



Programmierhandbuch

PS 2000 B Serie

2020er TFT-Modelle

Elektro-Automatik



**Achtung! Dieses Dokument gilt
nur für die in 2020 überarbeiteten
Modelle der PS 2000 B Serie
mit farbigem TFT-Display**

Doc ID: PG2DE
Revision: 8
Stand: 23.06.2020

INHALTVERZEICHNIS

| | |
|--|----------|
| 1. ALLGEMEINES | 3 |
| 1.1 Einleitung..... | 3 |
| 1.2 Treiberinstallation | 3 |
| 1.2.1 Windows..... | 3 |
| 1.2.2 Linux, MacOS und andere | 3 |
| 1.3 Begriffe | 4 |
| 2. ALLGEMEINES ZUR KOMMUNIKATION MIT DEM GERÄT | 4 |
| 2.1 Aufbau der Kommunikation | 4 |
| 2.2 Serielle Übertragungsparameter | 4 |
| 2.3 Werteumrechnung | 4 |
| 2.3.1 Istwerte umrechnen..... | 4 |
| 2.3.2 Sollwerte umrechnen | 4 |
| 2.4 Vorgehensweise | 5 |
| 2.5 Effektive Auflösung beim Programmieren | 5 |
| 3. NACHRICHTENFORMATE | 6 |
| 3.1 Altes, binäres Format | 6 |
| 3.1.1 Telegrammaufbau | 6 |
| 3.1.2 Der Startdelimiter im Einzelnen..... | 7 |
| 3.1.3 Das Maskenbyte für Objekt 54..... | 7 |
| 3.1.4 Telegrammbeispiele | 7 |
| 3.1.5 Mögliche Probleme beim Setzen von Zuständen..... | 8 |
| 3.1.6 Rückmeldungen und Kommunikationsfehler..... | 8 |
| 3.1.7 Fehlerbehandlung | 9 |
| 3.1.8 Objektliste | 9 |
| 3.2 ModBus RTU..... | 10 |
| 3.2.1 Vorwort..... | 10 |
| 3.2.2 Allgemeines zu ModBus RTU | 10 |
| 3.2.3 Über die Register-Liste | 10 |
| 3.2.4 Telegrammtypen..... | 11 |
| 3.2.5 Funktionen | 11 |
| 3.2.6 Sendetelegramm (Schreiben) | 11 |
| 3.2.7 Anfragetelegramm..... | 12 |
| 3.2.8 Antworttelegramm (Lesen)..... | 12 |
| 3.2.9 Die ModBus-Checksumme | 13 |
| 3.2.10 Kommunikationsfehler..... | 13 |
| 3.2.11 Beispiele für ModBus RTU-Telegramme | 14 |
| 3.3 SCPI..... | 16 |
| 3.3.1 Format der Soll- und Istwerte..... | 16 |
| 3.3.2 Syntax | 16 |
| 3.3.3 Befehlsverkettung | 16 |
| 3.3.4 Groß-/Kleinschreibung | 16 |
| 3.3.5 Langform und Kurzform | 17 |
| 3.3.6 Abschlußzeichen..... | 17 |
| 3.3.7 Kommunikationsfehler..... | 17 |
| 3.3.8 Standard-IEEE-Befehle..... | 17 |
| 3.3.9 Statusregister | 18 |
| 3.3.10 Adressierung der Ausgänge..... | 19 |
| 3.3.11 Sollwertbefehle..... | 20 |
| 3.3.12 Meßbefehle | 20 |
| 3.3.13 Zustandsbefehle..... | 20 |
| 3.3.14 Befehle für Schutzfunktionen | 21 |
| 3.3.15 Befehle für Einstellungsgrenzen | 21 |
| 3.3.16 Befehle zum Tracking-Modus..... | 22 |
| 3.3.17 Weitere Befehle..... | 22 |

1. Allgemeines

1.1 Einleitung

Diese Anleitung dient zur Erläuterung der Kommunikation mit einem Gerät der in 2020 überarbeiteten Serie PS 2000 B. Diese Geräte unterstützen drei Nachrichtenformate bzw. Befehlssprachen:

- Bisheriges (altes) binäres EA-Nachrichtenformat (identisch zu dem der vorherigen PS 2000 B Generation)
- SCPI
- ModBus RTU

Die Unterscheidung der Protokolle anhand des Formats einer Nachricht erfolgt automatisch und aufgrund von ein paar Regeln:

- Bei ModBus muß die Slave-Adresse im ersten Byte entweder 0x00 oder 0x01 sein, auch wenn sie gar nicht zur Adressierung des Gerätes dient (ModBus-Tools setzen meist Adresse 0x01 als Standard) -> ist das erste Byte 0x00 oder 0x01, dann wird die Nachricht als eine ModBus-RTU-Nachricht interpretiert und verarbeitet
- Beim binären Protokoll muß das zweite Byte einer Nachricht 0x00 oder 0x01 sein, um Ausgang 1 oder Ausgang 2 zu adressieren (bei einem Single-Modell wäre das Byte daher immer 0x00) -> ist das zweite Byte 0x00 oder 0x01, dann wird die Nachricht als eine binäre Nachricht interpretiert und verarbeitet
- Wenn das erste Byte einen Wert >1 enthält und das zweite Byte auch, dann wird die Nachricht als ein string-basierter SCPI-Befehl interpretiert und verarbeitet
- Jede andere Situation wird als ungültige Nachricht interpretiert

Kommunikation erfolgt nur über den frontseitigen USB-Port, der die einzige digitale Schnittstelle darstellt. Der USB-Treiber erzeugt auf der PC-Seite für das Gerät einen virtuellen COM-Port (VCP), über den die Kommunikation stattfindet. Die Verwendung des virtuellen COM-Ports vereinfacht die Ansteuerung des Ports auf ein Minimum, weil er nicht konfiguriert werden muß bzw. jede Konfiguration funktioniert ist, also auch Standardwerte.

1.2 Treiberinstallation

1.2.1 Windows

Der Treiber kommt in einer installierbaren Datei auf USB-Stick mit dem Gerät. Sollte der Stick fehlen, ist der Treiber auch auf unserer Webseite zu finden. Er ist kompatibel zu allen Windows-Versionen ab 7 (außer Embedded).

Wir empfehlen, den Treiber zu installieren, bevor das Gerät das erste Mal mit dem PC verbunden wird, damit Windows diesen und nicht einen anderen Treiber installiert, was rein auf die Funktion des COM-Ports bezogen vorerst kein Problem darstellt. Spätestens wenn das Gerät mal in LabView eingebunden und unsere angebotenen VIs verwendet werden sollen, ist die korrekte Serienbezeichnung im Gerätemanager wichtig.

