EWIO$_2$
Modbus Router

METZ CONNECT GmbH | Im Tal 2 | 78176 Blumberg | Deutschland
Tel. +49 7702 533-0 | Fax +49 7702 533-433
Weitere Dokumentation siehe www.metz-connect.com

Description Modbus Router v1.0 for the EWIO$_2$

General

The programme ModbusRouter contains a Modbus/RTU master for a RS485 interface. It connects the master to the TCP/IP stack using the Modbus/TCP protocol. Several Modbus/RTU slaves can be controlled and queried by this master.

The EWIO$_2$ has two RS485 interfaces.
1. on the 4-pole terminals at the top of the unit, specially used for expansion modules.
2. on terminals A'- and B'+ of the wide terminal strip for general use.

Each interface needs its own Modbus/RTU master, so the Modbus router must run twice.

General information about the Modbus protocol

The Modbus protocol regulates the communication between devices in a network.
For this purpose, it defines messages (frames) that are exchanged between the devices, their coding, error handling, timing and other things. It is divided into an application layer and a data link layer.

The application layer describes the application-related functions. The most important ones are used for reading and writing bits and registers by one device (called client) in another device (called server).

The data link layer describes the underlying data transmission, here two protocols are important:
- Modbus/RTU for the connection via an RS485 interface.
- Modbus/TCP for the connection via a TCP/IP network.
The equivalents of client and server at this level are called master and slave.

The description below is highly abbreviated, complete ones are available from www.modbus.org/specs.php.

Example for the Modbus Application Layer

Modbus-function 3 (0x03) Read Holding Registers:

This function is used to read several consecutive registers.
The client sends the request, the server responds with Response or Error.

Request:

Byte 1	Function Code	0x03
Byte 2-3	Register Address	First Register
Byte 4-5	Register Quantity	Number of registers

Response:

Byte 1	Function Code	0x03
Byte 2	Byte Count	2 Bytes per register
Byte 3-4	Register Value	Value of the first register
Byte ...	Register Value	Value of the next register

Error:

Byte 1	Error Code	0x83
Byte 2	Exception Code	Fehlercode

If data consists of several bytes, the highest value is transmitted first.
An application frame can be a maximum of 253 bytes long.

Modbus/RTU

Terminating resistors:

In the EWIO$_2$ there are resistors with 220 Ohm against ground and 3.3V for the quiescent level:
3.3V * (150 / 2) / (150 / 2 + 220 * 2) = 0.48V.

Coding:

Byte 1	Slave Address
Byte 2-N	Application Layer Frame
Byte N+1	CRC Low Byte
Byte N+2	CRC High Byte

The Slave Address field is used to address the devices on the RS485 bus.

Broadcast to all slaves: 0 (the slaves do not answer)Individuelle Slave-Adressen: 1 … 247

Reserved addresses: 248 ... 255

The two CRC bytes are used to detect transmission errors.

Frame synchronisation:

A receiver must be synchronised to the transmitter at the other end of the connection.
The following frame timing specifications, which the transmitter must comply with, serve this purpose:

A pause between the bytes of a frame must not exceed a maximum duration.
- For up to 19200 baud: duration of 1.5 bytes (1.5 * 11 / baud rate)
- For higher baud rate: 0.75 ms

Between two frames there must be a pause with a minimum duration.
- For up to 19200 baud: duration of 3.5 bytes (3.5 * 11 / baud rate)
- With higher baud rate: 1.75 ms

For the receiver, the range in between applies accordingly as a tolerance for the time measurement.
On average, a pause of 2.5 bytes or 1.25 ms during reception leads to the detection of the end of the frame. With a repeater in the Modbus connection, this pause also leads to a reversal of the transmission direction.

Modbus/TCP

Coding:

Byte 1-2	Transaction Identifier	
Byte 3-4	Protocol Identifier	For Modbus value 0 applies.
Byte 5-6	Length	Bytes after Length, N - 6
Byte 7	Unit Identifier	
Byte 8-N	Application Layer Frame	

The Transaction Identifier is a number with which request and response can be assigned to each other if several requests are sent almost simultaneously over the same connection without waiting for the respective response.

The Unit Identifier corresponds to the Slave Address field in Modbus/RTU. The unit with the Modbus/TCP interface can contain a router to a Modbus/RTU interface. Then this field is used to address the Modbus/RTU slave.

Broadcast to all slaves: 0 (the slaves do not answer)
Individual slave addresses: 1 ... 247
Reserved addresses: 248 ... 254
Router is addressed: 255

Frame synchronisation:

A receiver must be synchronised to the transmitter at the other end of the connection. For this purpose, the first 6 bytes of a frame are received. Then the Length field is evaluated and a corresponding number of bytes are received again.

