



TECHNOLOGY MEETS BUSINESS

M4M Teltonika Integration

Prepared For: UP
Prepared By: Heinrich de Lange

CONFIDENTIAL

Executive Summary

This document serves as a guide to integrate Modbus meters, specifically ABB M4M30 meters with Teltonika RUT956 or TRB145 units via MQTT.

Source code for this project can be found here:

<https://github.com/heinrich321/abb-teltonika>

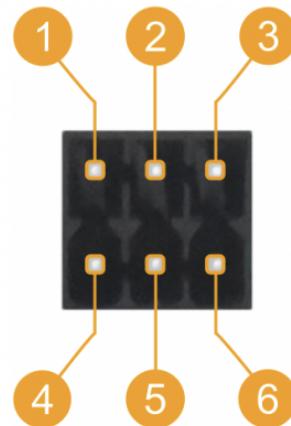
Table of Contents

Executive Summary	2
Wiring	4
ABB M4M30 Configuration	5
Communication options	6
Gateway Configuration	7
Overview	7
Teltonika Gateway Configuration	9
Command Structure	11
Example	12
Data to Server	15
Data Flow	15
Data to Server (MQTT - Decoded)	21
Overview	21
Example MQTT Messages	21
Topic	21
Payload	21
Data to Server (MQTT - Hex)	22
Overview	22
Topic	22
Payload	22

Wiring

The configuration used is **not** full duplex and therefore the same 2 wires are used as transmit and receive wires. This requires the Negative Driver & Receiver to be bridged as well as the Positive Driver and Receiver.

RS485 connector pinout		
Pin	Name	Description
1	D_N	Driver negative signal
2	R_N	Receiver negative signal
3	GND	Device ground
4	D_P	Driver positive signal
5	R_P	Receiver positive signal
6	NC	Power input 9-30 VDC



Bridging RS485 connector pins 1 & 2 as well as pin 4 & 5 will result in the below configuration:

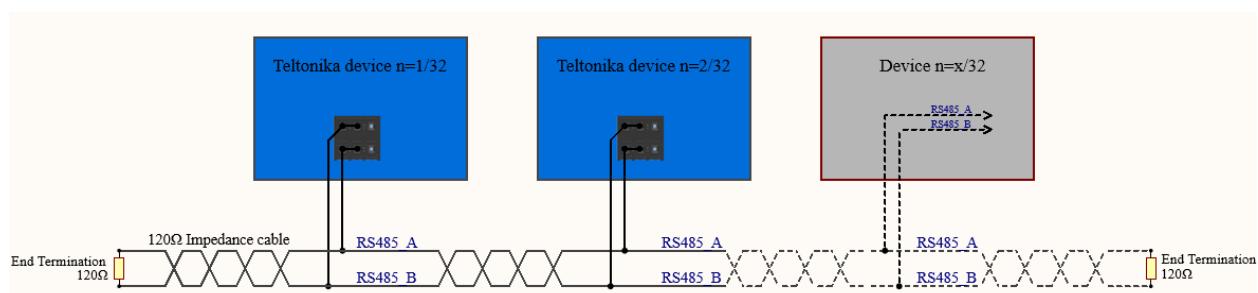


ABB M4M30 Configuration

Ensure that the communication settings on the M4M30 device align with the Teltonika settings. Navigate to the touchscreen interface to configure these parameters. For this integration guide, pay attention to the following crucial settings and their corresponding values:

