or if you wish the serial port (RS232 or RS485) to be used by an external device specified in the "External Devices" configuration section. For example, if you wish to use a temperature sensor or any other device specified in the "External Devices" menu, you must select this option.
- **Function: Serial – IP Gateway (TCP Server):** select this operating option if you wish to establish a transparent Serial – 4G/3G/2G gateway in TCP Server mode, i.e. a scenario where the TITAN-based device is listening on a specific TCP port, waiting to receive a connection to establish the gateway.
- **TCP Local Port:** listening TCP port for the Serial – 4G/3G/2G gateway
- **Timeout:** time (seconds) without data on the gateway to automatically close the socket.
- **TCP Local Priority Port:** priority TCP listening port for the Serial – 4G/3G/2G gateway. If there is an active connection on this port, connections on the TCP Local Port will not be accepted. Useful for reading electricity meters with IP connection priority.
- **Close IEC102 session:** select this option only if you are using the IP-Serial gateway to read meters with the IEC-60870-5-102 protocol. When this option is enabled, the Titan router analyses the circulating packets of the IEC-60870-5-102 protocol, obtaining the link address and measurement point of the meter. This information is used each time an IP-Serial gateway is initiated against the meter in order to close any session with the meter that may have been opened.
- **Temporary RS232 client:** Select this option if you want to activate a temporary socket when working in TCP Server mode but the IP/Serial gateway is not set up and you are receiving data through the serial port. If you activate this option, you must have the "Remote IP", "Remote TCP Port" and "ID String" parameters in the TCP Client section shown below correctly configured.
- **Temporary client Wakeup:** This option allows you to activate a temporary socket when working in TCP Server mode but without the IP/Serial gateway established and when the configured time arrives. If you activate this option, you must have the "Remote IP", "Remote TCP Port" and "ID String" parameters in the TCP Client section shown below correctly configured.
- **Temporary client time:** indicates the time a temporary socket is active.
- **Temporary client Random:** it is possible to set a random time in seconds for a temporary client that is added to the Wakeup time. This is only useful if you have a large number of computers connecting at the same time and you want to introduce a random time to distribute the number of simultaneous connections to a server.
- **SSL/TLS enabled:** if you need to secure the TCP Server gateways, you can do so using this option. If enabled, you must also configure the certificate section "Serial Settings→ SSL Certs".
- **Function: Serial – IP Gateway (TCP Client):** select this operating option if you wish to establish a transparent Serial – 4G/3G/2G gateway in TCP Client mode, i.e. a scenario where the TITAN-based device connects to a specific IP / TCP port to establish the serial – 4G/3G/2G gateway.
- **Remote IP:** IP address to which the TITAN-based device will connect
- **Remote TCP Port:** TCP port to which the TITAN-based device will connect
- **Reconnection time:** in the event of a connection failure or problems with connections, this indicates how many milliseconds should elapse between connection attempts. 0 = immediate reconnection. Please

careful with this value if you do not have a flat-rate SIM card or have a very limited monthly data allowance.

- **ID String:** string sent immediately after establishing the socket with the remote IP. This text will allow you to identify the device making the connection. For example, if you have 100 TITAN-based devices in this operating mode, it allows you to distinguish after a connection which of those 100 devices made the connection. It is possible to add the special tags [IMEI], [CR] and [LF], which will be replaced by the IMEI and the characters 0x13 and 0x10 respectively.
- **Function: Serial – IP Gateway (Modbus TCP / Modbus RTU):** select this operating option if you wish to establish a Serial – 4G/3G/2G gateway with Modbus TCP – Modbus RTU protocol conversion. Do not use this option if your control software uses Modbus RTU, as a normal TCP Server gateway would suffice. Use this option if your control software uses the Modbus TCP protocol. If you have any questions, please consult contact@webdyn.com.
- **Serial – IP Gateway (MQTT):** select this option if you wish to establish an IP-Serial gateway via MQTT protocol. To use this option, you must configure the MQTT client found in the "Other - MQTT" section.
- **TX Mqtt Topic:** The TITAN-based device will forward all data received on the serial port (data sent in RAW mode) to the MQTT topic defined in this parameter.
- **RX Mqtt Topic:** The TITAN-based device will subscribe to the MQTT topic defined in this parameter and forward all data received in that topic (data sent in RAW mode) through its serial port.
- **Function: Serial – IP Gateway (UDP):** select this operating option if you wish to establish a transparent IP – Serial gateway using the UDP protocol
- **Remote IP:** IP address of the remote server where the data will be sent via UDP
- **Remote UDP Port:** UDP port of the remote server
- **Local UDP Port:** listening UDP port of the device

Mobile

- Status
- Basic Settings
- Keep Online

Ethernet

- Basic Settings
- DHCP Server

Wifi

- Basic Settings
- DHCP Server

Firewall

- NAT
- Authorized IPs
- MAC Filter
- Routes

Serial Settings

- Serial Port1-RS232
- Serial Port2-RS485
- SSL Certificates

External Devices

- Logger configuration
- ModBus Devices
- Generic Serial Device
- Temperature Sensor
- IEC102 Meter
- GPS Receiver

VPN

- IPSec
- OpenVPN Client
- OpenVPN Server
- ZeroTier

Other

- AT Command
- DynDns
- Private DynDns
- Sms control
- Periodic Autoreset
- Time Servers
- Remote Console
- Snmp
- Tacacs+
- Ldap
- Mqtt
- Http / Https
- User Permissions
- Passwords Web UI
- CA Certificates
- Email Config
- ModBus Slave
- Titan Scripts**
- Connectivity tools
- Digital I/O
- Custom Skin
- Led Config
- Syslog
- Backup / Factory
- Firmware Upgrade
- Reboot
- Logout

Serial Gateway ▶ Com1 Settings

Baudrate: Baudrate of serial port

Data bits: Number of data bit

Parity: Parity

Stop bits: Number of stop bits

Flow Control: Flow control of serial port

Timeout ms: msec without serial data before sending (default: 50)

- ☐ **Allow local embedded AT commands** Ex.: <MTXTUNNEL>AT</MTXTUNNEL>
- ☐ **Allow remote embedded AT commands** Ex.: <MTXTUNNELR>AT</MTXTUNNELR>
- ☐ **Allow incoming GSM call (CSD Data Call)** Only **TCP Server** and **TCP Client** functions or **Nothing**. 2G (CSD) network required.

● Function: Nothing or used by External Device or Script

○ Function: Serial - IP Gateway (TCP Server)

TCP Local Port: Listening TCP Port (1 ... 65535)

Timeout: Seconds without data for closing. (0...7200) 0=not used.

TCP Local Priority Port: Listening TCP Local Priority Port (0 ... 65535). 0=not used.

Close IEC102 session ☐ Close IEC 60870-5-102 meter session

Temporal client RS232 ☐ Check if you need a temporal TCP Client when data is present at serial port.

Temporal client Wakeup DDHMM. Example: XX2200 starts a temporal client every day at 22:00

Temporal client time: Seconds for temporal client

Temporal client Random Seconds. Random time for temporal client

SSL/TLS enabled ☐ SSL/TLS Enabled (SSL Certs needed)

○ Function: Serial - IP Gateway (TCP Client)

Remote IP: Address of remote IP server

Remote TCP Port: Port number of remote server (1 ... 65535)

Reconnection time: Milliseconds between connection attempts

ID String: This identification String is sent in each connection (can be used for device identification)

SSL/TLS enabled ☐ SSL/TLS Enabled (CA Certs needed) (SSL Certs needed if client authentication)

○ Function: Serial - IP Gateway (ModBus TCP / ModBus RTU)

TCP Local Port: Listening TCP Port (1 ... 65535). Normally 502

SSL/TLS enabled ☐ SSL/TLS Enabled (SSL Certs needed)

○ Function: Serial - IP Gateway (MQTT)

TX Mqtt Topic:

RX Mqtt Topic:

All data received by serial will be retransmitted to this MQTT topic

All data received by IP in this MQTT topic will be retransmitted by the serial port.

○ Function: Serial - IP Gateway (UDP)

Remote IP: Address of remote IP server (can be included several IP between ';')

Remote UDP Port: Remote UDP port (1 ... 65535)

Local UDP Port: Local UDP port (1 ... 65535)

SAVE CONFIG

4.5.2 Serial Settings – SSL Certificates

The IP-Series gateways of TITAN-based equipment can be established using SSL/TLS connections, both in client TCP connections and server TCP connections. When using SSL/TLS connections, it is necessary to configure the corresponding certificate section.

The Serial Gateway - SSL/TLS Certificates (TCP SERVERS) section refers to certificates when the COM port has been configured as a TCP Server socket.

The screenshot shows the webdyn interface for configuring SSL/TLS certificates for TCP servers. The interface includes a sidebar with navigation options: Mobile, Ethernet, Wifi, Firewall, Serial Settings, and External Devices. The main content area is titled 'Serial Gateway > SSL/TLS Certificates (TCP SERVERS)'. It contains two sections: 'Certificate for SSL/TLS Servers (PEM format)' and 'Certificates for authorized SSL/TLS clients (if needed) (PEM format)'. The first section has fields for 'Server certificate' and 'Server KEY', each with a file selection button, an 'Upload' button, and a 'not uploaded' status. Below these is a 'DELETE CERTIFICATE' button. The second section has three rows for 'Client certificate 1', 'Client certificate 2', and 'Client certificate 3', each with similar file selection, upload, and status fields. Below these is a row of three 'DELETE CERTIFICATE' buttons. At the bottom, there is a 'Client authentication' checkbox (checked) and a 'Check for client authentication' label, followed by a 'SAVE CONFIG' button.

- **Server Certificate:** Certificate (PEM format) to be used by TITAN-based equipment for any service configured as a TCP Server.
- **Server KEY:** Private key of the certificate used by the TITAN-based device when configured by TCP Server.
- **Client authentication:** If this box is checked, the TITAN-based device will only allow clients using authorised client certificates to connect to its TCP Server sockets.

- **Client certificate 1:** If using "Client authentication", this field allows you to enter authorised certificate 1.
- **Client certificate 2:** If using "Client authentication", this field allows you to enter authorised certificate 2.
- **Client certificate 3:** If using "Client authentication", this field allows you to enter authorised certificate 3.

The Serial Gateway section→ SSL/TLS Certificates (TCP CLIENT) refers to certificates when the COM port has been configured as a TCP Client socket

Certificates for TCP Client sockets are only required when the remote TCP server (where the TITAN-based device will connect) requires certificate authentication.

Serial Gateway ▶ SSL/TLS Certificates (TCP CLIENT)

Certificate for SSL/TLS TCP Client Serial Port 1 (PEM format)

Client Port1 certificate: Ninguno archivo selec. **not uploaded**

Client Port1 KEY: Ninguno archivo selec. **not uploaded**

Certificate for SSL/TLS TCP Client Serial Port 2 (PEM format)

Client Port2 certificate: Ninguno archivo selec. **not uploaded**

Client Port2 KEY: Ninguno archivo selec. **not uploaded**

Serial Gateway ▶ SSL/TLS Certificates ▶ Tools

With this tool you can generate a pair of files certificate.crt and certificate.key for your tests.

Process Status:

- **Client Port1 Certificate:** Certificate (PEM format) to be used by the TITAN-based device when a Port1 serial port acts as a TCP client with SSL/TLS with certificate identification
- **Client Port1 KEY:** Private key of the certificate used by the TITAN-based device when a Port1 serial port acts as a TCP client with SSL/TLS with certificate identification.
- **Client Port2 Certificate:** Certificate (PEM format) to be used by the TITAN-based device when a Port2 serial port acts as a TCP client with SSL/TLS with certificate identification.

- *Client Port2 KEY*: Private key of the certificate used by the TITAN-based device when a Port2 serial port acts as a TCP client with SSL/TLS with certificate identification.

The Serial Gateway section→ SSL/TLS Certificates→ is a utility that allows you to conveniently generate a pair of .crt and .key key pair conveniently for performance testing.

4.6 External Devices

This section is used to configure the internal data logger, as well as external serial peripherals (temperature probes, Modbus RTU or Modbus TCP devices, generic RS232-485 devices, etc.).

4.6.1 External Devices >> Logger configuration

If you need the TITAN-based equipment to collect data from external devices (Modbus devices, temperature sensors, generic serial devices, GPS receivers, W-Mbus devices, IEC-102 meters, etc.) to send them to an external server, you must first configure the internal logger, i.e. how to collect the data, where and how to send it. That is what this section is for.

This section allows you to configure the parameters related to the internal data logger. You can send the data to a server via HTTP/HTTPS, FTP/FTPS or MQTT/MQTTS. In all cases, the data will be sent in JSON format.

General parameters:

The screenshot shows the 'webdyn' interface, powered by TITAN. The left sidebar contains navigation links for Mobile, Ethernet, and Wifi. The main content area is titled 'External Devices > Logger'. It contains the following configuration fields:

- ID:** A text input field containing 'ID-869101054287764'. A note indicates it is optional for device identification.
- Send mode:** A dropdown menu set to 'FIFO'. A note indicates it is the normal mode.
- Time format:** A dropdown menu set to 'unix (yyyy-mm-ddTHH:mm:ss)'. A note explains it is the format used in timestamp logger data.
- Use script:** A checkbox that is checked. A note explains it is for customized JSON using a script.
- Use array:** A checkbox that is checked. A note explains it is for sending more than one JSON per transmission.

- **ID:** Optional. This is a text parameter that allows you to enter an identifying string that is included within the JSON.
- **Send mode:** FIFO or LIFO. In FIFO mode, data is transmitted to the server in order of arrival, i.e. the first data to enter the data logger is the first to be sent. LIFO mode is the opposite, with the last data received being the first to be sent. LIFO mode can be useful in applications where you always want to view data on screen in real time. For example, a device may be without coverage for several days, storing large amounts of data without being able to transmit it. Once coverage returns, it may take some time for all the accumulated data to be sent to the server. In LIFO mode, by sending the latest data collected first, the data can be quickly displayed on a screen in real time while the rest of the previously captured data arrives later.
- **Time format:** the JSON data object always includes a timestamp (date and time) with the moment when the saved data was recorded. This parameter allows you to specify the format of that timestamp.

- Use script: If this box is ticked, before the TITAN-based device stores collected data (e.g. a Modbus reading) in its internal memory (datalogger) in the default JSON format of the TITAN-based device, this data can be formatted to be stored in another format. This can be very useful for sending data to platforms that require a specific structure. Please refer to the application note related to 'JSON Transformer Scripts' for more information.

For example, Modbus records read by TITAN-based equipment would be stored in the following standard format:

```
{"IMEI":"869101054287806","TYPE":"MODB","TS":"2022-06-03T11:38:01Z","ID":"TRES","A":"192.168.1.28:502","ST":"1","N":"6","V":[1,2,3,4,5,6],"P":"ID0001"}
```

However, we may wish to modify this structure, add new custom fields, etc.

```
{"data":{"IMEI":"869101054287806","TYPE":"MODB","TS":"2022-06-03T11:38:01Z","ID":"THREE","A":"192.168.1.28:502","ST":"1","N":"6","V":[1,2,3,4,5,6],"P":"ID0001","myField1":123,"MyField2":456}}
```

This format change can be done using "JSON Transformer Scripts".

- Use array: By default, TITAN-based equipment sends data to a server (MQTT, HTTP) record by record, i.e., one transmission per record sent. If you select the "Use array" option, the data is sent in array form, allowing up to 100 readings to be sent simultaneously and greatly speeding up the data transmission process.

For example, if this box is not selected, the data is transmitted in the standard JSON format of the TITAN-based equipment.

```
{"IMEI":"869101054287806","TYPE":"MODB","TS":"2022-06-03T11:38:01Z","ID":"TRES","A":"192.168.1.28:502","ST":"1","N":"6","V":[1,2,3,4,5,6],"P":"ID0001"}
```

If the box is selected, the data is transmitted in the form of an array with between 1 and 100 elements, depending on the data stored inside the data logger.

```
[{"IMEI":"869101054287806","TYPE":"MODB","TS":"2022-06-03T11:38:01Z","ID":"TRES","A":"192.168.1.28:502","ST":"1","N":"6","V":[1,2,3,4,5,6],"P":"ID0001"}, {"IMEI":"869101054287806","TYPE":"MODB","TS":"2022-06-03T11:39:01Z","ID":"THREE","A":"192.168.1.28:502","ST":"1","N":"6","V":[1,2,3,4,5,6],"P":"ID0001"}, {"IMEI":"869101054287806","TYPE":"MODB","TS":"2022-06-03T11:40:01Z","ID":"THREE","A":"192.168.1.28:502","ST":"1","N":"6","V":[1,2,3,4,5,6],"P":"ID0001"}, ... ]
```

HTTP mode

The screenshot shows the 'Communication mode: WEB PLATFORM (HTTP REST)' configuration page. On the left is a sidebar with navigation links: Basic Settings, DHCP Server, Firewall (selected), NAT, Authorized IPs, Serial Settings, and External Devices. The main area contains the following settings:

- Enabled:** A checkbox that is currently unchecked. To its right, the text 'Communication mode HTTP enabled' is displayed.
- Mode:** A dropdown menu currently set to 'HTTP GET (JSON)'. To its right, the text 'Method of sending data' is displayed.
- Custom parameters:** A text input field. To its right, the text 'Optional. Ex: &a=1&b=2 only for "HTTP GET/PUT (PARAMETERS)" modes' is displayed.
- Custom header1:** A text input field. To its right, the text 'Optional. Custom header1. For example: Content-type;application/json' is displayed.
- Custom header2:** A text input field. To its right, the text 'Optional. Custom header2. For example: IDENTITY_KEY;YOUR_KEY' is displayed.
- Custom header3:** A text input field. To its right, the text 'Optional. Custom header3.' is displayed.
- Server:** A text input field. To its right, the text 'Destination URL. Example: www.mydomain.com/set.asp?data=' is displayed.
- Server Username:** A text input field. To its right, the text 'Optional. Blank if no server authentication required' is displayed.
- Server Password:** A text input field. To its right, the text 'Optional. Blank if no server authentication required' is displayed.

- **Enabled:** Click to enable the mode for sending data to a web platform via HTTP/HTTPS
- **Mode:** data transfer mode. You can select between HTTP GET (JSON), HTTPS GET (JSON), HTTP POST (JSON), HTTPS POST (JSON)
- **Custom parameters:** allows you to add optional parameters for HTTP GET methods
- **Custom header1, Custom header2, and Custom header3:** Allows you to add headers to HTTP/HTTPS requests. Many web platforms require you to include a header with an identification token. You can configure this in this section.
- **Server:** Full URL for sending data collected in the data logger. For example www.metering.es/json/set.asp?data=
- **Server Login:** If your platform has restricted access, enter your username here.
- **Server Password:** If your platform has restricted access, enter the user password here.

FTP Mode

- Snmp - Tacacs+ - Mqtt - Http / Https - User Permissions - Passwords Web UI - CA Certificates - Email Config - ModBus Slave - Titan Scripts - Connectivity tools - Digital I/O - Custom Skin - Led Config	Enabled: ☒ Communication mode FTP enabled FTP prot.: FTPS FTP / FTPS protocol FTP Server: Destination FTP Server. Example: ftp.mydomain.com FTP port: FTP server port. Default 21 FTP Path: FTP path. Example: /dev/plcs/ FTP Username: FTP Username FTP Password: FTP Password FTP File Period: day FTP File Period (one file every minute, hour, day)
--	--

- **Enabled:** Click to enable the mode for sending data to an FTP server (the data file format is JSON).
- **FTP prot.:** Allows you to select the transfer protocol, choosing between FTP or the secure FTPS protocol.
- **FTP Server:** FTP server for sending data (IP or DNS)
- **FTP port:** FTP server port, normally 21.
- **FTP Path:** Path within the server where the data dump will be performed.
- **FTP Username:** The username of an account on your FTP server that allows writing.
- **FTP Password:** The password for an account on your FTP server that allows writing
- **FTP File Period:** How often do you want TITAN to send the accumulated data file to your server (every day, every hour, or every minute)

MQTT Mode

- Sms control - Periodic Autoreset - Time Servers - Remote Console - Snmp - Tacacs+ - Mqtt	Communication mode: MQTT Enabled: ☐ Communication mode MQTT enabled MQTT Topic MQTT Topic. Example: [IMEI]/logger Note: Other>MQTT menu must be configured
--	---

- **Enabled:** Click to enable the mode for sending data to an MQTT broker (all data is sent in JSON format).
- **MQTT Topic:** MQTT topic to which the data stored in the logger will be sent

Real Time Status.

Allows you to view the status of the internal data logger of the TITAN-based device and view data pending transmission. It can also be useful for some applications where you do not want to transmit data to a remote server.

Z-Other > Logger Registers (pending of transmission)

```
{
  "IMEI": "869101054287806",
  "TYPE": "MODBUS",
  "TS": "2022-06-03T11:54:01Z",
  "ID": "UNO",
  "A": "192.168.1.28:502",
  "ST": "1",
  "N": "14"
},
{
  "IMEI": "869101054287806",
  "TYPE": "MODBUS",
  "TS": "2022-06-03T11:54:01Z",
  "ID": "DOS",
  "A": "192.168.1.28:502",
  "ST": "1",
  "N": "5"
},
{
  "IMEI": "869101054287806",
  "TYPE": "MODBUS",
  "TS": "2022-06-03T11:54:01Z",
  "ID": "TRES",
  "A": "192.168.1.28:502",
  "ST": "1",
  "N": "4"
}
```

Number of listed registers: 3
Size of listed registers: 669 bytes of data
Total registers in Logger: 3

Show: Text filter:

From this section, you can also delete all data pending transmission from the TITAN-based device by clicking the "DELETE ALL PENDING REGISTERS FROM LOGGER" button. You can also filter by text.

Additional notes

- Once you have finished configuring, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the device for the new changes to take effect.
- When using HTTP or MQTT transmission mode, data is sent to your server as it is collected. In the event of a lack of 4G/3G/2G coverage or problems with data transmission (e.g. a remote server crash), TITAN-based equipment will store the data internally for later transmission when communication returns to normal.
- When using FTP transmission mode, the data is stored in a JSON record structure file and sent according to the selected transmission interval (minute, hour, day). In the event of a lack of GSM coverage or problems with data transmission (e.g. remote server failure), the TITAN-based device will store the data internally for later transmission when communication returns to normal.
- The name of the FTP file created on your server will have the following format:

IMEI-year-month-day.txt → If the selected data transmission period is "every day".

IMEI-year-month-day-hour.txt → If the selected data sending period is "every hour".

IMEI-year-month-day-hour-minute.txt→ If the selected data sending period is "every minute".

- If you are using MQTT mode, remember to configure the "OTHER→ Mqtt" section correctly. In this section, you must enter all the configuration parameters necessary to connect the TITAN-based equipment to your MQTT broker

4.6.2 External Devices – Temperature Sensor

If your TITAN-based device model supports an MTX-Temp-RS232 temperature probe, you can send the temperature periodically to a web platform, send an SMS alert message when the temperature is above or below a certain threshold, etc.

- *Enabled*: Check this box if you have an MTX-Temp-RS232 temperature probe connected to a serial port.
- *Serial Port*: Select the serial port on the device where the MTX-Temp-RS232 temperature probe is connected.
- *Interval*: If you want to send temperature readings to your web server periodically, specify the interval here in minutes.
- *Logger*: If you want to use the internal logger to store temperature data (for subsequent sending to a web platform), you must select this option. When should this option not be enabled? For example, when you only want to send an SMS alert for a temperature outside the range.
- *Alarms enabled*: Select this option if you want to activate alarms for temperatures outside the range.
- *Max Temperature*: Maximum temperature at which a high temperature alarm is triggered.
- *Text alarm on (max)*: Alarm text sent when a high temperature alarm is triggered.
- *Text alarm off (max)*: Alarm text sent when a high temperature alarm is deactivated.
- *Min Temperature*: Minimum temperature at which a low temperature alarm is triggered.
- *Text alarm on (min)*: Alarm text sent when a low temperature alarm is triggered.
- *Text alarm off (min)*: Alarm text sent when a low temperature alarm is deactivated.
- *Telephone numbers*: Telephone numbers (separated by semicolons ";") to which SMS alarm messages are to be sent.
- *Current temperature*: indicates the current temperature in real time, if an MTX-Temp-RS232 temperature sensor is connected.

- ★ **Mobile**
 - Status
 - Basic Settings
 - Keep Online
- ★ **Ethernet**
 - Basic Settings
- ★ **Wifi**
 - Basic Settings
 - DHCP Server
- ★ **Firewall**
 - NAT
 - Authorized IPs
- ★ **Serial Settings**
 - Serial Port1-RS232
 - Serial Port2-RS485
 - SSL Certificates
- ★ **External Devices**
 - Logger configuration
 - ModBus Devices
 - Generic Serial Device
 - Temperature Sensor
 - IEC102 Meter
 - W-MBus
 - GPS Receiver
- ★ **Other**
 - AT Command
 - DynDns

External Devices ▶ Temperature Sensor

Enabled: ☐

Communication Port: Serial Port 1

Interval: 5

Logger: ☐

Enable Temperature sensor

Select the connected port

Reading period (minutes)

Check if logger must be used

Please, configure logger before using this option

Alarms Enabled: ☐

Max Temperature: 50

Text alarm on (max):

Text alarm off (max):

Min Temperature: 0

Text alarm on (min):

Text alarm off (min):

Phone numbers:

Enable alarm of temperature sensor

Select high temperature

Text of SMS when alarm activated (max temp)

Text of SMS when alarm deactivated (max temp)

Select low temperature

Text of SMS when alarm activated (min temp)

Text of SMS when alarm deactivated (min temp)

Phone numbers for SMS alarm message. (separated with ;)

Current temperature: ERROR °C

Real time value

SAVE CONFIG
REFRESH

Additional notes.

- Once the configuration is complete, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the equipment for the new changes to take effect.
- The hysteresis used by the device is 1 degree. For example, if you set the high temperature to 50°C, the alarm will go off when it reaches 50°C, but the alarm will not be considered to have disappeared until it drops to at least 49°. This prevents continuous SMS messages from being sent when the temperature is right at the selected threshold.
- Example of the format of the JSON frame sent:

```
{ "IMEI": "358884050088207", "TS": "25/12/2014 17:40:09", "TYPE": "TEMP",
  "P": "1234", "TEM": 25.5, "TEMH": 0, "TEML": 0 }
```

Where:

IMEI: device identification number. Unique for each device
 TS: timestamp DD:MM:YYYY HH:MM:SS

TYPE: frame type. In this case, temperature

P: Logger ID field (External Devices > Logger configuration) TEM:
temperature
TEMH: high temperature alarm (0=no, 1=yes) TEMPL:
low temperature alarm (0=no, 1=yes)

4.6.3 External Devices – Modbus Devices

Basic Mode

From firmware version 5.3.6.25 onwards, it is possible to choose between "**Basic Mode**" (the only mode available until firmware 5.3.6.24) and "Expert Mode".

TITAN-based devices are designed to read, store and send records from external Modbus RTU and Modbus TCP devices to an external server (via HTTP/HTTPS, FTP or MQTT/MQTTS). In other words, you can schedule periodic readings of up to 40 Modbus RTU and/or TCP devices, selecting the reading records and then sending the readings to a server using a JSON object.

- **Enabled:** Check this box if you have one or more Modbus RTU devices connected to a serial port and want to read Modbus logs autonomously.
- **Serial Port:** Allows you to select the RS232 or RS485 serial port on the TITAN-based device where the Modbus RTU device is connected.
- **Logger:** If you wish to use the internal logger to store the Modbus records read (for subsequent sending to a web platform), you must select this option.
- **Device name:** Identifying name of a Modbus RTU or Modbus TCP device.
- **Address:** Modbus RTU address of the device to be read. In the case of a Modbus TCP device, the IP and port must be specified (format IP:port).
- **Command:** Modbus read command.
- **Start:** initial read register.
- **Num Words:** number of records to read
- **Reg type:** type of record to read
- **Period:** reading period, i.e. how often the set of records should be read.

Important: See application note AN27 (ANV6_27-Router-TITAN-Modbus-RTU-TCP-Concentrator-multimap_HTTP.pdf and ANV6_32-Router-TITAN-Modbus-RTU-TCP-Concentrator-multimap_MQTT.pdf) for detailed information and examples of how to use this functionality on TITAN-based devices.

- ★ **Mobile**
 - Status
 - Basic Settings
 - Keep Online
- ★ **Ethernet**
 - Basic Settings
- ★ **Wifi**
 - Basic Settings
 - DHCP Server
- ★ **Firewall**
 - NAT
 - Authorized IPs
- ★ **Serial Settings**
 - Serial Port1-RS232
 - Serial Port2-RS485
 - SSL Certificates
- ★ **External Devices**
 - Logger configuration
 - ModBus Devices
 - Generic Serial Device
 - Temperature Sensor
 - IEC102 Meter
 - W-MBus
 - GPS Receiver
- ★ **Other**
 - AT Command
 - DynDns
 - Private DynDns
 - Sms control
 - Periodic Autoreset
 - Time Servers
 - Remote Console
 - Snmp
 - Tacacs+
 - Mqtt

External Devices ▶ ModBus RTU / TCP

Settings successfully updated. Changes take effect after reboot

Enabled: ☒ Enable Modbus Devices

Serial Port: Serial Port 2 Select the connected serial port if needed

Logger: ☒ Check if logger must be used
Please, configure logger before using this option

SAVE CONFIG VIEW LOG

Dev. name / ID	Addr.	Command	Start @	Num word/bit	Reg Type	Period		
UNO	1	0x03	1	10	WORD	1	Del	Test
DOS	192.168.1.28:502	0x03	10	5	WORD	1	Del	Test

Device name / ID: Insert the device name or ID

Address: Modbus RTU address or IP:port address

Command: 0x01 Modbus read command

Start: Address of the first register

Number Words / Bits: Words for command 0x03/0x04. Bits for 0x01/0x02

Reg Type: WORD Type of registers for command 0x03/0x04

Period: 1 Read period (minutes)

SAVE DEVICE (Max 40 modbus devices)

Additional notes.

- Once the configuration is complete, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the device for the new changes to take effect.
- If you wish to create a new Modbus RTU device, you must complete the form and click on the "SAVE DEVICE" button.
- An example of the format of the JSON frame stored with the readings to be sent to a server is as follows:
 - {"IMEI":"354740050367237", "TS":"17/02/201419:02:46", "TYPE":"MODB", "P":"1234", "ST":1, "A":1, "V": [1,2,3,4,5,6,7,8,9,10]}

Where:

IMEI: device identification number. Unique for each device
TS: timestamp DD:MM:YYYY HH:MM:SS

TYPE: data type. In this case, Modbus

P: Logger ID field (External Devices > Logger configuration) ST:
initial record

A: Modbus address of the device read V:
array with the registers read

- We recommend reading the chapter on available AT commands, as it is possible to read and change Modbus register values via AT commands, from the web configuration environment, from Remote Console (Telnet), SSH, or from SMS, etc.
- In the "Start" and "Num Words" fields, it is possible to set non-consecutive registers from firmware version 3.26 onwards. See application note AN27 (ANV6_27-Router-TITAN-Modbus-RTU-TCP-Concentrator-multimap_HTTP.pdf) and AN32 ANV6_32-Router-TITAN-Modbus-RTU-TCP-Concentrator-multimap_MQTT.pdf for more information and examples.

