Modbus TCP

sailer
Energieeffiziente Warmwasser-Systeme

Technische Dokumentation

SAILER GmbH
Zementwerkstraße 17A
D-89584 Ehingen

Tel.: 07391 / 5002 0
Fax.: 07391 / 5002 29

www.SailerGmbH.de

info@SailerGmbH.de

Diese Bedienungsanleitung ist Teil des Produkts.

- Bedienungsanleitung vor Gebrauch sorgfältig lesen,
- während der Lebensdauer des Produkts aufbewahren,
- an jeden nachfolgenden Besitzer oder Benutzer des Produktes weitergeben.

Version der Schnittstelle: 0.11Stand 11.2011

Inhalt dieser Dokumentation

1 Allgemeines zu Modbus TCP

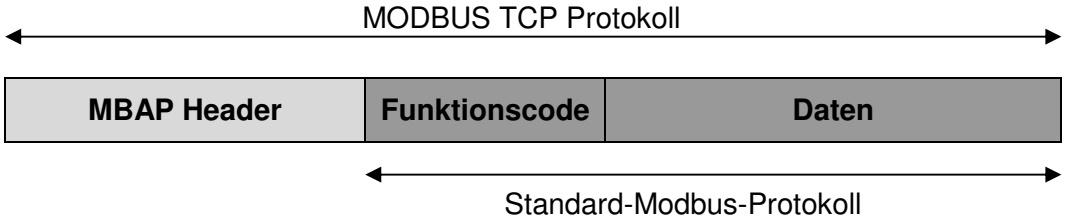
Modbus TCP ermöglicht die Übertragung von Modbusnachrichten über ein TCP/IP-Netzwerk. Modbus TCP befindet sich zur Zeit in der Phase der Festlegung als Norm (IEC PAS 62030 (pre-standard)).
In die Regler ist ein Modbus-Server (entspricht einem Modbus-Slave) implementiert, mehrere Clients (entspricht einem Modbus-Master) können sich zeitgleich verbinden. Vorliegendes Dokument beschreibt das implementierte Modbus TCP Protokoll im Allgemeinen und bezogen auf die spezielle Anwendung im FRIWASTA Manager und FRIWASTA Master.

2 Verbindungsaufbau mit einem Modbus-Server

Die Modbus-Server akzeptieren auf Port 502 Anfragen von Modbus-Clients. Bis zu drei Modbus-Clients können sich zeitgleich mit einem Server (Regler) verbinden. Alle weiteren Anfragen werden abgewiesen.

3 Protokollaufbau Modbus TCP

Das Protokoll setzt sich aus dem Standard-Modbus-Protokoll und dem MBAP (Modbus Application) Header zusammen.



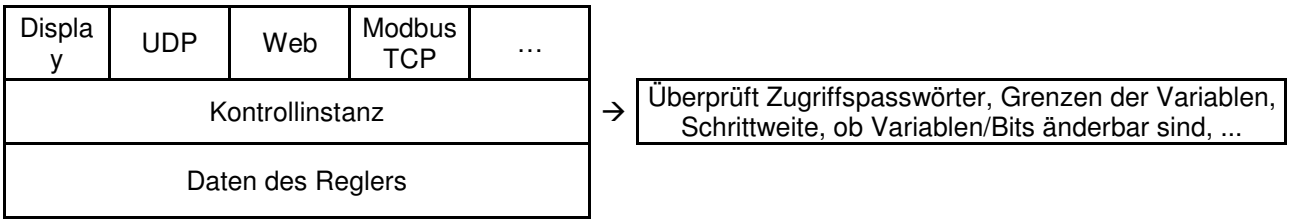
Das Standard-Modbus-Protokoll setzt sich aus dem Funktionscode (1 Byte) und den Daten zusammen. Die Daten werden abhängig vom Funktionscode interpretiert. Die implementierten Funktionen werden an anderer Stelle im Detail beschrieben.