- Baud Rate: 38400
- Parity: Even
- Bits: 8
- Stop Bit: 1

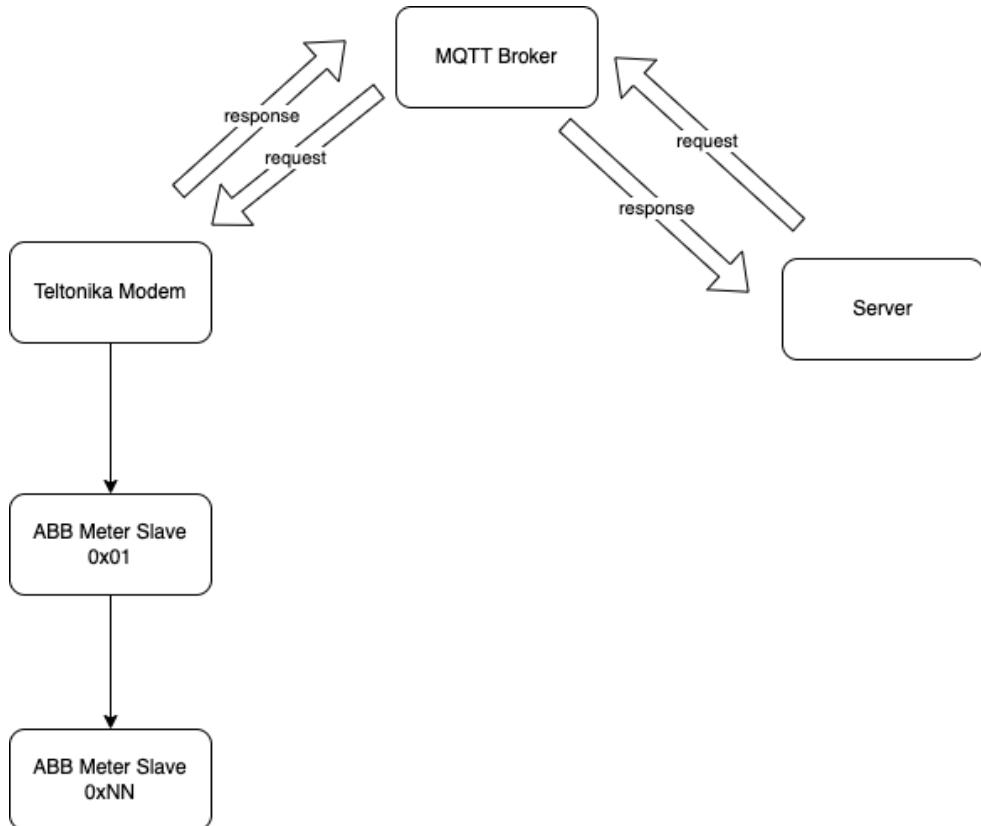
Communication options

There are two main ways of configuring communication between an RS485 Modbus RTU device and an MQTT client. The first option is the “Gateway” configuration. This configuration allows Modbus commands to be sent by an external MQTT client and is then executed by the Teltonika device and the response is sent back.

The second option is to use the data-to-server functionality in combination with a running Modbus Serial Client on the Teltonika device. In this configuration the Teltonika device periodically reads all of the registers configured from the slave devices and then periodically publishes this data to an MQTT topic for MQTT clients to consume.

Gateway Configuration

Overview



The gateway configuration creates a bridge over MQTT between the server and the Modbus devices. Requests are sent to the /request topic and responses are sent back on the /response topic. These topics are configurable.

This configuration can be used if it is not known what parameters should be read in advance and transfers the logic for reading & decoding the Modbus packets to the server side.

An example of this implementation can be found here :
<https://github.com/heinrich321/abb-teltonika>

The gateway_client.py example will poll all the registers configured in m4m30.py at a set interval over MQTT and decode the responses.

OCTOCO

www.OCTOCO.ltd

OCTOCO CONSULTING (PTY) LTD.
VAT REG. NUMBER: 4110293455

Teltonika Gateway Configuration

The screenshot displays the OCTOCO Teltonika Gateway Configuration interface. On the left, a sidebar lists various services: Status, Network, Services (with Modbus, Modbus TCP Server, Modbus Serial Server, Modbus TCP Client, Modbus Serial Client, and MQTT Modbus Gateway highlighted with a red oval), and System (Modbus TCP over Serial Gateway and Events Reporting). The main content area shows two configuration sections:

- MQTT GATEWAY**: Settings include Host (broker.hivemq.com), Port (1883), Request topic (m4m-example/request), Response topic (m4m-example/response), QoS (Exactly once (2)), Username (Username), Password (Password), Client ID (teltonikam4m), Keepalive (5), and Use TLS/SSL (off).
- SERIAL GATEWAY CONFIGURATION**: Settings include Password (Password), Client ID (teltonikam4m), Keepalive (5), and Use TLS/SSL (off).

Below these, a table lists a single instance (Device ID 1, Device rs485, off) with edit and delete icons. A new instance can be added via the "ADD NEW INSTANCE" section, which includes fields for DEVICE ID (circled in red) and DEVICE NAME (rs232), with an ADD button (circled in red) and a SAVE & APPLY button.

