Fahrschein für den Modbus

Modbus ist ein Kommunikationsprotokoll, welches als Quasi-Standard in der Automatisierungstechnik gilt. Es basiert auf binärer Datendarstellung und nutzt einen Master-Slave-Mechanismus (Client-Server-Architektur). Das Prinzip der Datenübertragung dabei ist einfach: Adresse, Befehlscode, Längenangabe, Dateninhalt und Checksumme bilden die Pakete. Dies lässt sich universell nutzen und erklärt die große Verbreitung.

Doch warum kommt es immer wieder zu Unstimmigkeiten in Sachen Modbus-Kommunikation?

Wer ist dieser Groß-Ender?

Nein, bei Modbus geht es nicht um Rotwild. Hier geht es nicht um Zwölfender, sondern um die Reihenfolge der Datendarstellung, also um die *„Endianness"* bzw. *„byte order"*.

Gerade weil Modbus so ein einfaches und universelles Protokoll ist, hat man dabei mit unterschiedlichen Rechner- und Speicherarchitekturen zu tun. Hier kommt die Endianness ins Spiel.

Modbus nutzt 16 bit-Register zum Datenaustausch. Die kleinste Einheit ist hier also 16 bit bzw. 2 Byte groß. In 8 bit-Architekturen werden daher zwei Speichereinheiten zusammengefasst. Hier stellt sich zum ersten Mal die Frage der Reihenfolge.

Wenn zwei Teilwerte zu einem Gesamtwert zusammengefügt werden, wie hier zwei 8 bit-Werte zu einem 16 bit-Wert, muss man sich über deren Reihenfolge Gedanken machen. Nehmen wir beispielsweise den Wert 4660 (oder hexadezimal 0x1234). Dieser 16 bit-Wert besteht aus den 8 bit-Werten 0x12 und 0x34. Im Speicher können diese aber so abgelegt sein:

Speicheradresse	Big-Endian	Little-Endian
1	0x12	0x34
2	0x34	0x12

Big-Endian bedeutet hier, dass der höchstwertige Teil an der niedrigsten Speicheradresse steht und somit die Darstellung sich an der klassischen, arabischen Zahlennotation orientiert. Die vorn stehenden Ziffern (z.B. Tausender) haben eine höhere Wertigkeit als die hinten (z.B. Einer).

Little-Endian bedeutet, dass der niederwertigste Teil an der niedrigsten Speicheradresse steht. Dies widerspricht der klassischen, arabischen Zahlennotation, folgt jedoch eher der Verarbeitungsreihenfolge bei Rechnungen, nämlich, dass wie bei der schriftlichen Addition an der Einer-Stelle begonnen wird. Arithmetisch kann dies Vorteile bringen und wird daher in allen x86-kompatiblen Rechnerarchitekturen verwendet, also auch Ihrem PC.

Als Analogie wird oft Datum und Uhrzeit angegeben. Das Datum ist im deutschsprachigem Raum Little-Endian, also der niederwertigste Wert Tag zuerst: 01.02.2003. Die Uhrzeit hingegen ist Big-Endian: 12:34.

Für den Modbus ist dies relevant weil fast immer eine serielle Kommunikation mit 8 Bit Datenbreite genutzt wird. Selbst bei Modbus TCP wird der Dateninhalt wie bei einer seriellen Kommunikation angeordnet.

Vermutlich wegen der Lesbarkeit im Datenstrom, vielleicht auch wegen der Architektur früherer SPSen, nutzt Modbus die Big-Endian-Darstellung. Daher wird das höherwertige Byte zuerst übertragen: erst die 0x12 und dann die 0x34. Trifft Modbus so auf eine typische x86-Architektur am PC kommt es zum Konflikt, da dieser zuerst die 0x34 im Speicher hat oder in den Speicher schieben will. Hier muss gedreht bzw. vertauscht werden (***„swapping"***).

Für 16 bit-Werte muss man sich also die Reihenfolge der Einzelbytes anschauen und ggf. swappen. Geht man auf größere Wertebereiche, wie 32 bit-Werte oder gar 64 bit-Werte, potenziert sich diese Variabilität. Jedoch wird man nicht alle Kombinationen finden, da die Architekturen sich in der Regel in ihren Endianness treu bleiben.