Nach der erfolgreichen Installation kann das Gerät verbunden werden. Windows sollte daraufhin das Gerät im System installieren. Zwecks Überprüfung kann man den Windows Gerätemanager aufrufen (in Windows 10 über Rechtsklick auf den Startmenü-Button und in der Auswahl) und unter „Anschlüsse“ nach einem „PS 2000 Series“ Gerät schauen:

 **PS 2000 Single (COM8)** oder PS 2000 Triple (COM8), je nach Modell

Hinweis: Die COM-Port-Nummer hier ist ein Beispiel. Windows vergibt für jedes dieser Geräte einen neuen COM-Port, den es sich für das Gerät „merkt“.

Hinweis: Falls mehrere Geräte am PC angeschlossen sind, erscheinen auch mehrere „PS2000 Single“ im Gerätemanager.

1.2.2 Linux, MacOS und andere

Wir bieten keinen proprietären Treiber für andere Systeme als Windows an, das ist aber auch nicht unbedingt nötig. Es gibt generische Treiber für zumindest Linux und MacOS. Sollte die automatische Erkennung nicht funktionieren, wäre der manuell zu wählende Hardwaretyp CDC (communication device class). Es entsteht durch den generischen Treiber nur eine Problematik bei Verwendung von LabView: unsere VIs funktionieren nicht bzw. nur zum Teil, weil das Such-VI, das zum Finden und Erkennen dient, auf Daten aus der hier nicht vorhandenen Windows Registry angewiesen ist.

1.3 Begriffe

Telegramm / Nachricht = Kette von hexadezimalen Bytes, mit unterschiedlicher Länge. Wird entweder zum Gerät gesendet oder vom Gerät empfangen. Ein Teil des Telegramms (Datenbereich) repräsentiert entweder hexadezimale Werte oder ASCII-Strings.

SCPI = **S**tandard **C**ommands for **P**rogrammable Instruments, eine international standardisierte, text-/stringbasierte Befehlssprache

ModBus / ModBus RTU = Binäres, international bekanntes, spezifiziertes Datenübertragungsformat, das auf verschiedenen Subformaten wie RTU basiert

2. Allgemeines zur Kommunikation mit dem Gerät

2.1 Aufbau der Kommunikation

Die Kommunikation mit dem zu steuernden Gerät basiert auf diesen zwei Telegrammtypen:

a) **Sendung**: es wird ein Objekt bzw. ein Befehl gesendet, der z. B. einen Sollwert setzen soll. Sofern dies im momentanen Betriebszustand des Gerätes zulässig ist, wird der Befehl akzeptiert und ausgeführt. Das Gerät sendet dann als Antwort:

- beim alten, binären Format eine Fehlermeldung mit Fehlercode 0. Falls die Ausführung des Befehls nicht zulässig oder das Telegramm fehlerhaft ist, kommt eine Fehlermeldung mit einem Fehlercode ungleich 0.
- bei SCPI nichts
- bei ModBus eine Bestätigungsnachricht (unterschiedliche Formate, je nach Funktionscode)

b) **Anfrage**: es wird mittels eines Objekts eine Anfrage an das Gerät gesendet, worauf man eine Antwort erwartet, welche die angefragten Daten enthält. Bei einem Kommunikationsfehler käme stattdessen eine Fehlermeldung (altes, binäres Format / ModBus) oder gar nichts (SCPI).

2.2 Serielle Übertragungsparameter

Die Übertragung von Daten erfolgt trotz USB-Schnittstelle im Stil eines seriellen COM-Ports, also byteweise. Da der vom Treiber im System angelegte COM-Port ein virtueller ist, sind die seriellen Einstellparameter irrelevant. Der Treiber ignoriert diese. Je nach verwendeter IDE kann es also entfallen, den COM-Port konfigurieren zu müssen.

2.3 Werteumrechnung

Beim alten binären Format und bei auch ModBus werden Sollwerte und Istwerte als Prozentwerte übertragen. Ein Wert von 0x6400 (altes, binäres Format) bzw. 0xCCCC (ModBus) entspricht 100,00%. Die andere Obergrenze bei ModBus ist notwendig, damit diese Geräteserie bei ModBus kompatibel zu unserer Software und anderen Serien ist.

2.3.1 Istwerte umrechnen

Istwerte werden aus dem Gerät gelesen und liegen nach Erhalt der Antwort als hexadezimaler 16-Bit-Wert vor, der einen Prozentwert repräsentiert.

$$\text{Realer Istwert} = \frac{\text{Nennwert} * \text{Prozent-Istwert}}{\text{Umrechnungsfaktor}}$$

Umrechnungsfaktoren

- Altes binäres Format: 26500 oder 0x6400
ModBus: 52428 oder 0xCCCC

Beispiel: Der Spannungsnennwert des Gerätes ist 42V, der prozentuale Istwert kam als 0x2454 = 9300. Nach der Formel ergibt sich bei Umrechnung aus dem ModBus-Format der reale Istwert $42 * 9300 / 52428 = 7,45 \text{ V}$.

2.3.2 Sollwerte umrechnen

Sollwerte müssen als hexadezimaler 16-Bit-Prozentwert gesendet werden und sind daher vor dem Senden umzurechnen. Beim Auslesen verhält es sich wie bei Istwerten.

$$\text{Prozentualer Sollwert} = \frac{\text{Umrechnungsfaktor} * \text{Realer Sollwert}}{\text{Nennwert des Gerätes}}$$

Umrechnungsfaktoren

- Altes binäres Format: 26500 oder 0x6400
ModBus: 52428 oder 0xCCCC

Beispiel: der Sollwert soll 13,5 A sein, der Stromnennwert des Gerätes ist 20 A. Nach der Formel ergibt sich ein Prozent-Sollwert von $25600 * 13,5 / 20 = 17280 = 0x4380$, wenn für das alte binäre Format umgerechnet. Nachdem der Sollwert 0x4380 an das Gerät gesendet wurde, sollte ein 20 A-Modell die gewünschten 13,5 A am adressierten Ausgang einstellen.