Server

Modbus TCP Client

Modbus Client

MQTT Gateway

Modbus Serial

Events

RS485 DEVICES SERIAL GATEWAY CONFIGURATION

Client ID: teltonikam4m

Enable: off on

Device: rs485

Baud rate: 38400

Data bits: 8

Stop bits: 1

Parity: Even

Flow control: None

Full Duplex: off on

SAVE & APPLY

MQTT GATEWAY

Enable: on

Host: broker.hivemq.com

Port: 1883

Request topic: m4m-example/request

Response topic: m4m-example/response

QoS: Exactly once (2)

Username: Username

Password: Password

Client ID: teltonikam4m

Keepalive: 5

Use TLS/SSL: off on

Use TLS/SSL off on**▼ SERIAL GATEWAY CONFIGURATION**

DEVICE ID	DEVICE	
1	rs485	<input checked="" type="checkbox"/> off <input type="checkbox"/> on  

▼ ADD NEW INSTANCE

DEVICE ID	DEVICE NAME	 
<input type="text"/>	<input type="text" value="rs232"/> ▼	

Command Structure

Command Structure: 1 <COOKIE> <SERIAL_DEVICE_ID> <TIMEOUT> <SLAVE_ID> <MODBUS_FUNCTION> <FIRST_REGISTER> <REGISTER_COUNT>

1. Format version	1
2. Cookie	from 0 to $2^{64} - 1$
3. Serial device ID	a string used to identify a serial device. Must match with Device ID field in MQTT Gateway page Serial gateway configuration section
4. Timeout	timeout for Modbus connection, in seconds. Range [1..999].
5. Slave ID	Indicates to which slave request is sent
6. Modbus function	Modbus task type that will be executed. Possible values are: <ul style="list-style-type: none"> • 1 - read coils; • 2 - read input coils; • 3 - read holding registers; • 4 - read input registers; • 5 - set single coil; • 6 - write to a single holding register; • 15 - set multiple coils; • 16 - write to multiple holding registers.

7. First register	number (not address) of the first register/coil/input (in range [1..65536]) from which the registers/coils/inputs will be read/written to.
8. Registry count	1 - coil count (in range [1..2000]); must not exceed the boundary (first coil number + coil count <= 65537); 2 - input count (in range [1..2000]); must not exceed the boundary (first input number + input count <= 65537); 3 - holding register count (in range [0..125]); must not exceed the boundary (first register number + holding register count <= 65537); 4 - input register count (in range [0..125]); must not exceed the boundary (first register number + input register count <= 65537); 5 - coil value (in range [0..1]); 6 - holding register value (in range [0..65535]); 15 - coil count (in range [1..1968]); must not exceed the boundary (first coil number + coil count <= 65537); and coil values separated with commas, without spaces (e.g., 1,2,3,654,21,789); there must be exactly as many values as specified (with coil count); each value must be in the range of [0..1].

Example

This example reads the frequency register:

Parameter	Unit	Resolution	Data Type	Register (Hex)	Register (Dec)	Nr of quantities	Size	Nr of registers
Frequency	Hz	0.01	Unsigned	5B32	23346	1	1	1

Example Request:

Topic: m4m-example/request

Payload: 1 1 1 1 1 3 23347 1

Example Response:

