

Elmatic Sparrow NW10

Industrial Cellular VPN Router

Application Note 048

Modbus Master

Version: V1.0.0
Date: Oct 2021
Status: Confidential

Directory

1. Introduction.....	3
1.1 Overview.....	3
1.2 Compatibility.....	3
1.3 Version.....	3
1.4 Corrections.....	3
2. Topology.....	4
3. Transport via TCP.....	5
3.1 Configuration on Modbus Slave.....	5
3.2 Configuration on Modbus Poll.....	5
3.3 Configuration on Modbus Transport.....	7
4. Transport via FTP.....	10
5. Transport via MQTT.....	12

1. Introduction

1.1 Overview

This document contains information regarding the configuration and use of Modbus Master.

This guide has been written for use by technically competent personnel with a good understanding of the communications technologies used in the product, and of the requirements for their specific application.

1.2 Compatibility

This application note applies to:

Models Shown: Sparrow NW10 / Sparrow NW20

Firmware Version: V1.1.2 or newer

Other Compatible Models: None

1.3 Version

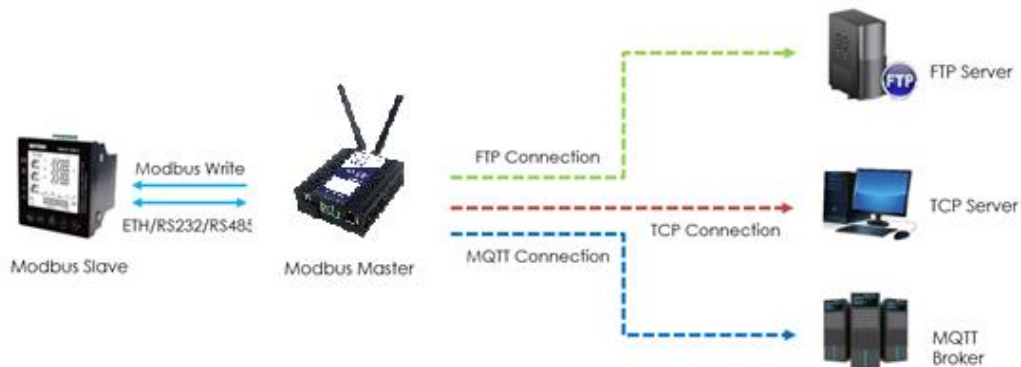
Updates between document versions are cumulative. Therefore, the latest document will include all the content of previous versions.

Release Date	Doc. Version	Firmware Version	Change Description
2021/09/30	V1.0.0	V1.1.2	First released

1.4 Corrections

Appreciate for corrections or rectifications to this application note, and if any request for new application notes please email to: elmark@elmark.com.pl