2.4 Vorgehensweise

Generell gilt:

- Überwachung (Monitoring), also nur Abfrage von Istwerten und Status, ist immer möglich. Das Gerät benötigt dazu keinen Fernsteuerbetrieb
- Setzen von Zuständen und Sollwerten (Controlling) erfordert die vorherige Aktivierung des Fernsteuerbetriebes (remote)
- Bei Triple-Modellen sind die Hauptausgänge 1 und 2 fernsteuerbar und müssen, wie bei manueller Bedienung, separat adressiert werden, außer bei aktiviertem Tracking-Modus wo man nur Ausgang 1 adressieren kann und Ausgang 2 folgt

Um ein Gerät zu **steuern** müssen Sie

1. immer zuerst den Fernsteuerbetrieb aktivieren (Objekt 54)

2. und können danach Sollwerte oder Status senden.

Der Fernsteuerbetrieb sollte verlassen werden, wenn er nicht mehr benötigt wird. Solange er aber aktiviert ist, kann das Gerät bzw. der adressierte Ausgang nicht oder nur bedingt manuell bedient werden. Der Modus wird stets auf der Front des Gerätes angezeigt.

2.5 Effektive Auflösung beim Programmieren

Die Soll- und Istwerte von Spannung und Strom, die transferiert werden können und letztendlich von der Leistungsstufen am DC-Ausgang umgesetzt bzw. dort erfaßt werden, haben eine programmierbare Auflösung für 0-100% und eine effektive Auflösung. Gleiches gilt für die Istwerte, die mit einfachen Meßkreisen am DC-Ausgang in erster Linie als Regelgröße erfaßt werden. „Einfach“ bedeutet hierbei, daß man ein Netzgerät nicht als Meßgerät betrachten bzw. damit vergleichen kann, weil Meßgeräte deutlich schneller und präziser messen.

Übersicht:

| Nachrichtenformat | Programmierbare Auflösung | Effektive Auflösung |
|-----------------------|--|--|
| Altes, binäres Format | 0 - 0x6400 = 25601 Schritte | 25600 Schritte |
| ModBus | 0 - 0xCCCC = 52429 Schritte | 25600 Schritte |
| SCPI | Theoretisch unendlich, jedoch ist die Anzahl von effektiven Nachkommastellen identisch mit dem Format des zugehörigen Wertes auf der Anzeige | Modellabhängig, z. B. 5 A = 500 Schritte, weil ein Modell mit 5 A Nennwert den Strom als 5.00 darstellt. |

| Nachrichtenformat | Erreichbare Auflösung der Istwerte |
|-----------------------|---|
| Altes, binäres Format | ≤1024 |
| ModBus | ≤1024 |
| SCPI | Model depending, e. g. 5 A = 500 steps (siehe oben) |

Die effektive Auflösung bestimmt die am DC-Ausgang erreichbare Schrittweite von Sollwerten. Diese errechnet sich für alle Werte als Schrittweite Spannung/Strom = Nennwert ÷ Auflösung. So hat beispielsweise ein PS 2384-05 B eine erreichbare Schrittweite beim Sollwert der Spannung und wenn mit ModBus gesendet von 84 V / 25600 = ca. 3 mV. Beim Strom wären es dann 5 A / 25600 = ca. 0,2 mA. Bei den Istwerten ist die erfaßte Auflösung hingegen deutlich geringer.

Zu dieser theoretisch erreichbaren Schrittweite kommen aber noch Gerätetoleranzen hinzu, die einen Stellwert verfälschen. Diese lassen sich aus den Angaben in den technischen Daten des Gerätes errechnen. Das PS 2384-10 B aus dem Beispiel oben hat bei der Spannung laut Handbuch eine Toleranz von max. 0,2% vom Nennwert. Das sind rechnerisch bis zu 168 mV. Wenn man also einen Sollwert von 24 V setzt, dürfte dieser durch die zulässige Toleranz zwischen 23,83 V und 24,17 V betragen und man könnte ihn, sofern extern nachgemessen und nicht genau genug, in etwa 3 mV-Schritten nachkorrigieren. Extern mit einem Multimeter nachmessen wäre die bessere Möglichkeit, weil auch der Istwert des Netzgerätes fehlerbehaftet ist.

Der vom Gerät auslesbare Istwert enthält die Toleranz (oder Fehler) bereits. Angenommen, man würde mit einem Multimeter nachmessen und die Spannung ist aktuell 24,1 V und man will näher an den Zielwert 24 V heran, könnte man per Software und durch die Schrittweite von etwa 3 mV bei diesem Gerät nachkorrigieren und den Istwert weiten an den Sollwert annähern.

3. Nachrichtenformate

3.1 Altes, binäres Format

3.1.1 Telegrammaufbau

Das längenvariable binäre Telegramm setzt sich aus diesen Bytes zusammen:

| Byte 0 | Byte 1 | Byte 2 | Variable Anzahl Bytes | 2 Bytes |
|-----------|-----------|------------|-----------------------|-----------|
| SD | DN | OBJ | DATEN | CS |

Byte 0: **SD** (start delimiter)

Der Startdelimiter kennzeichnet den Beginn des Telegramms und legt fest, ob die Nachricht eine Anfrage ist oder nicht.

Bits 3-0: Datenlänge -1 des Datenbereichs **DATEN** im Telegramm, welche variabel ist und bis zu 16 Bytes lang (Bytes 3-18). Die Datenlänge ist in erste Linie für eine Anfrage an das Gerät interessant, da sie vorgibt, wieviele Bytes angefragt werden. Das Maximum von 16 Bytes ergibt dann in den Bits 3 - 0 den Nibblewert 0xF.

Die Datenlänge beim Senden bzw. die zu erwartende Datenlänge bei einer Anfrage müssen hier angegeben werden. Die zu erwartende Datenlänge ist aus der Objektliste, Spalte 6 zu entnehmen.

Bit 4: Richtung

0 = Nachricht vom Gerät an den PC

1 = Nachricht vom PC an das Gerät

Bit 5:

1 = Muß für Senden oder Anfragen an das Gerät immer auf 1 gesetzt sein, ist in Antworten jedoch immer 0.