Topic: m4m-example/response

Response Payload: 1 OK 5010

MQTT GATEWAY

Enable

Host

Port

Request topic

Response topic

QoS

Username

Password 

Client ID

Keepalive

Use TLS/SSL

Use TLS/SSL

SERIAL GATEWAY CONFIGURATION

DEVICE ID

DEVICE

1

rs485



ADD NEW INSTANCE

DEVICE ID

DEVICE NAME



ADD

SAVE & APPLY

OCTOCO

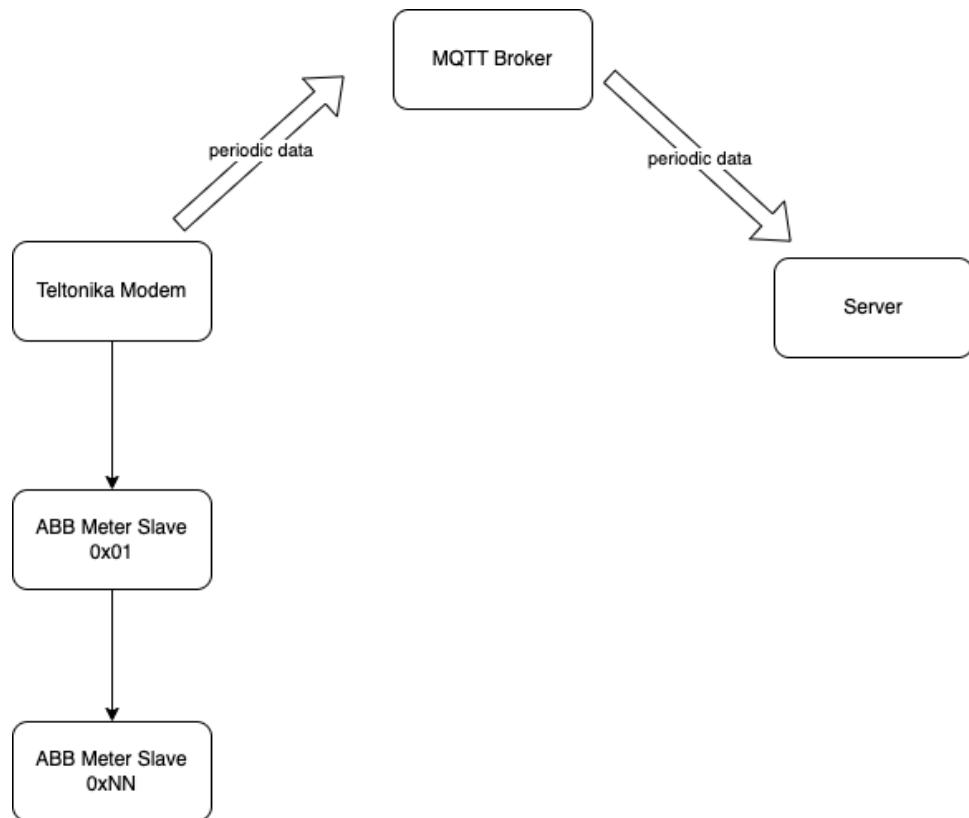
www.OCTOCO.ltd

OCTOCO CONSULTING (PTY) LTD.
VAT REG. NUMBER: 4110293455

Data to Server

Data Flow

The data to the server sends all data accumulated over a period of time to the MQTT topic at a set interval.



Navigate to Modbus Serial Client tab

The screenshot shows the Octoco software interface with the following navigation path:

- Network
- Services
- Data to Server
- Modbus** (highlighted)
- Modbus TCP Server
- Modbus Serial Server
- Modbus TCP Client
- Modbus Serial Client** (highlighted with a red circle)
- MQTT Modbus

The main configuration screen for "Modbus Serial Client" is displayed, containing the following sections:

- MODBUS CLIENT**: An "Enabled" toggle switch is set to "on".
- MODBUS SERIAL DEVICE CONFIGURATION**: A table with one row:

NAME	DEVICE
ABB	rs485

 Edit and delete icons are available for this row.
- ADD NEW INSTANCE**: Fields for "NEW CONFIGURATION NAME" (empty) and "DEVICE NAME" (set to "rs232") with an "ADD" button.
- MODBUS SERVER DEVICE CONFIGURATION**: A table with one row:

NAME	MODBUS SERIAL DEVICE	FREQUENCY	TIMEOUT
M4M	1	60	1

 Edit and delete icons are available for this row.

Add an Modbus client instance containing the settings for the slave devices

The screenshot shows the Octoco software interface with the following navigation path:

- Network
- Services
- Data to Server
- Modbus**
- Modbus TCP Server
- Modbus Serial Server
- Modbus TCP Client
- Modbus Serial Client** (highlighted with a red circle)
- MQTT Modbus Gateway
- Modbus TCP over Serial Gateway
- Events Reporting

The main configuration screen for "Modbus Serial Client" is displayed, containing the following sections:

- MODBUS SERIAL DEVICE CONFIGURATION**: A table with one row:

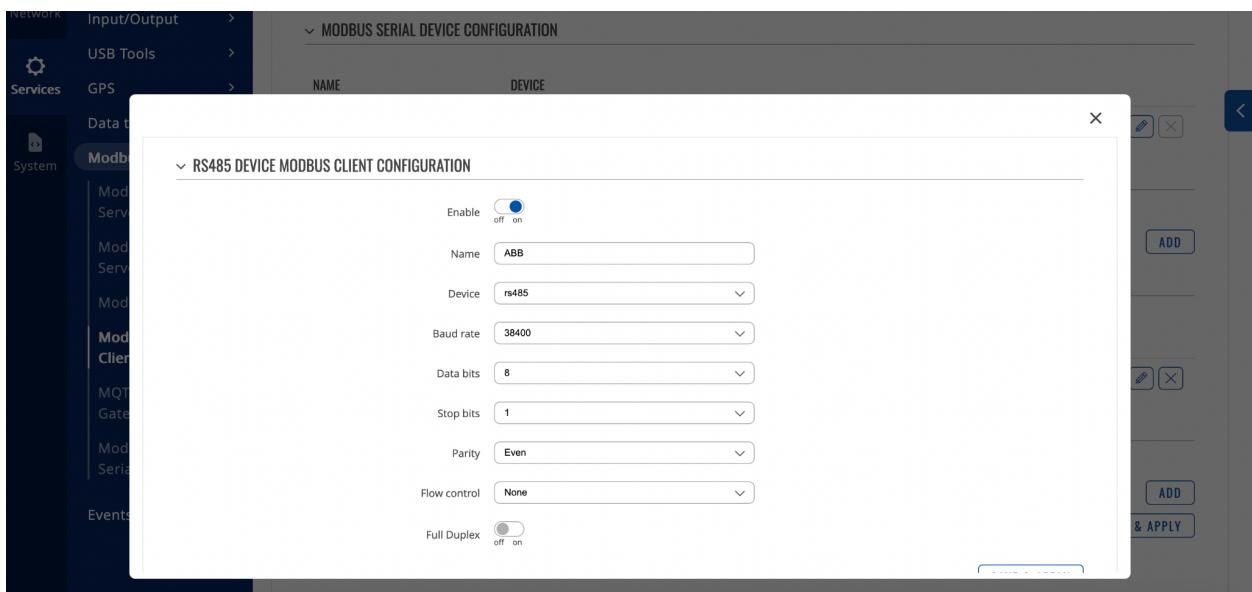
NAME	DEVICE
ABB	rs485

 Edit and delete icons are available for this row.
- ADD NEW INSTANCE**: A form with "NEW CONFIGURATION NAME" (highlighted with a red circle) and "DEVICE NAME" (set to "rs232"). An "ADD" button is highlighted with a red circle.
- MODBUS SERVER DEVICE CONFIGURATION**: A table with one row:

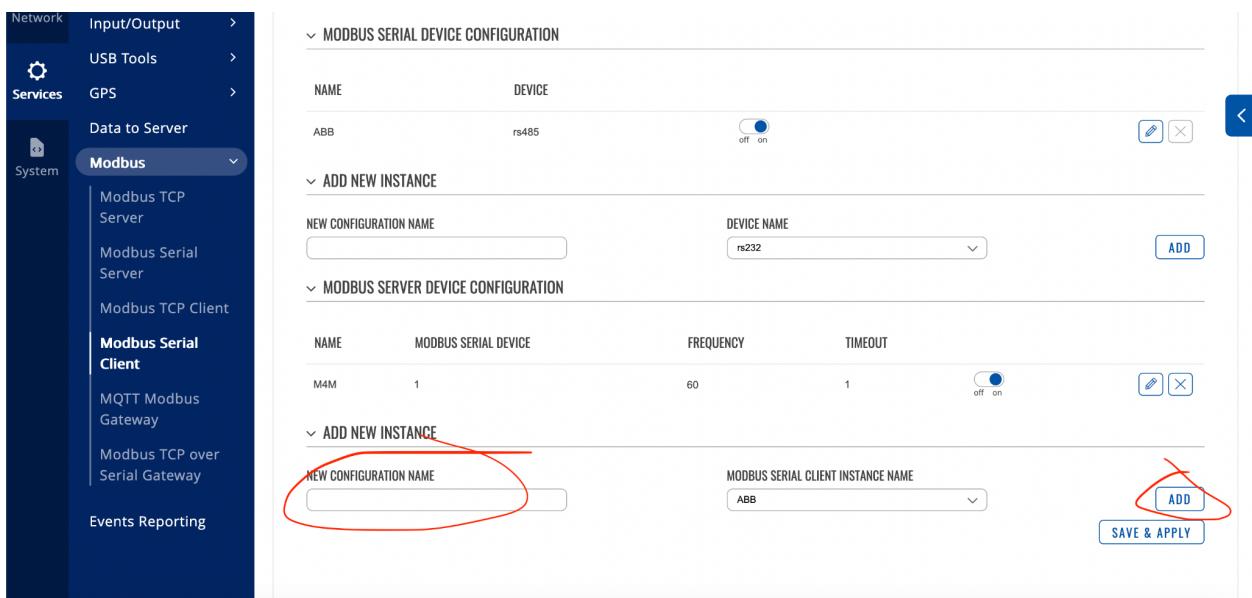
NAME	MODBUS SERIAL DEVICE	FREQUENCY	TIMEOUT
M4M	1	60	1

 Edit and delete icons are available for this row.
- ADD NEW INSTANCE**: A form with "NEW CONFIGURATION NAME" and "MODBUS SERIAL CLIENT INSTANCE NAME" (set to "ABB"). A "SAVE & APPLY" button is visible.

The configuration used for this specific ABB M4M Meter is the following:



Add an instance. This will be done for every slave device connected to the bus



Edit the configuration for the slave device

Network **Input/Output** >
Services **USB Tools** >
GPS >
Data to Server
Modbus >
System

Modbus TCP Server
Modbus Serial Server
Modbus TCP Client
Modbus Serial Client
MQTT Modbus Gateway
Modbus TCP over Serial Gateway
Events Reporting

MODBUS SERIAL DEVICE CONFIGURATION

NAME	DEVICE
ABB	rs485

ADD NEW INSTANCE

NEW CONFIGURATION NAME:
DEVICE NAME: **ADD**

MODBUS SERVER DEVICE CONFIGURATION

NAME	MODBUS SERIAL DEVICE	FREQUENCY	TIMEOUT
M4M	1	60	1

ADD NEW INSTANCE

NEW CONFIGURATION NAME:
MODBUS SERIAL CLIENT INSTANCE NAME: **ADD**

SAVE & APPLY

Configure the address of the slave, frequency of reads and timeouts

Network **Input/Output** >
Services **USB Tools** >
GPS >
Data to Server
Modbus >
Modbus Serial Client
Modbus Serial Server
Modbus TCP Client
Modbus TCP Client
MQTT Modbus Gateway
Modbus TCP over Serial Gateway
Events Reporting

MODBUS SERIAL DEVICE CONFIGURATION

NAME	DEVICE
ABB	rs485

SERVER DEVICE CONFIGURATION

Enabled: **on**
Name:
Serial device:
Server ID:
Number of timeouts:
Frequency:
Period:
Timeout:

REQUESTS CONFIGURATION

Configure the registers to be read. Pick the appropriate data type depending if you are doing server side decoding or decoding on the Teltonika device as described in the following sections.

NAME	DATA TYPE	FUNCTION	FIRST REGISTER NUMBER	REGISTER COUNT / VALUES	REMOVE BRACKETS
FREQ	Hex	Read holding registers (3)	23347	1	<input checked="" type="radio"/> off <input type="radio"/> on
ACTIVE_IMPORT	Hex	Read holding registers (3)	20481	4	<input checked="" type="radio"/> off <input type="radio"/> on
ACTIVE_EXPORT	Hex	Read holding registers (3)	20485	4	<input checked="" type="radio"/> off <input type="radio"/> on
ACTIVE_NET	Hex	Read holding registers (3)	20489	4	<input checked="" type="radio"/> off <input type="radio"/> on
REACTIVE_IMPORT	Hex	Read holding registers (3)	20493	4	<input checked="" type="radio"/> off <input type="radio"/> on
REACTIVE_EXPORT	Hex	Read holding registers (3)	20497	4	<input checked="" type="radio"/> off <input type="radio"/> on
REACTIVE_NET	Hex	Read holding registers (3)	20501	4	<input checked="" type="radio"/> off <input type="radio"/> on
APPARENT_IMPORT	Hex	Read holding registers (3)	20505	4	<input checked="" type="radio"/> off <input type="radio"/> on

Test your configuration to make sure you are getting the appropriate response from the slave.

Next navigate to the Data to Server tab to configure the data transport settings over MQTT.

The screenshot shows the Teltonika Networks interface. On the left sidebar under the 'SERVICES' section, the 'Data to Server' option is highlighted with a red circle. The main panel displays the 'DATA TO SERVER COLLECTIONS' section. It lists one collection named 'M4M' with 'Server type: MQTT'. Below it, there is a table for 'Modbus' data with 'Data type: Modbus' and 'Format type: Json'. There are edit and delete icons for each row. A '+ New data input' button is also present. The top right corner shows the device identifier 'RUT9M_R_00.07.05.4' and a 'View Settings' link.