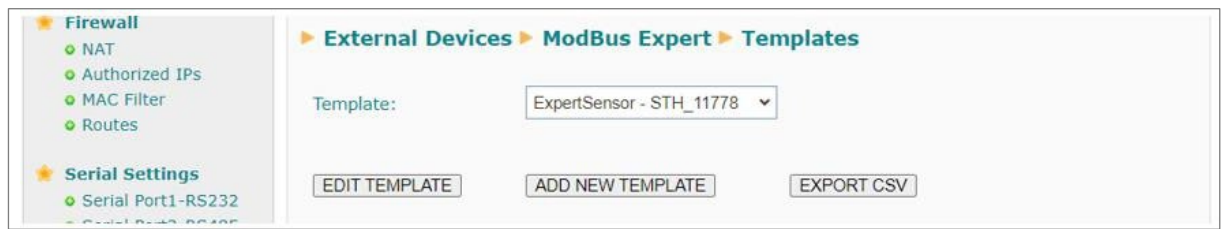
Expert Mode

The "Expert Mode" operating mode is very similar to "Basic Mode" but with significantly increased performance compared to its predecessor. The main screen is divided into three distinct sections. The first is shown in the following screen:

The screenshot shows the 'ModBus Expert' configuration page. On the left is a sidebar menu with categories: Mobile (Status, Basic Settings, Keep Online), Ethernet (Basic Settings, DHCP Server), and Wifi (Basic Settings, DHCP Server). The main content area is titled 'External Devices > ModBus Expert' and includes a 'Change to basic mode' button. It features three settings: 'Enabled' (checkbox), 'Serial Port' (dropdown menu set to 'Serial Port 2'), and 'Logger' (checkbox). Below these are 'SAVE CONFIG' and 'VIEW LOG' buttons. Explanatory text on the right states: 'Enable Modbus Devices', 'Select the connected serial port if needed', 'Check if logger must be used', and 'Please, configure logger before using this option'.

- **Enabled:** Check this box to activate the Modbus "Expert Mode" service. If the Modbus "Basic Mode" service is activated, it will be automatically deactivated as it is not possible to use both services simultaneously.
- **Serial Port:** Allows you to select the RS232 or RS485 serial port on the TITAN-based device where the Modbus RTU devices are connected. You can also choose the "None" option if you are only going to work with Modbus TCP devices. If you are going to work simultaneously with Modbus RTU and Modbus TCP devices, specify the serial port to which your Modbus RTU devices are connected ("Serial Port 1" for RS232 and "Serial Port 2" for RS485).
- **Logger:** Tick this box if you wish to store Modbus log readings in the Titan router's internal data logger for subsequent transmission to a WEB platform (HTTP, MQTT, FTP).

The following section shows the device templates. Device templates are useful if you want to add multiple devices of the same type to an installation. By creating a template, you can quickly add a new Modbus device based on a template without having to create the entire Modbus register map again.



- *Edit template button:* Allows you to edit the template indicated in the "Template" drop-down menu.
- *Add new template:* Allows you to create a new template.
- *Export CSV:* Allows you to export a template in CSV format. Useful for importing it into other computers.

The last section of the screen displays a table of the Modbus devices created.



This table shows the name of the device and whether it is a Modbus RTU or Modbus TCP address. Clicking on the "EDIT DEVICE" button for one of the devices will take you to the edit screen for the corresponding Modbus records for that device.

To add a new device to the list, simply choose a template from which to start creating it. If you do not want to select a template, simply select "No template" and click the "ADD NEW DEVICE FROM TEMPLATE" button.

This last section includes a button labelled "RESTART MODBUS SERVICE". If the Modbus Expert service is running, this allows you to reload the configuration of all devices and restart the service. This allows you to make changes to the Modbus device configurations (modifications, device deletion, new device registration) and restart the service without having to restart the Titan router.

Device editing screen:

The screenshot shows the 'Device editing screen' in the webdyn TITAN interface. The top header features the webdyn logo and the TITAN tagline 'Makes your APPLICATION happen'. A left sidebar contains a navigation menu with categories: Mobile (Status, Basic Settings, Keep Online), Ethernet (Basic Settings, DHCP Server), Wifi (Basic Settings, DHCP Server), Firewall (NAT, Authorized IPs, MAC Filter, Routes), and Serial Settings (Serial Port1-RS232, Serial Port2-RS485, SSL Certificates). The main content area is titled 'External Devices > ModBus Expert > Device'. It contains form fields for 'Name' (Device Name), 'Template Brand' (From original template (non-editable)), 'Template Model' (From original template (non-editable)), 'Address' (RTU or IP@ID:PORT), and 'Period' (Data will be saved each period (minutes)). Below these fields are buttons for 'SAVE DEVICE', 'DELETE DEVICE', and 'RETURN TO MODBUS PAGE'. A file upload section shows 'File: modbusd-3.csv' and a 'Select v[x] range:' dropdown set to '0 - 49'. A table displays the Modbus record map with columns: v[x], register, type, flip, com., name, units, and mode. The table has four rows for v[x] values 0, 1, 2, and 3. Each row includes a 'Period factor' dropdown (set to 1) and a 'Script' input field. The 'mode' column for all rows is set to 'Not used'.

v[x]	register	type	flip	com.	name	units	mode
0		UInt16	No	3			Not used
	Period factor	1	Script:				
1		UInt16	No	3			Not used
	Period factor	1	Script:				
2		UInt16	No	3			Not used
	Period factor	1	Script:				
3		UInt16	No	3			Not used
	Period factor	1	Script:				

The upper section of the previous screen allows you to configure the following parameters:

- **Name:** Device name
- **Template Brand:** Not editable. Imported from the template used.
- **Template Model:** Not editable. Imported from the template used.
- **Address:** Modbus RTU address of the device or IP address in the format IP@ID:PORT, where IP is the IP address of the Modbus TCP device, ID (optional) is its internal address (usually 1) and PORT is the TCP port.
- Valid examples: 192.168.1.100:502, [192.168.1.100@1:502](#) It is also possible to set the address to "0" if you wish to disable the device at any time (so that it cannot be read) without having to remove it.
- **Period:** Indicates the base period, in minutes, during which the Modbus records read will be stored inside the Titan router's data logger for subsequent transmission to a web platform. For example, if you specify 5, the records read are stored, by default, every 5 minutes.

The lower section shows the complete map of modbus records for the device:

v[x]	register	type	flip	com.	name	units	mode
0		UInt16 ▾	No ▾	3 ▾			Not used ▾
	Period factor	1 ▾	Script:				

- **V[0]:** This is an array where the value read by the register is stored. It is important for the "Script" section, as will be seen later.
- **Register:** The register number to be read must be specified. For example, 40000. If the register is of type String, the register address must be specified, followed by "_" and ending with the number of characters. For example, for a String of length 10 starting at register 42000, "42000_10" should be specified.
- **Type:** The type of Modbus register to be read must be specified. Available data types:
 - UInt16: 16-bit unsigned integer (1 WORD)
 - UInt32: 32-bit unsigned integer (2 WORD)
 - UInt64: 64-bit unsigned integer (4 WORD)
 - Int16: 16-bit integer (1 WORD)
 - Int32: 32-bit integer (2 WORD)
 - Int64: 64-bit integer (4 WORD)
 - Float: Float (32 bits, 2 WORD)
 - Double: Double (64 bits, 4 WORD)
 - String: String (X WORD)
 - Bit: Bit (1 WORD)
- **Flip:** allows the rotation of words or bytes of a data type, permitting all combinations.
 - No: No rotation is performed
 - W: Rotation of register words
 - B: Rotation of bytes in each word
 - W+B: Rotation of words and bytes.
 - W2: Rotation of 2 words in the register
 - W2+B: Rotation of 2 words and bytes

Example of rotation for a DOUBLE type register:

```
No: AB CD EF GH
W: GH EF CD AB
B: BA DC FE HG
W+B: HG FE DC BA
W2: EF GH AB CD
W2+B: FE HG BA DC
```

- **Com.:** specifies the register read command
 - 1: Read coil type registers
 - 2: Read bit type registers (discrete inputs)
 - 3: Reading holding registers
 - 4: Read input registers.
- **name:** (Optional). Text with the name of the register.
- **name:** (Optional). Text with the register units.

- **mode:** Indicates what should be done with the register. The options are as follows:

- Not used: The record will not be read by the router.

- Not saved: The log will be read, but the reading will not be stored in the data logger. This can be useful for certain types of logs that depend on others. For example, if a log X only indicates whether a third log has units in volts or millivolts, it may be necessary to read it, but not store the information in the data logger. In that case, the "Not saved" mode is appropriate.

- Instant: The record will be read, and when the specified time in the period arrives, the value read at that moment (instantaneous value) will be stored in the data logger.

- Average: The record will be read, and when the specified time in the period arrives, the average value recorded during the reading period will be stored in the data logger.

- Minimum: The register will be read, and when the specified time in the period arrives, the minimum value read during the reading period will be stored in the data logger.

- Maximum: The record will be read, and when the specified time in the period arrives, the maximum value read during the reading period will be stored in the data logger.

Note: the number of samples taken to generate the average values will vary depending on the number of devices and the records to be read. The Titan router will read all records cyclically at the highest possible speed. The greater the number of devices and the greater the number of readings to be taken, the lower the number of samples.

- **Period factor:** Numeric field that allows a period multiple for the record. For example, if the "Period" field has a value of 5, the readings will be stored in the data logger every 5 minutes by default. However, if a specific record is to be stored every 15 minutes, a "3" should be entered in the "Period factor" field. The data would be stored in the data logger 1 out of every 3 times every 5 minutes, i.e. every 15 minutes.

Script: (Optional). Small script in javascript (v5) that can be entered to operate on a read value. It allows access to the `v[x]` values and also allows access to the "mtx" object (see the TITAN SCRIPTS section for more information on the "mtx" object). Note: keep in mind that if you run a script on an int64 or uint64 value containing a very large integer, there may be a loss of precision, as JavaScript loses precision for integers greater than 9007199254740991 or less than -9007199254740991.

Examples:

v[x]	register	type	flip	com.	name	units	mode
0	40000	UInt16 ▾	No ▾	4 ▾	Voltage	Volts	Instant ▾
	Period factor	1 ▾	Script:	return v[0]/1000;			

The script "**return v[0]/1000;**" would mean that the value stored in the data logger would not be directly the value read from register 40000, but rather, before being stored in the data logger, it would be divided by 1000 to store the value in volts.

v[x]	register	type	flip	com.	name	units	mode
0	40000	UInt16 ▾	No ▾	4 ▾	Voltage	Volts	Instant ▾
	Period factor	1 ▾	Script:	if (v[1]==0) {return v[0];} else {return v[0]/1000;}			

The script:

```

If (V[1]==0) {
    return v[0];
} else {
    return v[0]/1000;
}

```

would mean that the value stored in the data logger would not be the value read from register 40000, but rather, before storing it, if register v[1] has the value "0", the value of v[0] would be stored directly. On the other hand, if v[1] has the value "1", the value v[0]/1000 would be stored.

In other words, scripts allow operations to be performed on records before they are stored in the data logger, even allowing interaction with the values of other records.

v[x]	register	type	flip	com.	name	units	mode
0	30000	Int16 ▾	No ▾	3 ▾	Temperature	Celsius	Average ▾
	Period factor	1 ▾	Script:	if (v[0]>=300) mtm.modbusTCPSetBit("192.168.1.10",502,5,1)			

Let us imagine that with this configuration, in record 30000, v[0], the average temperature is stored, which the sensor returns in tenths of a degree. For example, it returns 300 to indicate 30.0°C, 252 to indicate 25.2°C, etc. The aim is to store the average temperature in the data logger, but storing the value in degrees Celsius, which is why we will have to include a **"return v[0]/10"** in the script.

However, if the temperature is equal to or greater than 30°C, a value of "1" must be written to the Modbus 10000 bit register of a PLC that operates with Modbus TCP in order to activate a fan. It is also necessary that, if the temperature is below 29°C, a "0" be written in that register 10000 to stop the fan.

Therefore, the final script would look like this:

```

if (v[0]>=300)
    mtm.modbusTCPSetBit("192.168.1.10",502,5,10000,1);
elseif (v[0]<=290)    mtm.modbusTCPSetBit("192.168.1.10",502,5,10000,0);
return v[0]/10;

```

Note: From the script of one record, it is possible to refer to the value of another record. For example, from the script of record v[1], you can refer to record v[0]. V[0] refers to the value of the record without going through the script that may be present in record v[0]. In other words, if there is a script in record v[0] that says "return v[0]*2;", references to v[0] from the script of v[1] will be to the value of v[0] before it is multiplied by 2. If you want to obtain the value of v[0] after it has passed through the script (i.e., the value multiplied by 2), you must refer to **vs[0]** instead of **v[0]** in the v[1] script.

Storage format in Datalogger.

When Modbus records are stored in the data logger's memory, they are saved in the following JSON format, described with an example:

```
{ "IMEI": "865583042283167", "TYPE": "MODB2", "TS": "2023-10-03T11:46:00Z", "P": "TITAN", "ID": "Device2", "A": "2", "data": [{"R": 30000, "V": "30.0", "N": "Temperature", "M": "3", "U": "Celsius", "S": "OK"}, {"R": 30001, "V": "40.0", "N": "Humidity", "M": "3", "U": "%", "S": "OK"}]}
```

Where:

IMEI: router IMEI

TYPE: frame type

TS: timestamp of when the Modbus logs were read **P:** "ID" field in the "Logger" configuration section **ID:** "ID" field in the Modbus device section

A: Modbus address of the device

R: Register address

V: Register value

N: Record name

M: Recording mode (2: instantaneous value, 3: average value, 4: maximum value, 5: minimum value)

U: Units

S: Reading status (OK: correct reading, ERR: incorrect reading)

Additional notes.

- We recommend reading the application note "**ANV6_72-Router-Titan-Modbus-Expert**" for a better understanding.

4.6.4 External Devices - GPS Receiver

This menu allows you to configure the GPS positioning data for TITAN-based devices that use an internal GPS module or, if they do not have an internal GPS module, an NMEA GPS receiver connected to one of their RS232 or RS485 ports.

- *Enabled*: Check this box if you have an internal (or external) GPS module connected to the TITAN-based device you want to use.
- *Serial Port*: Select the serial port of the TITAN-based device to which the GPS is connected.
- *Interval*: Select the GPS reading interval in seconds (you will obtain a GPS position every few seconds). Set the minimum value to 30 seconds.
- *Logger*: tick this box if you want to record the GPS positions read in the internal logger so that you can send them to your web platform via HTTP/HTTPS, MQTT/MQTTS or FTP, for example. This is very useful if you want to use TITAN-based equipment to implement a GPS fleet control system.
- *System time*: tick this box if you want the system time to be synchronised with the time read from the GPS.
- *TCP port listener*: allows NMEA frames to be obtained via an IP connection made on the TITAN-based device. For example, if you open a socket against the IP of the TITAN-based device to the port specified at this point, you will be able to obtain NMEA positioning frames in real time.
- *TCP Client IP*: Allows you to specify an IP address to send NMEA frames to an external server.
- *TCP Client Port*: Allows you to specify the TCP port to which NMEA frames from the external server specified in the "TCP Client IP" parameter should be sent.
- *TCP Client Timeout*: specifies the number of seconds that the TITAN-based device should wait between detecting a socket closure and reopening it.

- ★ **Mobile**
 - ◆ Status
 - ◆ Basic Settings
 - ◆ Keep Online
- ★ **Ethernet**
 - ◆ Basic Settings
- ★ **Firewall**
 - ◆ Authorized IPs
- ★ **Serial Settings**
 - ◆ Serial Port1-RS232
 - ◆ Serial Port2-RS485
 - ◆ SSL Certificates
- ★ **External Devices**
 - ◆ Logger configuration
 - ◆ ModBus Devices
 - ◆ Generic Serial Device
 - ◆ Temperature Sensor
 - ◆ IEC102 Meter
 - ◆ GPS Receiver
- ★ **Other**
 - ◆ AT Command

▶ External Devices ▶ GPS Receiver

Enabled:	☐	Enable GPS Receiver
Serial Port:	Serial Port 1 ▼	Select the connected serial port
Interval:	60	Reading period for logger (seconds, minimum 30)
Logger:	☐	Check if logger must be used. Please, configure logger before using this option
System time:	☐	Check for System time synchronization
TCP port listener:	20030	Redirect NMEA data to TCP server socket port if greater than 0 (1 frame / second)
TCP Client IP:		If filled with IP or DNS, Titan will send NMEA raw data to this server
TCP Client Port:	20030	
TCP Client timeout:	5	If socket closes, will wait this seconds in order to re-open the socket.

SAVE CONFIG

Additional notes.

- Once the configuration is complete, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the device for the new changes to take effect.

4.6.5 External Devices – Generic Serial Device

From this section, you can configure a data logger for RS232 or RS485 serial devices. Basically, TITAN-based equipment will read and store the serial data collected by any of the aforementioned interfaces and send it to a web platform via HTTP/S, MQTT/S or FTP.

This is particularly suitable for collecting data from serial devices that transmit a data frame at regular intervals (temperature sensors, humidity sensors, serial alarms, etc.).

- **Enabled:** Check this box if you want to connect a serial device and have the TITAN-based device perform the data logger function.
- **Serial Port:** Select the serial port of the TITAN-based device where the serial device to be logged is connected.
- **Interval:** If you specify "0", it will collect all received serial frames. If you specify "1", it will collect one out of every two received frames. If you specify "2", it will collect one frame out of every three received frames.
- **Only changes:** Check this box if you only want to log frames that differ from the previous frame. For example, if you connect a temperature sensor, that sensor may be emitting the same serial data for quite some time until the temperature changes. If you check this box, only the serial data will be collected when it changes.
- **Logger:** tick this box if you want to record the serial data in the internal logger to send it later to your web platform via HTTP/HTTPS, FTP or MQTT/MQTTS.
- **TX period:** a number greater than 0 indicates the period (in seconds) in which the data frames "TX Frame1", "TX Frame2", "TX Frame3", "TX Frame4", "TX Frame5" will be sent through the serial port. This option allows you to "interrogate" serial devices with proprietary protocols.
- **TX FrameX:** indicates a data frame that will be resent via the serial port every X seconds specified in the "TX period" parameter. These frames must be specified in hexadecimal format. Example: 0102030A0D...

★ Mobile

- Status
- Basic Settings
- Keep Online

★ Ethernet

- Basic Settings

★ Wifi

- Basic Settings
- DHCP Server

★ Firewall

- NAT
- Authorized IPs

★ Serial Settings

- Serial Port1-RS232
- Serial Port2-RS485
- SSL Certificates

★ External Devices

- Logger configuration
- ModBus Devices
- Generic Serial Device
- Temperature Sensor
- IEC102 Meter
- W-MBus

▶ External Devices ▶ Generic Serial Device

Enabled:	☐	Enable Generic Serial Device
Serial Port:	Serial Port 1	Select the connected serial port
Interval:	0	0=save every serial frame
Only changes:	☐	Check for register every serial frame if different than previous one.
Mode:	Raw (Default)	Communication mode with Titan
Logger:	☐	Check if logger must be used. Please, configure logger before using this option
TX Period:	0	Period mof TX Frames (seconds 30 ... 86400. 0=no TX Data)
TX Frame1:		HEX Frame 1 (max length 256)
TX Frame2:		HEX Frame 2 (max length 256)
TX Frame3:		HEX Frame 3 (max length 256)
TX Frame4:		HEX Frame 4 (max length 256)
TX Frame5:		HEX Frame 5 (max length 256)

[SAVE CONFIG](#)

[VIEW LOG](#)

4.6.6 External Devices – IEC102 Meter

From this section, you can configure TITAN-based equipment to read an electricity meter with the IEC 60870-5-102 protocol.

- **Enabled:** Check this box if you want to connect an IEC-60870-5-102 protocol electricity meter that the TITAN-based device must query autonomously.
- **Communication Port:** Select the serial port of the TITAN-based device where the IEC-60870-5-102 electricity meter is connected.
- **Period:** period (in minutes) during which the IEC-60870-5-102 electricity meter will be interrogated by the TITAN-based device to read the instantaneous values.
- **ID Meter:** Enter an identifying text for the electricity meter.
- **Link Address:** link address of the electricity meter.
- **Meas. Address:** measurement point address of the electricity meter.
- **Password:** Password for the electricity meter.
- **Sign:** Select the box if you want the PAT/FPT, PAF1/FPF1, PAF2/FPF2, PAF3/FPF3 parameters to contain a negative value if energy is exported.
- **Optical probe:** Select the box if you are using an optical probe with echo (if you do not select this box and the optical probe has echo, the meter reading will not work).
- **Fiscal close:** Check the box if, in addition to reading instantaneous values, the TITAN-based device must read fiscal closes. If the box is checked, the TITAN-based device will read and store the meter closes every midnight.
- **Fiscal close days:** Indicate the number of days for reading the fiscal closes. Default is 31 days.
- **Logger:** tick this box if you want to record the serial data in the internal logger to send it later to your web platform via HTTP/HTTPS, FTP or MQTT/MQTTS.

Mobile

Status

Basic Settings

Keep Online

Ethernet

Basic Settings

Wifi

Basic Settings

DHCP Server

Firewall

NAT

Authorized IPs

Serial Settings

Serial Port1-RS232

Serial Port2-RS485

SSL Certificates

External Devices

Logger configuration

ModBus Devices

Generic Serial Device

Temperature Sensor

IEC102 Meter

W-MBus

GPS Receiver

External Devices

IEC102 Meter

Enabled:

☒

Enable Temperature sensor

Communication Port:

Serial Port 1

Select the serial port for reading

Period:

1 minute

Reading period for instant values

ID Meter:

ID00000

Customizable label for identification

Link address:

1

Link address of meter (1...65535)

Meas. address:

1

Measurement address (1...65535)

Password:

1

Password of meter (1...65535)

Fiscal close:

☒

Read and send Fiscal close data once per day (Contract I)

Fiscal close days:

31

Number of days (1 ... 31) Default 31

Logger:

☒

Logger must be used. Please, [configure logger](#) before using this option.

Recommended 'Register Size': 7000, 'Number of registers': 100

SAVE CONFIG

VIEW LOG

Additional notes.

- Once you have finished configuring, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the device for the new changes to take effect.
- If you also need to read the integrated totals (load curves), you will need to do so from the scripts section.
- The frames sent by the TITAN-based device have the following format:

 webdyn | 62

contact@webdyn.com | webdyn.com
V02.08 subject to change | Webdyn © by Flexitron

Example of a frame stored in the Logger with instantaneous values:

```
{"IMEI":"867962046823806","TYPE":"IEC102","TS":"2021-11-02T15:14:04Z","P":"1234","ID":"ID00000","VABA":0,"VABRI":0,"VABRC":0,"PAT":0,"PRT":0,"FPT":1000,"PAF1":0,"PRF1":0,"FPF1":1000,"PAF2":0,"PRF2":0,"FPF2":1000,"PAF3":0,"PRF3":0,"FPF3":1000,"IF1":0,"TF1":1337,"IF2":0,"TF2":1108,"IF3":0,"TF3":3}
```

Where:

IMEI:	device identification number based on TITAN
TYPE:	JSON type. In this case, type IEC102
TS:	timestamp of when the data was collected
P:	optional identification field (from the Logger)
ID:	Counter identifier
VABA:	Absolute active energy
VABRI:	Absolute inductive reactive energy
VABRC:	Absolute inductive reactive energy
PAT:	Total active power
PRT:	Total reactive power
FPT:	Total power factor
PAF1:	Active power phase I
PAF1Dir	Imported (0) / Exported (1)
PRF1:	Reactive Power Phase I
FPF1:	Phase I Power Factor
PAF2:	Active Power Phase II
PAF2Dir	Imported (0) / Exported (1)
PRF2:	Reactive Power Phase II
FPF2:	Phase II Power Factor
PAF3:	Active Power Phase III
PAF3Dir	Imported (0) / Exported (1)
PRF3:	Reactive Power Phase III
FPF3:	Phase III Power Factor
IF1:	Phase I Current
TF1:	Phase I Voltage
IF2:	Phase II Current
TF2:	Phase II Voltage
IF3:	Phase III Current
TF3:	Phase III Voltage

Other

AT Command
DynDns
Private DynDns
Sms control
Periodic Autoreset
Time Servers
Remote Console
Snmp
Tacacs+
Mqtt
Http / Https
User Permissions
Passwords Web UI
CA Certificates
Email Config
ModBus Slave
Titan Scripts
Connectivity tools
Digital I/O
Custom Skin
Led Config
Syslog
Backup / Factory
Firmware Upgrade
Reboot
Logout

External Devices ▶ IEC102 Meter ▶ Read Instant Values

TS:	TimeStamp (Titan time)
VabA:	Active absolute energy
VabRi:	Inductive Reactive Absolute Energy
VabRc:	Absolute Reactive Capacitive Energy
pat:	Total Active Power
prt:	Total Reactive Power
fpt:	Total power factor
paf1:	Active Power Phase I
prf1:	Reactive Power Phase I
fpf1:	Power Factor Phase I
paf2:	Active Power Phase II
prf2:	Reactive Power Phase II
fpf2:	Power Factor Phase II
paf3:	Active Power Phase III
prf3:	Reactive Power Phase III
fpf3:	Power Factor Phase III
if1:	Current Phase I
tf1:	Voltage Phase I
if2:	Current Phase II
tf2:	Voltage Phase II
if3:	Current Phase III
tf3:	Voltage Phase III

REFRESH DATA

FORCE READ (Force read will take 10 seconds)

- See application note **ANV6_51-Router-TITAN-Metering-IEC-60870-5-102.pdf**

Example of a frame stored in the Logger with fiscal closures:

```
{
  "IMEI": "869101057093607",
  "TYPE": "IEC102_CTAVM2",
  "TS": "09/06/2023 08:46:23",
  "P": "1234",
  "ID": "ID00000",
  "CTAVM2": [
    {
      "DO": 20,
      "EaA": 0,
      "EiA": 0,
      "CA": 2,
      "EaRi": 0,
      "EiRi": 0,
      "CRi": 2,
      "EaRc": 0,
      "EiRc": 0,
      "CRc": 2,
      "R7": 0,
      "C7": 128,
      "R8": 0,
      "C8": 128,
      "MPA": 0,
      "FMFA": "2023-01-09T12:58-00",
      "CMA": 2,
      "EPA": 0,
      "CE": 128,
      "DINI": "2022-12-01T00:00-00",
      "DEND": "2023-01-09T12:58-00"
    },
    {
      "DO": 21,
      "EaA": 0,
      "EiA": 0,
      "CA": 2,
      "EaRi": 0,
      "EiRi": 0,
      "CRi": 2,
      "EaRc": 0,
      "EiRc": 0,
      "CRc": 2,
      "R7": 0,
      "C7": 128,
      "R8": 0,
      "C8": 128,
      "MPA": 0,
      "FMFA": "2023-01-09T12:58-00",
      "CMA": 2,
      "EPA": 0,
      "CE": 0,
      "DINI": "2022-12-01T00:00-00",
      "DEND": "2023-01-09T12:58-00"
    },
    {
      "DO": 22,
      "EaA": 0,
      "EiA": 0,
      "CA": 0,
      "EaRi": 0,
      "EiRi": 0,
      "CRi": 0,
      "EaRc": 0,
      "EiRc": 0,
      "CRc": 0,
      "R7": 0,
      "C7": 128,
      "R8": 0,
      "C8": 128,
      "MPA": 0,
      "FMFA": "2023-01-09T12:58-00",
      "CMA": 0,
      "EPA": 0,
      "CE": 0,
      "DINI": "2022-12-01T00:00-00",
      "DEND": "2023-01-09T12:58-00"
    },
    {
      "DO": 23,
      "EaA": 0,
      "EiA": 0,
      "CA": 0,
      "EaRi": 0,
      "EiRi": 0,
      "CRi": 0,
      "EaRc": 0,
      "EiRc": 0,
      "CRc": 0,
      "R7": 0,
      "C7": 128,
      "R8": 0,
      "C8": 128,
      "MPA": 0,
      "FMFA": "2023-01-09T12:58-00",
      "CMA": 0,
      "EPA": 0,
      "CE": 0,
      "DINI": "2022-12-01T00:00-00",
      "DEND": "2023-01-09T12:58-00"
    }
  ]
}
```

```
{
  "DO":24,"EaA":0,"EiA":0,"CA":0,"EaRi":0,"EiRi":0,"CRi":0,"EaRc":0,"EiRc":0,"CRc":0,"R7":0,"C7":128,"R8":0,"C8":128,"MPA":0,"FMPA":"2023-01-09T12:58-00","CMA":0,"EPA":0,"CE":0,"DINI":"2022-12-01T00:00-00","DEND":"2023-01-09T12:58-00"},
  {"DO":25,"EaA":0,"EiA":0,"CA":0,"EaRi":0,"EiRi":0,"CRi":0,"EaRc":0,"EiRc":0,"CRc":0,"R7":0,"C7":128,"R8":0,"C8":128,"MPA":0,"FMPA":"2023-01-09T12:58-00","CMA":0,"EPA":0,"CE":0,"DINI":"2022-12-01T00:00-00","DEND":"2023-01-09T12:58-00"},
  {"DO":26,"EaA":0,"EiA":0,"CA":0,"EaRi":0,"EiRi":0,"CRi":0,"EaRc":0,"EiRc":0,"CRc":0,"R7":0,"C7":128,"R8":0,"C8":128,"MPA":0,"FMPA":"2023-01-09T12:58-00","CMA":0,"EPA":0,"CE":0,"DINI":"2022-12-01T00:00-00","DEND":"2023-01-09T12:58-00"}
}
```

}

Where:

IMEI:	TITAN-based device identification number	TYPE:
	JSON type. In this case, type IEC102_CTAVM2	TS:
	timestamp of when the data was collected	
P:	optional identification field (from the Logger)	
ID:	Meter identifier	
CTAVM2:	Data array with the readings obtained.	
DO:	Object address	
EaA:	Absolute active energy	
EiA:	Active Incremental Energy	
CA:	Active Energy Qualifier	
EaRi:	Absolute Inductive Reactive Energy	
EiRi:	Incremental Inductive Reactive Energy	Cri:
	Inductive Reactive Energy Qualifier	EaRc:
	Absolute Capacitive Reactive Energy	EiRc:
	Incremental Capacitive Reactive Energy	
CRc:	Capacitive Reactive Energy Qualifier	
R7:	Register 7 reserve	
C7:	Reserve Register 7 Qualifier	
R8:	Register 8 reserve	
C8:	Reserve Register 8 qualifier	
MPA:	Maximum Power	
FMPA:	Date of Maximum	
CMA:	Maximum Qualifier	
EPA:	Power Excesses	
CE:	Excess Qualifier	
DINI:	Start of period	
DEND:	End of period	

4.7 Other

From this section, it is possible to execute AT commands, use connectivity tools, update the device's firmware, and schedule and execute scripts. It also allows you to configure DynDNS, Autoreset, NTP time servers, Remote console (Telnet/SSH), SNMP, Tacacs+, LDAP, MQTT, Email, GPIOs, Syslog, CA certificates, Modbus Slave, appearance (logos) and LEDs.

4.7.1 Other – AT Command

From this section, it is possible to send an AT command directly to the internal modem of the TITAN-based device. For example, this can be useful for checking coverage, identifying nearby mobile phone cells, etc.

It is also possible to configure up to 5 special AT commands to configure the device at start-up.

- **AT Command:** AT command for real-time execution (e.g. AT+CSQ). Once you press the "SEND AT COMMAND" button, the AT command will be executed.
- **AT1, ... AT5:** initialisation AT commands.



The screenshot shows the webdyn web interface. At the top, there is a logo for 'webdyn' with the tagline 'powered by TITAN' and 'Flexitron group'. On the left side, there is a sidebar menu with categories: Mobile (Status, Basic Settings, Keep Online), Ethernet (Basic Settings), Wifi (Basic Settings, DHCP Server), Firewall (NAT, Authorized IPs), Serial Settings (Serial Port1-RS232, Serial Port2-RS485, SSL Certificates), and External Devices (Logger configuration, ModBus Devices, Generic Serial Device, Temperature Sensor, IEC102 Meter, W-MBus, GPS Receiver). The main content area is titled 'Other > AT Command'. It contains a text input field for 'AT Command:' with the value 'at+cops?'. Below it, the 'AT Command Response:' is displayed in a text area, showing the output: 'at+cops? +COPS: 0,0,"Movistar",7 OK'. At the bottom of the main content area, there are two buttons: 'SEND AT COMMAND' and 'COPY TO CLIPBOARD'.

Additional notes.

- If you have set initialisation AT commands, press the "SAVE CONFIG" button to save the changes.
- Remember that you must restart the device for the new changes to take effect.