Der MBAP-Header setzt sich aus folgenden Feldern zusammen:

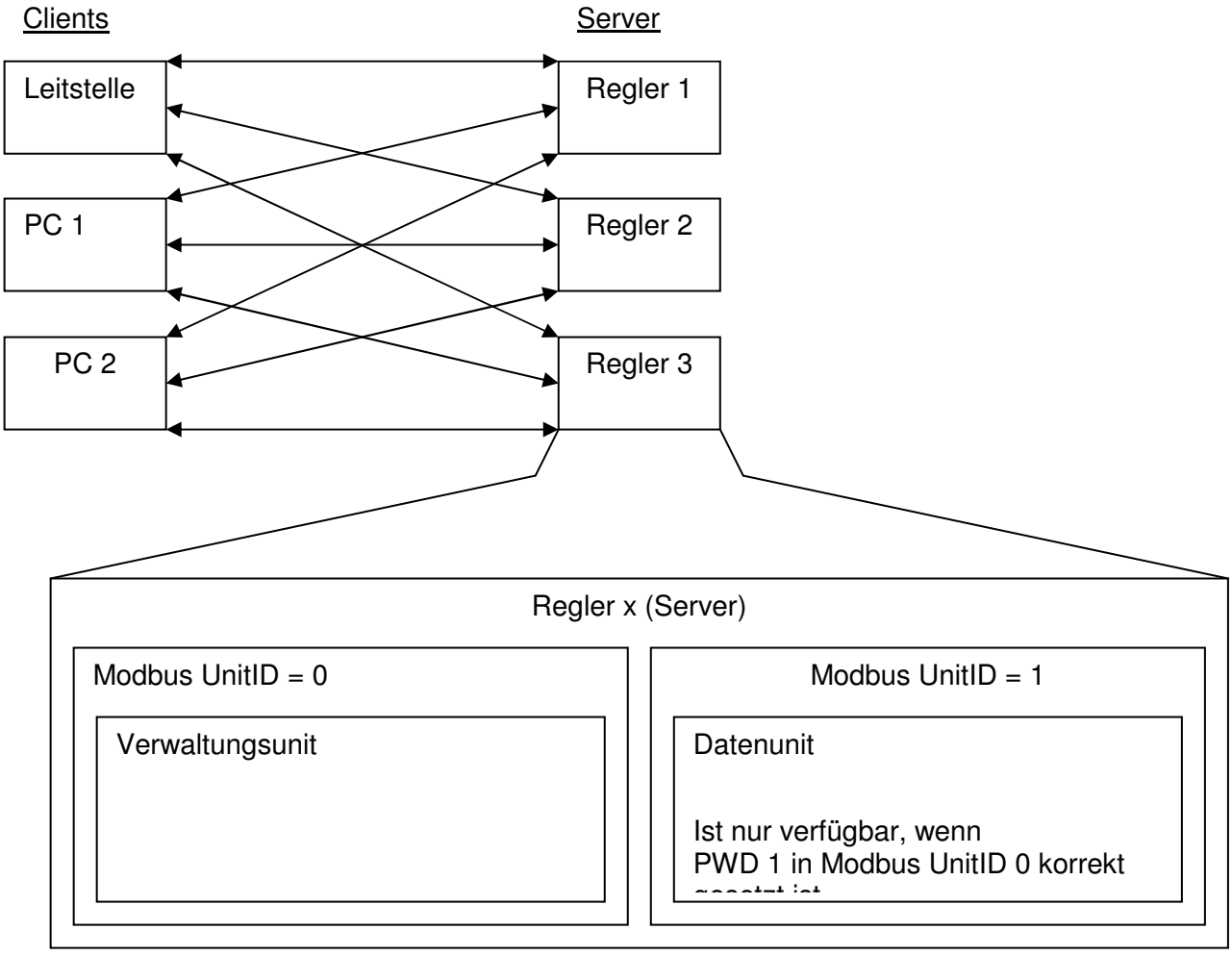
Feld	Länge [Byte]	Beschreibung	Client	Server	Besonderzeiten bei der Implementierung
Transaction Identifier	2	Identifikation einer Modbus-Anfrage, wenn zeitgleich mehrere Anfragen gesendet werden. Hiermit kann die Antwort der jeweiligen Anfrage zugeordnet werden.	Initialisiert vom Client	Der Server kopiert dieses Feld in die Antwort, damit diese zugeordnet werden kann.	
Protocol Identifier	2	0 = MODBUS Protokoll	Initialisiert vom Client	Der Server kopiert dieses Feld in die Antwort.	
Length	2	Anzahl der noch folgenden Bytes	Initialisiert vom Client (Anfrage)	Initialisiert vom Server (Antwort)	
Unit Identifier	1	Identifikation eines Slaves, der mit dem Server z.B. seriell verbunden ist.	Initialisiert vom Client	Der Server kopiert dieses Feld in die Antwort.	Unit ID = 0 →Virtueller Knoten 0, wird für die Verwaltungs- und Zugangssicherung verwendet Unit ID = 1 →Virtueller Knoten 1, wird für die Bereitstellung der Daten verwendet