Add a new instance and configure the MQTT topic and frequency that data should be sent to the specific topic.

This screenshot shows the detailed configuration for the 'M4M' collection. The 'COLLECTION CONFIGURATION' tab is active, displaying 'GENERAL SETTINGS' and 'ADVANCED SETTINGS' sections. Under 'GENERAL SETTINGS', the 'Enable' toggle is set to 'on', the 'Name' is 'M4M', and the 'Format type' is 'Json'. The 'SERVER CONFIGURATION' tab is also visible, showing fields for 'Type' (MQTT), 'Server address' (broker.hivemq.com), 'Port' (1883), 'Keepalive' (60), and 'Topic' (m4m-example). The top right corner shows the device identifier 'RUT9M_R_00.07.05.4' and a 'View Settings' link.

Data to Server (MQTT - Decoded)

Overview

The data to server MQTT method using pre-decoded as seen in the screenshot below:

NAME	DATA TYPE	FUNCTION	FIRST REGISTER NUMBER	REGISTER COUNT / VALUES	REMOVE BRACKETS
FREQ	16bit UINT, low byte first	Read holding registers (3)	23347	1	<input checked="" type="checkbox"/> off <input type="checkbox"/> on X

Uses functionality on the Teltonika modem to decode the Modbus response. This however has limitations such as only supporting up to 32-bit numbers. The ABB M4M meter utilizes 64-bit numbers for multiple of the parameters and as such will limit functionality. It can still be used but will need additional post-processing since the data will come through in an array as shown in the example.

Example MQTT Messages

Topic

m4m-example

Payload

```
{"Modbus":{"timestamp":1702826365,"date":"17/12/2023  
15:19:25","bdate":null,"server_id":1,"bserver_id":1,"addr":23317,"baddr":null,"full_add  
r":423317,"size":5,"data": "[0,0]","raw_data":null,"server_name":"M4M","name":"C  
URRENT_L3"}}
```

Data to Server (MQTT - Hex)

Overview

The hex data-type functionality on the Teltonika device leaves the Modbus response raw hex format and allows for more fine-grained control over post-processing.

An example of this implementation can be found here :

<https://github.com/heinrich321/abb-teltonika>

The modbus_client.py example subscribes to the data-to-server topic and parses the incoming hex data according to the register map as set out in m4m30.py.

NAME	DATA TYPE	FUNCTION	FIRST REGISTER NUMBER	REGISTER COUNT / VALUES	REMOVE BRACKETS
FREQ	Hex	Read holding registers (3)	23347	1	<input checked="" type="radio"/> off <input type="radio"/> on
ACTIVE_IMPORT	Hex	Read holding registers (3)	20481	4	<input checked="" type="radio"/> off <input type="radio"/> on
ACTIVE_EXPORT	Hex	Read holding registers (3)	20485	4	<input checked="" type="radio"/> off <input type="radio"/> on
ACTIVE_NET	Hex	Read holding registers (3)	20489	4	<input checked="" type="radio"/> off <input type="radio"/> on
REACTIVE_IMPORT	Hex	Read holding registers (3)	20493	4	<input checked="" type="radio"/> off <input type="radio"/> on
REACTIVE_EXPORT	Hex	Read holding registers (3)	20497	4	<input checked="" type="radio"/> off <input type="radio"/> on
REACTIVE_NET	Hex	Read holding registers (3)	20501	4	<input checked="" type="radio"/> off <input type="radio"/> on
APPARENT_IMPORT	Hex	Read holding registers (3)	20505	4	<input checked="" type="radio"/> off <input type="radio"/> on
APPARENT_EXPORT	Hex	Read holding registers (3)	20509	4	<input checked="" type="radio"/> off <input type="radio"/> on

Topic

m4m-example

Payload

```
{"Modbus": {"timestamp": 1702826551, "date": "17/12/2023 15:22:31", "bdate": null, "server_id": 1, "bserver_id": 1, "addr": 20513, "baddr": null, "full_add
```



r":420513,"size":18,"data":"\000000000003BCD1\","",raw_data":null,"server_name":"M4M","name":"APPARENT_NET"}]

www.OCTOCO.ltd

OCTOCO CONSULTING (PTY) LTD.
VAT REG. NUMBER: 4110293455