Bits 6+7: Sendungstyp

00= reserviert

01= Anfrage von Daten (PC->Gerät)

10= Antwort auf eine Anfrage (Gerät->PC)

11 = Senden von Daten (PC->Gerät)

Byte 1: **DN** (device node)

Hier ist zu unterscheiden, ob ein Single- oder Triple-Modell angesprochen werden soll. Dieser Wert wird benutzt um einen bestimmten DC-Ausgang zu adressieren. Single-Modelle haben nur den Ausgang 1. Bei den Triple-Modellen können Ausgang 1 (linke Anzeige) und Ausgang 2 (rechte Anzeige) adressiert werden, der dritte nicht. Der DN wird in einer Antwort 1:1 zurückgegeben, so daß sie eindeutig einem Ausgang zugeordnet werden kann. Folgende Regeln:

Ausgang 1: DN muß 0 sein (Single- oder Triple-Modell)

Ausgang 2: DN muß 1 sein (nur Triple-Modell)

In Hinsicht auf die automatische Unterscheidung der drei unterstützten Nachrichtenformate sei an dieser Stelle darauf hingewiesen, daß der DN immer nur 0 oder 1 sein sollte, damit die Erkennung auf „altes, binäres Format“ funktioniert.

Byte 2: **OBJ** (Objekt)

Die Kommunikationsobjekte eines Gerätes werden über die hier angegebene Zahl adressiert. In der Objektliste für die PS 2000 B Serie werden alle unterstützten Objekte aufgelistet und beschrieben. Siehe Abschnitt 3.1.8.

Byte 3 - 18: **DATEN**

Der Bereich **DATEN** kann 0-16 Bytes lang sein, die Länge des Telegramms variiert also. Bei einer Anfrage an das Gerät werden keine Daten übermittelt, der Datenbereich entfällt dann und ab Byte 3 folgt direkt die Checksumme. Nur bei einer Antwort (Netzgerät -> PC) oder beim Senden (PC -> Netzgerät) werden Daten übermittelt.

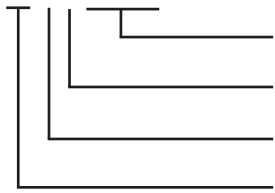
Letzte 2 Bytes: **CS** (check sum)

Die Position Prüfsumme (check sum) ist stets am Ende des Telegramms. Die Prüfsumme wird über die einfache Addition aller vorherigen Bytes des Telegramms gebildet. Sie ist zwei Bytes lang. Das Highbyte wird vor dem Lowbyte gesendet.

3.1.2 Der Startdelimiter im Einzelnen

Gemäß des Telegrammaufbaus (siehe oben) ist das erste Byte der Startdelimiter, der von der Richtung des Telegramms und dem Anfragetyp abhängig ist. Für dieses Beispiel nehmen wir einen SD von 0xF1. In Bits zerlegt sieht er so aus:

11 11 00 01



Bits 3...0: 0001 = Es werden 2 Bytes gesendet

Bit 4: 1 = Vom PC gesendet

Bit 5: 1 = Bei dieser Serie immer 1, wenn an das Gerät gesendet wird

Bit 7+6: 11 = Modus „Daten senden“

Dieser SD gibt vor, es wird etwas an das Gerät gesendet. Das Objekt und die Daten im Gerät bestimmen, was gesendet wird bzw. was das Gerät daraufhin macht. Alternativ zum bitweisen Zusammensetzen des Startdelimiters kann man sich das vereinfachen, indem man Hexwerte addiert:

SD = Sendungstyp + Cast-Typ + Richtung + Datenlänge

wobei Sendungstyp entweder

0xC0 Daten senden oder

0x40 Anfrage an das Gerät

0x80 Antwort vom Gerät

und Cast-Typ

0x20 Broadcast

und Richtung entweder

0x10 vom PC ans Gerät oder

0x00 vom Gerät an den PC

und die Datenlänge - 1 von

0x00...0x0F bis zu 16 Bytes am Stück

Vom obigen Beispiel ausgehend ergibt sich der SD mit 0xF1 aus 0xC0 + 0x20 + 0x10 + 0x01.

Der SD bei Antworten vom Gerät ist nicht derselbe wie bei der Anfrage, kann aber quasi komplett ignoriert bzw. könnte man die Datenlänge ablesen.

3.1.3 Das Maskenbyte für Objekt 54

Objekt 54 erfordert eine Maske, die immer zusammen mit dem eigentlichen Steuerbyte gesendet wird. Diese Maske ist in Spalte 6 der Objektliste angegeben (siehe separate PDF-Datei). Im Steuerbyte haben die einzelnen Bits unterschiedliche Funktionen, daher muß bestimmt werden, welches Bit geändert werden sollen. Dies legt die Maske mit einer 1 für das zu ändernde Bit fest.

Beispiel: man möchte Bit 0 des Steuerbytes verändern, also 0 oder 1 setzen, dann wäre das Steuerbyte 0x00 oder 0x01 und die Maske 0x01, entsprechend der Wertigkeit der Bits. Siehe Objektliste, Objekt 54.



Auch wenn die Maske zuläßt, mehrere Bits auf einmal zu verändern, so wird dringend empfohlen immer nur ein Bit pro Nachricht zu verändern, um Konflikte zu vermeiden.

3.1.4 Telegrammbeispiele

Hinweise: die unten dargestellten Hexwerte sind vereinfacht, ohne den sonst üblichen Präfix 0x.

Beispiel 1: Es sollen die Istwerte abgefragt werden (Objekt 71). Gemäß des oben erläuterten Aufbaus des Startdelimiters und des Telegrammaufbaus müßte das Abfragetelegramm dann so aussehen:

75 00 47 00 BC für ein Single-Modell bzw. Ausgang 1 eines Triple-Modells oder

75 01 47 00 BD für Ausgang 2 eines Triple-Modells

Die Antwort (Single-Modell oder Ausgang 1 eines Triple-Modells) könnte dann so aussehen:

85 00 47 01 01 64 00 1E 00 01 50 ⁽¹⁾

Das ergibt 42 V (grüner Wert = Istwert Spannung) und 1,8 A (blauer Wert = Istwert Strom) bei einem PS 2042-06B mit 42 V Nennspannung und 6 A Nennstrom. Der Status wird mit 0x0101 (pink) zurückgegeben und bedeutet „Fernsteuerung ein“, „DC-Ausgang ein“ und „Regelungsart CV“.

1) Für die Aufschlüsselung der Bytes siehe Objektliste, Objekt 71

Beispiel 2: Fernsteuerung aktivieren. Das erfordert u. A., daß die Anzeige zu dem jeweiligen Ausgang nicht im Menü-Modus ist. Anders als bei SCPI können über dieses Nachrichtenformat nicht beide Ausgänge eines Triple-Modells gleichzeitig in Fernsteuerung versetzt werden. Es müssen also zwei Nachrichten gesendet werden.