4 Zugriff auf interne Variablen

Der Zugriff auf die Daten des Reglers lässt sich als Schichtenmodel darstellen. Die oberste Schicht entspricht der Art des Zugriffs. In diese Schicht fällt auch die Modbus-TCP-Schnittstelle. Die darunter liegende Schicht prüft, ob der Zugriff auf die Daten des Reglers erlaubt ist und ob z.B. Variablengrenzen und Schrittweite eingehalten wird. In der untersten Schicht werden dann die Daten des Reglers gelesen und ggf. geschrieben.



5 Netzwerkübersicht



6 Unit ID 0

Der virtuelle Modbusknoten mit der Unit ID 0 dient dazu, Verwaltungsvariablen bereitzustellen und die Zugangssicherung zu verwalten.

Folgende Variablen sind in Unit ID 0 verfügbar:

Variablen-name	Format	Start bei Register	Anzahl Register	Beschreibung	Zugriffsbeschränkung
UDP-Sicherheit	bit	1	1	Aktiviert die UDP-Sicherheit, um unerlaubten Zugriff auf: - Download/Konfiguration (Port 8001) - Kommunikation (konfigurierbarer Port) zu verhindern.	Lesen immer Schreiben nur mit gesetztem PWD1
MAC-Adresse	byte-Array	40001	3	MAC-Adresse des Reglers. 6 Byte → 3 Register	Lesen immer Schreiben nicht möglich
Modbus-Name	char-Array	40004	10	Modbus-Name des Reglers. Dient nur der Anzeige, optional: 19 Zeichen + Nullterminierung	Lesen immer Schreiben nur mit gesetztem PWD1
PWD1	ulong	40014	2	Setzen des Modbuspasswortes PWD1 Das Passwort lautet bei Auslieferung als String „1234“, als ulong „0x31323334“ und dezimal „825373492“. Das Passwort kann im Reglermenü auf den Auslieferungszustand zurückgesetzt werden. PWD1 verliert seine Gültigkeit, wenn der TCP-Socket geschlossen wird.	Lesen nicht möglich Schreiben geht immer
PWD2	ulong	40016	2	Setzen des Berechtigungspasswortes PWD2 PWD2 verliert seine Gültigkeit, wenn der TCP-Socket geschlossen wird.	Lesen nicht möglich Schreiben nur mit gesetztem PWD1
Userlevel	uword	40018	1	Lesen des aktuellen Userlevels Userlevel = 0: Weder PWD 1 noch PWD 2 sind gesetzt, UnitID 1 ist gesperrt und der Name/PWD1 und die UDP-Sicherheit können nicht geändert werden. Userlevel = 1: PWD1 ist korrekt gesetzt, PWD2 ist nicht gesetzt, Unit ID 1 ist freigeschaltet und alle Variablen und Bits sind lesbar, die ohne Berechtigung änderbaren Variablen sind auch schreibbar. Userlevel >= 2: Beide PWD sind gesetzt, hier wird die aktuelle Berechtigungsebene zurückgegeben.	Lesen immer Schreiben nicht möglich
Modbus-Version	uword	40019	1	Version der Modbus-Implementierung	Lesen immer
PWD1 ändern	ulong	40020	2	Ändern des Modbuspasswortes PWD1 Die Werte 0x00000000 und 0xFFFFFFFF sind nicht zulässig.	Lesen nicht möglich Schreiben nur mit gesetztem PWD1

7 Unit ID 1

Der virtuelle Modbusknoten mit der Unit ID 1 ermöglicht den Zugriff auf die Daten des Reglers. Unit ID 1 ist nur dann verfügbar, wenn in Unit ID 0 das Passwort PWD1 korrekt gesetzt ist.

