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

Beschreibung Modbus-Router v1.0 für das EWIO$_2$

Allgemeines

Das Programm ModbusRouter enthält einen Modbus/RTU-Master für eine RS485-Schnittstelle. Es verbindet diesen mit dem TCP/IP-Stack, wobei dort das Protokoll Modbus/TCP genutzt wird. Mehrere Modbus/RTU-Slaves können von diesem Master gesteuert und abgefragt werden.

Das EWIO$_2$ hat zwei RS485-Schnittstellen:
1. An den 4-poligen Klemmen oben am Gerät, speziell genutzt für Erweiterungs-Module.
2. An den Klemmen A'- und B'+ der breiten Klemmleiste für allgemeine Verwendung.

Jede Schnittstelle braucht einen eigenen Modbus/RTU-Master, so dass der Modbus-Router zweifach laufen muss.

Allgemeines zum Modbus-Protokoll

Das Modbus-Protokoll regelt die Kommunikation zwischen Geräten in einem Netzwerk. Dazu definiert es Nachrichten (Frames), die zwischen den Geräten ausgetauscht werden, ihre Codierung, die Fehlerbehandlung, das Timing und anderes. Es ist in Application-Layer und Data-Link-Layer aufgeteilt.

Der Application-Layer beschreibt die anwendungsnahen Funktionen. Die wichtigsten dienen zum Lesen und Schreiben von Bits und Registern durch ein Gerät (Client genannt) in einem anderen Gerät (Server genannt).

Der Data-Link-Layer beschreibt die zu Grunde liegende Datenübertragung, hier sind zwei Protokolle von Bedeutung:
- Modbus/RTU für die Verbindung über eine RS485-Schnittstelle.
- Modbus/TCP für die Verbindung über ein TCP/IP-Netz.

Die Entsprechungen von Client und Server heißen auf dieser Ebene Master und Slave.

Die Beschreibung unten ist stark verkürzt, vollständige gibt es von www.modbus.org/specs.php.

Beispiel für den Modbus-Application-Layer

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

Diese Funktion wird verwendet, um mehrere aufeinanderfolgende Register zu lesen.
Der Client sendet den Request, der Server antwortet mit Response oder Error.

```
Request:
  Byte 1      Function Code          0x03
  Byte 2-3    Register Address       Erstes Register
  Byte 4-5    Register Quantity      Anzahl der Register

Response:
  Byte 1      Function Code          0x03
  Byte 2      Byte Count             2 Bytes je Register
  Byte 3-4    Register Value         Wert des ersten Registers
  Byte ...    Register Value         Wert des nächsten Registers

Error:
  Byte 1      Error Code             0x83
  Byte 2      Exception Code         Fehlercode
```

Wenn Daten aus mehreren Bytes bestehen, wird das höchstwertige zuerst übertragen.
Ein Application-Frame kann maximal 253 Bytes lang sein.

Modbus/RTU

Abschlusswiderstände:

An den Enden des RS485-Busses sind Widerstände mit 150 Ohm, um Reflektionen zu dämpfen.
Im Master sind Widerstände gegen Masse und Versorgung, um den Ruhepegel einzustellen.

Im EWIO$_2$ sind Widerstände mit 220 Ohm gegen Masse und 3,3V für den Ruhepegel:
3,3V * (150 / 2) / (150 / 2 + 220 * 2) = 0,48V.

Codierung:

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

Mit dem Feld Slave Address werden die Geräte am RS485-Bus adressiert.

Broadcast an alle Slaves: 0 (die Slaves antworten nicht)
Individuelle Slave-Adressen: 1 … 247

Reservierte Adressen: 248 … 255

Die beiden CRC-Bytes dienen zur Erkennung von Übertragungs-Fehlern.

Frame-Synchronisation:

Ein Empfänger muss auf den Sender am anderen Ende der Verbindung synchronisiert sein. Dazu dienen folgende Festlegungen zum Frame-Timing, die der Sender einhalten muss:

Eine Pause zwischen den Bytes eines Frames darf eine Höchstdauer nicht überschreiten.
- Bei bis zu 19200 Baud: Dauer von 1,5 Bytes (1,5 * 11 / Baudrate)
- Bei höherer Baudrate: 0,75 ms

Zwischen zwei Frames muss eine Pause mit einer Mindestdauer sein.
- Bei bis zu 19200 Baud: Dauer von 3,5 Bytes (3,5 * 11 / Baudrate)
- Bei höherer Baudrate: 1,75 ms

Für den Empfänger gilt entsprechend der Bereich dazwischen als Toleranz bei der Zeitmessung.
Im Mittel führt also eine Pause von 2,5 Bytes oder 1,25 ms beim Empfang zur Erkennung des Frame-Endes. Bei einem Repeater in der Modbus-Verbindung führt diese Pause auch zur Umsteuerung der Übertragungs-Richtung.