4.7.2 Other – DynDns

TITAN-based devices are compatible with the DynDNS and No-IP services. If you do not have a SIM card with a fixed IP and are interested in using the external DynDNS or NO-IP services, you can configure them in this section.

- **Enabled:** Check this box if you want to enable the use of DynDNS or NO-IP.
- **Server:** Specify the service server (members.dyndns.org or dynupdate.no-ip.com).
- **Domain:** enter the DNS you have created (for example, mydomain.dyndns.org)
- **Login:** your DynDNS or NO-IP account login
- **Password:** your DynDNS or NO-IP account password
- **Period:** period, in minutes, during which the current IP is refreshed on the DynDNS or NO-IP servers
- **Hide WAN IP:** if this box is selected, the IP will be detected by the DynDNS server. This must be enabled if you wish to use DynDNS with the Ethernet or Wi-Fi port of your TITAN-based device as your Internet connection.



The screenshot shows the 'webdyn' configuration interface, powered by TITAN. On the left is a sidebar menu with categories: Mobile (Status, Basic Settings, Keep Online), Ethernet (Basic Settings), Wifi (Basic Settings, DHCP Server), Firewall (NAT, Authorized IPs), and Serial Settings (Serial Port1-RS232, Serial Port2-RS485, SSL Certificates). The main content area is titled 'Other > DynDns'. It contains the following fields and options:

- Enabled:** A checkbox that is currently unchecked. To its right is the text 'Enable DynDNS'.
- Server:** A text input field with the placeholder text 'members.dyndns.org, dynupdate.no-ip.com, ...'.
- Domain:** A text input field with the placeholder text 'Your domain. For example myDomain.dyndns.org'.
- Login:** A text input field with the placeholder text 'Login of your account'.
- Password:** A text input field with the placeholder text 'Password of your account'.
- Period:** A numeric input field with the value '0'. To its right is the text 'Minutes. Public IP is sent every time it changes. But it is recommended force an update every 60 minutes.'
- Hide WAN IP:** A checkbox that is currently unchecked. To its right is the text 'Public IP detected by server (mandatory if Ethernet or Wifi gateways are used)'.

At the bottom left of the main content area is a button labeled 'SAVE CONFIG'.

Additional notes.

- Once you have finished configuring, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the device for the new changes to take effect.
- The IP address is updated on the DynDNS or NO-IP servers each time it changes. However, we recommend using the "Period" configuration parameter, for example, with a value of 60, so that whatever happens, it is sent every hour.

4.7.3 Other – Private DynDns

The Private DynDNS service allows the current IP address of the TITAN-based device and certain status values to be sent to your own server. Data can be sent either via HTTP (HTTP/S GET, HTTP/S POST) or by sending a specific frame via socket to a configurable TCP port and also via MQTT.

HTTP or TCP Socket Method

- **Enabled:** Check this box if you want to enable the use of Private DNS.
- **Mode:** you can choose between TCP socket and HTTP/S GET, HTTP/S POST and HTTP/S PUT.
- **Server:** the IP or DNS of the remote server
- **Server Login:** your web server login (if using "HTTP" mode)
- **Password:** your web server password (if using "HTTP" mode)
- **TCP Port:** TCP port if using "socket" mode
- **ID:** identification string (for both "http get" and "socket" modes)
- **Period:** period, in minutes, during which the current IP on your server is refreshed. Please note that regardless of this period, as soon as the public IP changes, it is sent.
- **Custom header1:** Custom HTTP header 1
- **Custom header2:** Custom HTTP header 2
- **Custom header3:** Custom HTTP header 3
- **IO Status:** enable if you also wish to send the status of digital inputs and outputs and counters.

MQTT method

- **Enabled:** Check this box if you want to enable the use of MQTT
- **Period:** period, in minutes, during which the current IP will be sent to your server via MQTT. Please note that regardless of this period, the public IP is also sent whenever it changes.
- **MQTT Topic:** MQTT topic to be used to report with the IP
- **MQTT QoS:** Quality of service used by MQTT for sending the IP
- **IO Status:** enable if you also want to send the status of digital inputs and outputs and counters.

- ★ **Mobile**
 - Status
 - Basic Settings
 - Keep Online
- ★ **Ethernet**
 - Basic Settings
- ★ **Wifi**
 - Basic Settings
 - DHCP Server
- ★ **Firewall**
 - NAT
 - Authorized IPs
- ★ **Serial Settings**
 - Serial Port1-RS232
 - Serial Port2-RS485
 - SSL Certificates
- ★ **External Devices**
 - Logger configuration
 - ModBus Devices
 - Generic Serial Device
 - Temperature Sensor
 - IEC102 Meter
 - W-MBus
 - GPS Receiver
- ★ **Other**
 - AT Command
 - DynDns
 - Private DynDns
 - Sms control
 - Periodic Autoreset
 - Time Servers
 - Remote Console
 - Snmp
 - Tacacs+
 - Mqtt

► **Other** ► **Private DynDns**

Use script: ☐ Check for customized json using Json Transformed Script in [Script](#) section.

Communication mode: WEB PLATFORM (HTTP REST)

Enabled: ☐ Enable Private DynDNS

Mode: socket ▼ Communication mode

Server: IP or DNS of remote server. Example: www.myweb.com/set.asp?data=

Server Login: User (only for "http get" mode)

Server Password: Password (only for "http get" mode)

TCP Port: TCP port of remote server (only "socket" mode)

ID: String for device identification

Period: Minutes. Public IP in sent every time it changes. But it is recommended force an update every 60 minutes

Custom header1: Optional. Custom header1. For example: Content-type:application/json

Custom header2: Optional. Custom header2. For example: IDENTITY_KEY;YOUR_KEY

Custom header3: Optional. Custom header3.

I/O status: ☐ Send I/O status and counters

Communication mode: MQTT

Enabled: ☐ Communication mode MQTT enabled

Period: Minutes. Public IP in sent every time it changes. But it is recommended force an update every 60 minutes

MQTT Topic: MQTT Topic. Example: [IMEI]/dns

MQTT QoS: 0 ▼ MQTT QoS for DNS. Normally 0

I/O status: ☐ Send I/O status and counters

Additional notes.

- Once the configuration is complete, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the device for the new changes to take effect.
- Example of the format of the JSON frame sent:

```
{
  "TYPE": "DNS",
  "IMEI": "867962046823806",
  "IP": "88.28.221.24",
  "P": "",
  "CSQ": 24,
  "MOD": "",
  "VER": "5.2.6.02",
  "IMSI": "214075536243578",
  "TECH": "4g",
  "TS": "2021-11-02T16:00:51Z",
  "CID": "214;07;21E0;13B6D0A",
  "RSSI": "-65",
  "RSRP": "-91",
  "RSRQ": "-6",
  "IO0": 0,
  "IO1": 0,
  "IO2": 0,
  "IO3": 0,
  "IO4": 0,
  "CO0": 0,
  "CO1": 5,
  "CO2": 0
}
```

Where:

TYPE: type of frame. In this case, DNS.

IMEI: device identification number. Unique for each device

P: Logger ID field (External Devices > Logger configuration) IP: IP of WAN interface (2G/3G)

CSQ: 0...31 (signal strength)
MOD: TITAN-based equipment model Ver:
FW Version
IMSI: SIM card IMSI TECH:
Technology used (2G, 3G, 4G) TS:
timestamp
CID: Identification of the mobile phone cell used
RSSI: Signal strength level.
RSRP: 3G RSRP 3G
RSRQ: 3G RSRQ
IOx: current value of digital input/output "x"
COx: current value of the pulse counter associated with digital input "x"

4.7.4 Other – SMS Control

This section allows you to configure the control of the device via SMS. For example, you can configure this section so that the device restarts or changes a digital output when an SMS is received, or specify the telephone numbers authorised to do so.

- **AT enabled:** tick this box if you want the TITAN-based device to accept AT commands via SMS, for example, to check coverage remotely, to perform a reset, change a configuration, etc.
- **AT header:** enter the header text for command SMS messages here. For example, if you enter the text "mtx" in this box, when you send an AT command via SMS, let's say the command "AT+CSQ" to check the device's coverage, you would need to send an SMS with the text "mtx AT+CSQ".
- **All phones:** tick this box if you want all phones to be able to send AT commands to the device. Do not tick this box if you want to specify authorised phone numbers.
- **Authorised Number X:** in these boxes you can specify up to 10 authorised telephone numbers.
- **Alias / ATCommand:** up to 10 aliases can be entered to execute SMS commands. Imagine you want to send an SMS to update a specific MODBUS register on an external device. For example, you can configure an ALIAS so that when the TITAN-based device receives the text "reg on", it actually executes the AT command internally: `at^mtxtunnel=setmodbus,1;5;16;2`, writing the value 2 in register 5 of the device with Modbus address 1.
- **Alias Result OK:** text that will be sent as a response if the execution of an ALIAS command is successful. If you wish, you can return a customised response for each ALIAS by indicating the response between tags: `<a1>Ok</a1><a2>Perfect</a2>...`
- **Alias Result ERROR:** text that will be sent as a response if the execution of an ALIAS command is unsuccessful. If you wish, you can return a customised response for each ALIAS by indicating the response between tags: `<a1>Error</a1><a2>Oops</a2>...`
- **SIM 1 Serv. Centre:** updates the SMS messaging centre telephone number to be used by SIM card 1 for sending SMS messages. Leave the field blank if you do not wish to change the default number.
- **SIM 2 Serv. Centre:** updates the SMS messaging centre phone number to be used by SIM card 2 (if your device has DUAL SIM) for sending SMS messages. Leave the field blank if you do not wish to change the default number.

- ★ **Mobile**
 - Status
 - Basic Settings
 - Keep Online
- ★ **Ethernet**
 - Basic Settings
- ★ **Wifi**
 - Basic Settings
 - DHCP Server
- ★ **Firewall**
 - NAT
 - Authorized IPs
- ★ **Serial Settings**
 - Serial Port1-RS232
 - Serial Port2-RS485
 - SSL Certificates
- ★ **External Devices**
 - Logger configuration
 - ModBus Devices
 - Generic Serial Device
 - Temperature Sensor
 - IEC102 Meter
 - W-MBus
 - GPS Receiver
- ★ **Other**
 - AT Command

Other ▶ SMS control

SMS function

AT : ☒ enabled Send AT Commands by SMS allowed (you can reboot the device, get IP Wan, get GSM RSSI, change configuration, ...)

AT header: Header of at commands

Authorized phone numbers: ☒ all phones All Phones are allowed

	Authorized number 1
	Authorized number 2
	Authorized number 3
	Authorized number 4
	Authorized number 5
	Authorized number 6
	Authorized number 7
	Authorized number 8
	Authorized number 9
	Authorized number 10

	ALIAS	AT COMMAND
Alias 1:		

Additional notes.

- Once you have finished configuring the settings, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the computer for the new changes to take effect.
- Refer to application note ANV6_3 for further information with examples. You will also learn how to add parameters to ALIASES.

4.7.5 Other – Periodic autoreset

From this section, you can configure a scheduled autoreset for TITAN-based equipment.

- **Autoreset not enabled:** enable this option if you do not want the TITAN-based equipment to reset automatically.
- **Autoreset every X hours:** enable this option if you want the TITAN-based device to autoreset every certain number of hours.
- **Number of hours:** if you choose to autoreset every X hours, you must specify X in this box, i.e. the number of hours after which the reset is performed. Specify 24 for a daily reset.
- **Auto-reset at specific time:** select this option if you want the TITAN-based equipment to auto-reset at a specific time of day
- **Time for auto-reset:** specific time for the daily auto-reset.
- **Auto-reset if router cannot obtain IP after X minutes:** highly recommended option for situations involving loss of context. For example, it allows you to specify the number of minutes after which the TITAN-based device should auto-reset if it cannot obtain an IP address.

The screenshot shows the webdyn configuration interface. At the top, the webdyn logo is displayed with the text "powered by TITAN" and "flexitron group". On the left, a sidebar menu lists various configuration categories: Mobile (Status, Basic Settings, Keep Online), Ethernet (Basic Settings), Wifi (Basic Settings, DHCP Server), Firewall (NAT, Authorized IPs), Serial Settings (Serial Port1-RS232, Serial Port2-RS485, SSL Certificates), and External Devices (Logger configuration, ModBus Devices, Generic Serial Device, Temperature Sensor, IEC102 Meter, W-MBus). The main content area is titled "Other > Periodic Autoreset". It contains four radio button options: "Autoreset not enabled" (selected), "Autoreset every X hours" (with a "Number of hour:" input field set to 12 and the text "Every X hours device will be rebooted"), "Autoreset at specific hour" (with an "Hour for autoreset" input field set to 0 and a range "0 ... 23"), and "Reset if router can't obtain IP after X minutes" (unchecked, with a "Time for reset" input field set to 60 and a range "5 ... 1440 min."). A "SAVE CONFIG" button is located at the bottom of the configuration area.

Additional notes.

- Once the configuration is complete, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the device for the new changes to take effect.

4.7.6 Other – Time Servers (NTP)

The TITAN-based device has a real-time clock inside that allows it to keep time even after a power loss lasting several hours. This internal clock needs to be synchronised periodically with time servers using the NTP protocol.

Time Servers (NTP)

- **Enabled:** tick this box if you wish to use NTP time servers.
- **NTP Server 1:** IP address or DNS of time NTP server 1
- **NTP Server 1 port:** port of NTP time server 1
- **NTP Server 2:** IP address or DNS of NTP time server 2
- **NTP Server 2 port:** port of NTP time server 2
- **Time Zone:** allows you to specify the time zone.

Time Servers (NTP) → Local Time Server.

- **Enabled:** tick this box if you want to use the Titan router as an NTP time server
- **NTP Server port:** NTP server port on the Titan router

The screenshot shows the webdyn configuration interface. At the top, there is a logo for 'webdyn' with the tagline 'powered by TITAN' and 'Flexitron group'. On the left, there is a sidebar menu with categories: Mobile (Status, Basic Settings, Keep Online), Ethernet (Basic Settings), Wifi (Basic Settings, DHCP Server), Firewall (NAT, Authorized IPs), and Serial Settings (Serial Port1-RS232, Serial Port2-RS485). The main content area is titled 'WAN Time Server (NTP)'. It contains the following settings:

Setting	Value	Description
Enabled:	☒	Enable NTP
NTP Server 1:	ntp.roa.es	IP or DNS address
NTP Server 1 port:	123	UDP port. Default 123
NTP Server 2:	es.pool.ntp.org	IP or DNS address
NTP Server 2 port:	123	UDP port. Default 123
Time zone:	Europe/Madrid	Select the timezone
Current Time:	02-08-2022 10:38:48	Current date & time of the system

At the bottom of the main content area, there is a 'SAVE CONFIG' button.

It is also possible to configure the Titan as an NTP time server. To do this, simply tick the enabled box and select the desired UDP port (default 123). This is useful, for example, to provide time to devices even without the need for a SIM card (as long as it is a Titan model with GPS).

★ **Serial Settings**

- Serial Port1-RS232
- Serial Port2-RS485
- SSL Certificates

★ **External Devices**

- Logger configuration
- ModBus Devices
- Generic Serial Device
- Temperature Sensor
- IEC102 Meter

★ **VPN**

- IPSec
- OpenVPN Client
- OpenVPN Server

★ **Other**

- AT Command
- DynDns

▶ **Other** ▶ **Time Servers (NTP)** ▶ **Local Time Server**

Enabled: ☒ Enable NTP Local Server

NTP Server port: Local UDP Port. Default 123

Additional notes.

- Once you have finished configuring, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the device for the new changes to take effect.

4.7.7 Other – Remote console (TCP Server)

If you ever need to perform a special operation on TITAN-based equipment using a "Telnet" or "SSH" connection, you can do so by configuring this section. In other words, this special connection also allows you to send AT commands to the TITAN-based device via a TELNET or SSH connection. This allows you to change the device configuration, switch a digital output, etc. See section 5 of this manual for a list of available AT commands.

- **Enabled:** tick this box if you wish to use this special connection.
- **TCP Port:** TCP listening port on the computer where you will need to establish the connection.
- **Login:** username that will be requested after establishing the connection.
- **Password:** User password that will be requested after entering the username.
- **SSH:** Tick this box if you want to use SSH instead of Telnet.



The screenshot shows the webdyn configuration interface. At the top, the webdyn logo is displayed with the text "powered by TITAN" and "flexitron group". On the left, a sidebar menu lists various settings: Mobile (Status, Basic Settings, Keep Online), Ethernet (Basic Settings), Wifi (Basic Settings, DHCP Server), and Firewall (NAT, Authorized IPs). The main content area is titled "Other > Remote Console (TCP Server)". It contains the following configuration options:

- Enabled:** A checkbox that is checked, with the label "Enable remote console".
- TCP port:** A text input field containing "20023", with the label "TCP port for remote console".
- Username:** A text input field containing "user", with the label "Username of your account".
- Password:** A text input field, with the label "Password of your account (min 8 char)".
- SSH:** An unchecked checkbox, with the label "Enable SSH security".

At the bottom of the configuration area, there is a "SAVE CONFIG" button.

Additional notes.

- Once you have finished configuring, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the device for the new changes to take effect. **If you enable/disable the SSH box, be sure to re-enter the password as well.**
- You can access the remote console both locally (Ethernet or Wi-Fi) and remotely via a 4G/3G/2G connection.

4.7.8 Other – SNMP

TITAN-based devices feature SNMP protocol. Using SNMP, you can perform SET and GET operations from standard SNMP applications.

- *Enabled*: Select whether you want to enable the SNMP service on TITAN-based equipment.
- *SNMP Version*: You can choose between SNMPv2c and SNMP v3.
- *UDP port*: The standard UDP port for SNMP is 161, but you can specify a custom one.
- *Custom OID*: Allows you to change the Enterprise-Product OID from the default value (.45711.1.1) if you want to adjust it to your corporate values.
- *Community*: The password required to execute SET and GET commands. Only necessary for SNMPv2.
- *Username*: Username when using SNMPv3 (not required for SNMPv2).
- *Auth Password* : Authentication password for SNMPv3 (not required for SNMPv2).
- *Priv Password* : Privacy password for SNMPv3 (not required for SNMPv2).
- *Auth Protocol* : Protocol for authentication (MD5 or SHA)
- *Priv Protocol* : Encryption protocol (DES, AES128)
- *Traps – Enabled*: Enables traps from TITAN-based equipment
- *Traps – UDP Port*: Allows you to specify the port for SNMP TRAPS.
- *Traps – IP*: IP address for sending SNMP TRAPS.
- *Traps – Community*: Community field for sending TRAPS (only for SNMPv2c)
- *Alarm OS*: Enables the device operating system alarm traps in the event of a critical failure.
- *Number traps alarm on*: Indicates the number of traps that will be sent when an alarm trap is activated.
- *Number of alarm traps off*: Indicates the number of traps that will be sent when an alarm trap is deactivated.
- *Trap period*: Indicates the period, in seconds, between sending alarm traps.

- ★ **Mobile**
 - Status
 - Basic Settings
 - Keep Online
- ★ **Ethernet**
 - Basic Settings
- ★ **Wifi**
 - Basic Settings
 - DHCP Server
- ★ **Firewall**
 - NAT
 - Authorized IPs
- ★ **Serial Settings**
 - Serial Port1-RS232
 - Serial Port2-RS485
 - SSL Certificates
- ★ **External Devices**
 - Logger configuration
 - ModBus Devices
 - Generic Serial Device
 - Temperature Sensor
 - IEC102 Meter
 - W-MBus
 - GPS Receiver
- ★ **Other**
 - AT Command
 - DynDns
 - Private DynDns
 - Sms control
 - Periodic Autoreset
 - Time Servers
 - Remote Console
 - Smm

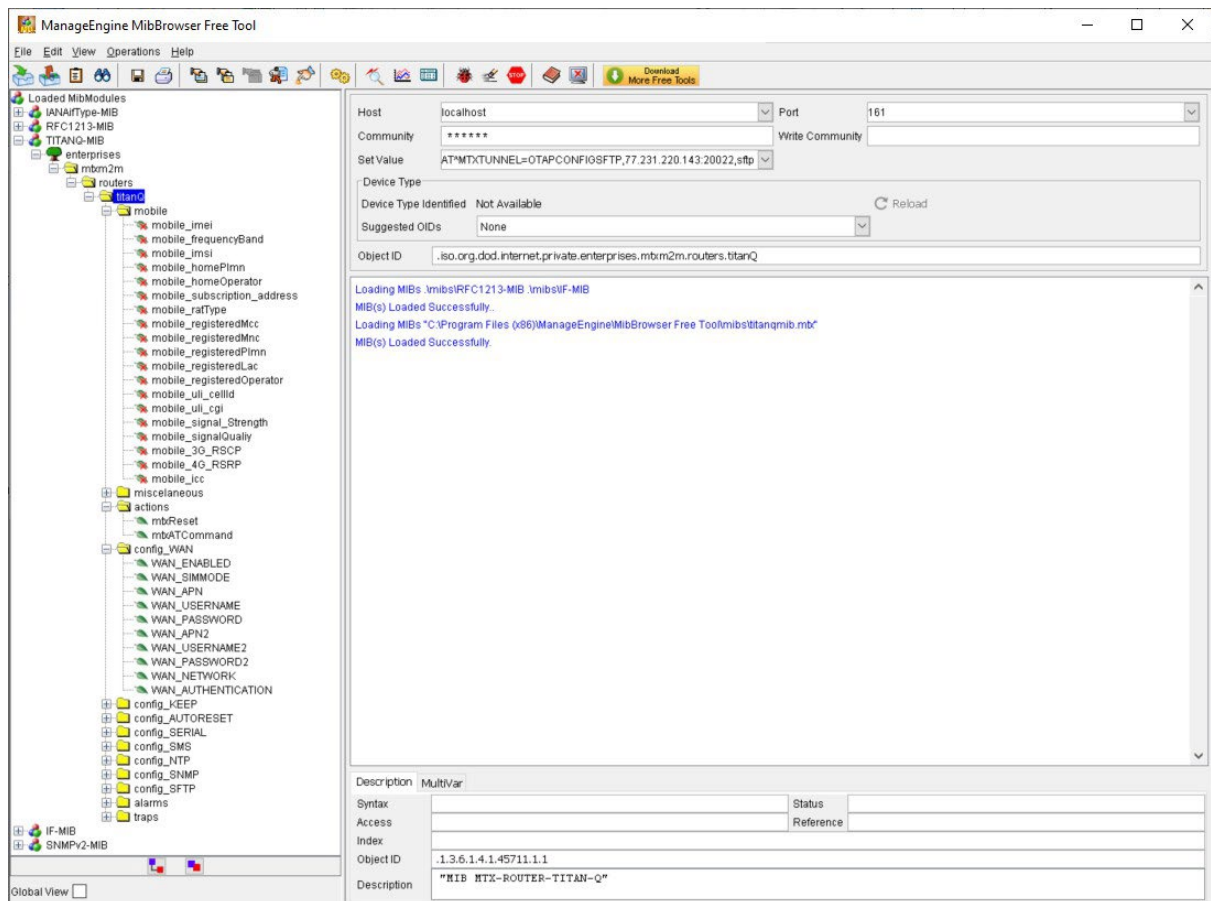
Other ▶ SNMP

Enabled:	☒	Enable SNMP v2c
SNMP Version:	SNMPv2c	SNMPv2 or SNMPv3
UDP Port:	161	Default UDP port 161
Custom OID:	.45711.1.1	Enterprise-Product OID. Default: .45711.1.1
Community:	public	Only SNMPv2. Password for GET and SET commands
Username:		Only SNMPv3.
Auth Password:		Only SNMPv3 (min 8 char)
Priv. Password:		Only SNMPv3 (min 8 char)
Auth Protocol:	MD5	Only SNMPv3.
Priv Protocol:	DES	Only SNMPv3.
Engine ID:	AUTO	Only SNMPv3. "AUTO" or custom HEX
Traps Enabled:	☒	Enable Traps
Traps - UDP Port:	162	Default UDP port 162
Traps - IP:	192.168.1.20	IP for sending traps
Traps - Community:	public	Only SNMPv2. Community for traps
Alarm OS:	☐	Enable trap for Oper. System alarm
Number traps alarm ON:	10	Number the traps sent when an alarm is activated. 0 ... 1440
Number traps alarm OFF:	5	Number the traps sent when an alarm is deactivated. 0 ... 1440
Trap period:	60	Period between traps (10...3600 sec)

Additional notes.

- Once you have finished configuring, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the device for the new changes to take effect.
- On the previous screen, there is a link with the text "Click here for download MIB". Download the file to obtain the MIB with the OIDs.
- Using SNMP, you can monitor the device, for example, by querying the device to obtain the RSSI (coverage), date and time, operating time, whether it is working in 4G/3G/2G mode, or the operator used.
- A very interesting option is the "Action" section. Here you will see that you can do two things. One is to reset the device remotely. The other is that you can execute an AT command remotely via SNMP (by writing to the log you will execute the AT command and by reading it you will get the response).
- By sending AT commands you can do anything via SNMP, from reading a Modbus device connected to the device, changing configuration parameters

, reading a GPS position, reading a digital input, changing the status of a digital output, etc.



4.7.9 Other – TACACS+

TITAN-based devices have the option of Tacacs+ authentication. This feature will enable external authentication for HTTP, Telnet, and SSH services on TITAN-based devices. To use this feature, you must configure this section.

- **Server:** IP address or DNS of the Tacacs+ server
- **KEY:** Encryption password.



The screenshot shows the webdyn TITAN configuration interface. At the top, the webdyn logo (a stylized 'w' with a blue and green wave) and the flexitron group logo are on the left, and the TITAN logo with the tagline "Makes your APPLICATION happen" is on the right. A left sidebar contains navigation links: Mobile (Status, Basic Settings, Keep Online), Ethernet (Basic Settings, DHCP Server), and Wifi (Basic Settings, DHCP Server). The main content area is titled "Other > TACACS+" and contains two input fields: "Server:" and "KEY:". To the right of the "Server:" field is the text "Remote server IP or DNS". To the right of the "KEY:" field is the text "Key for Tacacs+". Below the input fields is a "SAVE CONFIG" button.

Additional notes

- The TACACS+ service on Titan routers uses authentication and authorisation.
- For authorisation, the Titan router will search for the services "webdyn_http_admin", "webdyn_http_user", "webdyn_http_guest", "webdyn_telnet_admin", "webdyn_ssh_admin".
- Refer to application note ANV6_74, which explains in detail, with an example, how to configure the Titan router and its TACACS+ server.
- Once the configuration is complete, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the device for the new changes to take effect.

4.7.10 Other – MQTT

TITAN-based devices can work as MQTT clients by connecting to an MQTT/MQTT broker. Configuring the TITAN-based device as an MQTT client allows you to send the data collected in the internal data logger (data from sensors, Modbus devices, etc.) via MQTT. You must configure this section appropriately if you select MQTT transmission in the LOGGER section.

- *Enabled*: Select whether you want to enable the MQTT Client service.
- *Username*: MQTT username. Leave blank if not used.
- *Password*: MQTT password. Leave blank if not used.
- *ID*: Device identification field
- *QoS*: Quality of service (0,1,2)
- *KeepAlive*: Seconds for keepalive (300 recommended)
- *Persistence*: Select whether you want the data to be persistent
- *AT Topic*: Topic 1 to which the TITAN-based device will subscribe. The AT commands sent to this topic will be executed on the device.
- *AT Resp Topic*: Responses to AT commands received by MQTT by the TITAN-based device in the "AT Topic" topic will be sent to this Topic via MQTT.
- *AT Topic 2*: Topic 2 to which the TITAN-based device will subscribe. AT commands sent to this topic will be executed on the device.
- *AT Resp Topic 2*: Responses to AT commands received by MQTT by the TITAN-based device in the "AT Topic 2" topic will be sent to this Topic via MQTT.
- *AT Topic 3*: Topic 3 to which the TITAN-based device will subscribe. AT commands sent to this topic will be executed on the device.
- *AT Resp Topic 3*: Responses to AT commands received by MQTT by the TITAN-based device in the "AT Topic 3" topic will be sent to this Topic via MQTT.
- *MQTT Script Topic 1*: MQTT topic in which a customisable script will be executed upon receiving data. Refer to the scripts section of this manual and the application notes for more information.
- *MQTT Script Topic 2*: MQTT topic in which a customisable script will be executed upon receiving data. Refer to the scripts section of this manual and the application notes for more information.
- *Client Certificate*: If using MQTTS with client authentication, you will need to insert the client certificate .key file (PEM format) in this field.

- Serial Port2-485
- SSL Certificates

★ **External Devices**

- Logger configuration
- ModBus Devices
- Generic Serial Device
- Temperature Sensor
- IEC102 Meter
- W-MBus
- GPS Receiver

★ **Other**

- AT Command
- DynDns
- Private DynDns
- Sms control
- Periodic Autoreset
- Time Servers
- Remote Console
- Snmp
- Tacacs+
- Mqtt
- Http / Https
- User Permissions
- Passwords Web UI
- CA Certificates
- Email Config
- ModBus Slave
- Titan Scripts
- Connectivity tools
- Digital I/O
- Custom Skin
- Led Config
- Syslog
- Backup / Factory
- Firmware Upgrade
- Reboot
- Logout

► **Other** ► **MQTT Client**

Enabled: ☐ Enable MQTT client

Destination MQTT Broker. Examples:
tcp://test.mosquitto.org:1883
ssl://test.mosquitto.org:8883 (certificate needed)
ssl://test.mosquitto.org:8884 (certificates needed)

MQTT Broker:

MQTT Username: MQTT Username (blank if not used)

MQTT Password: MQTT Password (blank if not used)

MQTT ID: Device identification

MQTT Qos: MQTT Quality Of Service (0 ... 2)

MQTT Keepalive: Seconds for keepalive (30 ... 3600)

MQTT Persistence: ☐ Data persistence

MQTT AT Topic: This topic will be subscribed for receiving AT Commands (usefull for individual device)

MQTT AT Resp Topic: This topic will be used for publishing the AT Command Responses of AT Topic

MQTT AT Topic 2: This topic will be subscribed for receiving AT Commands (usefull for groups)

MQTT AT Resp Topic 2: This topic will be used for publishing the AT Command Responses of AT Topic 2

MQTT AT Topic 3: This topic will be subscribed for receiving AT Commands (usefull for all devices)

MQTT AT Resp Topic 3: This topic will be used for publishing the AT Command Responses of AT Topic 3

MQTT Script Topic 1: When data is received in this topic the 'Topic Script' will be executed.

MQTT Script Topic 2: When data is received in this topic the 'Topic Script' will be executed.

SAVE CONFIG

Additional notes.

- Once you have finished configuring, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the device for the new changes to take effect.
- Please note that it is possible to write the text [IMEI] instead of the IMEI number. In other words, if the IMEI is 012345678912345, it would be the same to use the TOPIC /0123456789012345/TEST as the topic
/[IMEI]/TEST
- Remember that if you use MQTTS, you may need to include the CA certificate used by the MQTT broker in the "Other→ CA Certificates" section.
- In previous versions of TITAN firmware, responses to AT commands in any topic were sent to a single topic. If you want the same behaviour, you can set the same value in the topics "AT Resp Topic", "AT Resp Topic 2", "AT Resp Topic 3".

4.7.11 Other – HTTP/HTTPS

HTTP configuration and HTTPS activation for the configuration environment.

- **HTTP Port:** indicates the TCP port for remote configuration via HTTP. For example, if you specify 8080, the configuration URL will be <http://x.x.x.x:8080>. The default standard port is 80, but if you wish to perform NAT to the TCP port 80 of an internal ETH device (IP camera, PLC), you will need to change it, for example, to 8080. The HTTP service cannot be used remotely if HTTPS is enabled.
- **HTTPS Enabled.** Enables the HTTPS service (certificates are automatically generated after restarting).
- **HTTP Port:** indicates the TCP port for HTTPS configuration. Default is 443.
- **HTTPS Key and Cert.** You can use your own certificate for the TITAN-based computer's HTTPS server. In this field, you can enter the CERTIFICATE and KEY in the same file (PEM format).