F1 00 36 10 10 01 47 für ein Single-Modell bzw. Ausgang 1 eines Triple-Modells oder

F1 01 36 10 10 01 48 für Ausgang 2 eines Triple-Modells

Eine erfolgreiche Ausführung des Befehls würde dann entweder mit

80 00 FF 00 01 7F für ein Single-Modell bzw. Ausgang 1 eines Triple-Modells oder

80 01 FF 00 01 80 für Ausgang 2 eines Triple-Modells

Im anderen Fall kann irgendein anderer Fehlercode kommen, wie beispielsweise

80 00 FF 05 01 80, der mit Code 0x05 („Falsche Adresse für Ausgang“) besagt, daß bei einem Single-Modell versucht wurde, den nicht vorhandenen Ausgang 2 anzusprechen.



Hexwerte müssen binär übertragen werden, nicht als ASCII-String!

3.1.5 Mögliche Probleme beim Setzen von Zuständen

Mit dem Objekt 54 wird der Fernsteuerbetrieb (Remote) aktiviert/deaktiviert oder der adressierte Ausgang des Gerätes ein- bzw. ausgeschaltet. Die Verwendung des Objektes und der Maske lassen es zwar zu, daß im Steuerbyte beides gleichzeitig gesetzt/zurückgesetzt werden kann, dies ist aber nicht zu empfehlen!

Setzen des Eingangs/Ausgangs erfordert, daß Fernsteuerung bereits aktiv ist, und sollte daher nach der Aktivierung von Remote durch erneutes Senden des Objektes 54 geschehen. Natürlich nur mit dem entsprechenden Bit. Beim Beenden des Fernsteuerbetriebes umgekehrt genauso.

Würden man beide Bits gleichzeitig setzen, könnte es vorkommen, daß das Gerät zuerst den Ausgang/Eingang setzen will, bevor Remote aktiviert wird und würde das mit einer Fehlermeldung quittieren. Daher ist es vielleicht sinnvoll, nach dem Setzen des Ausgangs/Eingangs dessen Status durch das Objekt 70 zurückzulesen.

3.1.6 Rückmeldungen und Kommunikationsfehler

Beim Senden von Werten oder Zuständen schickt das Gerät eine Bestätigung (positive acknowledge), wenn der Befehl fehlerfrei ankam, in Ordnung war und ausgeführt wurde. Die Antwort kommt als Fehlermeldung mit Objekt-nummer **0xFF** und mit Fehlercode **0x00**.

Bei einem Telegrammfehler (der Anwender hat ein falsches Telegramm gesendet) oder in Abhängigkeit vom Zustand des Gerätes kann bzw. wird eine Fehlermeldung zurückgegeben, die einen Code aus der Tabelle enthält:

| Fehlercode | | |
|------------|------|---|
| Hex. | Dez. | Beschreibung |
| 00 | 0 | Kein Fehler |
| 03 | 3 | Prüfsumme nicht korrekt |
| 04 | 4 | Startdelimiter falsch |
| 05 | 5 | Falsche Adresse für Ausgang |
| 07 | 7 | Objekt nicht definiert |
| 08 | 8 | Objektlänge nicht korrekt |
| 09 | 9 | Schreib-Leserechte verletzt, kein Zugriff |
| 0F | 15 | Gerät ist in "Lock" Modus |
| 30 | 48 | Obere Grenze des Objektes überschritten |
| 31 | 49 | Untere Grenze des Objektes unterschritten |

Legende

| | |
|--|----------------------|
| | Kommunikationsfehler |
| | Userfehler |

Beispiel: wenn man mit Objekt 50 bei einem Gerät die Spannung setzen will und das Gerät nicht in Fernsteuerung ist, dann würde sich das Fehlertelegramm **80 00 FF 09 01 88** ergeben. Der Fehlercode **0x09** besagt, daß das Objekt 50 im momentanen Zustand nicht beschrieben werden kann.

Erläuterungen zu einigen Fehlercodes:

Code 0x7: die Objektnummer im Telegramm an das Gerät ist dem Gerät unbekannt. Hinweis: die Objektnummern sind nicht alle aufeinanderfolgend.

Code 0x8: die Länge des Datenfeldes im Telegramm ist in der Objektliste definiert. Dieser Fehler kommt z. B., wenn ein Sollwert (immer 2 Bytes bei Typ „Int“) gesendet werden soll, das Datenfeld aber nur ein Byte enthielt. Selbst wenn der Startdelimiter die richtige Telegrammlänge enthält, dies dient zusätzlich zum Schutz, daß falsche Werte gesetzt werden.

Code 0x9: zusätzlich zu dem obigen Beispiel kann dieser Code auch bedeutet, daß wurde versucht, ein Objekt zu schreiben (=setzen), das laut Objektliste nur lesbar ist (Typ: read only (ro)).

Code 0xF: es wurde versucht die Fernsteuerung des adressierten Ausgang zu aktivieren, während die zugehörige Anzeige gerade im Menü-Modus ist.

3.1.7 Fehlerbehandlung

Mögliches Problem: Es wurden mehrere Anfragen gestellt, aber nicht alle wurden beantwortet

Wahrscheinlichste Ursache: Die Anfragen wurden womöglich zu schnell nacheinander gestellt. Es ist zwischen zwei Befehlen eine gewisse Zeit zu warten. Für ein Gerät der Serie PS 2000 B ist ein zeitlicher Mindestabstand von **50 ms** zwischen zwei Telegrammen empfohlen.

Mögliches Problem: Sollwerte oder Status werden nicht gesetzt

Mögliche Ursachen:

- Das Gerät bzw. der adressierte Ausgang befindet sich nicht im Fernsteuerbetrieb. Sollte zu Fehlercode 0x09 führen.
- Die gesendeten Werte sind falsch, d.h. zu groß. Es wird dann eine Fehlermeldung gesendet. Oder der Wert wird akzeptiert, kann aber für den jeweiligen Zustand nicht umgesetzt werden (Spannungssollwert gesendet, während Gerät in Strombegrenzung ist). In dem Fall käme keine Fehlermeldung.

3.1.8 Objektliste

Siehe separates PDF [object_list_ps2000b_de_en.pdf](#).