To detect synchronisation errors, the Protocol Identifier and Length fields are checked. The Protocol Identifier must always be 0 for Modbus and Length may only be 1...254. For the client, the transaction identifiers of the request and response must be the same. In addition, a timeout must be used if a frame is only received incompletely.

After a synchronisation error, the connection must be terminated, because a resynchronisation is not possible with the data present in the frames.

Special features of the Modbus router

Baud rate setting

Baud rate and parity of the Modbus router can be set in several ways. There is

- a default setting, that can be configured in the web-interface of the EWIO$_2$,
- the holding register 0.

The default setting can be changed at any time with Modbus functions. In addition, the baud rate and parity of Modbus slaves can be automatically set to the current values by METZ CONNECT. Holding registers 10-12 are used for this purpose.

Error handling

The Modbus/TCP connection is protected against transmission errors by the TCP protocol. Faulty Ethernet transmissions are thus automatically repeated.

This protection is not available for Modbus/RTU connections. The application programme, in this case the Modbus router, is responsible for repeating faulty transmissions. If, for example, an electromagnetic interference changes the transmitted data, there is first a parity, framing or CRC error at the receiver. Such an erroneous frame must not be used further, but discarded. If the slave does not answer at all, there is a timeout at the router.

The router has the possibility to repeat the Modbus/RTU transmission after an error. The timeout and the maximum number of repetitions after the first transmission can be configured in Modbus registers.

Exception Codes

A transmission error or timeout is reported by the router in an error frame.
In the error frame, bit 7 of the function code is set in the first byte, which makes the byte an error code. The second byte contains an exception code that describes the cause of the error:

Error from server:

1 Illegal Function Code Unknown code in function or sub-function.
2 Illegal Data Address A register address is invalid.
3 Illegal Data Value Inconsistent coding in register number, number of bytes or data value

Error from router:

10 Gateway Path Unavailable The Modbus/RTU interface is not working.
11 Gateway Target Device The slave device does not respond or responds incorrectly. Failed to Respond (timeout, parity, framing or CRC error)

Other exception codes are defined, but do not come from our devices so far.

Modbus register

Registers 0-3 are only used for communication with the extension module drivers. They are periodically checked and written by the drivers. Other programmes should therefore not write to these registers. The drivers are operated via the shared library mc-io-api.

Holding register 0:

The value in the register is coded almost the same as in the Modbus slaves of METZ CONNECT GmbH. The default setting is 19200 baud and even parity.

Bits 8-15 contain the constant 0x00 when reading.
Bits 8-15 contain the constant 0x53 when writing
Bits 4-7 contain the code for the parity and stop bits.
Bits 0-3 contain the code for the baud rate.

The constant 0x53 serves as a magic number to protect against accidental writing.
Only with this value will the write command continue to be executed.

Code	1	2	3	4				
Parity	Even	Odd	None	None				
Stopbits	1	1	2	1				
Code	1	2	3	4	5	6	7	8
Baudrate	1200	2400	4800	9600	19200	38400	57600	115200

Holding register 1:

The register contains the timeout in milliseconds that is provided at the Modbus/RTU interface for the response of a slave. The time is measured from the end of the command to the slave to the beginning of the response from the slave.
The value range is 0...1000 ms.
The default setting is 100 ms.

Holding register 2:

The register contains the number of repetitions if the slave does not respond to the commands.
The value range is 0...10.
The default setting is 2 (for a total of up to 3 attempts).

Special case: The extension module driver writes 0x7F into the high byte when it is started for synchronisation, but 0 is always read.

Holding register 3:

Is no longer used.

Holding register 4:

The register contains a counter for timeout errors, i.e. when the slave does not answer the commands. It is also counted here if the answer is shorter than 3 bytes.
The value range is 0...65535.
The default setting is 0.

Holding register 5:

This register contains a counter for parity, framing and CRC errors, i.e. when the slave's response has been altered, e.g. by electromagnetic interference.
The value range is 0...65535.
The default setting is 0.

Register for automatic setting of baud rate and parity for a Modbus slave from METZ CONNECT GmbH

Registers 10-12 are only used for communication with the driver for expansion modules. They are periodically checked and written by the driver. Other programs should therefore not write to these registers. The driver is operated via the shared library mc-io-api.

The automatic setting is supported by both Modbus routers for any slave addresses. So far it is only carried out for the extension modules (address 1-6).

When the extension module driver is started, an interrupted slave setting ends immediately.

Holding register 10:

The slave address is written into the register (rotary switch of the unit).

Holding register 11:

The Modbus router then tries to identify the slave at most twice, with all possible combinations of baud rate and parity. If this is successful, the baud rate and parity in the slave are set to the current values specified in Holding Register 0.

The Modbus router reports the end of the process in this register.
Value 0: The slave was successfully set.
Value 2: The slave could not be found or set.

Holding register 12:

In this register, the Modbus router reports the current work step as a progress indicator. The entire process takes up to 9 seconds. Values: 0...48.