Modbus/TCP

Codierung:

```
Byte 1-2    Transaction Identifier
Byte 3-4    Protocol Identifier          Für Modbus gilt Wert 0
Byte 5-6    Length                       Bytes nach Length, N - 6
Byte 7      Unit Identifier
Byte 8-N    Application Layer Frame
```

Der Transaction Identifier ist eine Nummer, mit der Request und Response einander zugeordnet werden können, falls mehrere Requests fast gleichzeitig über die gleiche Verbindung geschickt werden, ohne auf die jeweilige Response zu warten.

Der Unit Identifier entspricht dem Feld Slave Address bei Modbus/RTU. Das Gerät mit der Modbus/TCP-Schnittstelle kann einen Router zu einer Modbus/RTU-Schnittstelle enthalten. Dann dient dieses Feld zur Adressierung des Modbus/RTU-Slaves.

Broadcast an alle Slaves: 0 (die Slaves antworten nicht)
Individuelle Slave-Adressen: 1 … 247
Reservierte Adressen: 248 … 254

Router wird adressiert: 255

Frame-Synchronisation:

Ein Empfänger muss auf den Sender am anderen Ende der Verbindung synchronisiert sein. Dazu werden die ersten 6 Bytes eines Frames empfangen. Dann wird das Feld Length ausgewertet und noch einmal entsprechend viele Bytes empfangen.

Um Synchronisations-Fehler zu erkennen, werden die Felder Protocol Identifier und Length geprüft. Der Protocol Identifier muss für Modbus immer 0 sein und Length darf nur 1…254 sein. Beim Client müssen die Transaction Identifier von Request und Response gleich sein. Außerdem muss ein Timeout verwendet werden, falls ein Frame nur unvollständig empfangen wird.

Nach einem Synchronisations-Fehler muss die Verbindung beendet werden, denn eine Neusynchronisation ist mit den in den Frames vorhandenen Daten nicht möglich.

Besonderheiten des Modbus-Routers

Baudraten-Einstellung

Baudrate und Parität des Modbus-Routers können auf mehrere Arten eingestellt werden. Es gibt

- eine Grundeinstellung, die im Web-Interface des EWIO$_2$ konfiguriert werden kann,
- das Holding-Register 0.

Die Grundeinstellung kann jederzeit mit Modbus-Funktionen geändert werden. Außerdem können Baudrate und Parität von Modbus-Slaves von METZ CONNECT automatisch auf die aktuellen Werte eingestellt werden. Dazu dienen die Holding-Register 10-12.

Fehlerbehandlung

Die Modbus/TCP-Verbindung wird vom TCP-Protokoll vor Übertragungs-Fehlern geschützt. Fehlerhafte Ethernet-Übertragungen werden damit automatisch wiederholt.

Bei Modbus/RTU-Verbindungen ist dieser Schutz nicht vorhanden. Das Anwendungs-Programm, hier also der Modbus-Router, ist dafür verantwortlich, dass fehlerhafte Übertragungen wiederholt werden. Wenn z.B. eine elektromagnetische Störung die übertragenen Daten verändert, gibt es zuerst einen Paritäts-, Framing- oder CRC-Fehler beim Empfänger. Solch ein fehlerhafter Frame darf nicht weiter verwendet werden, sondern muss weggeworfen werden. Falls der Slave gar nicht antwortet, gibt es ein Timeout beim Router.

Der Router hat die Möglichkeit, nach einem Fehler die Modbus/RTU-Übertragung zu wiederholen. Das Timeout und die maximale Anzahl der Wiederholungen nach dem ersten Senden ist in Modbus-Registern konfigurierbar.

Exception Codes

Ein Übertragungs-Fehler oder Timeout wird vom Router in einem Error-Frame gemeldet. Im Error-Frame ist im ersten Byte das Bit 7 des Function Code gesetzt, das Byte wird damit zum Error Code. Im zweiten Byte folgt noch ein Exception Code, der die Fehler-Ursache beschreibt:

Fehler vom Server:

1	Illegal Function Code	Unbekannter Code in Funktion oder Subfunktion
2	Illegal Data Address	Eine Register-Adresse ist ungültig
3	Illegal Data Value	Inkonsistente Codierung bei Register-Anzahl, Byte-Anzahl oder Datenwert

Fehler vom Router:

10	Gateway Path Unavailable	Die Modbus/RTU-Schnittstelle funktioniert nicht
11	Gateway Target Device Failed to Respond	Das Slave-Gerät antwortet nicht oder fehlerhaft (Timeout, Paritäts-, Framing- oder CRC-Fehler)

Andere Exception Codes sind definiert, kommen von unseren Geräten aber bisher nicht.