Folgende Variablenbereiche sind in Unit ID 1 verfügbar:

Variablen-gruppe	Format	Start bei Register	Anzahl Register	Beschreibung	Zugriffsbeschränkung
Interne Bits des Reglers	Bit	1	Abhängig von den verfügbaren Bits des Reglers	Zugriff auf die internen Bits des Reglers. (Siehe Liste der Arbeitsbits in Anhang A2 und Liste der Einstellbits in Anhang A3) Registernummer = BitNr+1	Lesen nur mit gesetztem PWD1 Schreiben abhängig von der Applikation und dem aktuellen Userlevel
Ausgänge des Reglers	Bit	9001	Abhängig von den verfügbaren Ausgängen des Reglers	Zugriff auf die Ausgänge des Reglers. Jeder Ausgang wird durch zwei Bits repräsentiert: 1. Bit: Betriebsmodus 0 = Automatik-Betrieb 1 = Manueller Betrieb 2. Bit: Aktueller Zustand Register 9001: Betriebsmodus Ausgang 1 Register 9002: Zustand Ausgang 1 Register 9003: Betriebsmodus Ausgang 2 Register 9004: Zustand Ausgang 2 ...	Lesen nur mit gesetztem PWD1 Schreiben abhängig von der Applikation und dem aktuellen Userlevel
Digitale Eingänge des Reglers	Bit	10001	Abhängig von den verfügbaren Eingängen des Reglers	Zugriff auf die digitalen Eingänge des Reglers. Registernummer = EingangsNr+10001	Lesen nur mit gesetztem PWD1 Schreiben nicht möglich
Messwerte (Temp.-Fühler)	word	30001	Abhängig von den verfügbaren Fühlern des Reglers	Zugriff auf die Fühlerwerte des Reglers. Für jeder Fühler gibt es ein Register, die Skalierung beträgt [1/10 °C]. Registernummer = TempNr+30001	Lesen nur mit gesetztem PWD1 Schreiben nicht möglich
Variablen	long bzw. ulong	40001	Abhängig von den verfügbaren Variablen des Reglers	Zugriff auf die komplette Variablentabelle des Reglers. Jede Variable belegt zwei Register, Variablen, die mehr als vier Bytes (z.B. Arrays, Strings, Schaltuhren, ...) benötigen, werden nicht unterstützt. Bei einem Zugriff auf diese Variablen wird der Wert 0 gelesen. Einzelne Variablen müssen als ulong interpretiert werden, wie die Variablen zu interpretieren sind ist der Tabelle im Anhang A1 zu entnehmen. Registernummer = VarNr*2+40001	Lesen nur mit gesetztem PWD1 Schreiben abhängig von der Applikation und dem aktuellem Userlevel

8 Implementierte Kommandos

8.1 Kommandoübersicht

Folgende Kommandos sind implementiert:

Kommando	Code	Verwendung	Besonderheiten
Read Coils/internal Bits	0x01	Unit 0: Lesen des Sicherheitsbits für UDP Unit 1: Lesen der internen Bits Lesen der Ausgangsmodi / -zustände	
Read input status	0x02	Unit 0: Keine Verwendung Unit 1: Lesen der digitalen Eingänge	
Read holding/output Registers	0x03	Unit 0: Lesen von MAC-Adresse, Modbus-Name, Userlevel und Modbus-Version Unit 1: Lesen von Variablen	Beim Lesen der MAC-Adresse müssen immer alle 3 Register in einem Befehl gelesen werden. Beim Lesen des Modbus-Namens müssen immer alle 10 Register in einem Befehl gelesen werden.
Read Input Registers	0x04	Unit 0: Keine Verwendung Unit 1: Lesen der Werte der Temperaturfühler	
Write single Coil/Bit	0x05	Unit 0: Schreiben des Sicherheitsbits für UDP Unit 1: Schreiben der internen Bits Schreiben der Ausgangsmodi / -zustände	
Write multiple Coils/Bits	0x0F	Unit 0: Keine Verwendung Unit 1: Schreiben der Ausgangsmodi / -zustände	Ausgangsbits können nur in 2-Bit-Blöcken geschrieben werden (Modus und Zustand). Wird auf den Modus „Automatik“ geschaltet, entfällt die Prüfung, ob der Ausgang verfügbar ist, damit bei gleichzeitigem Zugriff auf mehrere Ausgänge kein Fehler zurückgegeben wird, wenn dazwischen ein nicht verfügbarer Ausgang liegt.
Write multiple Registers	0x10	Unit 0: Schreiben von Modbus-Name, PWD1, PWD2 und Ändern von PWD1 PWD1 und PWD2 können auch innerhalb eines Schreibbefehles gesetzt werden. Unit 1: Schreiben von Variablen	Beim Schreiben des Modbus-Namens müssen immer alle 10 Register in einem Befehl geschrieben werden.

8.2 Kommandos

Im Folgenden werden die verwendeten Befehle im Detail beschrieben.

8.2.1 Read Coils/internal Bits (0x01)

Mit diesem Befehl kann ein oder mehrere zusammenhängende Bits gelesen werden. Das LSB des ersten Bytes beinhaltet das Bit der Startadresse. Die weiteren Bits werden aufsteigend in das erste Byte übernommen. Das 9. Bit wird in das LSB des zweiten Bytes gelegt usw. Sind beim letzten Byte noch Bits unbelegt, werden diese als Null übertragen.

Dabei gilt: Startadresse = Startregisternummer – 1

Anfrage:

Funktionscode	1 Byte	0x01
Startadresse	2 Bytes	0x0000 bis 0xFFFF
Anzahl der Coils/Bits	2 Bytes	1 bis 2000 (0x7D0)

Antwort:

Funktionscode	1 Byte	0x01
Anzahl der Bytes	1 Byte	N
Status der Coils/Bits	N Bytes	

Fehler:

Funktionscode	1 Byte	0x81
Fehlercode	1 Byte	01, 02, 03 oder 04

8.2.2 Read input status (0x02)

Mit diesem Befehl kann ein oder mehrere zusammenhängende Eingänge gelesen werden. Das LSB des ersten Bytes beinhaltet den Eingang an der Startadresse. Die weiteren Eingänge werden aufsteigend in die Bits des ersten Byte übernommen. Der 9. Eingang wird in das LSB des zweiten Bytes gelegt usw. Sind beim letzten Byte noch Bits unbelegt, werden diese als Null übertragen.

Dabei gilt: Startadresse = Startregisternummer – 10001

Anfrage:

Funktionscode	1 Byte	0x02
Startadresse	2 Bytes	0x0000 bis 0xFFFF
Anzahl der Eingänge	2 Bytes	1 bis 2000 (0x7D0)

Antwort:

Funktionscode	1 Byte	0x02
Anzahl der Bytes	1 Byte	N
Status der Eingänge	N Bytes	

Fehler:

Funktionscode	1 Byte	0x82
Fehlercode	1 Byte	01, 02, 03 oder 04

8.2.3 Read holding/output Registers (0x03)

Mit diesem Befehl kann ein oder mehrere zusammenhängende Register gelesen werden. Die Registerwerte sind immer 16 Bit breit, das erste gesendete Byte ist das High-Byte, gefolgt vom Low-Byte.

Dabei gilt: Startadresse = Startregisternummer – 40001

Anfrage:

Funktionscode	1 Byte	0x03
Startadresse	2 Bytes	0x0000 bis 0xFFFF
Anzahl der Register	2 Bytes	1 bis 125 (0x7D)

Antwort:

Funktionscode	1 Byte	0x03
Anzahl der Bytes	1 Byte	2 x N
Registerwert	N x 2 Bytes	

Fehler:

Funktionscode	1 Byte	0x83
Fehlercode	1 Byte	01, 02, 03 oder 04

8.2.4 Read Input Registers (0x04)

Mit diesem Befehl kann ein oder mehrere zusammenhängende Messwertregister gelesen werden. Die Registerwerte sind immer 16 Bit breit, das erste gesendete Byte ist das High-Byte, gefolgt vom Low-Byte.