The screenshot displays the webdyn configuration interface, powered by TITAN. The sidebar on the left contains navigation links: Mobile (Status, Basic Settings, Keep Online), Ethernet (Basic Settings), Wifi (Basic Settings, DHCP Server), Firewall (NAT, Authorized IPs), Serial Settings (Serial Port1-RS232, Serial Port2-RS485, SSL Certificates), and External Devices. The main content area is titled 'Other > Http/Https'. It features the following settings:

- HTTP port:** A text input field containing '80'. A tooltip indicates: 'HTTP port for remote WAN connections. Not used if HTTPS is enabled.'
- HTTPS enabled:** A checkbox that is currently unchecked. A tooltip indicates: 'Enable HTTPS Web Server. First certificate is generated automatically.'
- HTTPS port:** A text input field containing '443'. A tooltip indicates: 'HTTPS port for remote WAN connections.'

Below these settings is a 'SAVE CONFIG' button. Further down, there is a section titled 'Custom KEY and Certificate for HTTPS WebServer (PEM format)'. It includes a label 'HTTPS Key and Certificate', a file selection button 'Seleccionar archivo', a status 'Ninguno archivo selec.', an 'Upload' button, and a red status indicator 'not uploaded'.

Additional notes.

- Once the configuration is complete, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the computer for the new changes to take effect.
- When you try to connect to the TITAN-based computer via HTTPS, your browser will probably display a warning message regarding the self-signed digital certificate. This is completely normal.

4.7.12 Other – User Permissions

From this section, you can configure the permissions that the "user" and "guest" users will have access to. The configuration options that you do not select will not appear in the menu on the left of the configuration environment when you log in to the device with the "user" or "guest" user.

★ Mobile 🔍 Status 🔍 Basic Settings 🔍 Keep Online ★ Ethernet 🔍 Basic Settings 🔍 DHCP Server ★ Wifi 🔍 Basic Settings 🔍 DHCP Server ★ Firewall 🔍 NAT 🔍 Authorized IPs 🔍 MAC Filter 🔍 Routes ★ Serial Settings 🔍 Serial Port1-RS232 🔍 Serial Port2-RS485 🔍 SSL Certificates ★ External Devices 🔍 Logger configuration 🔍 ModBus Devices 🔍 Generic Serial Device 🔍 Temperature Sensor 🔍 IEC102 Meter 🔍 GPS Receiver ★ VPN 🔍 IPSec 🔍 OpenVPN Client 🔍 OpenVPN Server 🔍 ZeroTier ★ Other 🔍 AT Command 🔍 DynDns 🔍 Private DynDns 🔍 Sms control 🔍 Periodic Autoreset 🔍 Time Servers 🔍 Remote Console 🔍 Snmp 🔍 Tacacs+ 🔍 Ldap 🔍 Mqtt 🔍 Http / Https 🔍 User Permissions 🔍 Passwords Web UI	Other ▶ User Permissions <table><tr><td>Mobile</td><td>Basic Settings</td><td>☐</td></tr><tr><td></td><td>Keep Online</td><td>☐</td></tr><tr><td>Ethernet</td><td>Basic Settings</td><td>☐</td></tr><tr><td></td><td>DHCP Server</td><td>☐</td></tr><tr><td>Wifi</td><td>Basic Settings</td><td>☐</td></tr><tr><td></td><td>DHCP Server</td><td>☐</td></tr><tr><td>Firewall</td><td>NAT</td><td>☐</td></tr><tr><td></td><td>Authorized IPs</td><td>☐</td></tr><tr><td></td><td>MAC Filter</td><td>☐</td></tr><tr><td></td><td>Routes</td><td>☐</td></tr><tr><td>Serial Settings</td><td>Serial Port 1</td><td>☐</td></tr><tr><td></td><td>Serial Port 2</td><td>☐</td></tr><tr><td></td><td>SSL Certificates</td><td>☐</td></tr><tr><td>External devices</td><td>Logger config.</td><td>☐</td></tr><tr><td></td><td>Modbus devices</td><td>☐</td></tr><tr><td></td><td>Generic serial</td><td>☐</td></tr><tr><td></td><td>Temperature sen.</td><td>☐</td></tr><tr><td></td><td>IEC102 meter</td><td>☐</td></tr><tr><td></td><td>GPS Receiver</td><td>☐</td></tr><tr><td>Vpn</td><td>IPSec</td><td>☐</td></tr><tr><td></td><td>OpenVPN Client</td><td>☐</td></tr><tr><td></td><td>OpenVPN Server</td><td>☐</td></tr><tr><td></td><td>ZeroTier</td><td>☐</td></tr><tr><td></td><td>All plugins</td><td>☐</td></tr></table>	Mobile	Basic Settings	☐		Keep Online	☐	Ethernet	Basic Settings	☐		DHCP Server	☐	Wifi	Basic Settings	☐		DHCP Server	☐	Firewall	NAT	☐		Authorized IPs	☐		MAC Filter	☐		Routes	☐	Serial Settings	Serial Port 1	☐		Serial Port 2	☐		SSL Certificates	☐	External devices	Logger config.	☐		Modbus devices	☐		Generic serial	☐		Temperature sen.	☐		IEC102 meter	☐		GPS Receiver	☐	Vpn	IPSec	☐		OpenVPN Client	☐		OpenVPN Server	☐		ZeroTier	☐		All plugins	☐
Mobile	Basic Settings	☐																																																																							
	Keep Online	☐																																																																							
Ethernet	Basic Settings	☐																																																																							
	DHCP Server	☐																																																																							
Wifi	Basic Settings	☐																																																																							
	DHCP Server	☐																																																																							
Firewall	NAT	☐																																																																							
	Authorized IPs	☐																																																																							
	MAC Filter	☐																																																																							
	Routes	☐																																																																							
Serial Settings	Serial Port 1	☐																																																																							
	Serial Port 2	☐																																																																							
	SSL Certificates	☐																																																																							
External devices	Logger config.	☐																																																																							
	Modbus devices	☐																																																																							
	Generic serial	☐																																																																							
	Temperature sen.	☐																																																																							
	IEC102 meter	☐																																																																							
	GPS Receiver	☐																																																																							
Vpn	IPSec	☐																																																																							
	OpenVPN Client	☐																																																																							
	OpenVPN Server	☐																																																																							
	ZeroTier	☐																																																																							
	All plugins	☐																																																																							

- CA Certificates
- Email Config
- ModBus Slave
- Titan Scripts**
- Connectivity tools
- Digital I/O
- Custom Skin
- Led Config
- Syslog
- Backup / Factory
- Firmware Upgrade
- Reboot
- Logout

Other

DynDNS	☐
Private DynDNS	☐
SMS Control	☐
Periodic autoreset	☐
Time Servers	☐
Remote Console	☐
Snmp	☐
Tacacs+	☐
LDAP	☐
Mqtt	☐
Http / Https	☐
CA-Certificates	☐
Email config	☐
Modbus Slave	☐
Connectivity tools	☐
Digital I/O	☐
Custom Skin	☐
Led Config	☐
Syslog	☐
Backup / Factory	☐
Firmware Upgrade	☐
Reboot	☐

SAVE CONFIG

4.7.13 Other – Passwords

There are three user names for accessing the device settings. The "admin" user gives you access to all device settings, and the "user" user gives you access to the settings you select (very useful if you customise the device with your logos). From this section, you can change the password for both users. There is also a "guest" user that allows you to access the same menus as the "user" user but without being able to make configuration changes.

The screenshot displays the 'webdyn' web interface, powered by TITAN and Flexitron group. The main content area is titled 'Other > Password Web UI'. It contains two sections: 'Administrator' and 'General User'. Each section has input fields for 'Username', 'Password', and 'Re enter Password', with corresponding help text. Below the input fields are buttons labeled 'SAVE ADMIN PASS' and 'SAVE USER PASS'. A sidebar on the left lists various configuration categories: Mobile (Status, Basic Settings, Keep Online), Ethernet (Basic Settings), Wifi (Basic Settings, DHCP Server), Firewall (NAT, Authorized IPs), Serial Settings (Serial Port1-RS232, Serial Port2-RS485, SSL Certificates), and External Devices (Logger configuration, ModBus Devices, Generic Serial Device, Temperature Sensor, IEC102 Meter, W-MBus, GPS Receiver).

Additional notes.

- Once you have finished configuring, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the device for the new changes to take effect.

4.7.14 Other – CA Certificates

All services that use secure connections under SSL/TLS against HTTPS or MQTTS servers need to have the CA Root certificate used by that server in order to verify the server's certificate. From this section, you can enter up to 2 custom CA Root servers (in PEM format).

It is also possible to select the "Allow all certificates" option. This option is not recommended unless you have a secure connection (IPSec, OpenVPN, SIM with private APN, etc.) as the certificate of a remote server will not be checked.

webdyn powered by **TITAN**
flexitron group

Mobile
Status
Basic Settings
Keep Online

Ethernet
Basic Settings

Wifi
Basic Settings
DHCP Server

Firewall
NAT
Authorized IPs

Serial Settings
Serial Port1-RS232
Serial Port2-RS485
SSL Certificates

External Devices
Logger configuration
ModBus Devices
Generic Serial Device
Temperature Sensor
IEC102 Meter
W-MBus
GPS Receiver

Other > CA-Root Certificates

Custom CA-Root Certificates (PEM format)

User CA-Root-1
Seleccionar archivo Ninguno archivo selec. Upload **uploaded**
Certificate fingerprint (SHA1):
06:90:7E:A4:EC:A0:F7:70:28:12:DC:51:1F:ED:62:A7:99:AC:71:9A

User CA-Root-2
Seleccionar archivo Ninguno archivo selec. Upload **not uploaded**

DELETE User CA-Root-1 DELETE User CA-Root-2

CA-Root Certificates > Options

Allow all certificates ☐

Unsecure option. This option is not recommended if your device is not running over a secure connection (like IPSec, OpenVPN, SIM card with private APN, ...)

SAVE CONFIG

Additional notes.

- TITAN-based equipment has an internal list of the most common CA Root certificates. If you need to use a certificate that is not available, you can enter it here.

4.7.15 Other – Email Configuration

TITAN-based devices allow notifications to be sent by email. They also allow emails to be sent via AT commands (for example, if you have a PLC and want to send an SMS or email message, you can do so by sending an AT command to the TITAN-based device).

The TITAN-based device must be configured in advance to be able to send emails. In other words, you must configure the SMTP server to be used in this section.

- **Enabled:** Enables the email service.
- **SMTP Server:** Specifies the SMTP server address (e.g. or)IP or DNS of the SMTP server (e.g. smtp.gmail.com, smtpout.secureserver.net, etc.).
- **SMTP Port:** TCP port of the SMTP server
- **Authentication:** select whether the SMTP server requires authentication (usually yes)
- **TLS:** select whether the SMTP server requires TLS
- **User:** username for the source email account
- **Password:** Password for the source email account
- **Email address:** Source email address from which emails will be sent

The screenshot displays the 'webdyn' configuration interface, powered by TITAN. The left sidebar lists various settings categories: Mobile (Status, Basic Settings, Keep Online), Ethernet (Basic Settings), Wifi (Basic Settings, DHCP Server), Firewall (NAT, Authorized IPs), and Serial Settings (Serial Port1-RS232, Serial Port2-RS485, SSL Certificates). The main content area is titled 'Other > Email configuration' and contains the following fields:

Field	Value	Description
Enabled:	☒	Enable Email
Smtp Server:	smtp.gmail.com	Ex: smtp.gmail.com, smtpout.se, cureserver.net , ...
Smtp Port:	25	TCP port for SMTP Server (25, 465, 587, ...)
Authentication:	☒	Check if authentication is required
TLS:	☒	Check if TLS is required
User:	Webdyn	User of email account
Password:	*****	Password of email account
Email address:	webdyn@matrix.es	Email address for sending emails (Ex: myemail@gmail.com)

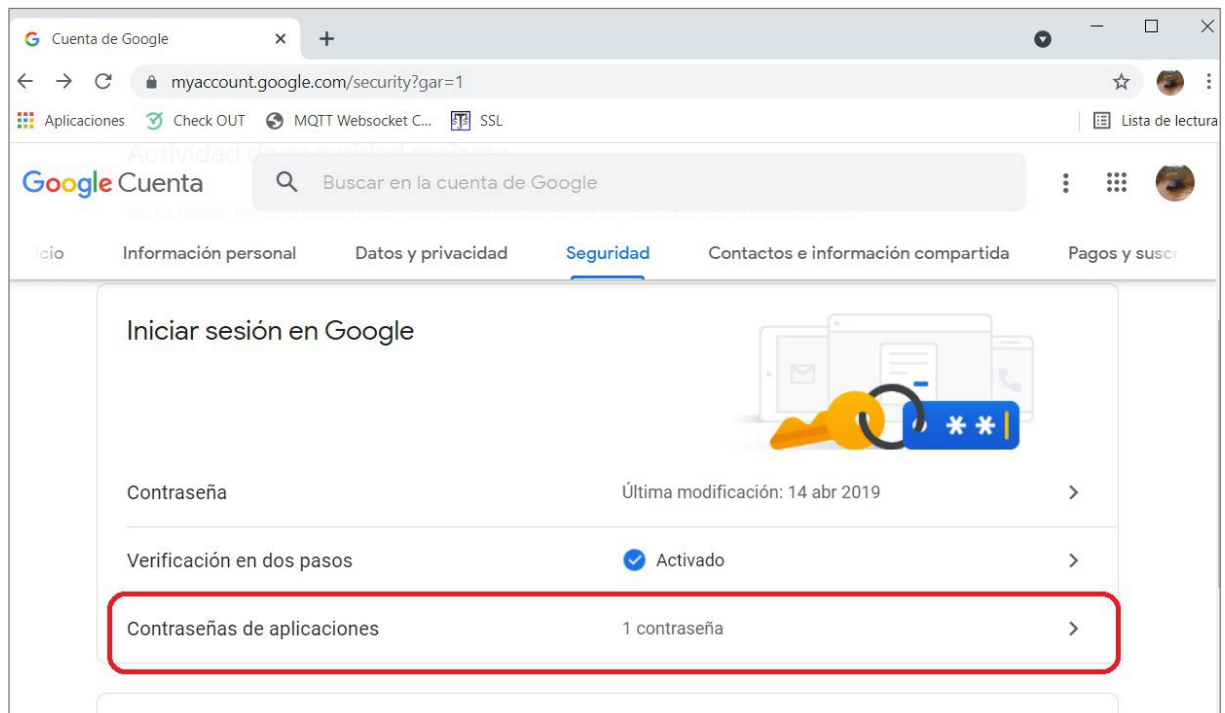
A 'SAVE CONFIG' button is located at the bottom of the configuration area.

Additional notes.

- Once you have finished configuring the settings, click on the "SAVE CONFIG" button to save the changes.

Remember that you must restart your computer for the new changes to take effect.

- Please note that if you use the GMAIL SMTP server, you will need to create an "application password". To do this, go to your "Google Account Manager", select the "Security" menu and add an "Application Password".



4.7.16 Other – Modbus Slave

TITAN-based devices can be configured to act as a Modbus TCP Slave and/or Modbus RTU Slave device. For example, using the Modbus protocol, you can remotely switch (via 4G/3G/2G, Ethernet or Wi-Fi) the device's digital outputs, check the status of the digital inputs, as well as send SMS messages, receive SMS messages or even send emails via Modbus.

Below is the memory table with the Modbus addresses of the TITAN-based device registers. The supported Modbus commands are: 0x03 for reading and 0x10 for writing.

Register ID	R/W	Possible values	Description
1	R	0 ... 32635	Firmware version
2	R	0 ... 32635	Firmware subversion
6	R	0...31	GSM signal RSSI
7	R	2,3,4	Technology used (2=2g,3=3g,4=4g)
8	R	1.....31	Day
9	R	1.....12	Month
10	R	2000 2099	Year
11	R	0 ... 23	Hour
12	R	0 ... 59	Minutes
13	R	0 ... 59	Seconds
20	W	1	Performs a reset
30	RW	0.1	GPIO 0 (if exists)
31	RW	0.1	GPIO 1 (if present)

32	RW	0.1	GPIO 2 (if present)
33	RW	0.1	GPIO 3 (if present)
34	RW	0.1	GPIO 4 (if present)
98	W	0 ... 32635	Length of AT command to be executed
99	R	0 ... 32635	Length of the AT command response
100...354	W	ASCII	AT command text (in ASCII)
500...754	R	ASCII	AT command response text (in ASCII)
1000	R/W	0.1	1 indicates a new SMS has been received. After reading the SMS, a PLC must write a 0.
1001	R	0 ... 18	Length of the telephone that sent the SMS
1002...1019	R	ASCII	Telephone number that sent the SMS
1020	R	0 ... 160	Length of the SMS message text
1021–1180	R	ASCII	SMS message text
1021...1180	R	ASCII	SMS message text
10000	R/W	0 ... 65535	Register of 10000
10001...19998	RW	ASCII	User registration from 10001 to 19998

19999	RW	0 ... 65535	User register 19999
-------	----	-------------	---------------------

AT commands via Modbus protocol

Another very interesting feature of TITAN-based devices is that they allow AT commands to be executed via Modbus protocol. For example, if you have a PLC that acts as a Modbus TCP Master, you can send a command to the device via Modbus TCP protocol to execute an AT command, allowing you to obtain data such as coverage, send an SMS, obtain the time, reset the device, etc.

The following describes how to execute an AT command via Modbus

Procedure for executing an AT command via Modbus

The procedure for executing an AT command via Modbus is very simple. The best way to understand it is through an example. Let's imagine that we want to execute the AT+CSQ command to obtain coverage

1.- We write the AT command, in ASCII, starting from register 100.

Register ID	Value	Description
100	65	ASCII of the letter: A
101	84	ASCII code for the letter: T
102	43	ASCII of the character: +
103	67	ASCII of the letter: C
104	83	ASCII code for the letter: S
105	81	ASCII code for the letter: Q

2.- The AT+CSQ command has 6 characters, so to execute the AT command, we write a 6 in register 98. After writing to register 98, the AT command will be executed immediately.

Register ID	Value	Description
98	6	Size of the command to be executed

3.- We check the result of executing the AT command by reading register 99. The value read will be the length of the response. A value of 0 indicates that there is still no response (the execution, which usually takes less than 1 second, has not finished). A value >0 indicates the length of the response.

Register ID	Value	Description
99	28	Command response size

4.- Finally, we read the 28 records containing the response starting from record 500.

Record ID	Value	Description
500	65	A
501	84	T
502	43	+
503	67	C
504	83	S
505	81	Q
506	13	/r
507	13	/r
508	10	/n
509	43	+
510	67	C
511	83	S
512	81	Q
513	58	:

514	32	[space]
515	49	1
516	54	6
517	44	,
518	57	9
519	57	9
520	13	/r
521	10	/n
522	13	/r
523	10	/n
524	79	O
525	75	K
526	13	/r
527	10	/n

In the case of this AT command, the relevant registers are 515 and 516, which indicate a coverage of a value of "16".

P.S. Remember that if you want to use this feature to send an SMS, you must use the AT command indicated in this guide: AT^MTXTUNNEL=SMS,telephone,message

Procedure for receiving an SMS

The procedure for receiving SMS messages via Modbus TCP is very simple. Your PLC must periodically check register 1000. If the value read is a "1", it means that a new SMS message has arrived. You must then read it using the relevant registers 1001-1180. Once read, you must write a "0" in register 1000.

The screenshot shows the webdyn configuration interface, powered by TITAN and part of the flexitron group. The left sidebar contains a menu with categories: Mobile (Status, Basic Settings, Keep Online), Ethernet (Basic Settings), Wifi (Basic Settings, DHCP Server), Firewall (NAT, Authorized IPs), Serial Settings (Serial Port1-RS232, Serial Port2-RS485, SSL Certificates), and External Devices (Logger configuration, ModBus Devices, Generic Serial Device, Temperature Sensor, IEC102 Meter, W-MBus, GPS Receiver). The main content area is titled 'Other > ModBus TCP Slave' and includes settings for enabling the router as a Modbus TCP Slave (checked), setting the ModBus TCP Port to 502, and a 'SAVE CONFIG' button. Below this is the 'Other > ModBus RTU Slave' section, which is currently disabled. It includes fields for ModBus RTU address (1) and ModBus COM Port (Serial Port 2). At the bottom, there is a 'GENERIC' table with the following data:

@Modbus Register	Register name	R / W	Comments
1	Version	R	Firmware version
2	Subversion	R	Firmware subversion

Additional notes.

- Once you have finished configuring, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the device for the new changes to take effect.
- Refer to the Modbus log table in the configuration menus of the TITAN-based device itself, in the Other > Modbus Slave section.

4.7.17 Other – TITAN Scripts

TITAN-based devices allow you to run a user-programmable script in JAVASCRIPT (ECMAScript 5). <https://www.w3schools.com/jsref/default.asp> From this script, you can access different features of the TITAN-based device to make small customisations (such as reading devices through the serial port, reading digital inputs, changing a digital output, reading a Modbus device, sending an SMS message, sending an email, an MQTT message to a specific topic, send an SNMP TRAP, save and read files on the device, etc. (See the various examples of TITAN-based devices for a better understanding).

The TITAN Scripts configuration screen looks like this:

Other > Titan Scripts v2

Example 1.- Hello World Load Example

```
mtx.println("EXAMPLE 1 - Running...");  
  
//EXAMPLE: HOW TO SHOW DEBUG INFO  
  
var i;  
for (i=0;i<10;i++)  
{  
    //***** PRINT "Hello World" BY CONSOLE  
    mtx.println("Hello World: " + i);  
  
    //***** PAUSE 1 SECOND  
    mtx.pause(1000);  
}  
  
mtx.println("EXAMPLE 1 - End");
```

Run Script Stop Script Save Script Delete Script Encrypt Script

Other > Titan Scripts v2 > Console

Run console Stop console Clear console

Other > Titan Scripts v2 > Options

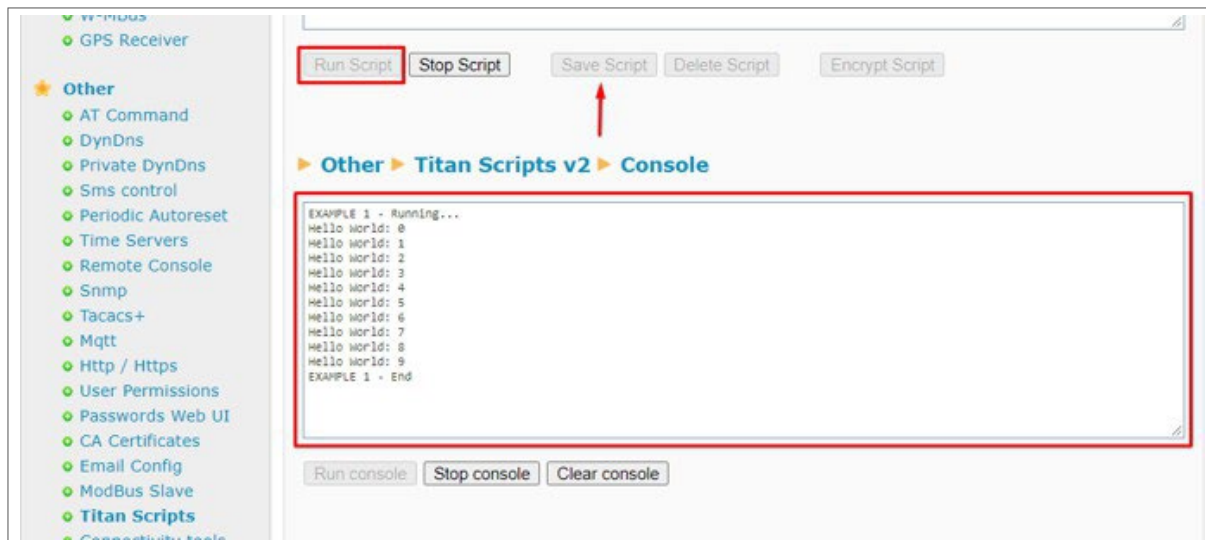
Autostart: ☐ Autostart script after Titan power up

Period: Once after Power Up Start script period. (Normally 'Once after power up')

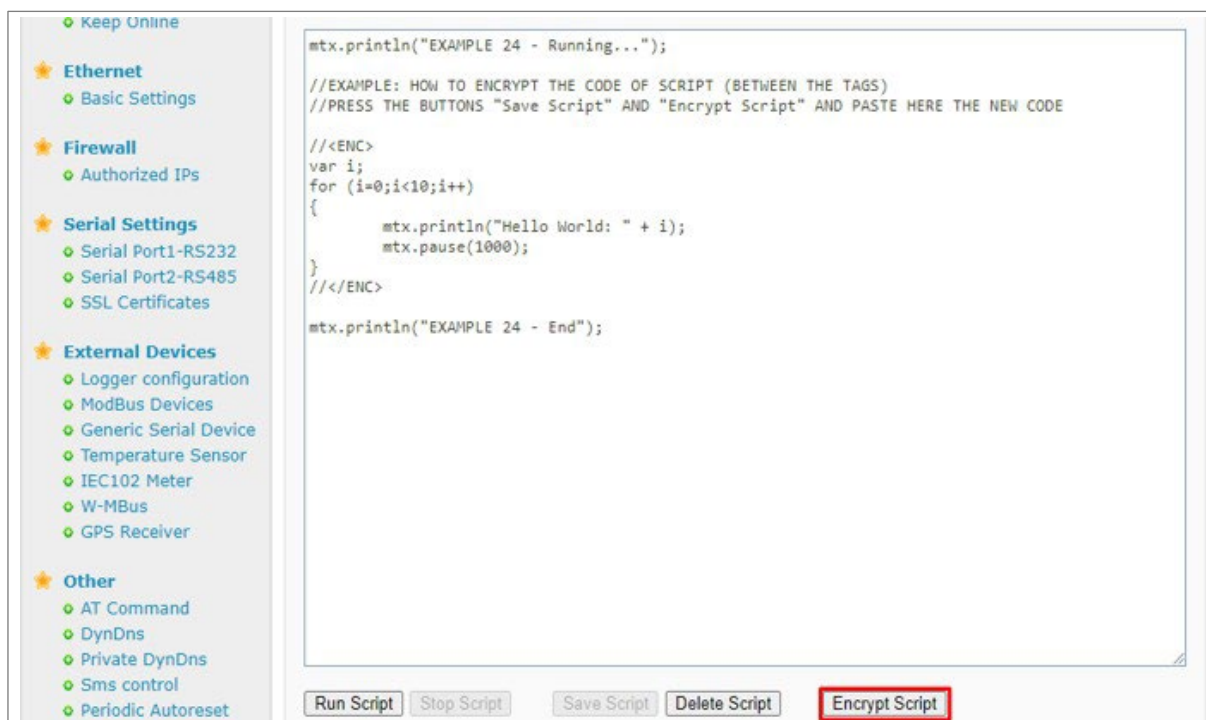
SAVE CONFIG

The top drop-down menu contains multiple examples with all the features you can use. To load an example, select the desired example from the drop-down menu (we recommend the "Hello World" example the first time) and then click the "Load Example" button. After doing so, the source code for the example will appear on the screen.

Then press the "Save Script" button to save the example code in the execution memory of the TITAN-based device. After that, you can press the "Run Script" button to execute the script, and in a few seconds the traces of the "mtx.println()" programme will appear in the console section.



If you wish to encrypt the script code, click on the "Encrypt Script" button and a pop-up window will appear with a new code.



After pasting this new code into the script, the TITAN-based computer will be able to execute it in the same way.

The screenshot displays the TITAN Scripts v2 web interface. On the left is a sidebar menu with categories: Ethernet (Basic Settings), Firewall (Authorized IPs), Serial Settings (Serial Port1-RS232, Serial Port2-RS485, SSL Certificates), External Devices (Logger configuration, ModBus Devices, Generic Serial Device, Temperature Sensor, IEC102 Meter, W-MBus, GPS Receiver), and Other (AT Command, DynDns, Private DynDns, Sms control, Periodic Autoreset, Time Servers, Remote Console, Snmp, Tacacs+, Mqtt, Http / Https, User Permissions, Passwords Web UI, CA Certificates, Email Config, ModBus Slave, Titan Scripts, Connectivity tools). The main area shows a script editor for 'EXAMPLE 24'. The code contains a comment about encryption and a long base64-encoded string. Below the editor are buttons: Run Script, Stop Script, Save Script, Delete Script, and Encrypt Script. At the bottom, there's a 'Console' section showing the output of the script: 'EXAMPLE 24 - Running...' followed by 'Hello World: 0' through 'Hello World: 9', and 'EXAMPLE 24 - End'.

For more information about script encryption, see examples 24 and 25.

TITAN Script configuration options.

Once you have developed and are running your script, you will want it to run autonomously every time the TITAN-based computer restarts. To do this, you will need to tick the "Autostart" box. Once this box is checked, you must specify the autostart method. The usual method is "Once after Power Up", meaning that the script will run once after the TITAN-based computer starts up. This is optimal for most applications of the type:

```
while (true) {  
  
    ...  
  
}
```

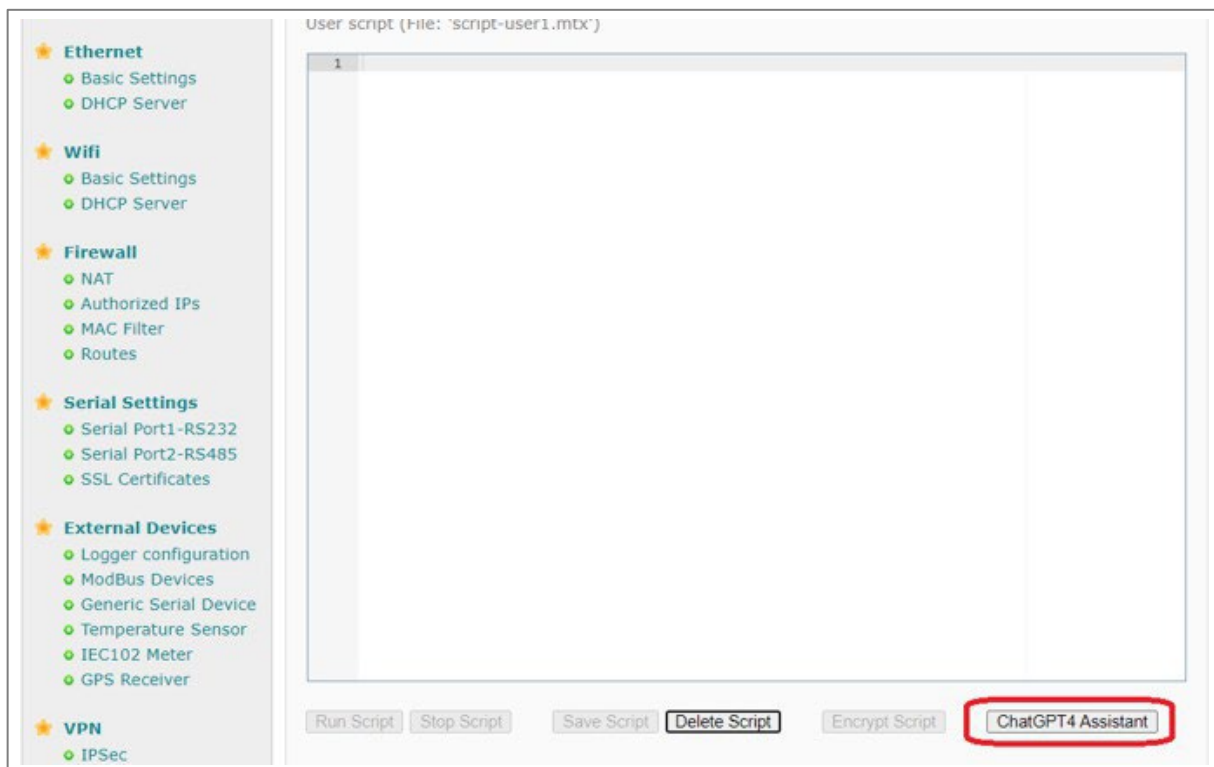
in which the code is executed cyclically. However, you may find it more convenient for the script to run automatically at certain intervals (every 1 minute, 2 minutes, 5 minutes, etc.) to perform certain tasks. For example, imagine a case where you need a script to read a Modbus RTU device every 15 minutes and send the reading to an MQTT server. In this case, you may find it more convenient to run the script every 15 minutes.