Aber nun konkret ein Beispiel: Nehmen wir die Zahl 305419896 oder hexadezimal 0x12345678. Die Bytereihenfolge kann dann so aussehen:

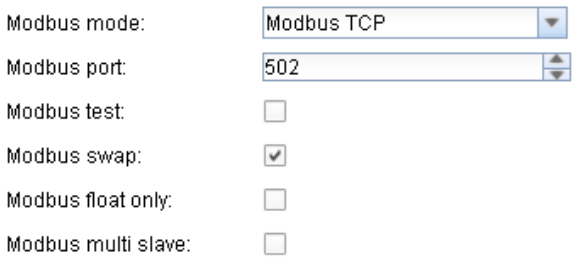
Reihen-folge	Modbus	Big-Endian*	Big-Mix*	Little-Mix*	Little-Endian
1	0x12	0x12	0x34	0x56	0x78
2	0x34	0x34	0x12	0x78	0x56
3	0x56	0x56	0x78	0x12	0x34
4	0x78	0x78	0x56	0x34	0x12

*Phantasienamen

Und somit ergibt sich schnell die Möglichkeit, dass eine Software auf einer SPS oder einem PC die Zahl nicht als 0x12, 0x34, 0x56, 0x78 im Modbus überträgt, sondern sei es aus Unachtsamkeit oder falschem Sachverstand eben als 0x78, 0x56, 0x34, 0x12 (dezimal 2018915346) oder gar eine der Mischformen. Da ist es gut, wenn man die Möglichkeit des Swappens hat, zumindest um von Big auf Little umzustellen.

Bei unseren Modbus-Geräten können Sie das Swapping von Big-Endian auf Little-Endian umstellen. Dafür dient

der Schalter „Modbus swap“. Ist dieser aktiviert, dann wird die Little-Endian-Darstellung genutzt.



Alles auf 0?

Eine weitere Falle des Modbus ist die Adressierung der Register bzw. der Speicherstellen. In obigen Tabellen taucht diese bereits auf: 1, 2, 3, 4. Der erste Wert steht in Zeile 1 oder eben Adresse 1. Dies ist jedoch in „Modbusianisch“ falsch. Da es ein binäres Protokoll ist, zählt man im Modbus logisch ab 0. Der erste Wert steht daher im Register mit Adresse 0.

Will man also den ersten Wert bzw. das erste Register abfragen, muss man die Adresse 0 nutzen. Auf binärer Ebene ist das einfach. Jetzt kommt aber der Mensch ins Spiel, dessen 1. Wert eben an der 1 liegt. Daher nutzen viele Modbus-Komponenten diese menschliche Adressierung und beginnen ab 1 zu zählen.

Verwirrenderweise wird aus dieser 1 dann auf Modbus allerdings „automatisch“ eine 0 im Adressfeld und die Gegenstelle antwortet dann entsprechend mit der 0.

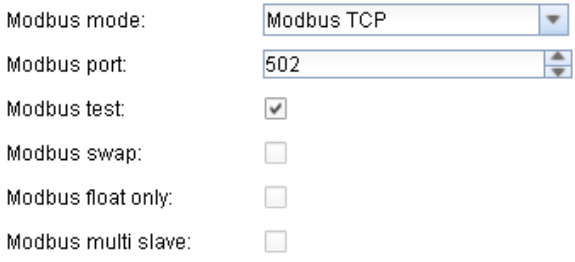
Hier müssen daher beide Seiten die gleiche Zählweise nutzen.

Bei unseren Geräten orientieren wir uns am Modbus und nutzen die Zählung ab 0. Daher haben zum Beispiel die ersten 10 Register auch die Adressen 0 bis 9. Eine Verschiebung um +1 ermöglichen wir nicht.

Falls die Gegenstelle ab 1 zählt, müssen Sie daher in dieser jeweils die Adresse um 1 im Gegensatz zu der Angabe in unseren Geräten erhöhen.

Wie kann ich das prüfen?

Speziell aus diesen beiden Aspekten heraus haben wir einen sogenannten Test-Modus in unsere Geräte integriert:



Hierbei wird ein statischer Antwort-Datensatz aktiviert, welcher zur Prüfung der Gegenstelle genutzt werden kann. Nur wenn die Zahlen exakt mit den Vorgaben aus unserem Handbuch übereinstimmen, dann stimmen Endianness und Zählweise überein.

