MODBUS TCP OVER ETHERNET

1. Introduction

Senquip devices feature an integrated Modbus RTU interface that communicates over RS232 or RS485. However, many industrial applications use Modbus TCP over Ethernet or Wi-Fi for networked communication.

Modbus TCP over Wi-Fi can be implemented on a Senquip device without the need for an adapter. This is covered in APN0024 – Modbus TCP In a Script.

This application note demonstrates how a Senquip device can interface with a remote Modbus TCP sensor using a cost-effective RS485-to-Ethernet converter. The PUSR USR-TCP232-304 RS485-to-Ethernet converter was selected for its affordability and widespread availability.

Figure 1 - PUSR USR-TSP232-304

2. Introduction to Modbus TCP 3.

2.1. What is Modbus TCP?

Modbus TCP is a widely used industrial communication protocol that enables Ethernet-based data exchange between devices such as sensors, controllers, and monitoring systems. It is an extension of the Modbus RTU protocol, which operates over RS232 or RS485, but instead utilizes TCP/IP as the transport layer.

- Key Features of Modbus TCP:
- Uses Ethernet/IP instead of serial communication.
- Follows a client-server architecture.
- Allows a single client to communicate with multiple servers (sensors, meters, or RTUs).
- Enables remote access over local networks or the internet.
- Uses port 502 as the default communication port.

2.2. Modbus TCP Client-Server Model

In Modbus TCP, devices follow a client-server model:

Client – Initiates communication by sending requests to read or write data. Examples: PLC, SCADA system, Senquip Device
Server – Holds data and responds to client requests. Examples: Sensors, meters, remote I/O modules

Most Modbus TCP sensors act as servers, waiting for a client (such as a Senquip QUAD) to request data at regular intervals. The client polls the sensor for measurements like temperature, pressure, or voltage and uses the received data for monitoring or control.

Example Communication Flow:

1. Client (Senquip QUAD) → Server (Modbus TCP Sensor): "Provide the value of register 1 (temperature reading)."
2. Server (Modbus TCP Sensor) → Client (Senquip QUAD): "Register 1 = 25°C"

This request-response cycle ensures that the client receives up-to-date sensor readings.

In some applications, a single Modbus TCP server may support multiple clients (e.g., both a SCADA system and a PLC requesting data from the same sensor). However, the most common scenario involves a single client polling multiple sensors.

2.3. Differences Between Modbus TCP and Modbus RTU

Since **Modbus TCP** runs over Ethernet, it supports multiple clients and servers, remote access, and faster communication speeds compared to **Modbus RTU**.

Feature	Modbus RTU (RS232/RS485)	Modbus TCP (Ethernet)
Physical Layer	RS232/RS485 (Serial)	Ethernet (TCP/IP)
Data Format	Binary (RTU)	TCP/IP packets
Addressing	Device ID (1-247)	IP Address
Communication	Master-Slave (single master)	Client-Server (multiple clients possible)
Speed	Typically, up to 115.2 kbps	Up to 1 Gbps or higher
Network Size	Limited (serial bus)	Scalable (IP network)

3. Connecting the RS485 to Ethernet Interface

In this example, the RS485-to-Ethernet converter was connected to the RS485 port of a Senquip QUAD, with an Ethernet cable linking the converter to a PC. The converter was powered using the included power adapter.

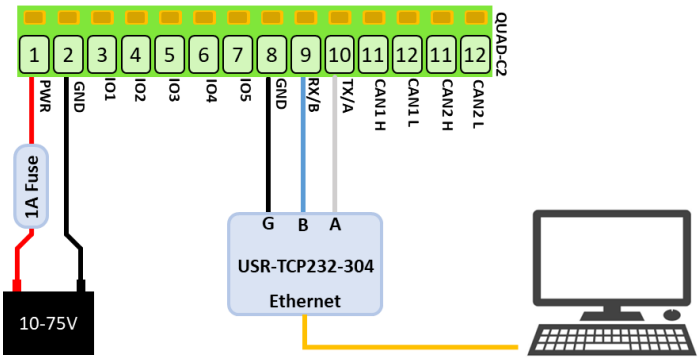


Figure 2 - RS485 to Ethernet Connections

The configuration software, USR-M0, shown in Figure 3, was downloaded from the PUSR website. The converter was assigned an available IP address (169.254.203.199) within the same subnet as the PC's Ethernet connection. The PC's