The "mtx" object in TITANScripts.

As indicated in the preceding paragraphs, TITANScripts allow you to program small scripts in JavaScript (ECMAScript 5) to interact with certain features specific to TITAN-based equipment. These features are accessed through certain functions of an "mtx" object. Each of these functions is described below. We strongly recommend studying the small examples included in the "Other→ TITAN Scripts" section for a better understanding.

Please note that TITAN Scripts are designed to perform minor customisations in the behaviour of TITAN-based equipment and are not designed to program any type of application as could be done in other languages such as Java, C++, Python, etc. Basically, everything you can do is contained in the examples provided.

We also recommend using "Titan Script Assistant". Starting with FW version 5.3.6.30, you will find a new button labelled "chatGPT assistant". This button will take you to a version of chatGPT trained with the customised functions of the Titan router and can be a useful aid in script development and programming troubleshooting, especially if you do not have strong programming skills. To use this assistant, you must have a chatGPT account for the use of chatGPT4 or higher. Please note that the use of artificial intelligence for programming may contain errors and that the code must be reviewed.



ChatGPT - Titan Script Assistant - Google Chrome

chat.openai.com/g-jtkS8VFEa-titan-script-assistant

 Titan Script Assistant



Titan Script Assistant

Advanced Technical Assistant in Script Programming for Titan Routers. You can: 1) Use the assistant to program your scripts 2) Use the assistant to fix errors in your scripts. Notice: ChatGPT can make mistakes. Please review the generated code.

By JOSE G. G.

Create a script that continuously checks if the value of register ...

Crea un script que compruebe continuamente si el registro 10 ...

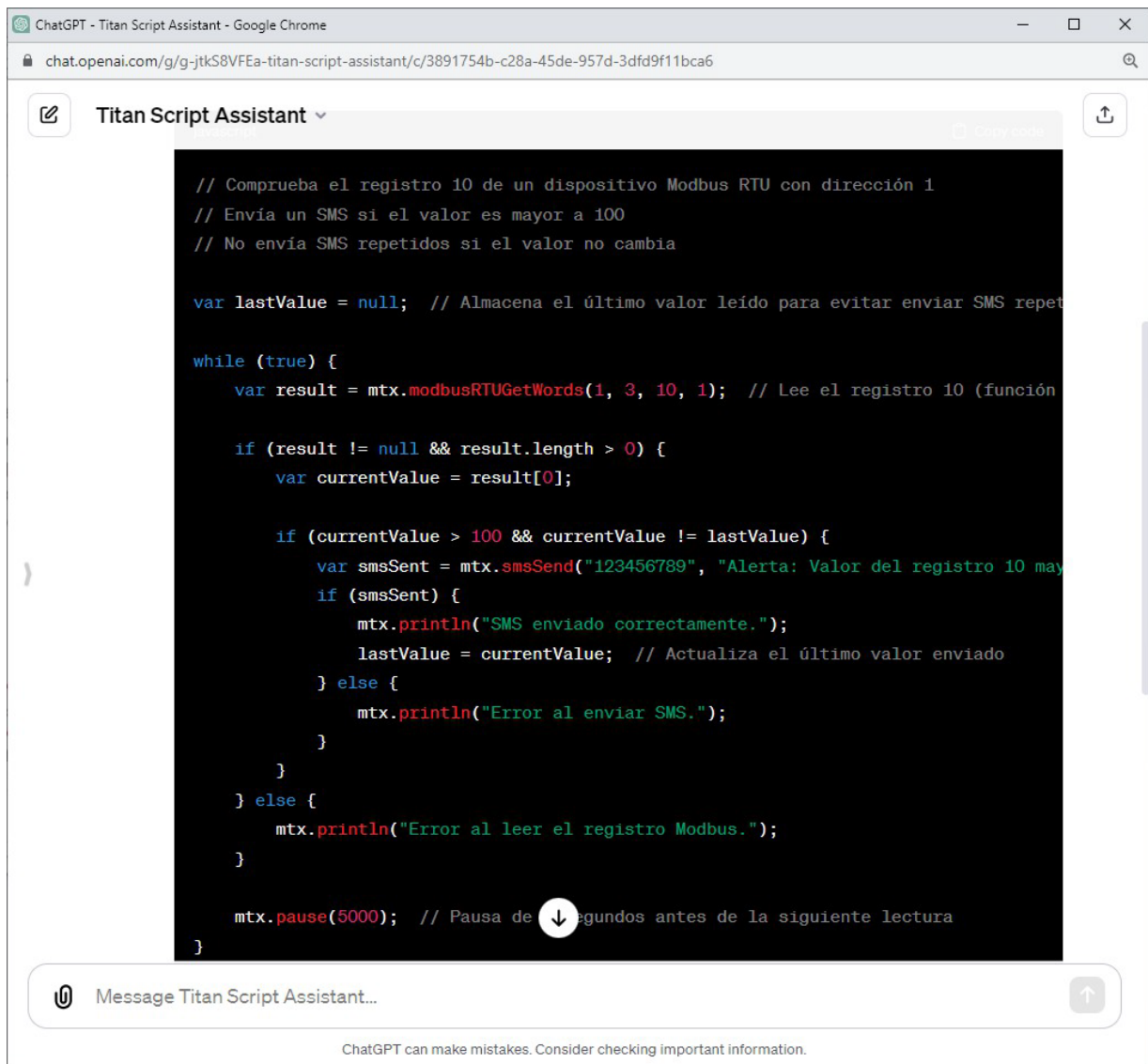
Créez un script qui vérifie continuellement si la valeur du registr...

创建一个脚本，持续检查Modbus RTU设备地址1的第10寄存器...

 Message Titan Script Assistant...



ChatGPT can make mistakes. Consider checking important information.



Refer to application note ANV6_76 for more information on "Titan Script Assistant".

Functions related to IOx DIGITAL INPUTS/OUTPUTS:

In the "Notes" for each function, you will find a list of examples of scripts available for use with that function. These examples are stored within the router itself via the "Other→ TITAN Scripts" menu.

- Function: `mtz.ioGet(idIO)`
 Description: allows you to obtain the status of a digital input/output.
 Parameters: `idIO`: int 0,1,2,3,4, ... (IO identifier)
 Response: int (0=input not activated, 1=input activated, -1=error)
 Notes: see examples: 3, 15, 16
- Function: `mtz.ioSet(idIO, value)`
 Description: allows you to change the status of a digital output.

Parameters: idIO: (int) 0,1,2,3,4, ... (IO identifier) value:
 (int) 0,1 (logical value of the digital output)
 Response: boolean (true=operation OK, false=operation ERROR)
 Notes: see examples: 16

- Function: `mtx.counterGet(idIO)`
 Description: allows you to obtain the value of the pulse counter associated with an input.
 Parameters: idIO: (int) 0,1,2,3,4, ... (digital input identifier)
 Response: long counter value
 Notes: see examples: 34
- Function: `mtx.counterSet(idIO, value)`
 Description: allows you to change the initial value of a pulse counter.
 Parameters: idIO: (int) 0,1,2,3,4, ... (IO identifier) value:
 (long) (counter value)
 Response: boolean (true=operation OK, false=operation ERROR)
 Notes: see examples: 34

Functions related to the router LEDs:

- Function: `mtx.ledSet(idLed, value)`
 Description: allows you to turn an LED on/off
 Parameters: idLed: (int) 0,1,2, ... (device LED identifier)
 value: (int) 0=off, 1=on, 2=slow flashing, 3=fast flashing.
 Response: boolean (true=operation OK, false=operation ERROR)
 Notes: see examples: 21

Functions related to AT COMMANDS:

- Function: `mtx.atSend(command, timeout)`
 Description: allows an AT command to be executed on the TITAN-based device itself.
 Parameters: command: (String) AT command to execute
 timeout: (int) timeout milliseconds
 Response: String: response to the AT command. "" =
 Error Notes: see examples: 2

MQTT-related functions:

- Function: `mtx.mqttIsConnected()`
 Description: Returns whether the TITAN-based device is connected to an MQTT broker.
 Parameters: None.
 Response: Boolean (true=Yes, false=No)
 Notes: see examples: 4, 35
- Function: `mtx.mqttSend(message, topic, qos)`
 Description: allows you to send an MQTT text message to a specific MQTT topic with a specific QoS
 Parameters: message: (String) Text message to be sent
 topic: (String) MQTT topic to which the message is to be sent
 qos: (int) QoS of the delivery (0,1,2)
 Response: boolean: (true=operation OK, false=operation ERROR)
 Notes: see examples: 4, 8, 35
- Function: `mtx.mqttSendArray(byte array, topic, qos)`
 Description: allows you to send an MQTT message from a byte array to a specific MQTT topic with a specific QoS
 Parameters: message: (byte array) Text message to send. topic: (String) MQTT topic to send the message to
 qos: (int) QoS of the message (0,1,2)
 Response: boolean: (true=operation OK, false=operation ERROR)
 Notes: see examples: 35
- Function: `mtx.mqttSubscribe(topic)`
 Description: TITAN scripts allow subscription to 1 MQTT topic in order to receive communications from a broker. This function allows you to subscribe to that topic.
 Parameters: topic: (String) MQTT topic where TITAN scripts will receive messages
 Answer: boolean: (true=operation OK, false=operation ERROR)
 Notes: see examples: 35
- Function: `mtx.mqttGetArray()`
 Description: function that allows you to check and receive whether data has arrived at the topic configured using the previous function `mtx.mqttSubscribe(topic)`
 Parameters: None
 Response: array bytes: (null=no data, !null=array of bytes with data)
 Notes: see examples: 35

Functions related to SNMP TRAPS:

- Function: `mtx.trapSend(oid, message, severity)`
 Description: sends an SNMP trap
 Parameters: oid: (String) OID of the SNMP trap
 message: (String) Text associated with the SNMP trap

severity (int) Trap severity (1, ... ,7)
 Response: boolean: (true=operation OK, false=operation ERROR)
 Notes: see examples: 5, 9

SMS-related functions:

- Function: `mtx.smsSend(phoneNumber, textMessage)`
 Description: sends an SMS message to a telephone number
 Parameters: `phoneNumber`: (String) Sending phone number
 `textMessage`: (String) Text of the SMS message
 Response: boolean: (true=operation OK, false=operation ERROR)
 Notes: see examples: 3, 10, 17, 18, 40
- Function: `mtx.smsRead()`
 Description: Checks whether a new SMS message has arrived.
 Parameters: None
 Response: String array: (null=no new SMS, data[0]=phone number
 data[1]=SMS message)
 Notes: see examples: 40

HTTP-related functions:

- Function: `mtx.httpRequest(url, headers, method, data, dataType)`
 Description: Sends an email to an email address. Parameters:
 `url`: (String) Connection URL
 `headers`: (HashMap) Custom HTTP headers
 `method`: (String) GET or POST
 `data`: (String) Data to be sent with the HTTP request
 `dataType`: (String) type of data to be sent with the HTTP request
 Notes: see examples: 39

Functions related to EMAILS:

- Function: `mtx.emailSend(address, subject)`
 Description: sends an email to an email address
 Parameters: `address`: (String) Destination email address
 `subject`: (String) Subject of the email sent
 Response: boolean: (true=operation OK, false=operation ERROR)
 Notes: see examples: 6, 11

Functions related to TELEGRAM:

- Function: `mtx.telegramSendMessage(apiToken, chatID,message)`
Description: sends a text message to a Telegram chat group
Parameters:
 `apiToken:` (String) Telegram bot API token
 `chatID:` (String) Chat identifier where to send the message
 `message:` (String) Text message to send
Response: `boolean:` (true=operation OK, false=operation ERROR)
Notes: see examples: 41, 42

Functions related to the SIM CARD:

- Function: `mtx.simID()`
Description: Returns the SIM card currently in use.
Parameters: None
Response: `int:` (-1=ERROR, 0=no SIM, 1=SIM1, 2=SIM2)
Notes: see examples: 18
- Function: `mtx.simInserted()`
Description: Returns whether the current SIM card is inserted.
Parameters: None
Response: `int:` (-1=ERROR, 0=not inserted, 1=inserted)
Notes: see examples: ---
- Function: `mtx.simStatus()`
Description: Returns the status of the current SIM card.
Parameters: None
Response: `int:` (-1=ERROR, 0=SIM ERROR, 1=SIM OK)
Notes: see examples: 18

Functions related to SERIAL PORTS:

- Function: `mtx.serialOpen(idPort,baudrate,databits,parity,stopbits,flowcontrol)`
Description: opens a serial port
Parameters:
 `idPort:` (int) serial port ID (1,2)
 `baudrate:` (int) speed (460800, 230400, ... , 1200, 600, 300)
 `databits:` (int) number of bits per data (8,7)
 `parity:` (int) 0=none, 1=odd, 2=even
 `stopbits:` (int) 1,2
 `flow control:` (int) 0=no, 1=hardware (if available)
Response: `boolean:` true: OK, false: ERROR
Notes: see examples: 19
- Function: `mtx.serialClose(idPort)`
Description: Closes a serial port
Parameters: `idPort:` (int) serial port ID (1,2)
Response: `boolean:` true: OK, false: ERROR

Notes: see examples: —

- Function: `mtx.serialRead(idPort)`
Description: Reads and returns data read by an RS232 or RS485 serial port.
Parameters: `idPort`: (int) serial port ID (1,2)
Response: `byte []`: array of bytes with the data read
 `null`: no data read.
Notes: see examples: 19
- Function: `mtx.serialWrite(idPort,data,start,length)`
Description: Sends an array of bytes through the serial port.
Parameters: `idPort`: (int) serial port ID (1,2)
 `data`: (`byte[]`) array of bytes for e
 `start`: (int) start index for sending the byte array length:
 (int) number of bytes to send.
Response: `boolean`: (true=operation OK, false=operation ERROR)
Notes: see examples: 19

MODBUS-related functions:

- Function: `mtx.modbusRTUGetWords(address, command, start, numRegisters)`
Description: Reads and returns the Modbus registers of a Modbus RTU device.
Parameters: `address`: (int) Modbus RTU address (1 ... 254)
 `command`: (int) Modbus command 3 or 4
 `start`: (int) initial register address to read
 `numRegisters`: (int) number of registers to read (1 ... 64)
Answer: `int []`: array of WORDS with the data read (values 0 ... 65535) `null`:
 Reading ERROR.
Notes: see examples: 7, 8, 9, 10, 11, 12, 22, 30
- Function: `mtx.modbusRTUSetWords(address, command, start, values[])`
Description: writes Modbus registers to a Modbus RTU device.
Parameters: `address`: (int) Modbus RTU address (1 ... 254)
 `command`: (int) Modbus command 16 or 6
 `start`: (int) initial register address to write to
 `values[]`: (`int[]`) array with the WORDS to be written
Response: `boolean`: (true=operation OK, false=operation ERROR)
Notes: see examples: 12
- Function: `mtx.modbusRTUGetBits(address, command, start, numBits)`
Description: Reads and returns the Modbus bit type registers of a Modbus RTU device.
Parameters: `address`: (int) Modbus RTU address (1 ... 254)
 `command`: (int) Modbus command 1 or 2

- start: (int) initial bit register address to read (1 ... 65535)
 numBits: (int) number of bits to read (1 ... 64)
 Response: int []: array of BITS with the bits read (values 0,1)
 null: Reading ERROR
 Notes: see examples: 13
- Function: mtz.modbusRTUSetBit(address, command, start, value)
 Description: writes 1 Modbus bit-type register to a Modbus RTU device.
 Parameters: address: (int) Modbus RTU address (1 ... 254)
 command: (int) Modbus command (5)
 start: (int) initial bit register address to write to
 values: (int) value to be written (0,1)
 Response: boolean: (true=operation OK, false=operation ERROR)
 Notes: see examples: 14
 - Function: mtz.modbusTCPGetWords(ip, port, command, start, numRegisters)
 Description: reads and returns the Modbus registers of a Modbus TCP device
 Parameters: ip: (String) IP address of the Modbus TCP device
 port: (int) TCP port of the Modbus TCP device
 command: (int) Modbus command 3 or 4
 start: (int) initial register address to read
 numRegisters: (int) number of registers to read (1 ... 64)
 Response: int []: array of WORDS with the data read (values 0 ... 65535) null:
 Read ERROR.
 Notes: see examples: —
 - Function: mtz.modbusTCPSetWords(ip, port, command, start, values[])
 Description: writes Modbus registers to a Modbus TCP device
 Parameters: ip: (String) IP address of the Modbus TCP device
 port: (int) TCP port of the Modbus TCP device
 command: (int) Modbus command 16 or 6
 start: (int) initial register address to write to
 values[]: (int[]) array with the WORDS to be written
 Response: boolean: (true=operation OK, false=operation ERROR)
 Notes: see examples: 22
 - Function: mtz.modbusTCPGetBits(ip, port, command, start, numBits)
 Description: Reads and returns the Modbus bit type registers of a Modbus TCP device.
 Parameters: ip: (String) IP address of the Modbus TCP device
 port: (int) TCP port of the device Modbus TCP
 command: (int) Modbus command 1 or 2
 start: (int) initial bit register address to read (1 ... 65535)
 numBits: (int) number of bits to read (1 ... 64)
 Response: int []: array of BITS with the bits read (values 0,1)
 null: Reading ERROR.
 Notes: see examples: —

- Function: `mtx.modbusTCPSetBit(ip, port, command, start, value)`
 Description: writes 1 Modbus bit-type register to a Modbus TCP device
 Parameters: `ip`: (String) IP address of the Modbus TCP device
 `port`: (int) TCP port of the Modbus TCP device
 `command`: (int) Modbus command (5)
 `start`: (int) initial bit register address to write values:
 (int) value to be written (0,1)
 Response: `boolean`: (true=operation OK, false=operation ERROR)
 Notes: see examples: ---
- Function: `mtx.modbusSaveWord(idRegister, registerValue)`
 Description: writes a word type value to the user memory area. Parameters:
 `idRegister`: (int) register address (10000 ... 19999)
 `registerValue`: (int) register value (0 ... 65535)
 Response: `boolean`: (true=operation OK, false=operation ERROR)
 Notes: see examples: ---
- Function: `mtx.modbusSaveDWord(idRegister, registerValue)`
 Description: writes a double word value to the user memory area Parameters:
 `idRegister`: (int) register address (10000 ... 19999)
 `registerValue`: (long) register value (0 ... 4294967295)
 Response: `boolean`: (true=operation OK, false=operation ERROR)
 Notes: see examples: 49
- Function: `mtx.modbusReadWord(idRegister)`
 Description: Reads a word type value from the user memory area.
 Parameters: `idRegister`: (int) register address (10000 ... 19999)
 Response: `int`: value of the register read (0 ... 65535)
 -1: error
 Notes: see examples: —
- Function: `mtx.modbusReadDWord(idRegister)`
 Description: Reads a double word value from the user memory area. Parameters:
 `idRegister`: (int) register address (10000 ... 19999) Response:
 `long`: value of the register read (0 ... 4294967295)
 -1: error
 Notes: see examples: —

Functions related to the MODBUS EXPERT section:

- Function: `mtx.modbusExpertGetStatus(deviceName)`
 Description: Returns whether the status of a device in the "Modbus Expert" section is correct or not.

incorrect (whether it is reading the Modbus logs correctly or not).

Parameters: deviceName: (String) Device name
Response: boolean: true: correct status
false: error

Notes: see examples: --- (Application note 78)

- Function: `mtx.modbusExpertGetReg(deviceName, registerAddress)`
Description: returns the current value of the Modbus register with address `registerAddress` of the device named `deviceName` from the devices in the "Modbus Expert" section
Parameters: deviceName: (String) Name of the device
registerAddress (int) Address of the Modbus register
Response: String: Value of the register read
null: error
Notes: see examples: --- (Application note 78)
- Function: `mtx.modbusExpertSetReg(deviceName, registerAddress, value)`
Description: sets the current value of the Modbus register with address `registerAddress` of the device named `deviceName` from the devices in the "Modbus Expert" section
Parameters: deviceName: (String) Name of the device
registerAddress (int) Address of the Modbus register
value (String) Value of the register to be set
Response: Boolean: true: operation executed correctly
false: operation executed with error
Notes: see examples: --- (Application note 78)
- Function: `mtx.modbusExpertGetV(deviceName, registerIndex)`
Description: Function identical to the "modbusExpertGetReg" function, but using the "registerIndex" parameter (the V index of the Modbus Expert section register) instead of the "registerAddress" parameter.
Parameters: deviceName: (String) Device name
registerIndex (int) Index of the device's Modbus register in the Modbus Expert section
Response: String: Value of the register read
null: error
Notes: see examples: ---
- Function: `mtx.modbusExpertSetV(deviceName, registerIndex, value)`
Description: Function identical to the "modbusExpertSetReg" function, but using the "registerIndex" parameter (the V index of the Modbus Expert section register) instead of the "registerAddress" parameter.
Parameters: deviceName: (String) Device name
registerIndex (int) Modbus register address
value (String) Value of the register to be set
Response: Boolean: true: operation executed correctly
false: operation executed with error
Notes: see examples: --- (Application note 78)

Functions related to IEC-102 counters:

- Function: `mtx.iec102AddMeter(idMeter,linkAddress,measAddress,password)`
 Description: Adds a meter to the meter manager. Maximum 100.
 Parameters: `idMeter` (String) meter identifier `linkAddress` (int) link address of the meter
 `measAddress` (int) meter measurement point address `password` (long) counter password

 Response: `boolean` `true`: counter added successfully. `false`: error
 Notes: see examples: 32, 47, 48, 49

- Function: `mtx.iec102DeleteAllMeters()`
 Description: Deletes all meters from the meter manager. Parameters:
 Response: `boolean` `true`: meters successfully deleted.
 `false`: error
 Notes: see examples: 32, 47, 48, 49

- Function: `mtx.iec102GetAllMeters()`
 Description: Returns a list of all counters stored in the counter manager.
 Parameters:
 Response: `ArrayList []` List with all counter identifiers. Notes: see examples: 48, 49

- Function: `mtx.iec102GetPresence()`
 Description: Checks the presence of a meter. Parameters:
 `idMeter` (String) meter identifier Response:
 `int` -2: not yet initialised
 -1: bus busy with reading 0:
 meter detected
 >0: error
 Notes: see examples: 47

- Function: `mtx.iec102SetInstantValues(idMeter)`
 Description: Executes the command to read the instantaneous values of the meter. Parameters: `idMeter` (String) meter identifier
 Response: `boolean`: (`true`=operation OK, `false`=operation ERROR)
 Notes: see examples: 32, 48, 69

- Function: `mtx.iec102GetInstantValues(idMeter)`
 Description: Reads the instantaneous values of the meter

Parameters: idMeter (String) meter identifier Response:
String JSON with the data read from the meter
"" ERROR
Notes: see examples: 32, 48, 69

- Function: mtX.iec102SetFiscalClose(idMeter, hourIni, minuteIni, dayIni, monthIni, yearIni, hourEnd, minuteEnd, dayEnd, monthEnd, yearEnd)

Description: executes the command to read fiscal closes.

Parameters: idMeter (String) meter identifier
hourIni (int) start hour (0 ... 23)
minuteIni (int) initial minute (0 ... 59)
dayIni (int) initial day (1 ... 31)
monthIni (int) initial month (1 ... 12)
yearIni (int) start year (22 ... 99)
hourEnd (int) end hour (0 ... 23)
minuteEnd (int) end minute (0 ... 59)
dayEnd (int) end day (1 ... 31)
monthEnd (int) end month (1 ... 12)
yearEnd (int) end year (22 ... 99)

Response: boolean: (true=operation OK, false=operation ERROR)

Notes: see examples: ---

- Function: mtX.iec102GetFiscalClose(idMeter)

Description: Reads the meter closing values

Parameters: idMeter (String) meter identifier Response:
String JSON with the data read from the meter
"" ERROR

Notes: see examples: —

- Function: mtX.iec102SetIntegratedTotal(idMeter, hourIni, minuteIni, dayIni, monthIni, yearIni, hourEnd, minuteEnd, dayEnd, monthEnd, yearEnd, mode, period)

Description: Executes the command to read the integrated totals (load curve).

Parameters: idMeter (String) meter identifier
hourIni (int) initial hour (0 ... 23)
minuteIni (int) initial minute (0 ... 59)
dayIni (int) initial day (1 ... 31)
monthIni (int) starting month (1 ... 12)
yearIni (int) starting year (22 ... 99)
hourEnd (int) end hour (0 ... 23)
minuteEnd (int) end minute (0 ... 59)
dayEnd (int) end day (1 ... 31)
monthEnd (int) end month (1 ... 12)
yearEnd (int) end year (22 ... 99)
mode (int) 0 = absolute, 1 = incremental
period (int) 0 = hourly, 1 = quarter-hourly

Response: boolean: (true=operation OK, false=operation ERROR)
Notes: see examples: 52

- Function: `mtx.iec102GetIntegratedTotal(idMeter)`
Description: Reads the integrated total values (load curve) from the meter.
Parameters: `idMeter` (String) meter identifier
Response: String JSON with the data read from the meter
"" ERROR
Notes: see examples: 52

GPS-related functions:

- Function: `mtx.gpsGetData()`
Description: Returns the current GPS position of the TITAN-based device.
Parameters: None
Response: String JSON with GPS data
"" ERROR (if no NMEA frames are detected)
Notes: see examples: 33

Functions related to the LOGGER:

- Function: `mtx.loggerWrite(data)`
Description: writes data to the internal logger of the TITAN-based device
Parameters: `data`: (String) data to be stored in the logger
Response: boolean true = operation successful
false = error (e.g., data length greater than the logger record size).
Notes: see examples: 26

Functions related to FILES:

- Function: `mtx.fileWrite(index, data, append)`
Description: writes data to a file on the TITAN-based computer
Parameters: `index`: (int) file index (0 ... 9)
`data`: (String) data to be written to the file
`append`: (boolean) true=adds the text to the existing file
false=deletes the file and writes the
text Response: boolean true = operation successful
false = error (for example, file data length greater than 100KB).
Notes: see examples: 27, 28, 30

- Function: `mtx.fileRead(index)`
 Description: Reads data from a file on the TITAN-based computer.
 Parameters: index: (int) file index (0 ... 9) Response: String data contained in the file Notes: see examples: 27, 29
- Function: `mtx.fileSize(index)`
 Description: returns the data length of a file Parameters: index: (int) file index (0 ... 9)
 Response: long file size (in bytes) Notes: see examples: 28, 30
- Function: `mtx.fileFtpDownload(tls, index, server, port, username, pass, remoteFile, timeout, addExtension)`
 Description: Receives a data file from TITAN from a remote server via FTP.
 Parameters: tls: (int) 0=FTP, 1=FTPS
 index: (int) index of the file (0 ... 9) where to store the data
 server: (String) IP or DNS of the remote FTP server
 port: (int) 0 ... 65535
 username: (String) FTP server username pass: (String) FTP server password
 remoteFile: (String) Remote file to be downloaded timeout: (int) Timeout in seconds to complete download
 addExtension: (String) If the file is downloaded, this extension is added
 Response: boolean true = operation successful
 false = error
 Notes: see examples: 29
- Function: `mtx.fileFtpUpload(tls, index, server, port, username, pass, remoteFile, timeout)`
 Description: Sends a TITAN data file to a remote server via FTP. Parameters: tls: (int) 0=FTP, 1=FTPS
 index: (int) file index (0 ... 9)
 server: (String) IP or DNS of the remote FTP server
 port: (int) 0 ... 65535
 username: (String) FTP server username pass: (String) FTP server password
 remoteFile: (String) Remote file to be downloaded timeout: (int) Timeout in seconds to complete the transfer
 Response: boolean true = operation successful
 false = error
 Notes: see examples: 28, 30
- Function: `mtx.fileFtpRename(tls, server, port, username, pass, currentFileName, newFileName, timeout)`
 Description: sends a TITAN data file to a remote server via FTP. Parameters: tls: (int) 0=FTP, 1=FTPS

server: (String) IP or DNS of the remote FTP server
 port: (int) 0 ... 65535
 username: (String) FTP server username
 pass: (String) FTP server password
 currentFileName (String) Current remote file name
 newFileName (String) New remote file name
 timeout: (int) Timeout in seconds to complete the operation
 Response: boolean true = operation successful
 false = error
 Notes: see examples: —

- Function: `mtx.scriptFtpDownload(tls, server, port, username, pass, remoteFile, timeout, addExtension)`
 Description: Downloads the script file from an FTP server.
 Parameters:
 - tls: (int) 0=FTP, 1=FTPS
 - server: (String) IP or DNS of the remote FTP server
 - port: (int) 0 ... 65535
 - username: (String) Username of the FTP server
 - pass: (String) FTP server password
 - remoteFile: (String) Remote script to be downloaded
 - timeout: (int) Timeout in seconds to complete download
 - addExtension: (String) If the file is downloaded, this extension is added
 Response: boolean true = operation successful
 false = error
 Notes: see examples: 31

MODEM-related functions:

- Function: `mtx.imeiCheck(imei, message)`
 Description: Checks whether the IMEI passed as a parameter is correct.
 Parameters:
 - imei: (String) IMEI to be checked
 - message: (String) message to return if incorrect
 Response: void
 Notes: see examples: 25
- Function: `mtx.imeiGet()`
 Description: Returns the IMEI of the internal modem of the TITAN-based device.
 Parameters: None
 Response: String The IMEI of the internal modem of the TITAN-based device.
 Notes: See examples: 30

OTAP-related functions:

- Function:** `mtx.otapFTPConfig(tls, server, port, user, pass, remoteConfigFile, secTimeout, addExtension);`

Description: Executes a remote configuration download command via FTP/FTPS. If the configuration is downloaded correctly, it is possible to rename the remote file (so that it is not downloaded again) and install the configuration.

Parameters:

 - `tls:` (int) 0=no TLS, 1=yes TLS
 - `server:` (String) FTP server
 - `port:` (int) FTP port
 - `user:` (String) FTP username
 - `pass:` (String) FTP password
 - `remoteConfigFile:` (String) remote file (e.g. path1/path2/file)
 - `sectimeout:` (int) timeout seconds
 - `addExtension:` (String) If downloaded, this extension is added to the remote file (so that it is not downloaded again)

Response: boolean true=process started, false=process not started

Notes: see examples: 37

- Function:** `mtx.otapFTPLastStatus()`

Description: Returns the current status of an OTAP process.

Parameters:

Response: int 0=none, 1=running, 2=OK, 3=ERROR

Notes: see examples: 37

Functions related to WATCHDOG:

- Function:** `mtx.watchdogSet(mode, timeout)`

Description: initialises the watchdog

Parameters:

 - `int mode:` (int) 0=mode reset
 - `timeout:` (int) Seconds to reset if no refresh (0, 30 ... 100000). 0=Disabled.

