Modbus Master RutOS configuration example

□

Contents

Summary

Modbus TCP Master section is used for configuring your router as a master device and other routers configured in this section as slave devices. A Modbus TCP master device can then request data from these Modbus TCP slaves.

First, let’s configure our second router which will be acting as a slave device.

Configure Modbus TCP slave device

In this example, we would be configuring our slave device to have a 192.168.1.2 internal IP address. This can be done by changing the IP address parameter on Network → WAN page:

Now we need to enable Modbus service for this slave device as we will open a specific port for letting through TCP communications (it is also possible to 'Allow remote access' if we wish to access this router from external WAN using its public IP). This is done on Services → Modbus page:

Field	Value	Description
Enable	off | on; default **off**	Turns Modbus TCP on or off.
Port	integer [0..65535]; default: **502**	TCP port used for Modbus communications.
Device ID	integer [0..255]; default: **1**	The device's Modbus slave ID. When set to 0, it will respond to requests addressed to any ID.
Allow Remote Access	off | on; default: **off**	Allows remote Modbus connections by adding an exception to the device's firewall on the port specified in the field above.

Keep persistent connection	off | on; default: **off**	Allows keep the connection open after responding a Modbus TCP master request.
Connection timeout	integer; default: **0**	Sets TCP timeout in seconds after which the connection is forcefully closed.
Enable custom register block	off | on; default: **off**	Allows the usage of custom register block.

Configuring Modbus TCP master device

Adding a new slave device

On the master device (in this example master device has 192.168.1.1 internal IP address) open Services → Modbus → Modbus TCP Master. To add a new slave, first, click the add button.

Once you have clicked that button, enter the following configuration so we can add the slave device:

Field	Value	Description
Enable	string; default: **none**	Slave device's name, used for easier management purposes.
Name	off | on; default: **off**	Turns communication with the slave device on or off.
Slave ID	integer [0..255]; default: **none**	Slave ID. Each slave in a network is assigned a unique identifier ranging from 1 to 255. When the master requests data from a slave, the first byte it sends is the Slave ID. When set to 0, the slave will respond to requests addressed to any ID.
IP address	ip; default: **none**	Slave device's IP address.
Port	integer [0..65535]; default: **none**	Slave device's Modbus TCP port.
Period	integer [1..86400]; default: **60**	Interval at which requests are sent to the slave device.
Timeout	integer [1..30]; default: **5**	Maximum response wait time.

Scroll down and click “Save and Apply”

After this, this is how your main Modbus Master panel should look:

Testing

For testing, if the functionality is working we can configure a request. A Modbus request is a way of obtaining data from Modbus slaves. The master sends a request to a slave specifying the function code to be performed. The slave then sends the requested data back to the Modbus master. You can create a maximum of 64 request configurations for each slave device. First, click the edit button:

Scroll down to the “Request configuration” section.

Write a name for the request and click the “Add” button.

And enter a data type, a function, first register, number of registers, if you need brackets or don’t and enable the request. In this case we are trying to get System uptime information so configure the next options:

Field	Value	Description
Name	string; default: **Unnamed Parameter**	Request name. Used for easier management purposes.
Data type	8bit INT | 8bit UINT | 16bit INT, high byte first | 16bit INT, low byte first | 16bit UINT, high byte first | 16bit UINT, low byte first | 32bit float, Byte order 1,2,3,4 | 32bit float, Byte order 4,3,2,1 | 32bit float, Byte order 2,1,4,3 | 32bit float, Byte order 3,4,1,2; default: **16bit INT, high byte first**	How read data will be stored.
Function	1 | 2 | 3 | 4 | 5 | 6 | 15 | 16; default: **3**	A function code specifies the type of register being addressed by a Modbus request. The codes represent these functions: • **1** - read Coil Status • **2** - read Input Status • **3** - read Holding Registers • **4** - read Input Registers • **5** - force Single Coil • **6** - preset Single Register • **15** - force Multiple Coils • **16** - force Multiple Registers
First Register	integer [0..65535]; default: **1**	First Modbus register from which data will be read.
Number of Registers	integer [1..2000]; default: **none**	Number of Modbus registers that will be read during the request.
Enabled	yes | no; default: **no**	Turns the request on or off.
Test	- (interactive button)	Generates a Modbus request according to given parameters in order to test the request configuration. You must first save the configuration before you can use the Test button.
Delete	- (interactive button)	Deletes the request.
Add	- (interactive button)	Adds a new request configuration.

Under the “Request Configuration” section there will be another section called “Request Configuration Testing” this is where you can test that the parameters that you set work properly. Just select the request that you want to test and click the test button and you should get an output.

Alarms

Alarms are a way of setting up automated actions when some Modbus values meet user-specified conditions. To configure these alarms, scroll to the end of the menu and click the “Add” button.

After clicking the “Add” button the next menu should come up:

Field	Value	Description
Enabled	yes | no; default: **no**	Turns the alarm on or off
Function code	Read Coil Status (1) | Read Input Status (2) | Read Holding Registers (3) | Read Input Registers (4); default: **Read Coil Status (1)**	Modbus function used in Modbus request.
Register	integer [0..65535]; default: **none**	Number of the Modbus coil/input/holding register/input register that will be read.
Condition	More than | Less than | Equal to | Not Equal to; default: **Equal to**	When a value is obtained it will be compared against the value specified in the following field. The comparison will be made in accordance with the condition specified in this field.
Value	various; default: **none**	The value against which the read data will be compared.
Action	SMS | Trigger output | Modbus Request; default: **SMS**	Action that will be taken if the condition is met. Possible actions: • **SMS** - sends and SMS message to a specified recipient(s). • **Trigger output** - changes the state of a specified output(s). • **Modbus Request** - sends a Modbus request to a specified slave.
SMS: Message	string; default: **none**	SMS message text.
SMS: Phone number	phone number; default: **none**	Recipient's phone number.
Trigger output: Output	Open collector output | Relay output | Both; default: **Open collector output**	Which output(s) will be triggered.
Trigger output: I/O Action	Turn On | Turn Off | Invert; default: **Turn On**	Action that will taken on the specified output.
Modbus Request: IP address	ip | host; default: **none**	Modbus slave's IP address.
Modbus Request: Port	integer [0..65535]; default: **none**	Modbus slave's port.
Modbus Request: Timeout	integer [1..30]; default: **5**	Maximum time to wait for a response.
Modbus Request: ID	integer [1..255]; default: **none**	Modbus slave ID.

Modbus Request: Modbus function	Read Coil Status (1) | Read Input Status (2) | Read Holding Registers (3) | Read Input Registers (4) | Force Single Coil (5) | Preset Single Register (6) | Force Multiple Coils (15) | Force Multiple Registers (16); default: **Force Single Coil (5)**	A function code specifies the type of register being addressed by a Modbus request.
Modbus Request: First register	integer [0..65535]; default: **none**	Begins reading from the register specified in this field.
Modbus Request: Number of registers	integer [0..65535]; default: **none**	The number of registers that will be read from the first register.