Adresse	Wert	Beschreibung	Dekodierter Wert
0	0xD080	Seriennummer des Geräts, oberes Word	0xD0800DC1: letzte Stellen der MAC-Adresse: 68:91:D0:80:0D:C1
1	0x0DC1	Seriennummer des Geräts, unteres Word	
2	0x0002	Version des Kommunikationsprotokolls des Geräts	2
3	0x0084	Version der Software des Geräts	0x84 = 132: Version 1.32
4	0x5CE5	Systemzeit des Geräts (Zeitstempel), oberes Word	0x5CE55EAC = 1559054252: Mittwoch, 22. Mai 2019, 16:37:32 GMT+2
5	0x5EAC	Systemzeit des Geräts (Zeitstempel), unteres Word	
6	0x0000	Leerfeld	
7	0x0100	Typfeld des Registersatzes im oberen Byte	0x01: Eintrag des Typs Gerät
8	0x0000	Leerfeld	
9	0x0000	Leerfeld	
10	0x00BC	Seriennummer des Zählers, oberes Word	0xBC614E = 12345678
11	0x614E	Seriennummer des Zählers, unteres Word	
12	0x0443	Herstellerkürzel des Zählers (siehe Abschnitt 10.3)	0x0443: ABC
13	0x0102	Version (oberes Byte) und Medium (unteres Byte) des Zählers	0x01: Version = 1, 0x02: Medium = 2 (Elektrizität)
14	0x5CE5	Auslesezeitpunkt des Zählers (Zeitstempel), oberes Word	0x5CE55EAC = 1559054252: Mittwoch, 22. Mai 2019, 16:37:32 GMT+2
15	0x5EAC	Auslesezeitpunkt des Zählers (Zeitstempel), unteres Word	
16	0x0000	Leerfeld	
17	0x0200	Typfeld des Registersatzes im oberen Byte	0x02: Eintrag des Typs Zähler
18	0x0000	Flags im unteren Byte	0x00: Zähler korrekt ausgelesen und alle Werte aktuell
19	0x0000	Leerfeld	
20	0x0000	Zählerwert (Ganzzahl), höchstes Word	0xBC614E = 12345678: Resultierender Zählerwert: 12345678 * 10^{-4} = 1234,5678 Wh
21	0x0000	Zählerwert (Ganzzahl)	
22	0x00BC	Zählerwert (Ganzzahl)	
23	0x614E	Zählerwert (Ganzzahl), niedrigstes Word	
24	0x449A	Zählerwert (Gleitkomma), oberes Word	0x449A522B = 1234,5677490234375 (Rundungsfehler bei *FLOAT32*)
25	0x522B	Zählerwert (Gleitkomma), unteres Word	
26	0xFFFC	Skalierungsfaktor (Exponent zur Basis 10)	0xFFFC = -4: Faktor = 10^{-4}
27	0x0005	Typfeld des Registersatzes im oberen Byte und Einheit im unteren Byte (siehe Tabelle 25)	0x00: Eintrag des Typs Zählerwert 0x05: Einheit = Wh
28	0x5CE5	Zeitpunkt des Zählerwerts (Zeitstempel), oberes Word	0x5CE55EAC = 1559054252: Mittwoch, 22. Mai 2019, 16:37:32 GMT+2
29	0x5EAC	Zeitpunkt des Zählerwerts (Zeitstempel), unteres Word	

Sofern Ihre Gegenstelle Float32 als Datenformat unterstützt, empfehlen wir stets die Prüfung anhand der Register 24 und 25.

Wenn die Zahl in Ihrer Gegenstelle nicht genau der 1234.5677490234375 entspricht (ja, es können bei Float32 kleine Abweichungen auftreten), stimmt etwas nicht. Würden Sie beispielsweise eine 237810783920322510848 angezeigt bekommen, würden Sie die Register 23 und 24 auslesen, was für eine Zählweise ab 1 spricht.

Für die Prüfung der Float32-Werte kann man u.a. folgende Seite nutzen: https://www.h-schmidt.net/FloatConverter/IEEE754.html