3.2 ModBus RTU

3.2.1 Vorwort

Wichtig! Zum weiteren Verständnis bitte unbedingt lesen:



Diese Serie erhält mit dem 2020er Facelift erstmal ModBus RTU als Kommunikationsprotokoll. ModBus ist zwar spezifiziert, die Spezifikation wurde aber von uns bis Anfang 2020 nicht vollständig eingehalten. Daher wurde in allen anderen Geräteserien mit ModBus-Support eine Umschaltung zwischen eingeschränkter und voller Kompatibilität zur Spezifikation eingeführt, wobei der eingeschränkte Modus die Standardeinstellung nach Auslieferung ist, um kompatibel zu früheren Firmwares zu bleiben. Das wurde 1:1 für die neue Generation PS 2000 B mit TFT-Anzeige übernommen und bedeutet, daß wenn man beginnen möchte ModBus zu verwenden, der Modus einmal auf „Voll“ umgeschaltet werden muß. Er wird dann dauerhaft gespeichert. Diese Einhaltung der Spezifikation kann über Register 10013 zwischen Modus „Voll“ und „Limitiert“ (Standard) umgeschaltet werden. Unterschiede:

- „Voll“ unterstützt die nur die Slave ID / Adresse 1 und Antworten auf Funktion READ COILS haben das korrekte Format
- „Limitiert“ unterstützt nur Slave ID / Adresse 0, demnach muß man die Aktivierung von „Voll“ an Adresse 0 senden. Die Antwort auf eine READ COILS Anfrage wird dann immer 16 Coils zurückgeben.

Ab hier wird davon ausgegangen, das zu programmierende Gerät ist auf Modus „Voll“ gestellt.

3.2.2 Allgemeines zu ModBus RTU

Eine Nachricht oder Telegramm nach ModBus RTU-Protokoll besteht aus hexadezimalen Bytes, wovon das erste Byte, die sogenannte „slave ID“ hier **immer 1** sein muß, weil das Gerät keine einstellbare Adresse benötigt und diese daher fest auf 1 steht. Grund: das erste Byte des Telegramms wird auch zur Erkennung benutzt, welches Nachrichtenformat für die aktuelle Nachrichten verwendet wurde. Siehe dazu auch „3. Nachrichtenformate“.

Das Format und die Länge eines Telegramms sind vorgegeben, wie unten näher erläutert.

3.2.3 Über die Register-Liste

Zusammen mit dieser Programmieranleitung wird eine sogenannte Register-Liste als PDF mitgeliefert, die eine Übersicht über die von PS 2000 B Gerät unterstützten Funktionen bei Fernsteuerung über ModBus bietet.

Die Liste erläutert, wie die Daten aufgebaut sind, welche für die jeweilige Funktion an ein bestimmtes Register geschickt werden müssen. Das hilft dem Programmierer bei der Implementation der Geräte-Kommunikation in eigene Applikationen. Anwender, die mit SCPI arbeiten, benötigen diese Liste im Allgemeinen nicht. Weiter hinten in diesem Dokument werden die Befehle der textbasierten SCPI-Sprache gesondert behandelt. Anwender, die das alte binäre Nachrichtenformat verwenden (siehe auch „3. Nachrichtenformate“), müssen eine andere Liste als Referenz verwenden, die PS 2000 B Objektliste.

3.2.3.1 Spalten „ModBus-Adresse“

Die hier aufgeführte Zahl ist als Adresse eines ModBus-Registers zu verstehen. Die Adresse wird in ModBus-Nachrichten in hexadezimaler Form verwendet.

3.2.3.2 Spalten „Funktion“

Die Köpfe der 5 Spalten rechts von den ModBus-Adreßspalten enthalten die Namen und Codes der ModBus-Funktionen, die unterstützt werden. Ein „x“ kennzeichnet für ein Register die zugehörige Funktion. Z. B. ist ein sog. Coil-Register meist schreib- und lesbar und daher mit den Funktionen „Read Coils (0x01)“ und „Write Single Coil (0x05)“ verknüpft.

3.2.3.3 Spalte „Datentyp“

| Datentyp | Länge | |
|----------|---------|---|
| char | 1 Byte | Einzelnes Byte, wird für Strings benutzt |
| uint(16) | 2 Bytes | Doppelbyte, auch genannt Wort oder vorzeichenloser 16-Bit-Integer |
| uint(32) | 4 Bytes | Doppelwort, auch genannt Long oder vorzeichenloser 32-Bit-Integer |
| float | 4 Bytes | Fließkommawert nach IEEE745 Standard |

3.2.3.4 Spalte „Zugriff“

Definiert für jedes Register, ob man auf das Register nur lesend, nur schreibend oder schreibend/lesend zugreifen kann.

R = Register kann nur gelesen werden

W = Register kann nur geschrieben werden bzw. würde beim Lesen keinen sinnvollen Wert zurückgeben

RW = Register kann geschrieben oder gelesen werden



Generell gilt: Schreiben auf ein Register, auf das geschrieben werden kann (Zugriff: W, RW), ist nur bei aktivierter Fernsteuerung möglich!

3.2.3.5 Spalte „Anzahl Register“

Bei ModBus ist ein Register immer zwei Bytes oder ein Vielfaches von zwei Bytes groß. Diese Spalte gibt an, wieviele Zwei-Byte-Werte die Register belegen. Der Wert ist jeweils die Hälfte des Wertes in der Spalte „Datenlänge in Bytes“.

3.2.3.6 Spalte „Daten“

Gibt zusätzliche Information zum Format der Daten, die in ein Register zu schreiben sind bzw. daraus gelesen werden können. Zwei, vier oder mehr Bytes können unterschiedlich interpretiert werden, je nach Datentyp.

3.2.4 Telegrammtypen

Grundsätzlich wird zwischen **Anfrage-Telegrammen**, **Sende-Telegrammen** und **Antwort-Telegrammen** unterschieden. Ein Anfrage-Telegramm veranlaßt das Gerät ein Antwort-Telegramm zu schicken, während Sende-Telegramme nur ein 1:1 Echo verursachen, als Bestätigung des Empfangs auf der Gegenseite.

3.2.5 Funktionen

Das zweite Byte einer ModBus-Nachricht ist die sogenannte **Funktion (FN)**, blau markiert in den Aufbaudarstellungen unten). Die Funktion bestimmt, ob etwas geschrieben (WRITE) oder gelesen (READ) werden soll. Weiterhin wird beim Schreiben und Lesen unterschieden, ob man ein bzw. mehrere ganze Register schreiben/lesen will.