Response: boolean true=OK, false: ERROR

Notes: see examples: 20

- Function:** `mtx.watchdogKick()`

Description: refreshes the watchdog so that it does not restart the modem. **Parameters:** None

Response: boolean true=OK, false: ERROR

Notes: see examples: 20

Time-related functions:

- Function: mtz.pause(mseconds)
Description: Creates a pause of the configured milliseconds.
Parameters: mseconds (long) milliseconds to wait
Response: boolean true = successful operation
false = error
Notes: see examples: 1, 2, 3, 4, 5, ... , 35
- Function: mtz.timeStamp(format)
Description: returns the date and time of the device
Parameters: format: (String) "utc" = UTC time (YYYY-MM-DDTHH:NN:SSZ)
"std" = LOCAL time (DD/MM/YYYY HH:NN:SS) "iso8601"
= LOCAL time (YYYY-MM-DDTHH:NN:SS)
Response: String date and time of the device
Notes: see examples: 28, 30
- Function: mtz.timeGetYear()
Description: returns the year
Parameters: None
Response: int Current year with four digits
Notes: see examples: ---
- Function: mtz.timeGetMonth()
Description: returns the month
Parameters: None
Answer: int current month (1=January ...
12=December)
Notes: see examples: ---
- Function: mtz.timeGetDay()
Description: returns the day
Parameters: None
Response: int current day (1 ...
31)
Notes: see examples: ---
- Function: mtz.timeGetHour()
Description: returns the time
Parameters: None
Response: int current hour (0 ... 23)
Notes: see examples: ---
- Function: mtz.timeGetMinute()
Description: returns the minutes
Parameters: None
Response: int current minute (0 ... 59)
Notes: see examples: ---
- Function: mtz.timeGetSecond()
Description: returns the seconds
Parameters: None

Answer: int current second (0 ... 59)
Notes: see examples: ---

Functions related to Sockets:

- Function: `mtx. socketOpen (idSocket,ssl,ip,port)`
Description: Opens a TCP client socket against an IP/DNS
Parameters: `idSocket` (int) socket identifier (0 ... 9)
`ssl` (int) 0=no, 1=yes
`idSocket` (String) IP or DNS address of the remote server TCP
`port` (int) TCP port of the remote server (0 ... 65535)
Response: `boolean` true = operation successful
false = error
Notes: see examples: 45, 46
- Function: `mtx. socketClose (idSocket)`
Description: Closes a TCP client socket.
Parameters: `idSocket` (int) socket identifier (0 ... 9)
Response: `boolean` true = operation successful
false = error
Notes: see examples: 45, 46
- Function: `mtx. socketIsOpen(idSocket)`
Description: Checks whether a TCP client socket has an established connection.
Parameters: `idSocket` (int) Socket identifier (0 ... 9)
Response: `boolean` true = yes, a connection has been established
false = no, there is no connection established.
Notes: see examples: 45, 46
- Function: `mtx. socketSend(idSocket, data[])`
Description: sends data through a TCP(IP) socket
Parameters: `idSocket` (int) socket identifier (0 ... 9)
`values[]`: (byte[]) array with the bytes to be sent
Response: `boolean`: (true=operation OK, false=operation ERROR)
Notes: see examples: 45, 46
- Function: `mtx. socketIsData(idSocket)`
Description: Checks whether there is data pending to be read in a TCP Client socket.
Parameters: `idSocket` (int) socket identifier (0 ... 9)
Response: `boolean` true = there is pending data
false = no pending data
Notes: see examples: 45, 46

- **Function:** `mtx. sockeRead(idSocket)`
Description: Reads the bytes available on a TCP client socket.
Parameters: `idSocket` (int) socket identifier (0 ... 9)
Response: `byte[]` Array of bytes with the received data
null: error
Notes: see examples: 45, 46

Data conversion functions:

- **Function:** `mtx.convert2WordToUnsignedInt32 (word1, word2)`
Description: Converts 2 words to 32-bit unsigned integer format.
Parameters: word1 (int) 0 ... 65535
word2 (int) 0 ... 65535
Response: unsigned int320 ... 4294967295
Notes: see examples: 14c
- **Function:** `mtx.convert2WordToSignedInt32 (word1, word2)`
Description: Converts 2 words to 32-bit signed integer format.
Parameters: word1 (int) 0 ... 65535
word2 (int) 0 ... 65535
Response: signed int32 -2147483648 ... 2147483647
Notes: see examples: 14c
- **Function:** `mtx.convert2WordToFloat (word1, word2)`
Description: converts 2 words to float format
Parameters: word1 (int) 0 ... 65535
word2 (int) 0 ... 65535
Response: Float -3.4028235x10^38 ... 3.4028235x10^38
Notes: see examples: 14c
- **Function:** `mtx.convert4WordToSignedInt64 (word1, word2, word3, word4)`
Description: converts 4 words to 64-bit signed integer format
Parameters: word1 (int) 0 ... 65535
word2 (int) 0 ... 65535
word3 (int) 0 ... 65535
word4 (int) 0 ... 65535
Answer: String -9223372036854775808 ... 9223372036854775807
Notes: see examples: 14c
The value is returned in String format because JavaScript does not support 64-bit integers. Useful for forming JSONs to send to platforms.
- **Function:** `mtx.convert4WordToDouble (word1, word2, word3, word4)`
Description: Converts 4 words to 64-bit Double format
Parameters: word1 (int) 0 ... 65535
word2 (int) 0 ... 65535
word3 (int) 0 ... 65535

	word4	(int)	0 ... 65535
Answer:	Double		-4.9x10 ³²⁴ ... 1.7976931348623157x10 ³⁰⁸
Notes:	see examples: 14c		

Functions for communicating with the chatGPT API

- Function:** `mtx.chatGPTInit(apiURL,apiKey)`

Description: Initialises the communication settings with the chatGPT API.

Parameters: `apiURL (String)` chatGPT API URL
`apiKey (String)` API KEY for the chatGPT API

Response: `boolean` `true` = operation successful
 `false` = error

Notes: see examples: 57, 58, 59

- Function:** `mtx.chatGPTGetTextResponse(apiModel,numTokens,prompt)`

Description: performs a query to chatGPT

Parameters: `apiModel (String)` AI model to be used from chatGPT
`numTokens (integer)` Maximum number of tokens to use
`prompt (String)` Query text for chatGPT

Response: `String` ChatGPT response to the question.
 "ERROR" = error

Notes: See examples: 57, 58, 59

- Function:** `mtx.chatGPTGetImageResponse(apiModel,numTokens,prompt,localImage, imageURL, urlUsername, urlPassword)`

Description: Queries chatGPT about an image from a URL. That URL can be A local image taken from a webcam that returns a URL with the captured image or an image from the internet. The difference is that in the local image, the image has to be uploaded to ChatGPT.

Parameters: `apiModel (String)` AI model to be used from chatGPT
`numTokens (int)` Maximum number of tokens to use
`prompt (String)` Query text for chatGPT about the image
`localImage (boolean)` "true" if the image is local and must be uploaded to chatGPT or "false" if the image is on the Internet and accessible by chatGPT.
`imageURL (String)`: URL to the image
`urlUsername (String)`: if necessary, the username to access the image with basic HTTP authentication.
`urlPassword (String)`: if necessary, the password to access the image with basic HTTP authentication.

Response: String ChatGPT response to the question.
"ERROR" = error
Notes: see examples: 57, 58, 59

MISCELLANEOUS functions:

- Function: `mtx.hashSHA256(text)`
Description: Returns the SHA256 hash of a text.
Parameters: `text (String)` Text for which the hash is to be calculated
Response: String SHA256 hash of the specified text.
Notes: see examples: ---
- Function: `mtx.mobileGetIP()`
Description: Returns the IP address associated with Mobile WAN.
Parameters: None
Response: String "" = None, "x.x.x.x"
Notes: see examples: —
- Function: `mtx.powerStatus()`
Description: Returns whether the router has external power. Useful for devices with internal batteries or supercapacitors.
Parameters: None
Response: int 0=no, 1=yes, -1=error
Notes: see examples: 6
- Function: `mtx.println(data)`
Description: Sends text data to the debug console
Parameters: `data (String)` text data to be written to the console
Response: void
Notes: see examples: 1, 2, 3, 4, 5, ... , 35
- Function: `mtx.ping(ip, timeout)`
Description: Generates a ping to a remote IP or DNS
Parameters: `ip (String)` Remote IP or DNS
`timeout (int)` timeout in milliseconds
Response: boolean true = ping successful
false = ping error
Notes: see examples: 17
- Function: `mtx.base64Encode(data)`
Description: Encodes a byte array in base64.
Parameters: `data (byte[])` array of bytes to be encoded
Response: String data encoded in base64

"" = error

Notes: see examples: 26

- Function: `mtx.stringToByteArray(data)`
Description: converts a string into a byte array
Parameters: `data` (String) data to be converted to a byte array
Response: `byte` array of bytes
 `null` = error

Notes: see examples: 51

- Function: `mtx.byteArrayToString (data, init, length)`
Description: converts a byte array to a String
Parameters: `data` (byte[]) byte array
 `init` (int) position
 `length` (int) number of bytes to convert
Response: `String` String with the conversion
 `""` = error

Notes: see examples: —

- Function: `mtx.configParamGet(paramName)`
Description: Reads the value of a Titan configuration parameter.
Parameters: `paramName` (String) Name of the configuration parameter
Response: `String` Value of the configuration parameter.

Notes: see examples: —

- Function: `mtx.configParamSet(paramName,paramValue)`
Description: Writes the value of a Titan configuration parameter
Parameters: `paramName` (String) Name of the configuration parameter
 `paramValue` (String) Value of the configuration parameter
Response: `boolean` `true` = operation successful
 `false` = error

Notes: see examples: —

List of examples.

In the router's web configuration environment, in the "Other→ TITAN Scripts" section, you will find a series of script examples. Below is a brief description of each one.

Example 1.- Hello World

Basic example of how to write debug traces to standard output.

Example 2.- Request AT Command

Example showing how to send AT commands to the device's internal modem (to obtain coverage, identification of the mobile phone cell used, etc.)

Example 3.- Send SMS when GPIO change

Example of how to send an SMS message when a change is detected in a digital input on the device.

Example 4.- Send MQTT message to broker

Example of how to send an MQTT message to the broker to which the device is connected.

Example 5.- Send SNMP TRAP

Example of how to send an SNMP TRAP.

Example 6.- Send EMAIL when power fails

Example of how to send an email. In this example, the email is sent when a power failure is detected. Example only valid for router models with an internal battery or supercap.

Example 7.- Read Modbus RTU word register

Example of how to read a Modbus RTU WORD type register from TITAN scripts.

Example 7b.- Read Modbus TCP word register

Example of how to read a Modbus TCP WORD register from TITAN scripts.

Example 8.- Read Modbus RTU word register and send by MQTT

Example showing how to read a Modbus WORD type register and send the read data to an MQTT topic.

Example 9.- Read Modbus RTU word register and send TRAP

Example showing how to read a Modbus WORD type register and send an SNMP TRAP based on the value of the register read.

Example 10.- Read Modbus RTU word register and send SMS

Example showing how to read a Modbus WORD type register and send an SMS based on the value of the register read.

Example 11.- Read Modbus RTU word register and send EMAIL

Example showing how to read a Modbus WORD register and send an EMAIL based on the value of the register read.

Example 12.- Write Modbus RTU word registers

Example showing how to write to a Modbus WORD register. Example 13.- Read

Modbus RTU bit register

Example showing how to read a Modbus bit register.

Example 14.- Write Modbus RTU bit register

Example showing how to write to a Modbus bit register.

Example 14b.- Read Modbus RTU word registers, save data in LOGGER for scheduled sending

Example showing how to read Modbus RTU registers from an external device via RS485 serial port, create a JSON with a custom data structure, and store that JSON in the Titan datalogger to be sent using the configured method (MQTT, HTTP, FTP).

Example 14c.- Convert Modbus word registers into Unsigned Int32, Signed Int32, Float, Int64

Example showing how to read Modbus RTU records from an external device via RS485 serial port as WORD and transform them into 32-bit unsigned integer, 32-bit signed integer, float, and 64-bit signed integer types.

Example 15.- Read all Digital Inputs

Example showing how to read all the IOs (digital inputs and/or outputs) of the equipment. Only valid for equipment that has digital inputs and/or outputs.

Example 16.- Write Digital Output

Example showing how to change the status of a digital output on the device. Only valid for devices that have digital outputs.

Example 17.- Ping IP and send SMS if there is a problem with Ping

Example showing how to ping a device connected to the equipment and send an SMS if there are problems with the ping, for example to detect a malfunction in a connected device.

Example 18.- Send SMS Alarm when Backup SIM 2 is being used

Example that sends an SMS alert when the secondary SIM card comes into operation. Only valid for devices with DUAL SIM.

Example 19.- Read and Write data from SERIAL PORT

Example showing how to read and write to a serial port. Example valid only for equipment with one or more RS232 and/or RS485 serial ports

Example 20.- Use of WATCHDOG

Example showing how to use the watchdog in TITAN Scripts. Restart in case of script malfunction.

Example 21.- Use of LEDs for custom indications

Example showing how to use LEDs in a customised way, managing their switching on and off from a script.

Example 22.- TITAN as MODBUS concentrator saving data into internal memory

Example showing how to use the device as a Modbus hub, saving the data read to internal memory. For example, it is possible to autonomously read different Modbus RTU sensors and store the records read in different positions within the device's memory

device and, from a remote location, read the data from the records stored within the device via Modbus TCP.

Example 23.- Exchanging data through AT Commands sent by SMS, MQTT, Telnet, SSH, HTTP ...

Example showing how to exchange data using AT commands from scripts sent by SMS, MQTT, Telnet, etc. In other words, you can send an AT command via SMS, MQTT, Telnet, etc. to the device and, from the script you have programmed, you can receive this data and act accordingly.

Example 24.- Encrypting the code of Script

Example showing how to encrypt the code of a script. For example, if you are going to supply a third party with a device with a pre-programmed script and do not want to show the source code, you can encrypt the script as shown in this example.

Example 25.- Encrypting the code of Script and avoiding the Piracy of the Script

Example showing how to encrypt script code. For example, if you are going to supply a third party with equipment containing a pre-programmed script and do not want to reveal the source code so that it cannot be copied or pirated, you can encrypt the script as shown in this example.

Example 26.- Saving DATA into the internal TITAN LOGGER.

Example showing how to store data in the device's internal memory. This allows you to take advantage of the device's automatic mechanisms with its memory to send data to an MQTT broker, HTTP server, etc.

Example 27.- Use of user FILES for reading, writing and appending data.

Example showing how to create user files for writing, reading and appending data.

Example 28.- Uploading user FILES by FTP

Example showing how to send user files via FTP. Example 29.-

Downloading user FILES via FTP

Example showing how to receive user files via FTP. Example 30.- Read

Modbus RTU, save data into FILE, and send it via FTP

Example showing how to read Modbus RTU data from a script, save the data to a user file, and send that file via FTP.

Example 31.- Download and install new version of SCRIPT from FTP Server

Example showing how to remotely update a user script.

Example 32.- Read instant values of IEC-60870-5-102 Meter

Example showing how to read instant values from an energy meter with IEC-60870-5-102 protocol from a script.

Example 33.- Read GPS data from TITAN

Example showing how to obtain the GPS position from a script. Only valid for devices that have GPS.

Example 34.- Read digital COUNTERS

Example showing how to read the value of pulse counters. Only valid for devices that have digital inputs.

Example 35.- Subscribe to MQTT topic. Read and send byte array data

Example showing how to subscribe to an MQTT topic to receive an array of data.

Example 36.- IO Line Passing by MQTT. Send digital input status to remote digital output.

Example designed for use with two TITAN-based devices that have digital inputs/outputs. When a change of state occurs in a digital INPUT on one device, the state of a digital OUTPUT on the remote device will change. In other words, it shows how to replicate a digital input on a remote digital output.

Example 37.- Check and download by FTPS a remote router configuration.

This example shows how a TITAN-based device can download a configuration backup file via FTPS. The router downloads the file, renames it on the remote server (so it does not need to be downloaded again) and installs the downloaded configuration.

Example 38.- Download a remote file with AT inside via FTPS and execute it

This example shows how a TITAN-based device can download a file containing AT commands via FTPS. Once the file has been downloaded, the TITAN-based device will execute the AT commands read from it.

Example 39.- Send data by HTTP GET and HTTP POST with custom headers

Demonstrative example of how to make HTTP/HTTPS requests using the GET and POST methods, also allowing the use of custom headers.

Example 40.- Read all received SMS from SCRIPT

Example of how to read all SMS messages received on a TITAN-based device from a script. Once the SMS message has been received, the script can obtain the sender's phone number and the associated text message.

Example 41.- Send TELEGRAM message when a GPIO changes.

Example of how to send a text message to a chat group in the Telegram messaging application when a digital input on the TITAN-based device changes. See the application notes for more information.

Example 42.- Send TELEGRAM message when a MODBUS register changes.

Example of how to send a text message to a chat group in the Telegram messaging app when the value of a Modbus register read by TITAN-based equipment via Modbus RTU from a device connected via RS485 port changes. See the application notes for more information.

Example 43.- Read CellID and obtain approximate latitude and longitude from a third party.

Example of how to obtain longitude and latitude from information obtained from the mobile phone cell used by the modem. A third-party API is also used.

Example 44.- Read the cheapest energy hour from the REE website by HTTP REST API.

Example of how to download hourly rates for the regulated electricity market in Spain from REE (Red Eléctrica de España) via HTTP GET (REST API). Once the hourly energy prices have been downloaded, the time with the lowest energy price is sought. The digital output of the equipment will be activated only during that hour (for example, to activate an electric hot water heater) in order to save as much as possible on the electricity bill.

Example 45.- Read and send data through a TCP client SOCKET.

Example of how to establish a TCP client socket against a server and how to send and receive data through it.

Example 46.- Read and send data through a TSSL/TLS CP client SOCKET.

Example of how to establish a TCP client socket with SSL/TLS security against a server and how to send and receive data through it.

Example 47.- Check presence of IEC-60870-5-102 Meter and send SMS alarm if not detected

Example of how to periodically check for the presence of an IEC-60870-5-102 meter. In the event of a presence failure, it will send an SMS alarm message to notify of the situation.

Example 48.- Read instant values of several IEC-60870-5-102 Meters and store data in LOGGER

Example showing how to read instantaneous values from several energy meters using the IEC-60870-5-102 protocol from a script. The data will be stored in the internal LOGGER so that it can be sent to a WEB platform via HTTP, MQTT or FTP.

Example 49.- Configuration of IEC-60870-5-102 to MODBUS TCP/RTU Gateway

Example of IEC-60870-5-102 to Modbus converter. The example shows how to read the instantaneous values of a meter with IEC-60870-5-102 protocol and store them in the user memory area so that they can be read via Modbus TCP or Modbus RTU using an external device (e.g. a PLC).

Example 50.- Communication with ANDROID app by MQTT

Example showing how to interact from a script with an Android app designed by the user. It is essential to consult application note 61.

Example 51.- Forward all received SMS messages via RS232 serial port

Example showing how to receive an SMS message from a script and forward the message content directly via the RS232 serial port.

Example 52.- Read INTEGRATED values of IEC-60870-5-102 Meters and store data in LOGGER

Example showing how to read the load curves (integrated values) from an IEC-60870-5-102 meter every day at 2am and store the data in the LOGGER memory so that the data is sent automatically as configured (MQTT, HTTP, FTP).

Example 53.- Configuration for DLMS/COSEM to MODBUS TCP/RTU Gateway

Example of DLMS/COSEM to Modbus converter. The example shows how to read the instantaneous values of a meter with DLMS/COSEM protocol and store them in the user memory area so that they can be read via Modbus TCP or Modbus RTU using an external device (e.g. a PLC).

Example 54.- Read instant values of a DLMS COSEM meter and store data in LOGGER

Example showing how to read instantaneous energy values using the DLMS/COSEM protocol from a script. The data will be stored in the internal LOGGER so that it can be sent to a WEB platform via HTTP, MQTT or FTP.

Example 55.- Configure and join with a single ZEROTIER VPN

Example showing how to configure and connect to a Zerotier VPN.

Example 56.- Configure and join with several ZEROTIER VPNs

Example showing how to configure and connect to three Zerotier VPN networks simultaneously.

Example 57.- Ask CHATGPT any question

Example showing how to use the chatGPT API to ask a textual question and obtain an answer. In this example, the question asks about the country where the device is located, given certain GPS coordinates.

Example 58.- Ask CHATGPT about what it sees in a remote IMAGE

Example showing how to use the chatGPT API to ask a question about analysing an image from an Internet URL. In this example, chatGPT is asked about an image depicting a traffic jam.

Example 59.- Ask CHATGPT about what it sees in a local IMAGE

Example showing how to use the chatGPT API to ask a question about analysing a local image taken from a webcam.

JSON Transformer Function Script

The JSON Transformer script function allows you to change the data format before sending it to a WEB platform via HTTP, MQTT or FTP. This way, you can adjust the format of the data sent by the TITAN-based device to the platform you need. See application note ANV6_57-Router-TITAN-Json_Transformer_Script for complete information on this feature.

MQTT Topic Function Script

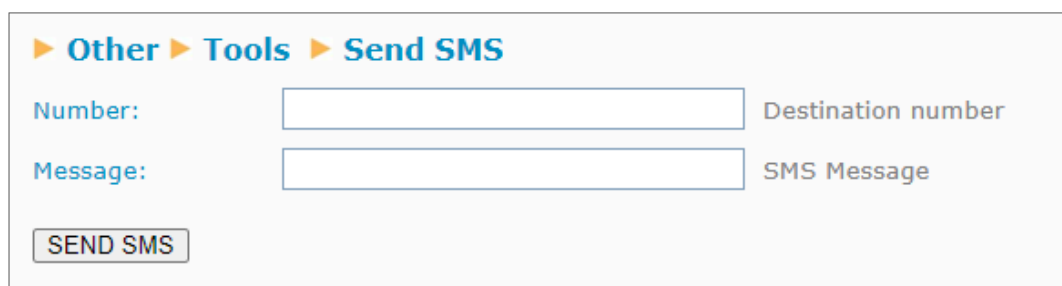
This feature allows you to run a script when the TITAN-based device receives data in certain MQTT topics. This allows you to receive and process data received from a platform.

Web MQTT regardless of their format. Refer to application note ANV6_58-Router-TITAN-Mqtt_Topic_Function_Script for complete information on this feature.

4.7.18 Other Connectivity tools

Sometimes it is useful to know the number of the SIM card inserted in the TITAN-based device, or to check connectivity against a specific IP or DNS from the device's point of view, or to check connectivity against a TCP port on an external server, or simply to know if the email configuration section is correct. This section provides a series of basic tools to meet all these needs.

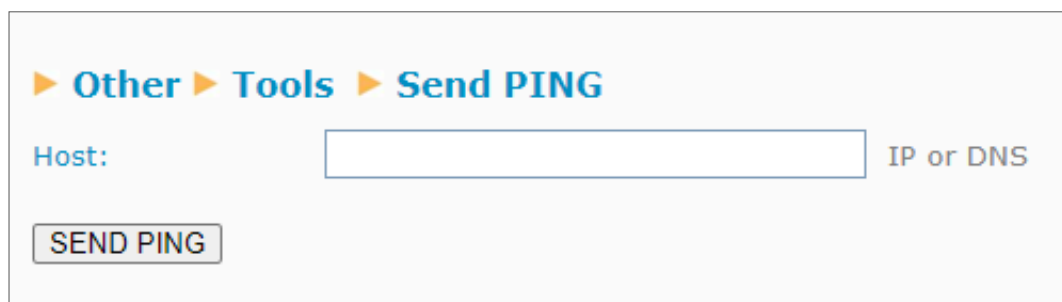
- "Send SMS" utility.



The screenshot shows a web interface for sending an SMS. At the top, there is a breadcrumb navigation: "Other > Tools > Send SMS". Below this, there are two input fields. The first is labeled "Number:" and has a text box next to it, with the label "Destination number" to the right. The second is labeled "Message:" and has a text box next to it, with the label "SMS Message" to the right. At the bottom left, there is a button labeled "SEND SMS".

As its name suggests, this utility allows you to easily send an SMS message from a TITAN-based computer to a mobile phone. It can be useful for quickly obtaining the phone number of the SIM card.

- "Send PING" utility.



The screenshot shows a web interface for sending a PING. At the top, there is a breadcrumb navigation: "Other > Tools > Send PING". Below this, there is a single input field labeled "Host:" with a text box next to it, and the label "IP or DNS" to the right. At the bottom left, there is a button labeled "SEND PING".

This utility allows you to perform a PING from the TITAN-based computer to a specific IP or DNS, allowing you to check the connectivity between both entities.

- "Send EMAIL" utility.

► Other ► Tools ► Send EMAIL

Email:

Email address destination

Subject:

Subject of email

Note: email settings need to be configured before using this tool

SEND EMAIL

This utility allows you to check the correct configuration for sending emails from the TITAN-based computer. To do this, you must first configure the "Other E→ Email" section.

- "Check TCP Connection" utility.

► Other ► Tools ► Check TCP Connection

IP or DNS:

IP or DNS of remote host

TCP port:

TCP port (1 ... 65535)

ASCII Data:

(Optional) Send ascii test data if connection is successful

CHECK CONNECTION

This utility allows you to check whether a TCP port is available for connection from the point of view of the TITAN-based device. To check this connectivity, the device creates a TCP Client socket that attempts to connect to the specified IP/DNS and TCP port. If the connection is successful, the result will be displayed as satisfactory. Optionally, it is possible to add an additional ASCII data field that the TITAN-based device will use to send through that socket once the connection is made.

The lower section of the screen displays a "Results" section showing the results of the previous tools. The following example shows the result of an attempt to connect to the address www.google.com and TCP port 80.

4.7.20 Other – Digital I/O

Many models of modems and TITAN-based equipment have digital inputs and/or outputs. These can be configured from this section.

The screenshot shows the webdyn configuration interface. At the top, the webdyn logo is displayed with the text "powered by TITAN" and "flexitron group". On the left, a sidebar menu lists various configuration categories: Mobile, Ethernet, Wifi, Firewall, Serial Settings, and External Devices. The main content area is titled "Other > Digital I/O". It contains an "Enable:" checkbox which is checked, and a "Period:" dropdown menu set to "Not Logging". Below these is a "SAVE CONFIG" button. Further down, there are two sub-sections: "Other > Digital I/O > Digital Inputs" and "Other > Digital I/O > Digital Outputs". Each sub-section has a "CONFIG" button. At the bottom, a reminder states: "Remember: you can also use I/O with 'Titan Scripts' for special functionality".

- **Enabled.** Activates the TITAN-based device's Input/Output service. It must be activated if you intend to use them.
- **Period.** Allows you to set a time period (in minutes) in which to read the digital inputs and outputs, as well as the pulse counters, and store the reading in the internal data logger of the TITAN-based device for subsequent sending to a platform via HTTP/S or MQTT/S.

Once the option has been enabled and the device restarted, you can configure the behaviour of the digital inputs and digital outputs by pressing the "CONFIG" button corresponding to each one. Remember that digital inputs and outputs can also be managed from TITAN Scripts.

Configuring digital inputs.

The digital input configuration screen looks like this.

The screenshot displays the webdyn configuration interface. The sidebar on the left lists various settings categories: Mobile (Status, Basic Settings, Keep Online), Ethernet (Basic Settings), Wifi (Basic Settings, DHCP Server), Firewall (NAT, Authorized IPs), Serial Settings (Serial Port1-RS232, Serial Port2-RS485, SSL Certificates), External Devices (Logger configuration, ModBus Devices, Generic Serial Device, Temperature Sensor, IEC102 Meter, W-MBus, GPS Receiver), and Other (AT Command). The main panel is titled 'Digital Input IO-0' and contains the following configuration options:

- Digital input mode:** A dropdown menu set to 'Normal'. A note says 'Select the mode of digital input IO-0'.
- Alarm Section:**
 - SMS:** A checkbox labeled 'enabled'. A note says 'When alarm, send SMS message'.
 - Phone numbers:** A text input field. A note says 'Phone numbers separated with ;'.
 - Text SMS Alarm On:** A text input field. A note says 'SMS text when alarm is activated'.
 - Text SMS Alarm Off:** A text input field. A note says 'SMS text when alarm is deactivated'.
 - Logger:** A checkbox. A note says 'Check if logger must be used for alarms. Please, [configure logger](#) before using this option'.
- SAVE CONFIG** button.
- Status Section:**
 - READ INPUT** button next to a display showing '0'. A note says 'Click for read the status of digital input IO-0'.

- **Digital input mode.** Indicates the behaviour of the digital input. You can choose between "normal" if you do not want to use the input for alarms, or choose "Alarm (pin 0→ 1)" or "Alarm (pin 1→ 0)" if you want to set an alarm to activate when the logic value of the digital input changes from 0 to 1 or vice versa.
- **SMS enabled.** This must be enabled if you want the TITAN-based device to send an SMS message when an alarm is activated or deactivated.
- **Phone numbers:** Enter the telephone numbers to which the SMS messages should be sent here. If you need them to be sent to more than one telephone number, you can enter several telephone numbers separated by ";".
- **Test SMS Alarm On:** Text of the SMS message to be sent when the alarm is activated.
- **Test SMS Alarm Off:** Text of the SMS message to be sent when the alarm is deactivated.
- **Logger enabled:** This must be enabled if you want the TITAN-based device to save the digital input change in the Logger so that it can be sent to a platform via HTTP/HTTPS, MQTT/MQTTS or FTP.

If the LOGGER is enabled, this is the JSON object that will be sent to the server.

```
{"IMEI":"867962046823806","TYPE":"GPIO","TS":"2021-11-03T08:16:00Z","ID":0,"VALUE":1,"DIR":"INPUT","P":"ID0001"}
```

Where:

IMEI: modem IMEI

TYPE: frame type. In this case, "GPIO"

TS: Timestamp of when the event occurred ID:
IO identifier (0, 1, ...)

VALUE: logical value of the digital input (0, 1)

DIR: indicates whether the GPIO is of type INPUT or

OUTPUT P: ID field configured in the LOGGER
section

It is also possible to read the value of the digital input in real time by pressing the "READ INPUT" button.

[▶ Other](#) [▶ Digital I/O](#) [▶ Digital Input IO-0](#) [▶ Status](#)

READ INPUT

1

Click for read the status of digital input IO-0

Digital output configuration.

The digital output configuration screen looks like this

 **webdyn**
flexitron group

powered by **TITAN**

★ **Mobile**

● Status

● Basic Settings

● Keep Online

★ **Ethernet**

● Basic Settings

★ **Wifi**

● Basic Settings

● DHCP Server

★ **Firewall**

● NAT

● Authorized IPs

[▶ Other](#) [▶ Digital I/O](#) [▶ Digital Output IO-1](#)

On:

SET ON

 Click for activate digital output IO-1

Off:

SET OFF

 Click for deactivate digital output IO-1

[▶ Other](#) [▶ Digital I/O](#) [▶ Digital Output IO-1](#) [▶ Status](#)

READ STATUS

0

Click for read the status of digital output IO-1

From this screen, you can change the status of a digital output and read its current status. Remember that you can also change the status of a digital output from TITAN Scripts.

4.7.21 Other – Custom Skin

TITAN-based equipment allows you to configure the header of the web configuration environment with your own logos, as well as its title and footer. This allows you, in a way, to give the appearance of a customised or white label product.

- **Set custom image:** select a 922 x 172 pixel image (gif type) for the header area.
- **Set custom labels:** enter the text you want to appear in the title and footer of the web configuration environment.

The screenshot displays the 'webdyn' web configuration interface, powered by TITAN and Flexitron group. The left sidebar lists navigation options: Mobile (Status, Basic Settings, Keep Online), Ethernet (Basic Settings), Wifi (Basic Settings, DHCP Server), Firewall (NAT, Authorized IPs), and Serial Settings (Serial Port1-RS232, Serial Port2-RS485). The main panel is titled 'Other > Custom Skin'. Under 'Set custom image (logo)', there is a file selection area with a button 'Seleccionar archivo' (showing 'Ninguno archivo selec.') and an 'Upload' button. Instructions specify selecting a 922x172 pixel image and pressing the upload button (or CTRL+F5 if the image is not visible). Under 'Set custom labels', there are two text input fields. The first field contains 'Intelligent Router - Web Panel Control' and is labeled 'Bottom text at web interface.'. The second field contains 'Intelligent Router' and is labeled 'Title text at web interface.'. A 'SAVE CONFIG' button is located at the bottom of the configuration area.

Additional notes.

- Once you have finished configuring, press the "SAVE CONFIG" button to save your changes.
- Please note that if you configure the equipment with your header and customised texts to be viewed by third parties, it makes no sense not to use the "Other > User Permissions" section. Use this section to prevent the "user" from accessing the "Other > Custom Skin" section.

4.7.22 Other – Custom LED

TITAN-based devices have several configurable LEDs. From this section, you can set the behaviour that best suits your needs.

- *Not used*: The LED will always be off, not in use.
- *IP*: The LED will turn on if the device has an IP address (WAN 4G/3G/2G).
- *RSSI*: The LED will flash when coverage is low and remain on without flashing when coverage is good. Off indicates critical or no coverage.
- *SIM Error*: The LED will light up when a SIM card fault is detected.
- *TITAN Scripts*: The LED will be used from the TITAN Scripts section of the device.