Ethernet IP address was determined using the ipconfig command from a command prompt. A static IP address must be used in this case as the PC is not running a DHCP server.

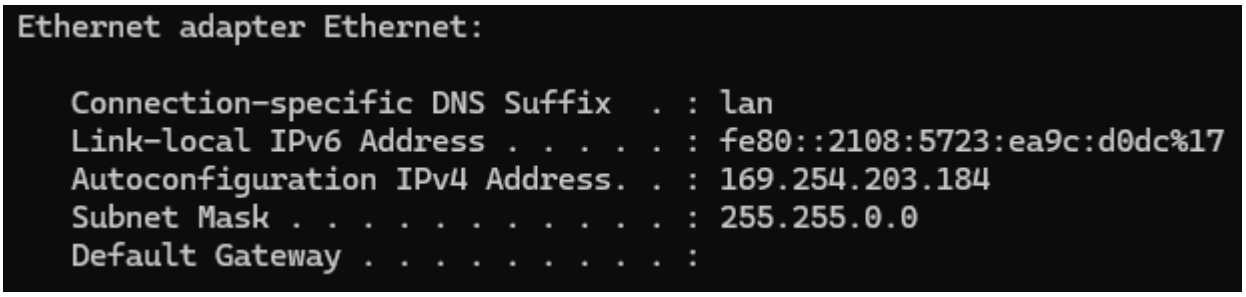


Figure 3 - PC IP Address on Ethernet Adapter

Modbus Slave was used as a Modbus TCP server to which the interface will connect.

The settings for the RS485 to ethernet interface are shown in Figure 4. The most important settings updates are:

3. The interface is setup as a TCP client.
4. The RemoteIP is set as the IP address of the PC running the Modbus Server software.
5. The port is specified as the port on which the Modbus server software will be listening.
6. The baud rate and serial port settings match the QUAD.
7. The ModBus Type is set as Modbus TCP/RTU.

The interface (shown in Figure 5) also has a built in webserver through which the device can be configured. The webserver can be accessed at the allocated IP address using username: admin and password: admin.

The settings for the Modbus Slave software are shown in Figure 6. The most important settings are:

1. Mode is set as TCP.
2. The port is set as 502 to match that configured on the interface.
3. A Slave ID of 1 is chosen.
4. Registers 1 and 2 are set as 12 and 13 respectively.

Modbus Slave offers a handy communications log in which incoming and sent data can be seen.

The serial port settings for the Senquip QUAD are shown in Figure 7 and Figure 8. The most important settings are:

1. The serial port settings must match those specified for the RS485 to ethernet interface.
2. The Mode is set to Modbus.
3. Two reads are configured for registers addresses 1 and 2 on a slave device with address 1.