2. Topology



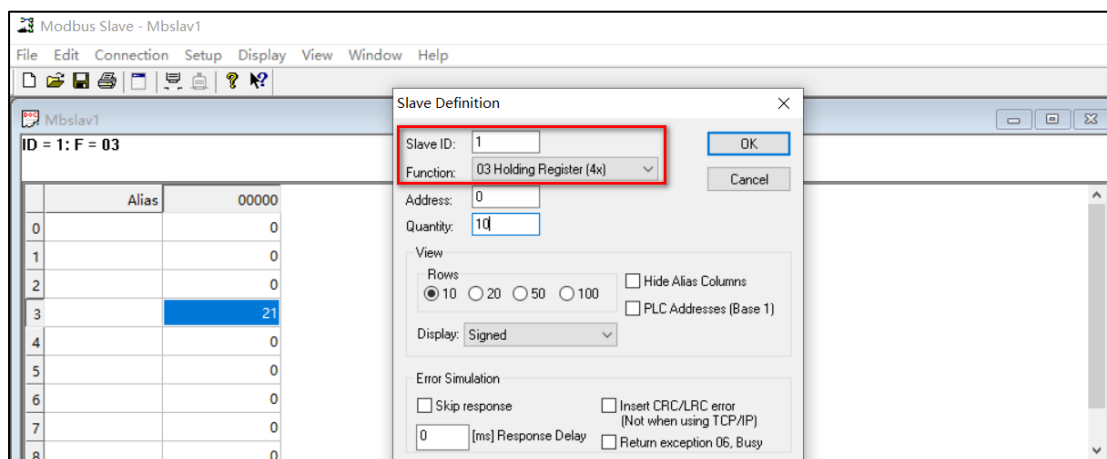
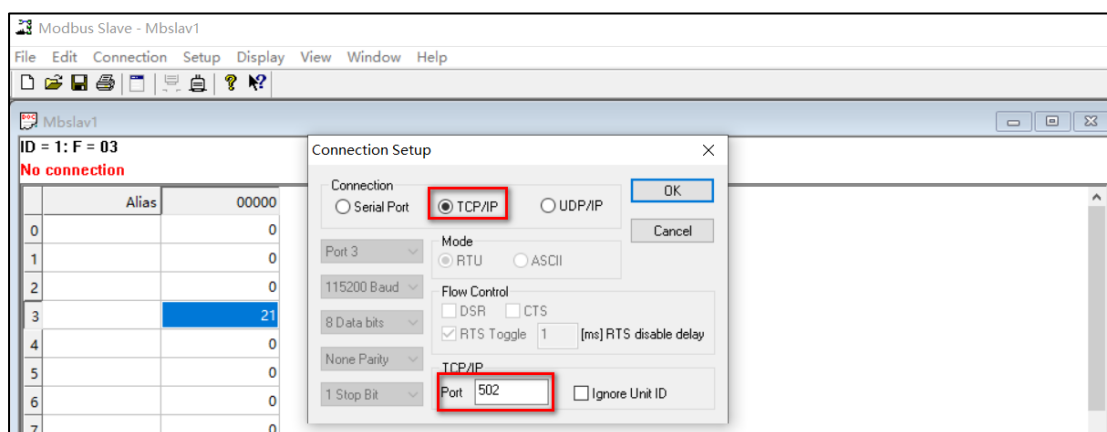
1. Sparrow Router runs as Modbus Master and connect to Modbus Slave via Ethernet, RS232 or RS485 interface.
2. Sparrow router poll the modbus data from modbus slave and send to the remote management center via TCP, FTP or MQTT protocol.
3. Sparrow as Modbus Master and write the register value or coil to Modbus Slave.

Note: For this Application Note, we will set the Connection Type as "TCP" as an example, which means that Sparrow(Modbus Master) will connect to the Modbus Slave and read the value via Ethernet port. Of course, it also works with Serial Port (RS232/RS485).

3. Transport via TCP

a) Configuration on Modbus Slave

1. Here we use “Modbus Slave” software to simulate the end device (Modbus Slave device), and the **TCP Port: 502, Slave ID: 1, Function Code: 03-Holding Register**, like below setting:



b) Configuration on Modbus Poll

1. Go to **Application>Modbus Master>Modbus Poll**, add a “Connection List” and specify the “Connection Type” as “TCP”, specify the “TCP Setting” to connect to Modbus Slave, like below:

Connection Settings

Connection List

Index	<input type="text" value="1"/>		
Enable	<input checked="" type="checkbox"/>		
Description	<input type="text" value="test"/>		
Scan Rate	<input type="text" value="100"/>	<input style="font-size: 8px; border: none; border-radius: 50%;" type="button" value="?"/>	
Reconnect Interval	<input type="text" value="60"/>	<input style="font-size: 8px; border: none; border-radius: 50%;" type="button" value="?"/>	
Response Timeout	<input type="text" value="1000"/>	<input style="font-size: 8px; border: none; border-radius: 50%;" type="button" value="?"/>	
Delay Between Polls	<input type="text" value="0"/>	<input style="font-size: 8px; border: none; border-radius: 50%;" type="button" value="?"/>	
Connection Type	<input type="text" value="TCP"/>		
Enable Show Status	<input checked="" type="checkbox"/>		
Enable Verbose Log	<input checked="" type="checkbox"/>		