Dabei gilt: Startadresse = Startregisternummer – 30001

Anfrage:

Funktionscode	1 Byte	0x04
Startadresse	2 Bytes	0x0000 bis 0xFFFF
Anzahl der Register	2 Bytes	1 bis 125 (0x7D)

Antwort:

Funktionscode	1 Byte	0x04
Anzahl der Bytes	1 Byte	2 x N
Registerwert	N x 2 Bytes	

Fehler:

Funktionscode	1 Byte	0x84
Fehlercode	1 Byte	01, 02, 03 oder 04

8.2.5 Write single Coil/Bit (0x05)

Mit diesem Befehl kann ein einzelnes Bit gesetzt werden. Soll das Bit auf logisch 0 gesetzt werden, muss der Bit-Wert 0x0000 enthalten, soll das Bit auf logisch 1 gesetzt werden, muss der Bit-Wert 0xFF00 enthalten. Alle anderen Werte sind ungültig.

Dabei gilt: Bit-Adresse = Registernummer – 1

Anfrage:

Funktionscode	1 Byte	0x05
Bit-Adresse	2 Bytes	0x0000 bis 0xFFFF
Bit-Wert	2 Bytes	0x0000 oder 0xFF00

Antwort:

Funktionscode	1 Byte	0x05
Bit-Adresse	2 Bytes	0x0000 bis 0xFFFF
Bit-Wert	2 Bytes	0x0000 oder 0xFF00

Fehler:

Funktionscode	1 Byte	0x85
Fehlercode	1 Byte	01, 02, 03 oder 04

8.2.6 Write multiple Coils/Bits (0x0F)

Mit diesem Befehl kann ein oder mehrere zusammenhängende Bits geschrieben werden. Das LSB des ersten Bytes beinhaltet das Bit der Startadresse. Die weiteren Bits werden aufsteigend im ersten Byte übergeben. Das 9. Bit wird in das LSB des zweiten Bytes gelegt usw. Sind beim letzten Byte noch Bits unbelegt, werden diese ignoriert.

Dabei gilt: Startadresse = Startregisternummer –1

Anfrage:

Funktionscode	1 Byte	0x0F
Startadresse	2 Bytes	0x0000 bis 0xFFFF
Anzahl der Coils/Bits	2 Bytes	1 bis 1968 (0x7B0)
Anzahl der Bytes	1 Byte	N
Bit-Werte	N Bytes	

Antwort:

Funktionscode	1 Byte	0x0F
Startadresse	2 Bytes	0x0000 bis 0xFFFF
Anzahl der Coils/Bits	2 Bytes	1 bis 1968 (0x7B0)

Fehler:

Funktionscode	1 Byte	0x8F
Fehlercode	1 Byte	01, 02, 03 oder 04

8.2.7 Write multiple Registers (0x10)

Mit diesem Befehl kann ein oder mehrere zusammenhängende Register geschrieben werden. Die Registerwerte sind immer 16 Bit breit, das erste gesendete Byte ist das High-Byte, gefolgt vom Low-Byte.

Dabei gilt: Startadresse = Startregisternummer – 40001

Anfrage:

Funktionscode	1 Byte	0x10
Startadresse	2 Bytes	0x0000 bis 0xFFFF
Anzahl der Register	2 Bytes	1 bis 123 (0x7B)
Anzahl der Bytes	1 Byte	2 x N
Registerwert	N Bytes	

Antwort:

Funktionscode	1 Byte	0x10
Startadresse	2 Bytes	0x0000 bis 0xFFFF
Anzahl der Register	2 Bytes	1 bis 123 (0x7B)

Fehler:

Funktionscode	1 Byte	0x90
Fehlercode	1 Byte	01, 02, 03 oder 04

9 Variablenliste