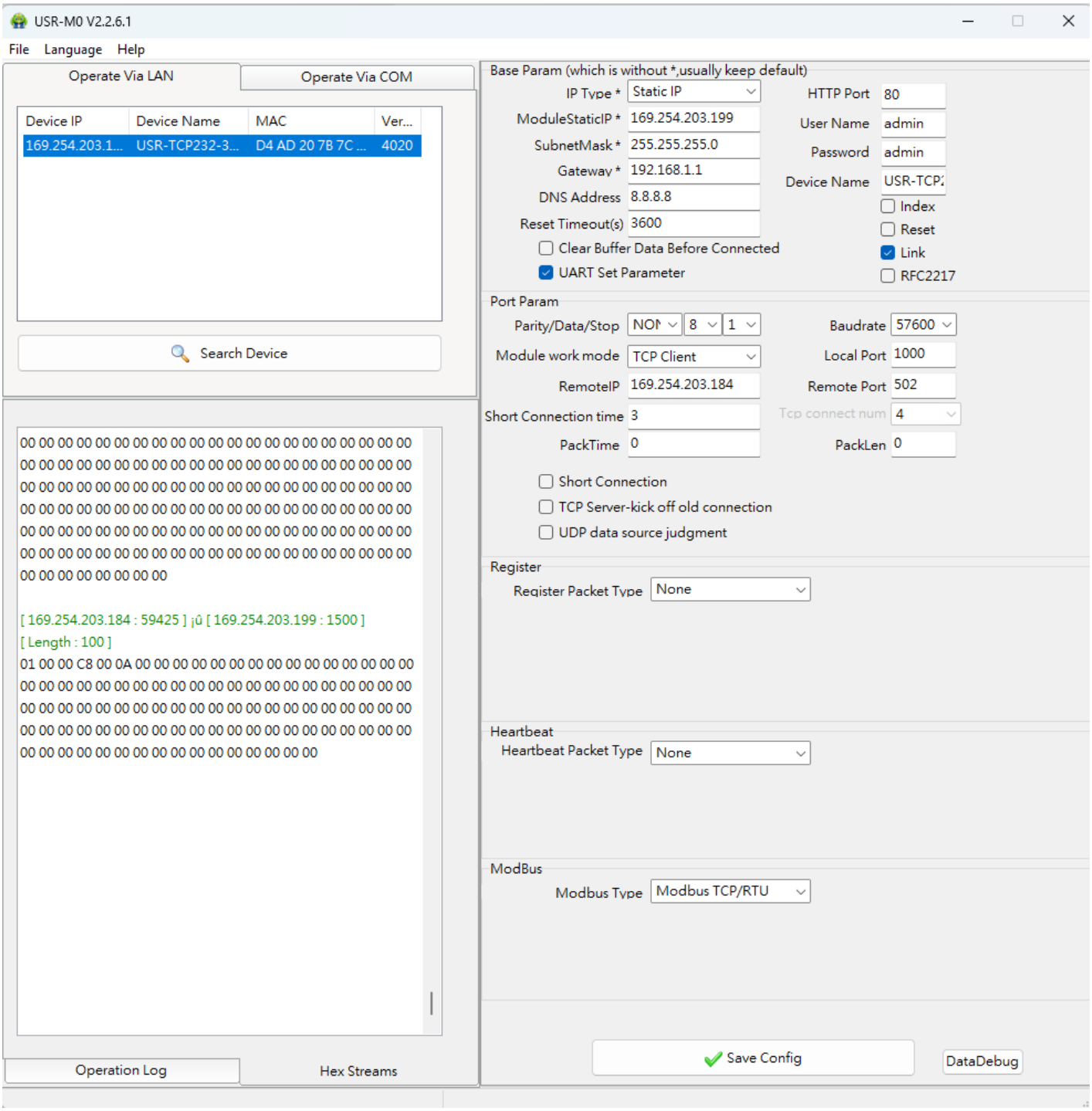


Figure 4 – RS485 to Ethernet Interface Settings

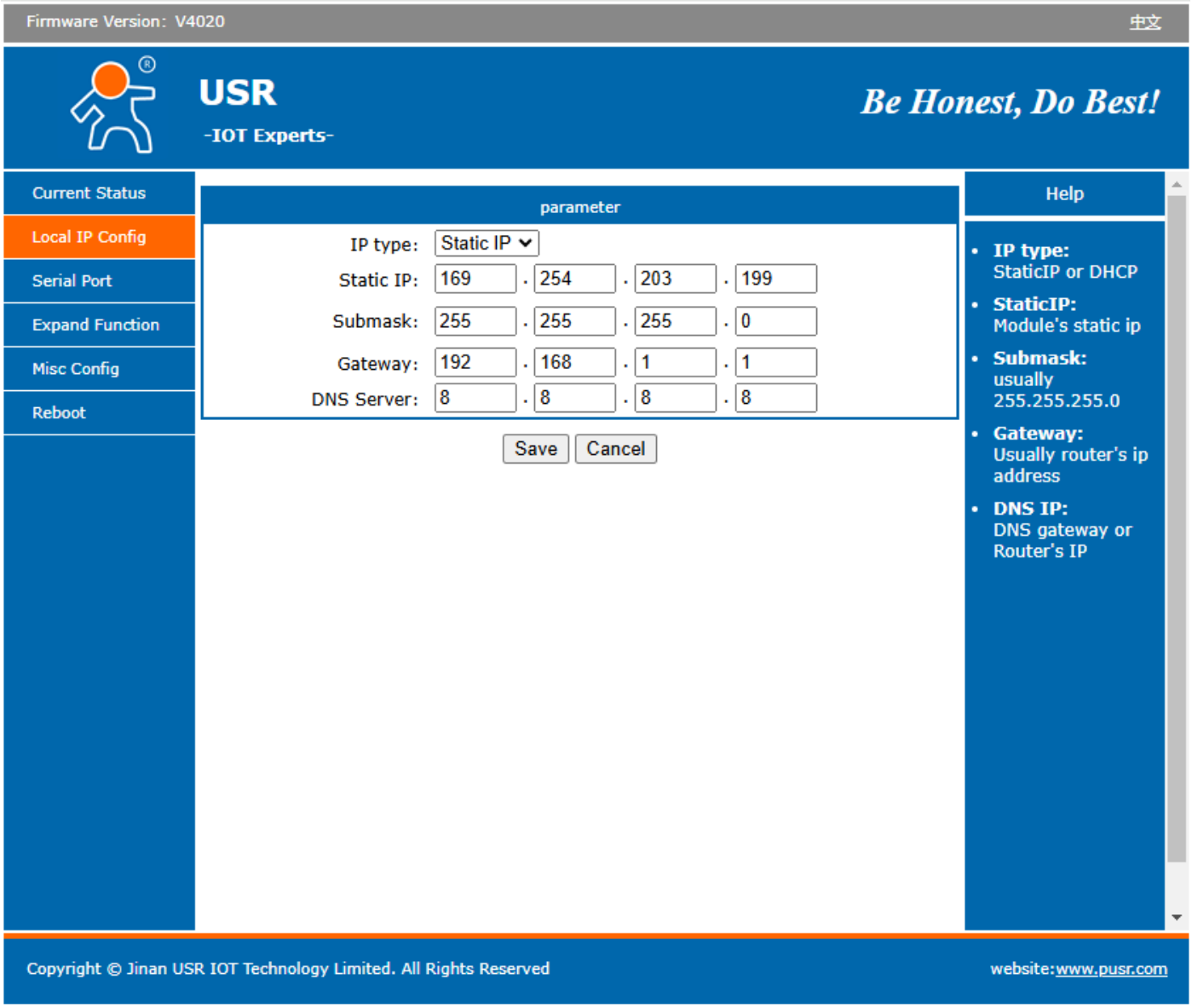


Figure 5 - Webserver

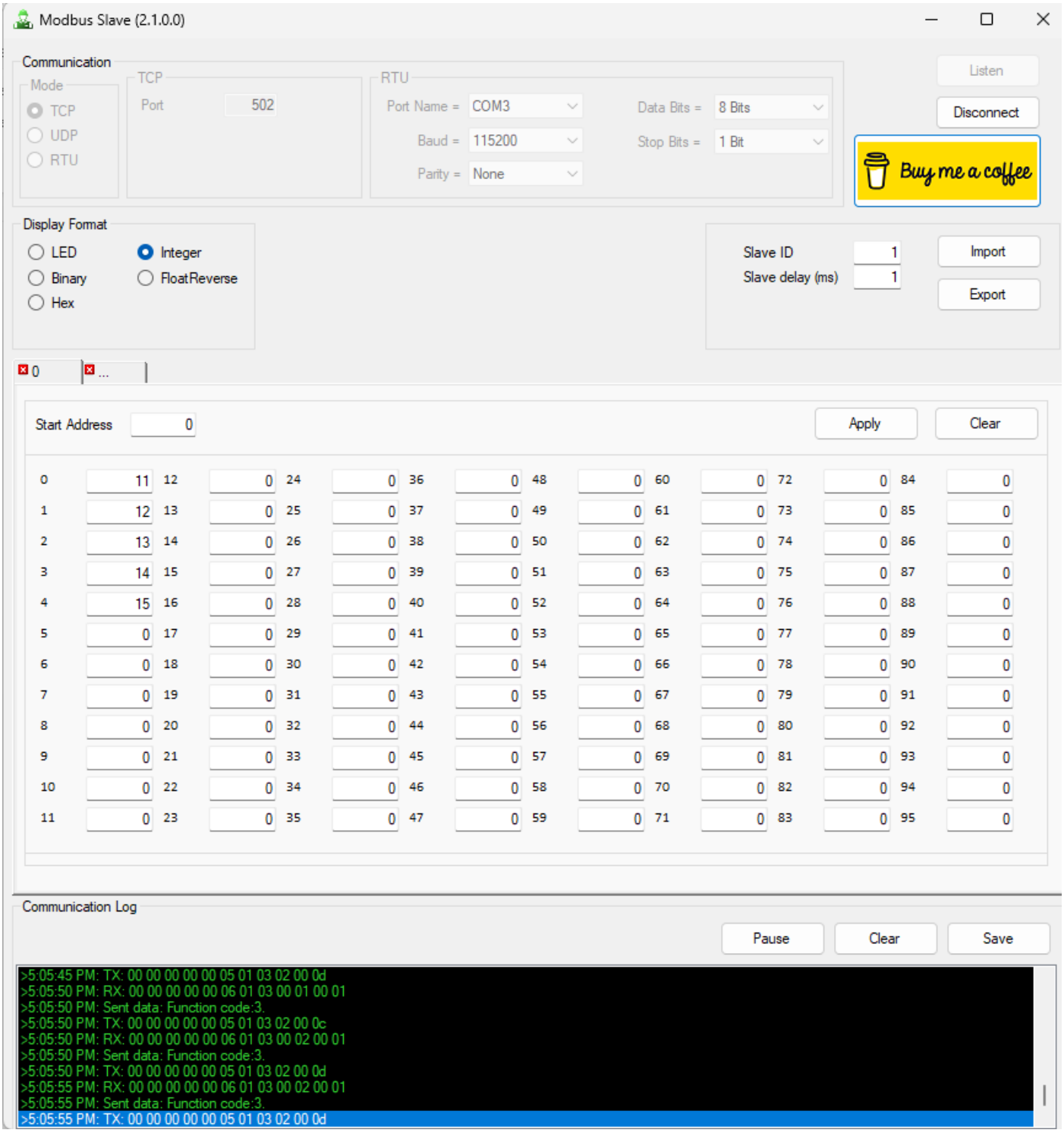


Figure 6 - Modbus Slave Settings

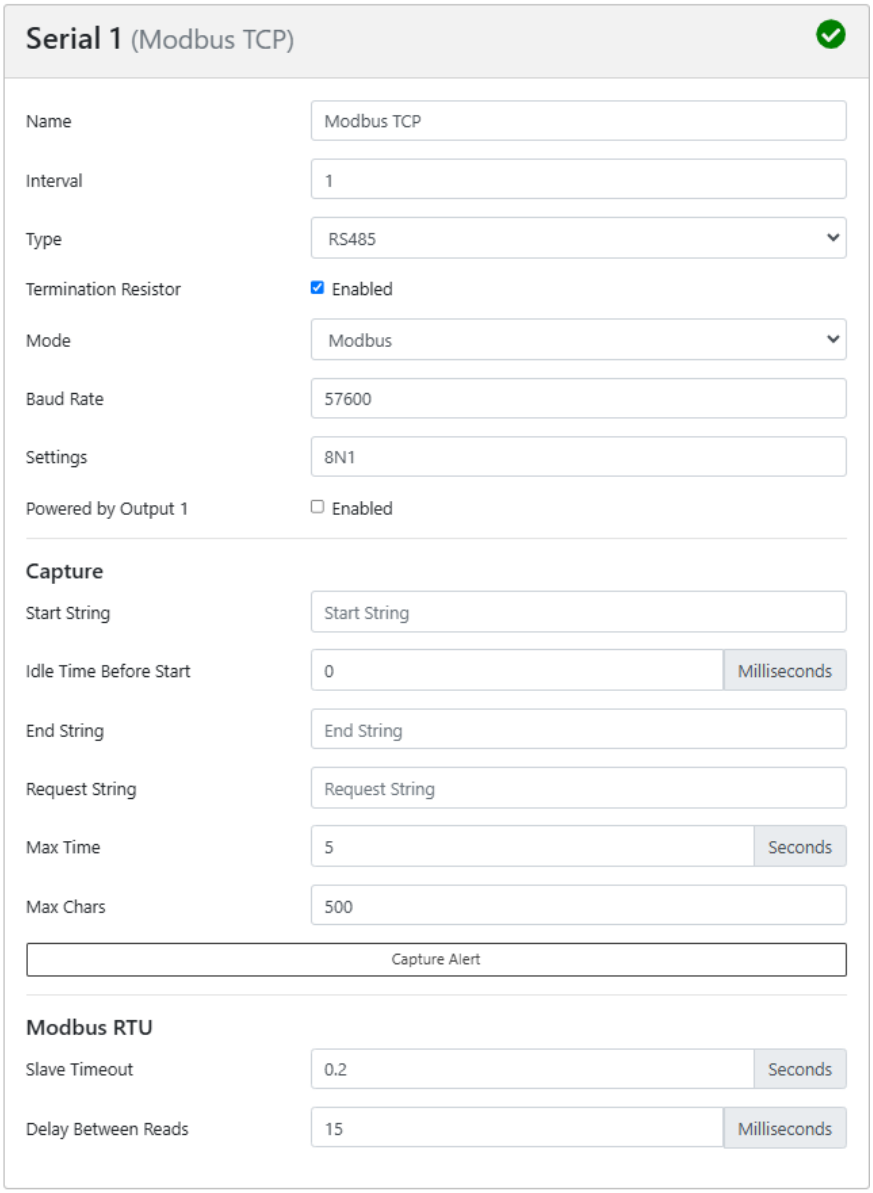