TCP Settings

Server Address	<input type="text" value="192.168.111.44"/>	
Server Port	<input type="text" value="502"/>	

← IP Address of Modbus Slave

Channel List

Index	Enable	Description	Slave ID	Function Code	Register Address	+

2. Click Save>Apply.
3. Enable "Channel List", and specify the Slave ID as " 1 ", Function Code as " **03-Holding -Register** ", Register Address to " 3 ", then it will poll the value from register address 3 of Modbus Slave:

Channel Settings

Channel List

Index	<input type="text" value="1"/>		
Enable	<input checked="" type="checkbox"/>		
Description	<input type="text" value="test"/>		
Slave ID	<input type="text" value="1"/>		
Function Code	<input type="text" value="03-Holding-Register"/>		
Register Address	<input type="text" value="3"/>		
Data type	<input type="text" value="Uint16"/>		
Data Endian	<input type="text" value="AB"/>		
Plus	<input type="text" value="0"/>	<input style="font-size: 8px; border: none; border-radius: 50%;" type="button" value="?"/>	
Subtract	<input type="text" value="0"/>	<input style="font-size: 8px; border: none; border-radius: 50%;" type="button" value="?"/>	
Divisor	<input type="text" value="1"/>	<input style="font-size: 8px; border: none; border-radius: 50%;" type="button" value="?"/>	
Multiplier	<input type="text" value="1"/>	<input style="font-size: 8px; border: none; border-radius: 50%;" type="button" value="?"/>	
Shift Right Bits	<input type="text" value="0"/>	<input style="font-size: 8px; border: none; border-radius: 50%;" type="button" value="?"/>	
Number Of Bits	<input type="text" value="16"/>	<input style="font-size: 8px; border: none; border-radius: 50%;" type="button" value="?"/>	
Keep Decimal Places	<input type="text" value="0"/>	<input style="font-size: 8px; border: none; border-radius: 50%;" type="button" value="?"/>	

- Click Save>Save>Apply. *(Note: This is a secondary list, it needs to double click save)*
- Go to **Application>Modbus Master>Status**, then we can check the router had read the value from Modbus Slave successfully.

Index	Description	Connection Index	Type	Slave ID	Register Address	Function Code	Status	Value
1	test1	1	TCP	1	3	3	read succeeded	21

c) Configuration on Modbus Transport

- Go to **Application>Modbus Transport>Modbus Transport**, enable “Connection List”, and specify the TCP server IP address and port to send the data to remote TCP server. The Data Format could be defined accordingly or set it as default.

Connection Settings

Connection List

Index: 1

Enable:

Description: TCP Setting

Protocol: TCP-Client

Server Address: 14.215.177.39

Server Port: 2000

Reconnect Interval: 60

Connection Timeout: 30

Enable Verbose Log:

Transport Data Settings

Data Location: NULL

Data Format: \$SERIAL_NUMBER,\$DATE,\$S

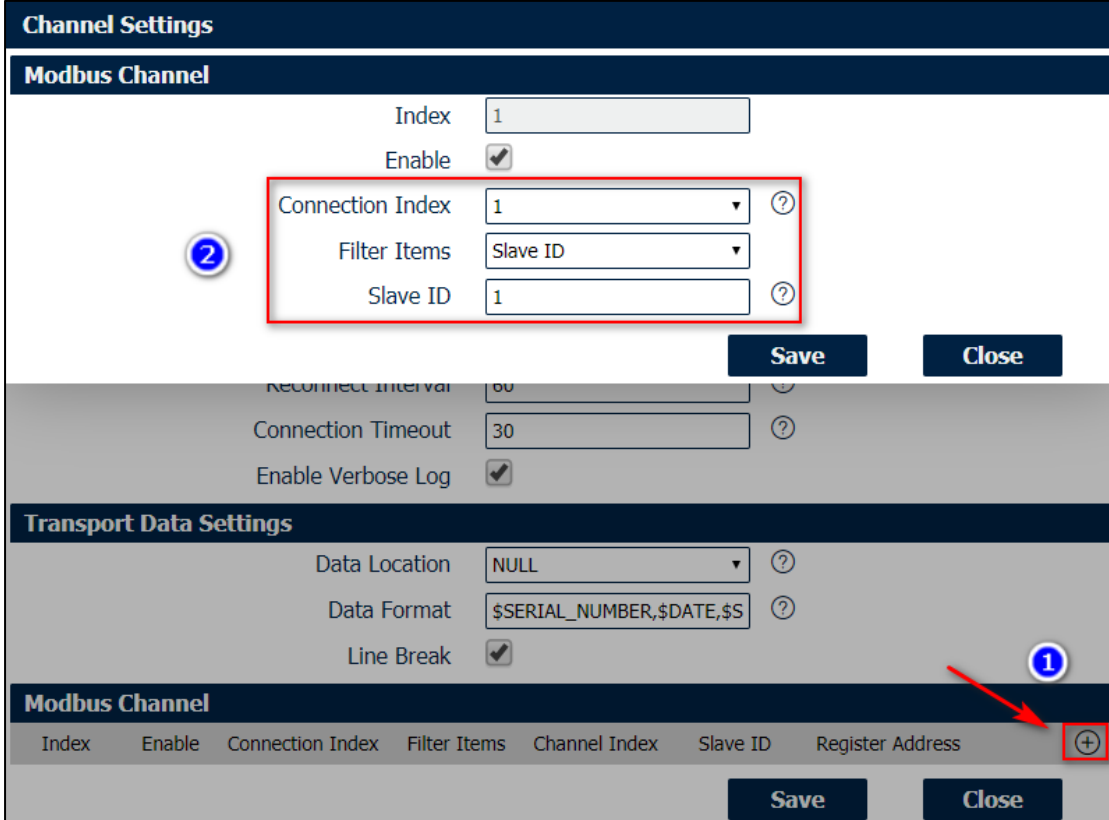
Line Break:

Modbus Channel

Index	Enable	Connection Index	Filter Items	Channel Index	Slave ID	Register Address

Save Close

- Enable “Modbus Channel”, Modbus Master will select the value send to the remote TCP server from Modbus Slave.



Channel Settings

Modbus Channel

Index: 1

Enable:

Connection Index: 1

Filter Items: Slave ID

Slave ID: 1

Save Close

Reconnect Interval: 60

Connection Timeout: 30

Enable Verbose Log:

Transport Data Settings

Data Location: NULL

Data Format: \$SERIAL_NUMBER,\$DATE,\$S

Line Break:

Modbus Channel

Index	Enable	Connection Index	Filter Items	Channel Index	Slave ID	Register Address