The screenshot shows the webdyn web interface. At the top, there is a logo for 'webdyn' with 'flexitron group' underneath and 'powered by TITAN' to the right. On the left, there is a sidebar with navigation links: 'Mobile' (Status, Basic Settings, Keep Online), 'Ethernet' (Basic Settings), and 'Wifi' (Basic Settings, DHCP Server). The main content area is titled 'OTHER > Led Configuration'. It contains three rows of configuration for LEDs: 'Led1' with a dropdown set to 'RSSI', 'Led2' with a dropdown set to 'IP', and 'Led3' with a dropdown set to 'Titan Scripts'. Each row has a 'Led behavior' label to its right. At the bottom of the configuration area is a 'SAVE CONFIG' button.

Additional notes.

- Once the configuration is complete, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the device for the new changes to take effect.

4.7.23 Other - SYSLOG

From this section, you can access an internal log of the TITAN-based device, where you can view the status of the device itself (coverage, network registration, operator, results of transmissions to servers , actions performed by users, etc.



The screenshot displays the webdyn interface, which is part of the Flexitron group and powered by TITAN. On the left, a sidebar menu lists various settings categories: Mobile (Status, Basic Settings, Keep Online), Ethernet (Basic Settings), Wifi (Basic Settings, DHCP Server), Firewall (NAT, Authorized IPs), Serial Settings (Serial Port1-RS232, Serial Port2-RS485, SSL Certificates), and External Devices (Logger configuration, ModBus Devices, Generic Serial Device, Temperature Sensor, IEC102 Meter, W-MBus, GPS Receiver). The main content area is titled 'Other > SYSLOG' and shows a log of system events. The log entries include timestamps, device identifiers, and messages such as 'Signal Level-4G', 'Registration-status: Registered', and 'Watchdog ack received'. At the bottom of the log area, there are three buttons: 'REFRESH LOG', 'SYSLOG CONFIG', and 'DELETE LOG'. A link below these buttons reads 'Click here for download current SYSLOG complete file'.

Clicking on the "SYSLOG CONFIG" button will take you to the SYSLOG configuration section. In this configuration section, you can set the following configurations.

- **Enabled Local:** "Disabled": SYSLOG will not be used. "RAM" (recommended option) will store SYSLOG in RAM. "FLASH" (only available on some TITAN-based devices) will store SYSLOG in flash memory. Only recommended for debugging and not for permanent use, as constant use can cause excessive wear on the FLASH memory.
- **Enabled Server 1:** Checking the box enables SYSLOG to be sent via UDP to a remote server.
- **Remote Server 1:** IP or DNS of remote server 1 to receive the SYSLOG.
- **Remote Port 1:** UDP port of remote server 1
- **Enabled Server 2:** Checking this box enables SYSLOG sending via UDP to a second remote server.
- **Remote Server 2:** IP or DNS of remote server 2 to receive SYSLOG

- **Remote Port 2:** USD port of remote server 2
- **MQTT Topic:** If an MQTT topic is specified, the syslog will be sent in real time to the specified MQTT topic. The Other section→ ☐ MQTT must be configured.
- **Network Log:** Checking the box adds the network status to the SYSLOG periodically.

webdyn powered by **TITAN**
flexitron group

Mobile

- Status
- Basic Settings
- Keep Online

Ethernet

- Basic Settings

Wifi

- Basic Settings
- DHCP Server

Firewall

- NAT
- Authorized IPs

Serial Settings

- Serial Port1-RS232
- Serial Port2-RS485
- SSL Certificates

External Devices

- Logger configuration
- ModBus Devices
- Generic Serial Device
- Temperature Sensor
- IEC102 Meter
- W-MBus
- GPS Receiver

Other > SYSLOG Config

Enabled Local: Enable Local Syslog

Enabled Server 1: ☐ Enable Remote Syslog Server 1

Remote Server 1: Remote Server 1 (DNS or IP)

Remote Port 1: Remote Server 1 UDP port

Enabled Server 2: ☐ Enable Remote Syslog Server 2

Remote Server 2: Remote Server 2 (DNS or IP)

Remote Port 2: Remote Server 2 UDP port

Mqtt topic: Only for debug purposes (mqtt config needed). Field blank = disabled

Mobile Network Log: ☒ Enable network log (rssi, tech, oper)

SAVE CONFIG

Additional notes.

- Once the configuration is complete, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the device for the new changes to take effect.

4.7.24 Other – Backup / Factory

You can make a complete backup of the device settings from this menu. You can save the settings to a file and restore them to the device when needed. It also allows you to reset the device to its factory settings.

- **Factory Settings button:** press this button if you want to restore the device to its factory settings.
- **"Download settings" button:** press this button to download the device settings to a file named "config.mtx".
- **"Select file" button:** to restore a saved configuration, after indicating the configuration file to use, press the "Upload" button to upload the file.

The screenshot shows the webdyn web interface. At the top, there is a logo for webdyn, which is a stylized 'w' made of three overlapping waves in grey, teal, and blue. To the right of the logo is the text 'webdyn' in a large, bold, black font, and below it, 'flexitron group' in a smaller font. Further to the right, it says 'powered by TITAN' in a teal font. On the left side, there is a sidebar menu with a star icon next to each category: 'Mobile' (with sub-items: Status, Basic Settings, Keep Online), 'Ethernet' (with sub-items: Basic Settings, DHCP Server), 'Firewall' (with sub-items: NAT, Authorized IPs, Routes), 'Serial Settings' (with sub-items: Serial Port1-RS232, Serial Port2-RS232, Serial Port3-RS485, Serial Port4-TTL, SSL Certificates), and 'External Devices' (with sub-items: Logger configuration, ModBus Devices, Generic Serial Device, Temperature Sensor). The main content area is titled 'Other > Backup / Factory'. It contains three sections: 1. 'Press the button for factory settings' with a 'Factory Settings' button. 2. 'Download full configuration' with a dropdown menu showing 'Download full configuration' and a 'Download configuration' button. 3. 'Select a config.mtx file, then press the button for upload the configuration file' with a file selection button labeled 'Seleccionar archivo' (showing 'Ninguno archivo selec.'), and an 'Upload' button.

4.7.25 Other - Reboot

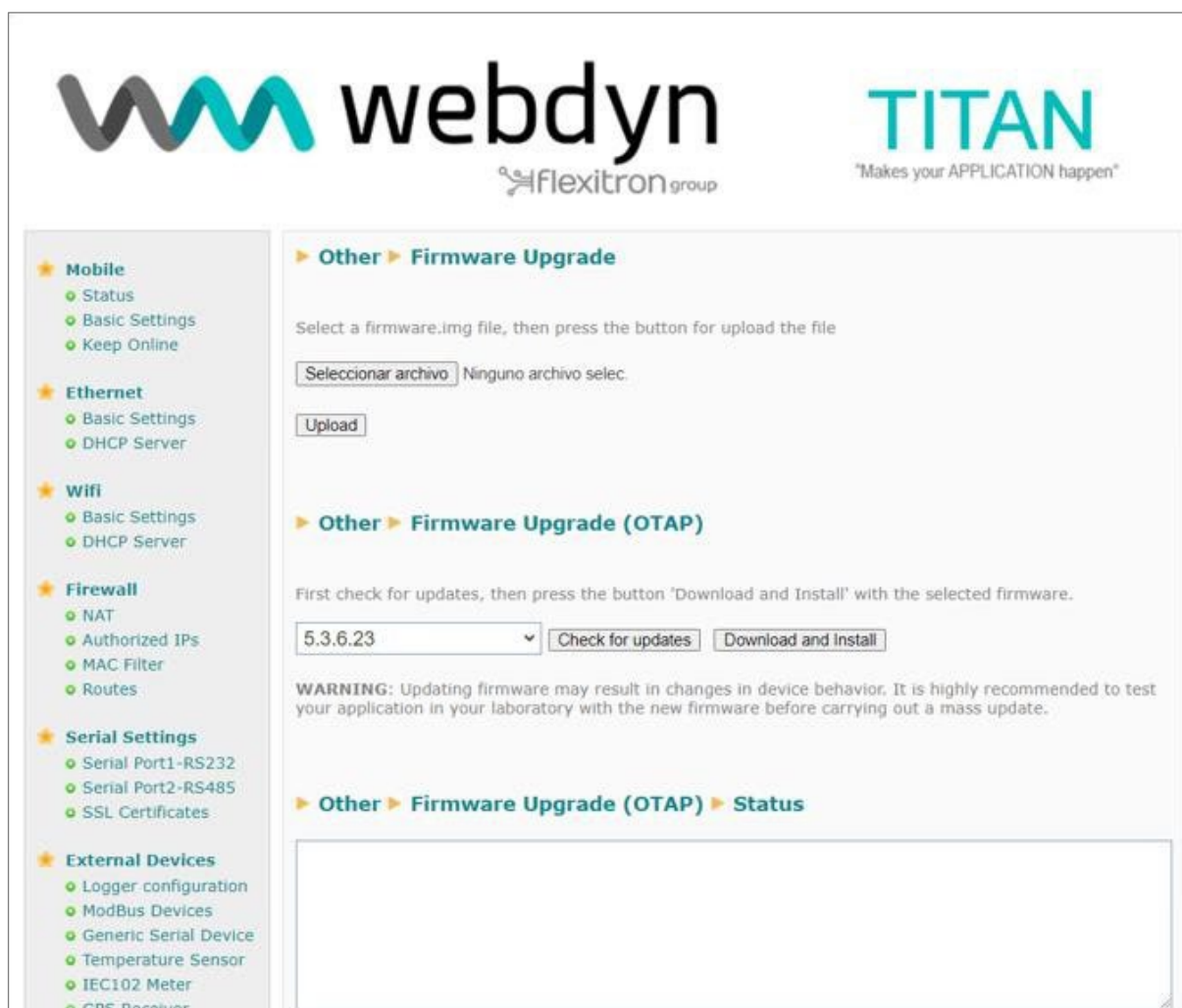
From this section, you can restart the device. For example, to apply changes after a configuration change. To do this, simply press the "Reboot" button.

Depending on the device, the "Power down" option may also be available. This option allows you to turn off a device model that has an internal battery, such as the MTX-Router-Q model. Before pressing the "Power down" button, you must disconnect the device from the power supply (220Vac) and then press the button. After a few seconds, all LEDs will remain off.



4.7.26 Other – Firmware Upgrade

From this section, you can update the Titan router's firmware in two ways: locally or remotely.



Firmware Upgrade

If you have the firmware file to update the device, click on the "select file" button, select the file, and then click on the "Upload" button. The process may take up to 2 minutes, and the Titan router may restart up to 2 times at the end of the process.

Firmware Upgrade (OTAP)

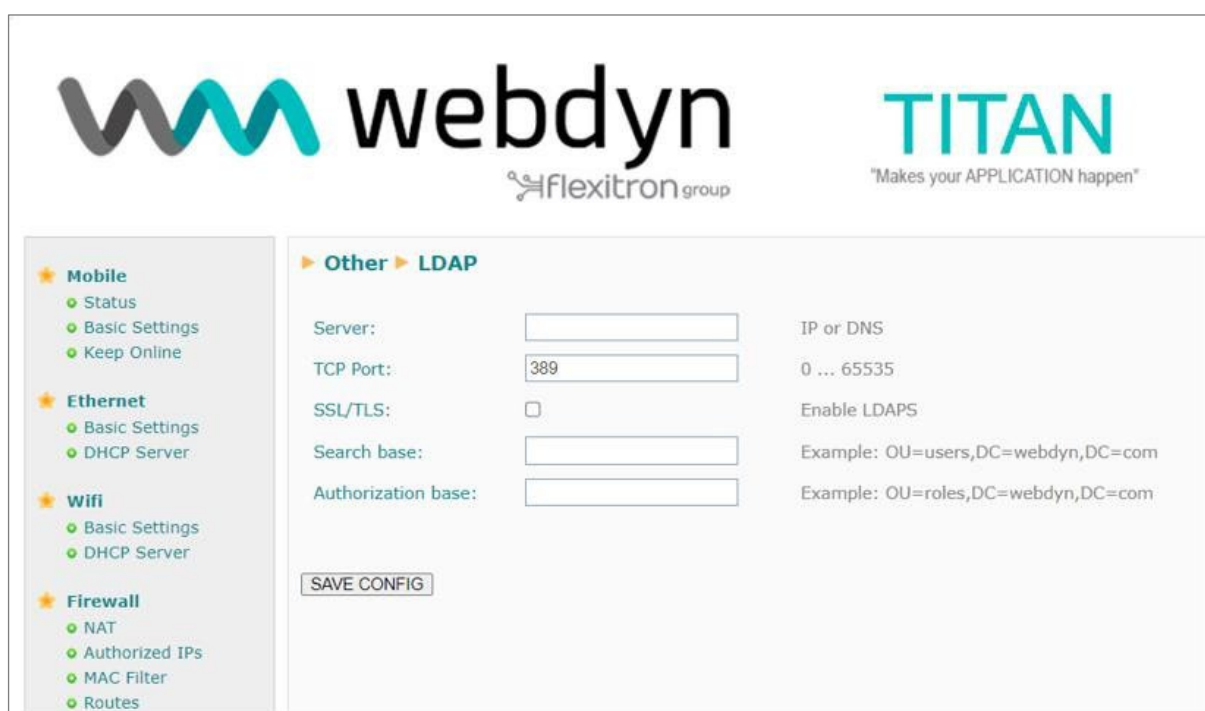
Another option for updating the device's firmware is through the OTAP process. If you intend to use this option, your Titan router must have Internet access, as the firmware will be downloaded from Webdyn's servers. To carry out the OTAP process, click on the "Check for updates" button. After a few seconds, if there are any firmware updates for your router, they will appear in the drop-down menu. Select the firmware version you wish to install and click on the "Download and Install" button to start the process.

Remember that any firmware change process on any device carries risks. It is highly recommended that you check that the scenario you have configured on the Titan router works correctly in the laboratory with the firmware version you intend to install before proceeding with mass firmware updates of devices located in remote locations.

4.7.27 Other – LDAP

TITAN-based devices have the option of LDAP authentication. This feature will enable external authentication for HTTP, Telnet, and SSH services on TITAN-based devices. To use this feature, you must configure this section.

- **Server:** IP address or DNS of the LDAP server
- **TCP Port:** TCP port of the LDAP server
- **SSL/TLS:** Tick this box if your LDAP server uses SSL/TLS
- **Search base:** Base DN for searching for users on the LDAP server
- **Authorisation base:** Base DN for checking authorisation on the LDAP server



The screenshot shows the Webdyn TITAN configuration interface. At the top, the Webdyn logo (a stylized 'w' with a blue and green wave) and the TITAN logo (in blue) are displayed. Below the logos, the text 'flexitron group' and 'Makes your APPLICATION happen' are visible. On the left side, there is a sidebar menu with categories: Mobile (Status, Basic Settings, Keep Online), Ethernet (Basic Settings, DHCP Server), Wifi (Basic Settings, DHCP Server), and Firewall (NAT, Authorized IPs, MAC Filter, Routes). The main content area is titled 'Other > LDAP'. It contains the following fields: 'Server:' with a text input box and the label 'IP or DNS'; 'TCP Port:' with a text input box containing '389' and the label '0 ... 65535'; 'SSL/TLS:' with a checkbox and the label 'Enable LDAPS'; 'Search base:' with a text input box and the label 'Example: OU=users,DC=webdyn,DC=com'; and 'Authorization base:' with a text input box and the label 'Example: OU=roles,DC=webdyn,DC=com'. At the bottom of the main content area, there is a 'SAVE CONFIG' button.

Additional notes.

- The LDAP service on Titan routers uses authentication and authorisation.
- For authorisation, the Titan router will search for a CN "webdyn_http_admin", "webdyn_http_user", "webdyn_http_guest", "webdyn_telnet_admin", "webdyn_ssh_admin" in a "memberOf" attribute.
- Refer to application note ANV6_75, which explains in detail, with an example, how to configure the Titan router and its LDAP servers.
- Once the configuration is complete, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the device for the new changes to take effect.

4.8 VPN

From this section, you can configure VPN services such as OpenVPN, IPSec, and ZeroTier.

4.8.1 VPN – OpenVPN Server

TITAN-based routers can act as OpenVPN servers. By establishing a VPN, all devices connected to the device's Ethernet bus, or even IP-RS232/485 gateways, will be securely accessible from the other end of the VPN network. A VPN infrastructure is very useful for circumventing most problems with proxies, firewalls, etc., especially when using SIM cards from an operator that assigns private IP addresses (10.x.x.x).

The screenshot shows the webdyn interface for configuring the OpenVPN Server. The interface is titled "webdyn" with the "flexitron group" logo and "powered by TITAN". On the left is a sidebar menu with categories: Mobile, Ethernet, Firewall, External Devices, VPN, and Other. The "VPN" category is selected, and the "OpenVPN Server" option is highlighted. The main configuration area is titled "VPN > OpenVPN Server" and contains the following settings:

- Enabled:** A checkbox that is currently unchecked.
- Mode:** A dropdown menu set to "Always On".
- Protocol:** A dropdown menu set to "udp".
- Port:** A text input field containing "1194".
- LZO compression:** A checkbox that is checked.
- Auth:** A dropdown menu set to "SHA512".
- Cipher:** A dropdown menu set to "AES-256-CBC".
- Server subnet:** An empty text input field.
- Server mask:** An empty text input field.
- Allow LAN access:** A checkbox that is unchecked.
- Client 1, Client 2, Client 3:** Each has a pair of empty text input fields for Subnet and Mask.

Help text is provided for several settings:

- Enable OpenVPN Server:** Always on or under request (ex, by SMS)
- Communication protocol:** Default tcp / udp port: 1194
- Enable LZO compression:** Normally yes.
- Authentication:**
- Cipher:** Example 10.8.0.0 (Titan will take 10.8.0.1)
- Example 255.255.255.0**
- Click for allow access to devices connected to MTX**
- Subnet and Mask:** (Ex: 192.168.1.0 / 255.255.255.0)

- **Enabled:** Tick this box if you wish to activate the OpenVPN service in Server mode.
- **Mode:** Allows you to set the VPN connection mode. A value of "Always On" will keep the VPN active at all times. A value of "Under Request" will enable the VPN for as long as you decide. For example, you may want to activate the VPN only at specific times for maintenance tasks. When you want to activate the VPN, simply send an AT command (see chapter 5 on AT commands) via SMS, Telnet, Webserver, SNMP or even Modbus TCP. Once the configured time has elapsed, the VPN will close.

- *Protocol*: You can set the establishment protocol. You can use either UDP or TCP protocol.
- *Port*: The standard port is 1194, but you can specify the port for the VPN that suits you best.
- *LZO compression*: tick the box if you wish to enable this option.
- *Auth*: allows you to choose the authentication method.
- *Cipher*: allows you to choose the encryption algorithm.
- *Server Subnet*: specify the subnet that will be created after the VPN is established. For example, if you specify the subnet 10.8.0.0, when the VPN is activated, the TITAN-based device will adopt the address 10.8.0.1.
- *Server Mask*: specify the network mask to be applied to the VPN
- *Allow Lan Access*: tick the box if you want to be able to remotely access devices connected to the device's Ethernet port. Do not tick it if you only want to be able to access the TITAN-based device and the IP-RS232/485 gateways it manages.
- Files required for OpenVPN

In order to use TITAN-based equipment as an OpenVPN server, you will need to create and upload the following files: ca.crt (certificate authority), server.crt (server certificate), server.key (server private key), dh1024.pem (Diffie Hellman parameters).

At the bottom of the screen, you will see that you can download some demo files that should only be used for testing purposes. Remember that in a real application, you should generate your own certificates.

Additional notes.

- Once you have finished the configuration, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the equipment for the new changes to take effect.
- Application notes (PDF documents external to this document) are available with examples of OpenVPN functionality with TITAN-based equipment.

4.8.2 VPN – OpenVPN Client

TITAN-based routers can act as OpenVPN clients. By establishing a VPN, all devices connected to the device's Ethernet bus, or even IP-RS232/485 gateways, will be securely accessible from the other end of the VPN network. A VPN infrastructure is very useful for circumventing most problems with proxies, firewalls, etc., especially when using SIM cards from an operator that assigns private IP addresses (10.x.x.x).

webdyn powered by **TITAN**
flexitron group

VPN > OpenVPN Client

Enabled: ☐ Enable OpenVPN Client

Mode: Always on or under request (ex, by SMS)

Protocol: Communication protocol

LZO compression: ☒ Enable LZO compression. Normally yes.

Auth: Authentication

Cipher: Cipher

IP Server: IP of remote OpenVPN Server

Port Server: Port of remote OpenVPN Server (def 1194)

Username: Username. Blank if not used

Password: Password. Blank if not used

Certificate Authority: Ninguno archivo selec. not uploaded

Client certificate: Ninguno archivo selec. not uploaded

Client KEY: Ninguno archivo selec. not uploaded

Client KEY: Ninguno archivo selec. not uploaded

- **Enabled:** Check this box if you want to activate the OpenVPN service in Client mode.
- **Mode:** Allows you to set the VPN connection mode. A value of "Always On" will keep the VPN active at all times. A value of "Under Request" will enable the VPN for as long as you decide. For example, you may want to activate the VPN only at specific times for maintenance tasks. When you want to activate the VPN, simply send an AT command (see chapter 5 on AT commands) via SMS, Telnet, Webserver, SNMP or even Modbus TCP. Once the configured time has elapsed, the VPN will close.
- **Protocol:** You can set the establishment protocol. You can use either UDP or TCP protocol.
- **Port:** The standard port is 1194, but you can specify the port for the VPN that suits you best.
- **LZO compression:** tick the box if you wish to enable this option.

- Auth: allows you to choose the authentication method.
- Cipher: allows you to choose the encryption algorithm.
- *IP Server*: enter the public IP address of the remote server that will act as the OpenVPN server
- *Files required for OpenVPN*

In order to use TITAN-based devices as OpenVPN clients, you will need to create and upload the following files: ca.crt (certificate authority), client.crt (client certificate), client.key (client private key).

At the bottom of the screen, you will see that you can download some demo files that should only be used for testing purposes. Remember that in a real application, you should generate your own certificates.

Additional notes.

- Once you have finished configuring, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the computer for the new changes to take effect.
- Application notes (PDF documents external to this document) are available with examples of OpenVPN functionality with TITAN-based equipment.

4.8.3 VPN – IPsec

TITAN-based routers feature IPSEC. By establishing a VPN, all devices connected to the device's Ethernet bus, or even IP-RS232/485 gateways, will be securely accessible from the other end of the VPN network. A VPN infrastructure is very useful for avoiding most problems with proxies, firewalls, etc., especially when using SIM cards from an operator that assigns private IP addresses (10.x.x.x).



- **Enabled:** Tick this box if you wish to activate the IPSec service.
- **ipsec.conf:** enter the configuration file for your IPSec VPN here. You can view and modify the examples at the bottom of the screen to suit your needs.
- **ipsec.secrets:** enter the contents of your IPSec VPN's ipsec.secrets file here. You can view and modify the examples at the bottom of the screen to best suit your needs.

TITAN-based computer acting as an IPSec client:

- **xca1-cert.pem:** file containing the CA root certificate for your IPSEC server
- **xclient1-cert.pem:** file containing the client certificate
- **xclient1-key.pem:** file containing the KEY

TITAN-based computer acting as an IPSec server:

- **ca-cert.pem:** CA root file for the TITAN-based computer
- **ca-key.pem:** file containing the CA KEY for the TITAN-based computer
- **server-cert.pem:** certificate file of the TITAN-based device
- **server-key.pem:** file containing the KEY for the TITAN-based device

- client1-cert.pem: file containing the certificate for client 1 authorised to connect
- client2-cert.pem: file containing the certificate for client 1 authorised to connect
- client3-cert.pem: file containing the certificate for client 1 authorised to connect

Additional notes.

- Once the configuration is complete, press the "SAVE CONFIG" button to save the changes. Remember that you must restart the device for the new changes to take effect.
- Application notes (PDF documents external to this document) are available with examples of IPSec functionality with TITAN-based equipment.

4.8.5 VPN - ZeroTier

Titan routers support the ZeroTier VPN service from FW version 5.3.6.25 onwards. ZeroTier allows you to implement a VPN very easily and quickly, enabling access to Titan routers and devices connected to them from any location. Please refer to application note AN71 for usage examples and more detailed information.

 **webdyn**
flexitron group

TITAN
"Makes your APPLICATION happen"

★ **Mobile**

- Status
- Basic Settings
- Keep Online

★ **Ethernet**

- Basic Settings
- DHCP Server

★ **Wifi**

- Basic Settings
- DHCP Server

★ **Firewall**

- NAT
- Authorized IPs
- MAC Filter
- Routes

★ **Serial Settings**

- Serial Port1-RS232
- Serial Port2-RS485
- SSL Certificates

★ **External Devices**

- Logger configuration
- ModBus Devices
- Generic Serial Device
- Temperature Sensor
- IEC102 Meter

★ **VPN**

▶ **VPN ▶ ZeroTier**

Enabled: ☐ Enable ZeroTier VPN service

TCP Port: TCP internal port (default 9993)

VPN1 Network ID: Network ID ZeroTier 1 (blank = not used)

VPN2 Network ID: Network ID ZeroTier 2 (blank = not used)

VPN3 Network ID: Network ID ZeroTier 3 (blank = not used)

▶ **VPN ▶ ZeroTier ▶ Status**

Device ID: ---

VPN1 Network name: ---

VPN1 IP & MAC: IP: --- MAC: ---

VPN1 interface: zt0

VPN1 status: ---

VPN2 Network name: ---

VPN2 IP & MAC: IP: --- MAC: ---

VPN2 interface: zt1

 webdyn | 150

contact@webdyn.com | webdyn.com
V02.08 subject to change | Webdyn © by Flexitron

- **Enabled:** Select this box if you wish to activate the ZeroTier service.
- **TCP Port:** TCP port for internal use by the Titan router. Keep the default TCP port 9993 if you do not need it for another service.
- **VPN1 Network 1:** ZeroTier identifier for VPN network 1
- **VPN1 Network 2:** ZeroTier identifier for VPN network 2
- **VPN1 Network 3:** ZeroTier identifier for VPN network 3

The screenshot displays the ZeroTier configuration page. On the left, a sidebar menu lists various system settings under 'Other', including AT Command, DynDns, Private DynDns, Sms control, Periodic Autoreset, Time Servers, Remote Console, Snmp, Tacacs+, Mqtt, Http / Https, User Permissions, Passwords Web UI, CA Certificates, Email Config, ModBus Slave, Titan Scripts, and Connectivity tools. The main content area is titled 'ZeroTier' and shows configuration for 'VPN3'. It includes fields for 'VPN3 Network name', 'VPN3 IP & MAC' (with sub-fields for IP and MAC), 'VPN3 interface' (set to 'zt2'), and 'VPN3 status'. A 'REFRESH' button is located below these fields. A breadcrumb trail indicates the path: 'VPN > ZeroTier > Tools'. At the bottom, there is a 'Device ID' field, a 'RESTART' button, and a note: 'Restart device ID. You will need to accept again the device in your zerotier account.'

- **Restart:** button that allows you to change the Device ID of the device on the ZeroTier network. If the Device ID is changed, you will need to go to the ZeroTier control panel (my.zerotier.com) and re-accept the device on the network.

5 AT commands

TITAN-based devices allow AT commands to be sent directly to the internal modem through multiple interfaces:

- 1.- Via a serial port.
- 2.- Via a 4G/3G/2G-Serial gateway (via IP WAN, Ethernet or Wi-Fi) using embedded AT commands.
- 3.- Via SMS
- 4.- Via Telnet (Remote Console, via 4G/3G/2G, Ethernet or Wi-Fi)
- 5.- Via SSH (Remote Console, via 4G/3G/2G, Ethernet or Wi-Fi)
- 5.- Via Webserver (via 4G/3G/2G, Ethernet or Wi-Fi)
- 7.- Via Modbus RTU (via an RS232 or RS485 serial port)
- 8.- Via Modbus TCP (via 4G/3G/2G, Ethernet or Wi-Fi)
- 9.- Via SNMP (via 4G/3G/2G, Ethernet or Wi-Fi)
- 10.- Via URL. Explained at the end of this point.

Therefore, under your responsibility, you may send AT commands to the device.

- **AT^MTXTUNNEL=REBOOT**

Action: Resets the TITAN-based device.

Result:

OK: Command executed successfully.
ERROR: Command executed with error

Example: AT^MTXTUNNEL=REBOOT
OK

- **AT^MTXTUNNEL=VERSION**

Action: Returns the firmware version of the TITAN device.

Result:

OK: Command executed successfully.
ERROR: Command executed with error

Example: AT^MTXTUNNEL=VERSION
5.2.6.17
OK

- **AT^MTXTUNNEL=GETIP**

Action: Returns the WAN IP address (2G/3G/4G)

Result:

OK: Command executed successfully.
 ERROR: Command executed with error

Example: AT^MTXTUNNEL=GETIP
 88.28.221.14
 OK

- **AT^MTXTUNNEL=GETIMEI**

Action: returns the IMEI of the internal modem

Result:

OK: Command executed successfully.
 ERROR: Command executed with error

Example: AT^MTXTUNNEL=GETIMEI
 869101054255506
 OK

- **AT^MTXTUNNEL=GETCELLID**

Action: Returns the mobile phone cell being used by the modem.

Result:

OK: Command executed successfully.
 ERROR: Command executed with error.

Example: AT^MTXTUNNEL=GETCELLID
 [4;214;07;219B;15F2D2A]
 OK

Where: 4 = technology used (2=2g,3=3)

214 = MCC (Mobile Country Code)

07 = MNC (Mobile Network Code)

07 = LAC (Local Area Code)

15F2D2A = CID (Cell ID)

AT commands related to messaging:

- **AT^MTXTUNNEL=TRAP,<OID>;<myMessage>;<mySeverity>**

Action: allows you to send an SNMP TRAP, with a specific OID, the corresponding associated message and the severity.

Parameters:

: OID of the trap to be sent by SNMP
<myMessage>: text message to be sent
<mySeverity>: 0 ... 7 severity of the trap

Result:

OK: Command executed successfully. Message sent.
ERROR: Command executed with error

Example: AT^MTXTUNNEL=TRAP,.1.3.6.1.4.1.45711.1.1.11.1.1;myMessage;5 OK

- **AT^MTXTUNNEL=SMS,<telephoneNumber>,<message>**

Action: allows you to send an SMS message to a specific telephone number.

Parameters:

<telephoneNumber>: telephone number to which the SMS message is to be sent
<message>: text message to send

Result:

OK: Command executed successfully. SMS message sent to the output queue.
ERROR: Command executed with error

Example: AT^MTXTUNNEL=SMS,+34677123456,burglar alarm OK

- **AT^MTXTUNNEL=EMAIL,<destinationAddress>,<subject>**

Action: allows you to send an email (subject only) to a specific email address. To send the email, you must first configure the "Other > Email Configuration" section.

Parameters:

<destinationAddress>: destination email address
<subject>: message to be sent by email

Result:

OK: Command executed successfully. Email sent.
ERROR: Command executed with error

Example: AT^MTXTUNNEL=EMAIL,jgallego@matrix.es,Alarma temperature
OK

- **AT^MTXTUNNEL=TELEGRAM,SEND,<tokenID>,<chatID>,<message>**

Action: allows you to send a Telegram text message to a specific chat in a Telegram group. You need to create a bot with your tokenID and obtain the chatID of the chat to which you want to send the message. See the application notes for more information.

Parameters:

<tokenID>: token identifier
 <chatID>: chat identifier
 <message>: message to send via Telegram

Result:

OK: Command executed successfully. Message sent.
 ERROR: Command executed with error

Example: AT^MTXTUNNEL=TELEGRAM,SEND,5416419553:ZZHof6enVx-
 y7uFQrL15cpf3QCHqNHp7Dtq,-730297962,My message
 OK

AT commands related to I/O:

- **AT^MTXTUNNEL=GETIO,<idIO>**

Action: returns the value of a digital input or output, where idIO is the IO identifier (0,1,2...)