Das hier beschriebene Protokoll unterstützt folgende Funktionen aus der ModBus-Spezifikation:

| Funktion | | Funktionsname | | Beschreibung | Anwendungsbeispiel |
|----------|-----|--------------------------|------|---|---|
| Hex | Dez | Lang | Kurz | | |
| 0x01 | 1 | READ COILS | RC | Läßt nur das Lesen von 1 Coil zu, weil die Coils nicht auf fortlaufenden Adressen liegen. | Zustand des Eingangs / Ausgangs abfragen |
| 0x03 | 3 | READ HOLDING REGISTERS | RHR | Lesen von n aufeinanderfolgenden Registern. Ergibt n*2 Bytes. | Gerätebezeichnung auslesen (1-40 Bytes) |
| 0x05 | 5 | WRITE SINGLE COIL | WSC | Schreiben einer Coil (TRUE/FALSE) eines booleschen Registers. | Gerät in Fernsteuerung umschalten |
| 0x06 | 6 | WRITE SINGLE REGISTER | WSR | Schreiben eines Registers. | Sollwert (U, I, P usw.) |
| 0x10 | 16 | WRITE MULTIPLE REGISTERS | WMR | Schreiben mehrerer aufeinanderfolgender Register. Kann jedoch nicht verwendet werden, um die Grenze eines Registerblocks hinweg zu schreiben, also wenn man z. B. mehrere Sollwerte wie U, I und P auf einmal schreiben wollte. | Mehrere Werte innerhalb eines Registerblocks oder den sog. Benutzertext schreiben |

3.2.6 Sendetelegramm (Schreiben)

Das Protokoll überprüft die plausible Länge des Telegramms beim Schreiben nur dahingehend, daß die maximale Länge des Registers nicht überschritten wird. Nach den Daten im Telegramm wird die Checksumme erwartet. Wenn z. B. die Daten nur die mindestens benötigten 2 Bytes lang sind und damit die Vorgaben für die gewählte Registeradresse erfüllen, würde die Checksumme ab dem 7. Byte erwartet. Stünde dort ein anderer Wert und die Checksumme ist erst weiter hinten im Telegramm, würde das Gerät einen Fehler zurückgeben.

Somit wird, egal ob das Telegramm zu kurz oder zu lang ist, ein Fehler zurückgegeben, weil die Checksumme nicht stimmt. Für einige Beispiele siehe „3.2.11. Beispiele für ModBus RTU-Telegramme“.



Die Daten in einer ModBus-Nachricht sind, mit Ausnahme der Checksumme, von links nach rechts zu lesen (big endian). Bei der 16 Bit-ModBus RTU-Checksumme werden Highbyte und Lowbyte jedoch vertauscht.

WRITE Single Register

| Byte 0 | Byte 1 | Bytes 2+3 | Bytes 4+5 | Letzte 2 Bytes |
|--------|--------|-----------|----------------------|--|
| ID | FN | Startreg. | Datenwort | CRC |
| 0x01 | 0x06 | 0...65535 | Zu schreibender Wert | Checksumme ModBus-CRC16 ⁽¹⁾ |

WRITE Multiple Registers

| Byte 0 | Byte 1 | Bytes 2+3 | Bytes 4+5 | Byte 6 | Bytes 7-253 | Letzte 2 Bytes |
|--------|--------|-----------|-----------|----------|-------------|--|
| ID | FN | Startreg. | Anzahl | Zähler | Datenbytes | CRC |
| 0x01 | 0x10 | 0...65535 | 0...123 | Anzahl*2 | Daten | Checksumme ModBus-CRC16 ⁽¹⁾ |

WRITE Single Coil

| Byte 0 | Byte 1 | Bytes 2+3 | Bytes 4+5 | Bytes 6+7 |
|--------|--------|-----------|--------------------|--|
| ID | FN | Register | Datenwort | CRC |
| 0x01 | 0x05 | 0...65535 | 0x0000 oder 0xFF00 | Checksumme ModBus-CRC16 ⁽¹⁾ |

3.2.7 Anfragetelegramm

Bei einer **Anfrage** an das Gerät wird eine möglichst unverzügliche Antwort erwartet, die - je nach Inhalt - von unterschiedlicher Länge, aber stets vom gleichen Aufbau ist. Es muß das Startregister angegeben werden, ab dem man die gewünschten Informationen lesen will, sowie die zu lesende Anzahl von Registern. Die Basis des ModBus-Datenformats ist ein 16-Bit-Wert, also eine Gruppe von 2 Bytes. Somit werden bei Anzahl = 1 genau zwei Bytes oder ein 16-Bit-Wert angefragt und bei Anzahl = 2 dann vier Bytes bzw. zwei 16-Bit-Werte usw. Bei READ COILS werden 1 Byte (= 1 Coil) oder 2 Byte (=16 Coils, falsche Antwortform in früheren Firmwares) zurückgegeben. Für einige Beispiele siehe „3.2.11. Beispiele für ModBus RTU-Telegramme“.

READ HOLDING REGISTERS

| Byte 0 | Byte 1 | Bytes 2+3 | Bytes 4+5 | Letzte 2 Bytes |
|--------|--------|-----------|---------------------------------------|--|
| ID | FN | Startreg. | Anzahl | CRC |
| 0x01 | 0x03 | 0...65535 | Anzahl zu lesender Register (1...125) | Checksumme ModBus-CRC16 ⁽¹⁾ |

READ COILS

| Byte 0 | Byte 1 | Bytes 2+3 | Bytes 4+5 | Letzte 2 Bytes |
|--------|--------|-----------|------------------|--|
| ID | FN | Startreg. | Anzahl | CRC |
| 0x01 | 0x01 | 0...65535 | Muß immer 1 sein | Checksumme ModBus-CRC16 ⁽¹⁾ |

3.2.8 Antworttelegramm (Lesen)

Eine Antwort vom Gerät kommt entweder nach einer Anfrage oder bei einem Kommunikationsfehler oder beim Setzen eines Wertes, Zustand o. ä., wenn das Setzen angenommen wurde.