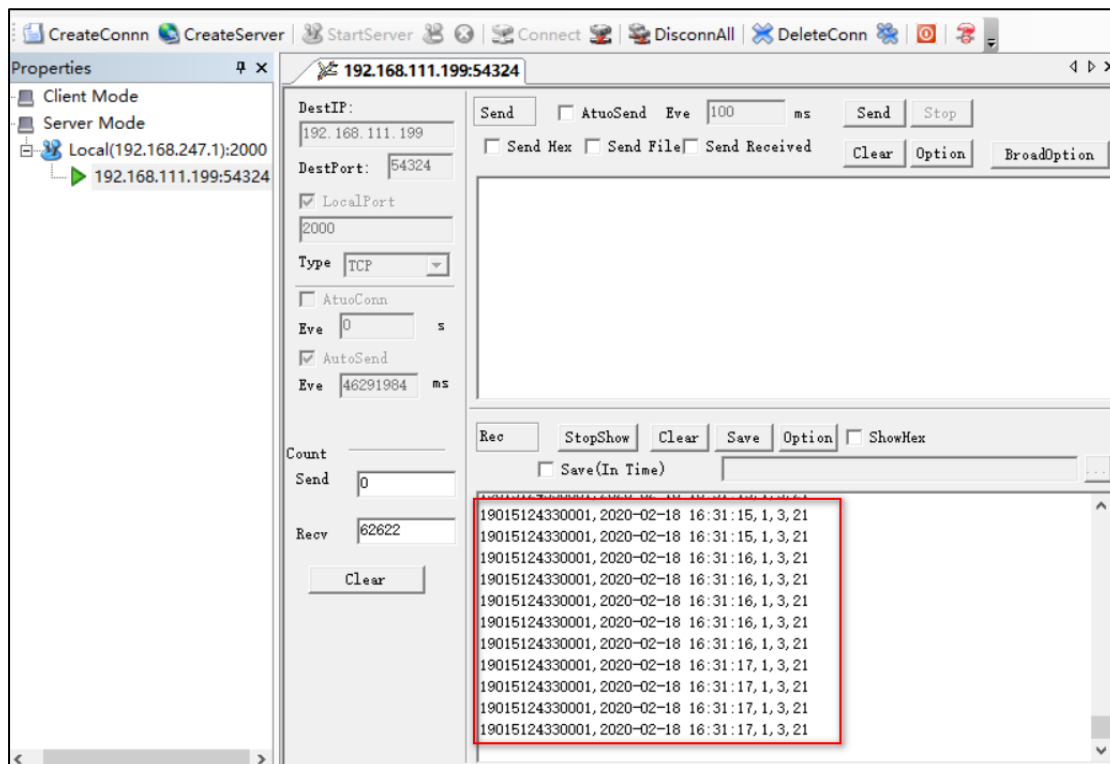
Save Close

3. Click Save>Save>Apply.

4. Go to **Application>Modbus Transport>Status**, Sparrow(Modbus Mater) had connected to the remote server successfully via TCP protocol.

<u>Status</u>		Modbus Transport	X.509 Certificate		
Connection Status					
Index	Enable	Description	Protocol	Status	Uptime
1	true	TCP Setting	TCP Client	Connected	00:02:35

5. Remote TCP Server received the data successfully.



4. Transport via FTP

1. Please refer to the “3.1 Configuration on Modbus Slave” and “3.2 Configuration on Modbus Poll” to finish and setting.
2. Go to **Application>Modbus Transport>Modbus Transport**, enable “Connection List”, and specify the FTP server IP address, port, username and password to send the data to remote FTP server. The File Name and Data Format could be defined accordingly or set it as default.

Connection Settings

Connection List

Index	<input type="text" value="1"/>
Enable	<input checked="" type="checkbox"/>
Description	<input type="text" value="FTP Setting"/>
Protocol	<input type="text" value="FTP"/>
Server Address	<input type="text" value="14.215.177.39"/>
Server Port	<input type="text" value="21"/>
Username	<input type="text" value="admin"/>
Password	<input type="text" value="adminftp"/>
Connection Timeout	<input type="text" value="30"/> ?
Try To Send	<input type="text" value="3"/> ?
Enable Verbose Log	<input checked="" type="checkbox"/>

Transport Data Settings

Data Location	<input type="text" value="NULL"/> ?
Add CSV File Title	<input checked="" type="checkbox"/>
File Name	<input type="text" value="\$SERIAL_NUMBER_\$DATE.cs"/> ?
Upload Interval	<input type="text" value="30"/> ?
Data Format	<input type="text" value="\$SERIAL_NUMBER,\$DATE,\$S"/> ?

3. Enable “Modbus Channel”, Modbus Master will select the value send to the remote FTP server from Modbus Slave.

Channel Settings

Modbus Channel

Index

Enable

Connection Index ?

Filter Items

Slave ID ?

Save **Close**

Try To Send ?

Enable Verbose Log

Transport Data Settings

Data Location ?

Add CSV File Title

File Name ?

Upload Interval ?

Data Format ?