Modbus-Register

Register 0-3 dienen zur nur Kommunikation mit den Erweiterungsmodul-Treibern. Sie werden von den Treibern periodisch geprüft und geschrieben. Andere Programme sollen deshalb nicht in diese Register schreiben. Die Treiber werden über die Shared-Library mc-io-api bedient.

Holding Register 0:

Der Wert im Register ist fast gleich codiert, wie bei den Modbus-Slaves von METZ CONNECT GmbH. Die Grundeinstellung im Auslieferungszustand ist 19200 Baud und gerade Parität.

Bits 8-15 enthalten die Konstante 0x00 beim Lesen
Bits 8-15 enthalten die Konstante 0x53 beim Schreiben
Bits 4-7 enthalten den Code für die Parität und Stopbits
Bits 0-3 enthalten den Code für die Baudrate

Die Konstante 0x53 dient als Magic-Number zum Schutz vor versehentlichem Schreiben.

Nur mit diesem Wert wird das Schreib-Kommando weiter ausgeführt.

Code	1	2	3	4
Parität	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:

Das Register enthält das Timeout in Millisekunden, das an der Modbus/RTU-Schnittstelle für die Antwort eines Slaves vorgesehen ist. Die Zeit wird gemessen vom Ende des Kommandos zum Slave bis zum Anfang der Antwort vom Slave.
Der Wertebereich ist 0…1000 ms.
Die Grundeinstellung ist 100 ms.

Holding Register 2:

Das Register enthält die Anzahl der Wiederholungen, falls der Slave auf die Kommandos nicht antwortet.
Der Wertebereich ist 0…10.
Die Grundeinstellung ist 2 (für insgesamt bis zu 3 Versuche).

Sonderfall: Der Erweiterungsmodul-Treiber schreibt bei seinem Start zur Synchronisation 0x7F ins High-Byte, gelesen wird aber immer 0.

Holding Register 3:

Wird nicht mehr verwendet.

Holding Register 4:

Das Register enthält einen Zähler für Timeout-Fehler, also wenn der Slave die Kommandos nicht beantwortet. Hier wird ebenfalls gezählt, wenn die Antwort kürzer als 3 Bytes ist.
Der Wertebereich ist 0…65535.
Die Grundeinstellung ist 0.

Holding Register 5:

Das Register enthält einen Zähler für Paritäts-, Framing- und CRC-Fehler, also wenn die Antwort des Slaves z.B. durch eine elektromagnetische Störung verändert wurde.
Der Wertebereich ist 0…65535.
Die Grundeinstellung ist 0.

Register für die automatische Einstellung von Baudrate und Parität bei einem Modbus-Slave von METZ CONNECT GmbH

Register 10-12 dienen zur nur Kommunikation mit dem Treiber für Erweiterungsmodule. Sie werden vom Treiber periodisch geprüft und geschrieben. Andere Programme sollen deshalb nicht in diese Register schreiben. Der Treiber wird über die Shared-Library mc-io-api bedient.

Die automatische Einstellung wird von beiden Modbus-Routern für beliebige Slave-Adressen unterstützt. Sie wird bisher nur für die Erweiterungsmodule ausgeführt (Adresse 1-6).

Beim Start des Erweiterungsmodul-Treibers endet eine unterbrochene Slave-Einstellung sofort.

Holding Register 10:

In das Register wird die Slave-Adresse geschrieben (Drehschalter des Geräts).

Holding Register 11:

In das Register wird der Wert 1 geschrieben, um die Einstellung des Slaves zu starten.

Der Modbus-Router versucht dann höchstens zweimal, bei allen möglichen Kombinationen von Baudrate und Parität, den Slave zu identifizieren. Wenn das Erfolg hatte, werden Baudrate und Parität im Slave auf die aktuellen Werte eingestellt, die in Holding Register 0 vorgegeben sind.

Der Modbus-Router meldet das Ende des Ablaufs in diesem Register.
Wert 0: Der Slave wurde erfolgreich eingestellt.
Wert 2: Der Slave konnte nicht gefunden oder eingestellt werden.

Holding Register 12:

In dem Register meldet der Modbus-Router den aktuellen Arbeitsschritt als Fortschrittsanzeige.
Der gesamte Ablauf dauert bis zu 9 Sekunden. Werte: 0…48.