Parameters:

<idIO>: 0, ... X number of I/O

Result:

OK: Command executed correctly, returning the status of the digital input or output.
 ERROR: Command executed with error

Example: Reading the status of digital input IO 0

AT^MTXTUNNEL=GETIO,0
 1
 OK

- **AT^MTXTUNNEL=SETIO,<idIO>,<value>**

Action: allows you to change the status of a digital output, where idIO is the IO identifier (0,1,2...) and value is the logical value of the output (0, 1)

Parameters:

<idIO>: 0, ... X number of I/O
<value>: 0.1 value to be set in the digital output

Result:

OK: Command executed correctly. Digital output updated.
ERROR: Command executed with error

Example: AT^MTXTUNNEL=SETIO,3,1
 OK

- **AT^MTXTUNNEL=GETCOUNTER,<idIO>**

Action: Returns the value of the pulse counter associated with a digital input, where idIO is the IO identifier (0,1,2...).

Parameters:

<idIO>: 0, ... X number of I/O

Result:

OK: Command executed correctly, returning the value of the pulse counter associated with the digital input
ERROR: Command executed with error

Example: AT^MTXTUNNEL=GETCOUNTER,0
 1022
 OK

- **AT^MTXTUNNEL=SETCOUNTER,<idIO>,<value>**

Action: allows you to change the value of a pulse counter associated with a digital input, where idIO is the IO identifier (0,1,2...) and value is the counter value

Parameters:

<idIO>: 0, ... X number of I/O
<value>: 0,1 value to be set in the counter associated with the digital input.

Result:

OK: Command executed correctly.
ERROR: Command executed with error

Example: AT^MTXTUNNEL=SETCOUNTER,3,1000
 OK

AT commands related to OpenVPN:

- **AT^MTXTUNNEL=OVPNS,minutes**

Action: if the VPN > OpenVPN Server section is configured as an OpenVPN

on demand, this AT command can be used to activate OpenVPN in server mode for the specified number of minutes.

Parameters:

<minutes>: 1 ... 525600 minutes

Result:

OK: Command executed successfully
ERROR: Command executed with error

Example: AT^MTXTUNNEL=OVPNS,5

(This example would activate the VPN for 5 minutes)

- **AT^MTXTUNNEL=OVPNC,<minutes>**

Action: if the VPN > OpenVPN Client section is configured as an on-demand OpenVPN, this AT command can be used to activate OpenVPN in client mode for the specified number of minutes.

Parameters:

<minutes>: 1 ... 525600 minutes

Result:

OK: Command executed successfully
ERROR: Command executed with error

Example: AT^MTXTUNNEL=OVPNC,5

(This example would activate the VPN for 5 minutes)

AT commands related to connectivity testing:

- **AT^MTXTUNNEL=COMMAND,<timeout>,<command>**

Action: allows certain special commands to be executed within the TITAN-based equipment. Currently, only the "ping" and "traceroute" commands are supported. A timeout (in seconds) and the command to be executed can be specified.

Parameters:

<timeout>: 1 ... 60 seconds timeout

<command>: ping, traceroute

Example: AT^MTXTUNNEL=COMMAND,5,ping -c 3 8.8.8.8

(This example would execute 3 PINGs to the IP 8.8.8.8 with a timeout of 5 seconds)



webdyn
flexitron group

powered by **TITAN**

- ★ **Mobile**
 - Status
 - Basic Settings
 - Keep Online
- ★ **Ethernet**
 - Basic Settings
- ★ **Wifi**
 - Basic Settings
 - DHCP Server
- ★ **Firewall**
 - NAT
 - Authorized IPs
- ★ **Serial Settings**

▶ Other ▶ **AT Command**

AT Command:

AT Command Response:

```

AT^MTXTUNNEL=COMMAND,5,ping -c 3 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_req=1 ttl=115 time=31.7 ms
64 bytes from 8.8.8.8: icmp_req=2 ttl=115 time=50.4 ms
64 bytes from 8.8.8.8: icmp_req=3 ttl=115 time=27.5 ms

--- 8.8.8.8 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 27.553/36.572/50.461/9.967 ms
OK

```

AT commands related to MODBUS:

- **AT^MTXTUNNEL=GETMODBUS,<modbusAddress>;<addressFirstRegister>;<numWords>;<command>**

Action: Returns the value of one or more Modbus registers from a device. Please note: modbus-related parameters are separated by ";" not ",",

Parameters:

<modbusAddress>: Modbus device address (1 ... 255) or IP address@dir:port

<addressFirstRegister>: address of the first register to be read (0 ... 65535)

<numWords>: number of Modbus registers to read (1 ... 64)

<command>: Modbus write command (5,6,15,16)

Result:

OK: Command executed correctly. Will return the words read separated by ","

ERROR: Command executed with error

Example: AT^MTXTUNNEL=GETMODBUS,192.168.1.200:502;1;1;3;3

sent from the web configuration environment (although it could also be sent by SMS or Remote Console (Telnet)), obtaining the values 20,21,22 as a response

- **AT^MTXTUNNEL=SETMODBUS,<modbusAddress>;<addressFirstRegister>;<command>;<data1>;<data2>;< ...dataX>**

Action: Sets the value of one or more Modbus registers on a device. Please note: Modbus-related parameters are separated by ";" not ",",

Result:

OK: Command executed correctly.
ERROR: Command executed with error.

Parameters:

<modbusAddress>: Modbus device address (1 ... 255) or IP@dir:port address

<addressFirstRegister>: address of the first register to be written (0 ... 65535)

<command>: Modbus write command (5,6,15,16)

-<data1>, .. <dataX>: values of the Modbus registers to be written (0 ... 65535)

Example: AT^MTXTUNNEL=SETMODBUS,1;3;16;10;11;12;13;14;15

Write the values 10, 11, 12, 13, 14, and 15 to the Modbus RTU device with address 1, starting at register 3 and using Modbus write command 16.

Example: AT^MTXTUNNEL=SETMODBUS,192.168.1.202@1:502;3;16;10;11;12;13;14;15

writes to the Modbus TCP device with IP address 192.168.1.202 using RTU address @1 and TCP port 502, starting at register 3 and using Modbus write command 16, values 10,11,12,13,14 and 15.

Example: AT^MTXTUNNEL=SETMODBUS,1;3;6;10

writes to the Modbus RTU device with address 1, in register 3 and using Modbus write command 6, the value 10

Example: AT^MTXTUNNEL=SETMODBUS,1;18;5;1

Write the value 1 to the Modbus RTU device with address 1, coil 18, using the Modbus write command 5.

Example: AT^MTXTUNNEL=SETMODBUS,1;25;15;1;0;1;0;1

writes to the Modbus RTU device with address 1, starting from coil 25 and using Modbus write command 15, the value of coils 1,0,1,0 and 1

AT commands related to system time:

- **AT^MTXTUNNEL=GETTIME**

Action: Returns the current system time in YYYY-MM-DDTHH:NN:SSZ (UTC) format

Result:

OK: Command executed successfully.
ERROR: Command executed with error

Example:

```
AT^MTXTUNNEL=GETTIME
2016-21-05T10:56:52Z
OK
```

- **AT^MTXTUNNEL=SETTIME,<dateAndHour>**

Action: Sets the current time.

Parameters:

<dateAndHour> date/time in UTC format YYYY-MM-DDTHH:NN:SSZ

Result:

OK: Command executed successfully
ERROR: Command executed with error

Example:

```
AT^MTXTUNNEL=SETTIME,2023-01-10T14:42:23Z OK
```

AT commands related to Titan device configuration:

- **AT^MTXTUNNEL=GETPARAM,<paramName>**

Action: allows you to read the value of any configuration parameter of TITAN-based equipment. For example, you can check the configuration of each parameter from a web platform or from a device connected to the TITAN's Ethernet or Wi-Fi port. If you double-click on any field on any TITAN configuration page, you will get the name of the field.

Parameters:

<paramName> name of the configuration parameter to be read

Result:

OK: Correct configuration parameter read

ERROR: Error reading the correct configuration parameter

Example:

```
AT^MTXTUNNEL=GETPARAM,WAN_APN
m2m.movistar.es
OK
```

- **AT^MTXTUNNEL=SETPARAM,<paramName>,<paramValue>**

Action: allows you to change the value of any configuration parameter on TITAN-based equipment. For example, you can change the configuration of almost any equipment parameter from a web platform or from a device connected to the Ethernet or Wi-Fi port. If you double-click on any field on any configuration page of the TITAN device, you will see the name of the field. To see the name of the field by double-clicking, you must first send the AT command:

```
AT^MTXTUNNEL=SETPARAM,MTX_SHOWNAMES,1
```

192.168.1.2 dice
Name: WAN_APN (Value: movistar.es)
Aceptar

flexitron group

Mobile Basic Settings

Mobile WAN: Enabled (IP active) Enable Wireless WAN interface

Sim Mode: SIM1 + SIM2 (backup) Sim selection

SIM1 APN: movistar.es APN of SIM card 1

SIM1 Username: MOVISTAR Username of SIM card 1

SIM1 Password: Password of SIM card 1

SIM1 Pin: PIN of SIM card 1

SIM2 APN: movistar.es APN of SIM card 2

SIM2 Username: MOVISTAR Username of SIM card 2

SIM2 Password: Password of SIM card 2

SIM2 Pin: PIN of SIM card 2

Authentication: Auto Authentication method

Parameters:

<paramName> name of the parameter of configuration to modify
<paramValue> New configuration parameter value

Result:

OK: The configuration parameter value was successfully changed.
ERROR: Error changing the configuration parameter value.

Example:

AT^MTXTUNNEL=SETPARAM,WAN_APN,m2m.movistar.es OK

- **AT^MTXTUNNEL=SETFILE,<fileName>,<length>,<{data_base64}>**

Action: Just as the AT^MTXTUNNEL=SETPARAM command allows you to change the value of a configuration parameter using an AT command, this parameter allows you to change the configuration of file-type parameters (scripts, certificates, operators, etc.). The file name appears next to the corresponding configuration boxes.

Parameters:

<fileName> file name
<length> data length of the data_base64 field, not counting the { }
<{data_base64}>: file data encoded in base64. The text must be indicated between { }

Result:

OK: The file was uploaded successfully.
 ERROR: Error executing the command.

Example:

For example, if you want to modify the file "operators.mtx" located in the Mobile section→ Basic Settings→ Operators



The AT command to be sent would be:

```
AT^MTXTUNNEL=SETFILE,operators.mtx,220,{MjE0MDY6YWlydGVsbmV0LmVzCjlxNDA3Om1vdml  
zdGFyLmVzLE1PVkITVEFSLE1PVkITVEFSCjlxNDA4OmludGVybmV0LmV1c2thbHRlbc5tb2JpCjlxN  
DE2OmludGVybmV0LnRlbgVjYWJsZS5lcwoyMTQxNzppbnRlcm5ldC5tdW5kby1yLmNvbQoyMTQxO  
DppbnRlcm5ldC5vbW8uY29tCg==}
```

Note: Some useful links for your tests are as follows: <https://www.base64encode.org/>
<https://wordcounter.net/character-count>

- **AT^MTXTUNNEL=GETFILEHASH,<fileName>**

Action: Returns the HASH of a file stored within the device.

Parameters:

<fileName> file name

Result:

{<HASH>}OK: HASH of the file.
 ERROR: Error executing the command.

Example:

For example, if you want to check the HASH of the file "operators.mtx"

```
AT^MTXTUNNEL=GETFILEHASH,operators.mtx
```

```
{06FD031D0CE8B58EA41A4E282047818E0EB74EFEEF3097F719968571DD7D8BC3} OK
```

AT commands related to IEC-60870-5-102 meters

- **AT^MTXTUNNEL=SETIEC102,<IDMETER>**

Action: By executing this command, it is possible to force the start of reading real-time values from a meter with the IEC-60870-5-102 protocol. The service in the "External Devices→ IEC102 Meter" section must be active and correctly configured.

Parameters:

: meter identifier. Tag configured in the "External Devices
→ IEC102 Meter" section or in the code of a script using the mtz. iec102AddMeter function

Result:

OK: The process has started.
ERROR: Error starting the process
BUSY: Serial bus busy with a CSD call, IP-RS232 gateway

Example:

```
AT^MTXTUNNEL=SETIEC102,ID0001
OK
```

- **AT^MTXTUNNEL=GETIEC102,<IDMETER>**

Action: read the result of executing the command AT^MTXTUNNEL=SETIEC102,<IDMETER>. The service in the "External Devices→ IEC102 Meter" section must be active and correctly configured.

Parameters:

: meter identifier. Tag configured in the "External Devices" section.
→ IEC102 Meter" or in the code of a script using the mtz. iec102AddMeter function

Result:

OK: Correct reading
ERROR: Error in reading or reading process not yet complete.

Example:

```
AT^MTXTUNNEL=GETIEC102,ID0001
{"IMEI":"869101054255506","TYPE":"IEC102","TS":"2023-01-
26T09:22:37Z","P":"MTX1","ID":"ID0001","VABA":0,"VABRI":0,"VABRC":0,"PAT":0,"PRT":0,"F
PT":1000,"PAF1":0,"PRF1":0,"FPF1":1000,"PAF2":0,"PRF2":0,"FPF2":1000,"PAF3":0,"PRF3":
0,"FPF3":1000,"IF1":0,"TF1":1156,"IF2":0,"TF2":1141,"IF3":0,"TF3":1} OK
```

- **AT^MTXTUNNEL=SETIEC102_CTAVM2, <IDMETER>, <hourIni>, <minuteIni>, <dayIni>, <monthIni>, <yearIni>, <hourEnd>, <minuteEnd>, <dayEnd>, <monthEnd>, <yearEnd>**

Action: By executing this command, it is possible to force the start of reading the fiscal locks (contract I) of a meter with IEC-60870-5-102 protocol. The service in the "External Devices→ IEC102 Meter" section must be active and correctly configured.

Parameters:

: meter identifier. Tag configured in the "External Devices→ IEC102 Meter" section or in the code of a script using the mtz function.

iec102AddMeter
 <hourIni>: start time of the interval (0 ... 23)
 <minuteIni>: starting minute of the interval (0 ... 59)
 <dayIni>: starting day of the interval (1 ... 31)
 <monthIni>: initial month of the interval (1 ... 12)
 <yearIni>: starting year of the interval (20 ... 99)
 <hourEnd>: end time of the interval (0 ... 23)
 <minuteEnd>: end minute of the interval (0 ... 59)
 <dayEnd>: end day of the interval (1 ... 31)
 <monthEnd>: end month of the interval (1 ... 12)
 <yearEnd>: end year of the interval (20 ... 99)

Result:

OK: The process has started.
 ERROR: Error starting the process
 BUSY: Serial bus busy with a CSD call, IP-RS232 gateway

Example:

AT^MTXTUNNEL=SETIEC102_CTAVM2,ID001,0,0,22,12,22,0,0,24,1,23 OK

• **AT^MTXTUNNEL=GETIEC102_CTAVM2,<IDMETER>**

Action: read of result of the execution of the command
 AT^MTXTUNNEL=SETIEC102_CTAVM2,<IDMETER>. The service in the "External Devices
 →IEC102 Meter" section must be active and correctly configured.

Parameters:

: meter identifier. Tag configured in the "External Devices" section.
 →IEC102 Meter" or in the code of a script using the mtz. iec102AddMeter function

Result:

OK: Correct reading
 ERROR: Error in reading or reading process not yet complete.

Example:

AT^MTXTUNNEL=GETIEC102_CTAVM2,ID001
 {"IMEI":"869101054255506","TYPE":"IEC102_CTAVM2","TS":"2023-01-
 26T09:41:29Z","P":"MTX1","ID":"ID001","CTAVM2":{"DO":20,"EaA":0,"EiA":0,"CA":2,"EaRi":0,"EiRi":0,
 "CRi":2,"EaRc":0,"EiRc":0,"CRc":2,"R7":0,"C7":128,"R8":0,"C8":128,"MPA":0,"FMPA":"2023-01-
 09T12:58-0","CMA":2,"EPA":0,"CE":128,"DINI":"2022-12-01T00:00-0","DEND":"2023-01-09T12:58-
 0"},"DO":21,"EaA":0,"EiA":0,"CA":2,"EaRi":0,"EiRi":0,"CRi":2,"EaRc":0,"EiRc":0,"CRc":2,"R7":0,"C7":1
 28,"R8":0,"C8":128,"MPA":0,"FMPA":"2023-01-09T12:58-0","CMA":2,"EPA":0,"CE":0,"DINI":"2022-12-
 01T00:00-0","DEND":"2023-01-09T12:58-
 0"},"DO":22,"EaA":0,"EiA":0,"CA":0,"EaRi":0,"EiRi":0,"CRi":0,"EaRc":0,"EiRc":0,"CRc":0,"R7":0,"C7":1
 28,"R8":0,"C8":128,"MPA":0,"FMPA":"2023-01-09T12:58-0","CMA":0,"EPA":0,"CE":0,"DINI":"2022-12-
 01T00:00-0","DEND":"2023-01-09T12:58-
 0"},"DO":23,"EaA":0,"EiA":0,"CA":0,"EaRi":0,"EiRi":0,"CRi":0,"EaRc":0,"EiRc":0,"CRc":0,"R7":0,"C7":1
 28,"R8":0,"C8":128,"MPA":0,"FMPA":"2023-01-09T12:58-0","CMA":0,"EPA":0,"CE":0,"DINI":"2022-12-
 01T00:00-0","DEND":"2023-01-09T12:58-
 0"},"DO":24,"EaA":0,"EiA":0,"CA":0,"EaRi":0,"EiRi":0,"CRi":0,"EaRc":0,"EiRc":0,"CRc":0,"R7":0,"C7":1
 28,"R8":0,"C8":128,"MPA":0,"FMPA":"2023-01-09T12:58-0","CMA":0,"EPA":0,"CE":0,"DINI":"2022-12-
 01T00:00-0","DEND":"2023-01-09T12:58-
 0"},"DO":25,"EaA":0,"EiA":0,"CA":0,"EaRi":0,"EiRi":0,"CRi":0,"EaRc":0,"EiRc":0,"CRc":0,"R7":0,"C7":1
 28,"R8":0,"C8":128,"MPA":0,"FMPA":"2023-01-09T12:58-0","CMA":0,"EPA":0,"CE":0,"DINI":"2022-12-

```
01T00:00-0","DEND":"2023-01-09T12:58-
0"},"DO":26,"EaA":0,"EiA":0,"CA":0,"EaRi":0,"EiRi":0,"CRi":0,"EaRc":0,"EiRc":0,"CRc":0,"R7":0,"C7":1
28,"R8":0,"C8":128,"MPA":0,"FMPA":"2023-01-09T12:58-0","CMA":0,"EPA":0,"CE":0,"DINI":"2022-12-
01T00:00-0","DEND":"2023-01-09T12:58-0"}} OK
```

- **AT^MTXTUNNEL=SETIEC102_CCINX2, <IDMETER>, <hourIni>, <minuteIni>, <dayIni>, <monthIni>, <yearIni>, <hourEnd>, <minuteEnd>, <dayEnd>, <monthEnd>, <yearEnd>, <absoluteIncremental>, <period>**

Action: By executing this command, it is possible to force a readout of the integrated totals (load curve) of a meter with IEC-60870-5-102 protocol. The service in the "External Devices→ IEC102 Meter" section must be active and correctly configured.

Parameters:

: meter identifier. Tag configured in the "External Devices" section.
 Devices→ IEC102 Meter" section or in the code of a script using the `mtx.iec102AddMeter` function
 <hourIni>: start time of the interval (0 ... 23)
 <minuteIni>: initial minute of the interval (0 ... 59)
 <dayIni>: initial day of the interval (1 ... 31)
 <monthIni>: initial month of the interval (1 ... 12)
 <yearIni>: start year of the interval (20 ... 99)
 <hourEnd>: end time of the interval (0 ... 23)
 <minuteEnd>: end minute of the interval (0 ... 59)
 <dayEnd>: final day of the interval (1 ... 31)
 <monthEnd>: final month of the interval (1 ... 12)
 <yearEnd>: final year of the interval (20 ... 99)
 <absoluteIncremental>: 0=absolute, 1=incremental
 <period>: 0=hourly, 1=quarterly

Result:

OK: The process has started.
 ERROR: Error starting the process
 BUSY: Serial bus busy with a CSD call, IP-RS232 gateway

Example:

```
AT^MTXTUNNEL=SETIEC102_CCINX2,ID001,15,0,19,1,23,20,0,19,1,23,1,0 OK
```

- **AT^MTXTUNNEL=GETIEC102_CCINX2,<IDMETER>**

Action: reading the result of executing the command
 AT^MTXTUNNEL=SETIEC102_CCINX2,<IDMETER>. The service in the "External Devices
 →IEC102 Meter" section must be active and correctly configured.

Parameters:

: meter identifier. Tag configured in the "External Devices
 →IEC102 Meter" section or in the code of a script using the `mtx. iec102AddMeter` function

Result:

OK: Correct reading
 ERROR: Error in reading or reading process not yet complete.

Example:

AT^MTXTUNNEL=GETIEC102_CCINX2,ID001

```
{
  "IMEI": "869101054255506",
  "TYPE": "IEC102_CCINX2",
  "TS": "2023-01-26T09:54:10Z",
  "P": "MTX1",
  "ID": "ID001",
  "CCINX2": {
    "DO": 11,
    "TI1": 0,
    "TI2": 0,
    "TI3": 0,
    "TI4": 0,
    "TI5": 0,
    "TI6": 0,
    "TI7": 0,
    "TI8": 0,
    "Q1": 0,
    "Q2": 0,
    "Q3": 0,
    "Q4": 0,
    "Q5": 0,
    "Q6": 0,
    "Q7": 128,
    "Q8": 128,
    "DATE": "2023-01-19T15:00-00",
    "DO": 11,
    "TI1": 0,
    "TI2": 0,
    "TI3": 0,
    "TI4": 0,
    "TI5": 0,
    "TI6": 0,
    "TI7": 0,
    "TI8": 0,
    "Q1": 0,
    "Q2": 0,
    "Q3": 0,
    "Q4": 0,
    "Q5": 0,
    "Q6": 0,
    "Q7": 128,
    "Q8": 128,
    "DATE": "2023-01-19T16:00-00",
    "DO": 11,
    "TI1": 0,
    "TI2": 0,
    "TI3": 0,
    "TI4": 0,
    "TI5": 0,
    "TI6": 0,
    "TI7": 0,
    "TI8": 0,
    "Q1": 0,
    "Q2": 0,
    "Q3": 0,
    "Q4": 0,
    "Q5": 0,
    "Q6": 0,
    "Q7": 128,
    "Q8": 128,
    "DATE": "2023-01-19T17:00-00",
    "DO": 11,
    "TI1": 0,
    "TI2": 0,
    "TI3": 0,
    "TI4": 0,
    "TI5": 0,
    "TI6": 0,
    "TI7": 0,
    "TI8": 0,
    "Q1": 0,
    "Q2": 0,
    "Q3": 0,
    "Q4": 0,
    "Q5": 0,
    "Q6": 0,
    "Q7": 128,
    "Q8": 128,
    "DATE": "2023-01-19T18:00-00",
    "DO": 11,
    "TI1": 0,
    "TI2": 0,
    "TI3": 0,
    "TI4": 0,
    "TI5": 0,
    "TI6": 0,
    "TI7": 0,
    "TI8": 0,
    "Q1": 0,
    "Q2": 0,
    "Q3": 0,
    "Q4": 0,
    "Q5": 0,
    "Q6": 0,
    "Q7": 128,
    "Q8": 128,
    "DATE": "2023-01-19T19:00-00",
    "DO": 11,
    "TI1": 0,
    "TI2": 0,
    "TI3": 0,
    "TI4": 0,
    "TI5": 0,
    "TI6": 0,
    "TI7": 0,
    "TI8": 0,
    "Q1": 0,
    "Q2": 0,
    "Q3": 0,
    "Q4": 0,
    "Q5": 0,
    "Q6": 0,
    "Q7": 128,
    "Q8": 128,
    "DATE": "2023-01-19T20:00-00"
  }
}
```

OK

Where DO: object address

Tlx, where x=1...8, the integrated totals, where:

Dirección	Objeto de Información
1	Totales Integrados de Activa Entrante.
2	Totales Integrados de Activa Saliente
3	Totales Integrados de Reactiva primer cuadrante
4	Totales Integrados de Reactiva segundo cuadrante
5	Totales Integrados de Reactiva tercer cuadrante
6	Totales Integrados de Reactiva cuarto cuadrante
7	Datos de reserva 1
8	Datos de reserva 2

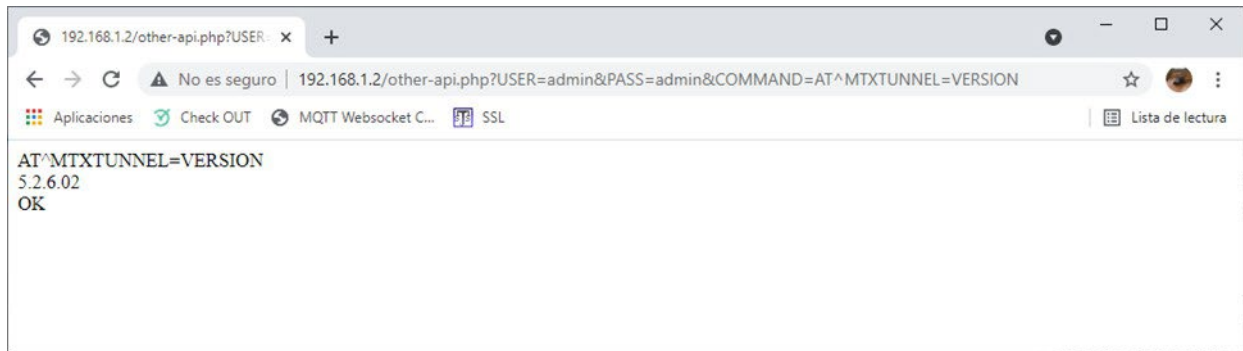
Qx, where x=1...8, the qualifiers, where:

Bit	Iden.	Descripción
7	IV	La lectura es válida (IV=0)
6	CA	Contador sincronizado durante el período (CA=1)
5	CY	Overflow (CY=1)
4	VH	Verificación horaria durante el período (VH=1)
3	MP	Modificación de parámetros durante el período (MP=1)
2	INT	Se produjo un intrusismo durante el período (INT=1)
1	AL	Período incompleto por fallo de alimentación en el período (AL=1)
0	RES	Reserva. Nótese su empleo en Tarificación.

For all other standard commands, please refer to the AT command manual for the internal GSM module of the TITAN-based device for information on the format and exact functionality of each of the module's commands. If you do not have the documentation, please request it at contact@webdyn.com

In addition to sending AT commands via SMS, Telnet, Modbus TCP, etc., it is possible to send AT commands via HTTP GET both locally and remotely. For example, to execute a command to read the temperature of the GSM module, simply make a call as follows:

`http://192.168.1.2/other-api.php?USER=admin&PASS=admin&COMMAND=AT^MTXTUNNEL=VERSION`



6 New features in firmware versions

- 6.17 - Possibility of autonomous reading of load curves for IEC-60870-5-102 meters from a script. New SCRIPT functions `mtx.iec102SetIntegratedTotal` and `mtx.iec102GetIntegratedTotal`.
- New example SCRIPT 52, showing how to read the load curves of an IEC-60870-5-102 meter.
 - New commands `AT^MTXTUNNEL=SETIEC102_CCINX2` and `AT^MTXTUNNEL=GETIEC102_CCINX2` for reading load curves for IEC-60870-5-102 meters
- 6.18 - Improvements in the management of TCP Client connections in the event of context loss.
- 6.19.- - MAC filtering from the "Firewall→ Mac Filter" section
- IEC60870-5.102. Added parameters (`patDir`, `paf1Dir`, `paf2Dir`, `paf3Dir`) to specify imported/exported energy in the JSON of instantaneous energy values.
 - IEC60870-5.102. New configuration parameter in the "External Devices→ Meter 102" section that allows assigning a sign to the parameters `pat/fpt`, `paf1/fpf1`, `paf2/fpf2`, `paf3/fpf3`
 - Aesthetic improvements to the Titan Scripts programming screen
 - HTTP PUT and HTTPS PUT methods added to the LOGGER and SCRIPTS sending options.
 - Modification of SCRIPT example 39 including the HTTP PUT method.
 - New `mtx.loggerSendingEnabled` function for the SCRIPTS section. This function allows you to enable/disable the service for sending data stored in the LOGGER.
 - New example SCRIPT 7b. Reading a Modbus TCP log.
 - New example SCRIPT 14b. Reading modbus RTU registers, customising the sending JSON, storing in LOGGER and scheduling sending within a specific time frame.
 - New command `AT^MTXTUNNEL=GETFILEHASH, <filename>` which allows you to read the HASH of an internal configuration file.
- 6.20.- - Fixed issue where special characters `^_` could not be sent via SMS
- Improved management of SMS sending from scripts
 - In the "ID String" field of client-type IP-Serial gateways, it is possible to specify the tags [IMEI], [CR] and [LF] so that they are automatically replaced by the IMEI and the special characters 0x13 and 0x10
 - New functions for scripts: `mtx.convert2WordToUnsignedInt32`, `mtx.convert2WordToSignedInt32`, `mtx.convert2WordToFloat32`, `mtx.convert4WordToSignedInt64` useful for Modbus data conversions.

- Added the Modbus write command 0x06
 - New functions for the `mtx.configParamGet` and `mtx.configParamSet` scripts that allow the router configuration to be read and changed from the scripts.
 - Added example scripts 53 and 54 for reading DLMS meters
- 6.21.-
- Ability to edit Modbus devices registered in the "External Devices → Modbus Devices" section
 - Support for the new WEBDYN-EXPERT-ROUTER device
- 6.22.-
- Support for autonomous meter reading with IEC-60870-5-102 protocol via an optical probe interface.
 - Support for the new WEBDYN-EASY-ROUTER device
- 6.24.-
- New firmware update feature via OTAP.
 - New configuration options for the SIM SWAP feature for devices with DUAL SIM.
 - Option to automatically close sessions with IEC-60870-5-102 protocol for TCP Server - RS232/RS485 gateways.
- 6.25.-
- New "ZeroTier" VPN feature. New application note AN71.
 - New "Modbus Expert" modbus device reading feature. New application note AN72.
 - New feature for configurations from a PC for mass production. New application note AN73.
- 6.30
- Authentication method for HTTP (GUI), Telnet and SSH via LDAP.
 - New artificial intelligence assistant to help with script development.
 - New script functions related to chatGPT
 - New script function for using HASH SHA256
 - Application note AN74 for using TACACS+ authentication
 - Application note AN74 for using LDAP authentication.
 - Application note AN75 explaining the artificial intelligence tool "Titan Script Assistant".
 - New sample scripts related to the chatGPT API for textual data and image analysis.
 - New UINT64 data type in the Modbus Expert section
 - New double word flip methods in the Modbus Expert section
 - New functions of the "mtx" object for scripts: `mtx.modbusExpertGetStatus`, `mtx.modbusExpertGetReg`, `mtx.modbusExpertSetReg`, `mtx.modbusExpertSetV`, `mtx.modbusExpertSetV`
 - New time format parameters for the `mtx.timeStamp()` function in Titan Scripts.

7 Sales contact

SPAIN

C/ Alejandro Sánchez 109
28019 Madrid

Telephone: +34.915602737
E-mail: contact@webdyn.com

FRANCE

26 Rue des Gaudines
78100 Saint-Germain-en-Laye

Telephone: +33.139042940
Email: contact@webdyn.com

INDIA

803-804 8th floor, Vishwadeep Building
District Centre, Janakpurt, 110058 Delhi

Telephone: +91.1141519011
E-mail: contact@webdyn.com

PORTUGAL

Av. Coronel Eduardo Galhardo 7-1°C
1170-105 Lisbon

Telephone: +351.218162625 E-
mail: comercial@lusomatrix.pt