Modbus Channel

Index	Enable	Connection Index	Filter Items	Channel Index	Slave ID	Register Address
						<input type="text" value="+"/>

Save **Close**

- Click Save>Save>Apply.
- Go to **Application>Modbus Transport>Status**, Sparrow(Modbus Mater) had connected to the remote server successfully via FTP protocol.

<u>Status</u>	Modbus Transport	X.509 Certificate			
Connection Status					
Index	Enable	Description	Protocol	Status	Uptime
1	true	FTP Setting	FTP	Sent Successfully	

- Remote FTP Server received the CSV file successfully.

名称	修改日期	类型	大小
19015124330001_2020-02-18_16-57-50.csv	2020/2/18 16:57	Microsoft Excel ...	1 KB
19015124330001_2020-02-18_16-58-21.csv	2020/2/18 16:58	Microsoft Excel ...	1 KB
19015124330001_2020-02-18_16-58-52.csv	2020/2/18 16:58	Microsoft Excel ...	1 KB
19015124330001_2020-02-18_16-59-23.csv	2020/2/18 16:59	Microsoft Excel ...	1 KB
19015124330001_2020-02-18_16-59-55.csv	2020/2/18 16:59	Microsoft Excel ...	1 KB

5. Transport via MQTT

1. Please refer to the “3.1 Configuration on Modbus Slave” and “3.2 Configuration on Modbus Poll” to finish and setting.
2. Go to **Application>Modbus Transport>Modbus Transport**, enable “Connection List”, and specify the MQTT Broker IP address, port, username and password to Publish the Topic with Modbus data to remote MQTT Broker. The Data Format could be defined accordingly or set it as default.

Connection Settings

Enable	<input checked="" type="checkbox"/>
Description	<input type="text" value="MQTT Setting"/>
Protocol	<input type="text" value="MQTT"/>
Server Address	<input type="text" value="192.168.111.93"/>
Server Port	<input type="text" value="1883"/>
Enable SSL	<input type="checkbox"/>
Username	<input type="text" value="mo_test"/>
Password	<input type="text" value="test123456"/>
Client ID	<input type="text"/> ?
Subscribe Topic	<input type="text"/> ?
Keepalive	<input type="text" value="60"/> ?
Reconnect Interval	<input type="text" value="60"/> ?
Connection Timeout	<input type="text" value="30"/> ?
Enable LWT	<input type="checkbox"/>
Enable Verbose Log	<input checked="" type="checkbox"/>

Transport Data Settings

Data Location	<input type="text" value="NULL"/> ?
Data Format	<input type="text" value="\$SERIAL_NUMBER,\$DATE,\$S"/> ?

3. Enable “Modbus Channel”, define the “Topic” to publish to MQTT Broker with Modbus data.

Channel Settings

Modbus Channel

Index

Enable

Publish Topic

Connection Index ?

Filter Items

Slave ID ?

Save **Close**

Connection Timeout ?

Enable LWT

Enable Verbose Log

Transport Data Settings

Data Location ?

Data Format ?

Line Break

Modbus Channel

Index Enable Connection Index Filter Items Channel Index Slave ID Register Address

Save **Close**

- Click Save>Save>Apply.
- Go to **Application>Modbus Transport>Status**, Sparrow(Modbus Mater) had connected to the remote MQTT broker successfully.

Status						
Modbus Transport						
X.509 Certificate						
Connection Status						
Index	Enable	Description	Protocol	Status	Uptime	
1	true	MQTT Setting	MQTT	Connected	00:23:04	

- Run the MQTT Client (MQTT Subscriber), to subscribe the topic just published to MQTT broker with modbus data. Then we can get the modbus data successfully.

Publish **Subscribe** Scripts Broker Status Log

test

test 237

test QoS 0

test 233 QoS 0

test 234 QoS 0

test 235 QoS 0

test 236 QoS 0

test 237 QoS 0

Topics Collector (0)

test 237

18-02-2020 18:27:36.66456286 QoS 0

19015124330001,2020-02-18 18:27:35,1,3,21

Payload decoded by