Figure 7 - Senquip QUAD Serial Port Settings

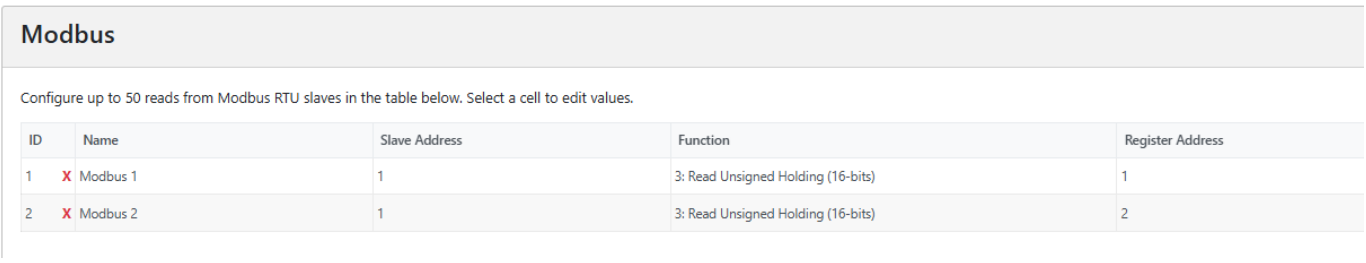
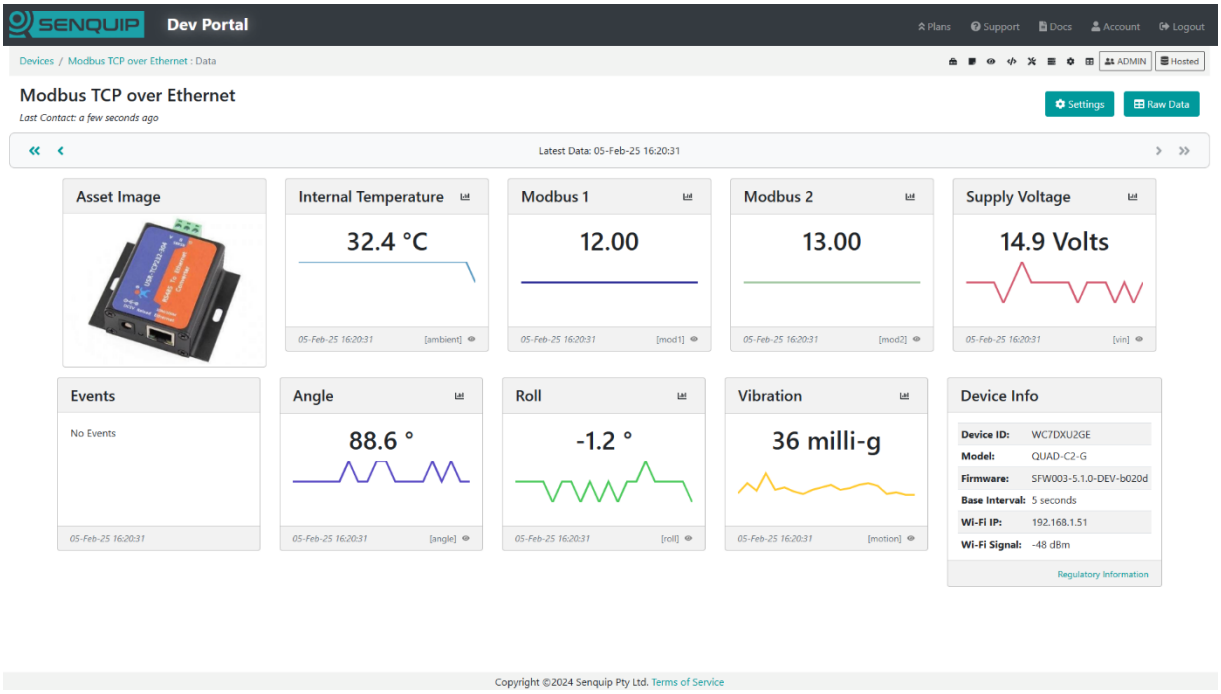


Figure 8 - Senquip QUAD Modbus Requests

4. Results

Modbus request packets can be seen arriving on the Modbus Slave Software. Responses are seen to be sent. Modbus reads 1 and 2 are shown on the Senquip Portal.



5. Conclusions

This application note demonstrates how a Senquip QUAD can communicate with a remote Modbus TCP sensor using an RS485-to-Ethernet converter. By configuring the PUSR USR-TCP232-304 as a TCP client, assigning a static IP address, and setting up the Modbus Slave software as a TCP server, seamless data exchange was established.

Successful Modbus requests were observed in the Modbus Slave software's communication log, confirming proper transmission and response of data packets. The Senquip Portal displayed the requested register values, verifying correct configuration and operation.

This approach provides a cost-effective solution for integrating RS485-based Modbus RTU devices into Ethernet-based Modbus TCP networks, enabling remote monitoring and control in industrial applications.