Erwartete Antwort bei WRITE SINGLE REGISTER:

| Byte 0 | Byte 1 | Bytes 2+3 | Bytes 4+5 | Letzte 2 Bytes |
|--------|--------|-----------|--------------------|--|
| ID | FN | Startreg. | Daten | CRC |
| 0x01 | 0x06 | 0...65535 | Geschriebener Wert | Checksumme ModBus-CRC16 ⁽¹⁾ |

Erwartete Antwort bei WRITE SINGLE COIL:

| Byte 0 | Byte 1 | Bytes 2+3 | Bytes 4+5 | Letzte 2 Bytes |
|--------|--------|-----------|--------------------|--|
| ID | FN | Startreg. | Daten | CRC |
| 0x01 | 0x05 | 0...65535 | Geschriebener Wert | Checksumme ModBus-CRC16 ⁽¹⁾ |

⁽¹⁾ Siehe „3.2.9. Die ModBus-Checksumme“

Erwartete Antwort bei WRITE MULTIPLE REGISTERS:

| Byte 0 | Byte 1 | Bytes 2+3 | Bytes 4+5 | Letzte 2 Bytes |
|--------|--------|-----------|------------------------------|--|
| ID | FN | Startreg. | Daten | CRC |
| 0x01 | 0x10 | 0...65535 | Anzahl geschriebene Register | Checksumme ModBus-CRC16 ⁽¹⁾ |

Erwartete Antwort bei READ HOLDING REGISTERS:

| Byte 0 | Byte 1 | Byte 2 | Bytes 3-253 | Letzte 2 Bytes |
|--------|--------|---------------------|---------------------|--|
| ID | FN | Datenlänge in Bytes | Daten | CRC |
| 0x01 | 0x03 | 2...250 | Angefragte Register | Checksumme ModBus-CRC16 ⁽¹⁾ |

Erwartete Antwort bei READ COILS:

| Byte 0 | Byte 1 | Byte 2 | Byte 3 | Letzte 2 Bytes |
|--------|--------|---------------------|----------------|--|
| ID | FN | Datenlänge in Bytes | Daten | CRC |
| 0x01 | 0x01 | 1 | 0x00 oder 0x01 | Checksumme ModBus-CRC16 ⁽¹⁾ |

Unerwartete Antwort (Kommunikationsfehler):

| Byte 0 | Byte 1 | Byte 2 | Letzte 2 Bytes |
|--------|----------------------|------------|--|
| ID | FN | | CRC |
| 0x01 | 0x80 + Funktionscode | Fehlercode | Checksumme ModBus-CRC16 ⁽¹⁾ |

3.2.9 Die ModBus-Checksumme

Die Checksumme am Ende eines ModBus RTU-Telegramms ist eine 16bit-Checksumme, aber laut Vorgabe der ModBus-Spezifikation anders berechnet als sonstige CRC16 und wird zudem noch **mit dem Low-Byte zuerst** angegeben. Informationen und Sourcecode zur Berechnung der Checksumme sind im Internet zugänglich, unter Anderem hier: http://www.modbus.org/docs/Modbus_over_serial_line_V1_02.pdf, Abschnitt 2.5.1.2.

3.2.10 Kommunikationsfehler

Kommunikationsfehler beziehen sich lediglich auf die digitale Kommunikation mit dem Gerät und sind nicht zu verwechseln mit Alarmen und sonstigen Ereignissen, die das Gerät melden kann. Sie werden sofort vom Gerät zurückgegeben, wenn z. B. versucht wird schreibend auf das Gerät zuzugreifen während es nicht in Fernsteuerung ist oder auch wenn z. B. über die Funktion READ HOLDING REGISTER versucht wurde, ein Zustands-Bit auszulesen, das laut Registerdefinition mit READ COILS gelesen werden müßte.

Bei einem Fehler sendet das Gerät die zuvor verwendete Funktionsnummer, jedoch um den Wert 0x80 erhöht, sowie einen Fehlercode zurück.

Übersicht der Funktionswerte in den Kommunikationsfehlertelegammen:

| FN-Fehler | Gehört zu |
|-----------|--------------------------|
| 0x81 | READ COILS |
| 0x83 | READ HOLDING REGISTERS |
| 0x85 | WRITE SINGLE COIL |
| 0x86 | WRITE SINGLE REGISTER |
| 0x90 | WRITE MULTIPLE REGISTERS |

Übersicht der Kommunikationsfehlercodes, die ein Gerät melden kann:

| Code | Fehlerbezeichnung | Erläuterung |
|------|---------------------------|--|
| 0x01 | 1 Falsche Funktionsnummer | Es wurde eine ModBus-Funktion im 2. Byte der Nachricht verwendet, die vom Gerät nicht unterstützt wird. Siehe „3.2.5. Funktionen“, welche unterstützt werden. Der Fehler tritt außerdem auf, wenn für den Zugriff auf eine Adresse die falsche Funktion verwendet wurde, z. B. READ Holding Registers beim Schreiben eines Bits, für das WRITE Single Coil richtig gewesen wäre. |

⁽¹⁾ Siehe „3.2.9. Die ModBus-Checksumme“

| Code | Fehlerbezeichnung | Erläuterung |
|------|-------------------|---|
| 0x02 | 2 | Adresse nicht definiert Es wurde versucht, schreibend oder lesend auf eine Adresse zuzugreifen, die für das Gerät nicht definiert wurde. Siehe separate ModBus-Register-Liste, welche Register/Adressen für die jeweilige Serie verfügbar sind, zu der Ihr Gerät gehört. |
| 0x03 | 3 | Fehlerhafte Daten oder Datenlänge falsch Es wurden Daten mit falscher Länge oder falsche Daten gesendet. Zum Beispiel erfordert ein Sollwert immer 2 Bytes. Wenn stattdessen nur 1 Byte gesendet würde, wäre die Nachricht zu kurz und wenn 3 Bytes (vor der Checksumme) gesendet würden, wäre die Nachricht zu lang. „Fehlerhafte Daten“ könnte z. B. ein zu hoher Sollwert sein, also wenn ein Sollwert als max. 0xCCCC definiert ist und man würde 0xE000 schicken. |
| 0x04 | 4 | Execution Befehl nicht ausführbar, situationsabhängig |
| 0x05 | 5 | CRC CRC16-Checksumme am Ende der ModBus RTU-Nachricht ist falsch. Das kann durch eine falsche Bytereihenfolge (High-Byte zuerst statt Low-Byte zuerst) oder durch falsche Berechnung entstehen. |
| 0x07 | 7 | Zugriff verweigert Zugriff auf eine Adresse generell nicht erlaubt, oder nur lesend oder nur schreibend oder Zugriff zwar schreibend erlaubt, Gerät aber nicht in Fernsteuerung bzw. in Fernsteuerung durch eine andere Schnittstelle |
| 0x17 | 23 | Gerät in Lokal-Modus Dieser Fehler kommt, wenn irgendeine Nachricht gesendet wird, die steuernd auf das Gerät zugreifen will, während die Fernsteuerung gesperrt ist (lokal). |

Beispiel: Sie versuchen, das Gerät in Fernsteuerung zu versetzen und bekommen statt des erwarteten Echos Ihres Befehls eine Antwort wie diese: 01 85 07 03 52. Das ist eine Fehlermeldung. An der Position der Funktion steht der Wert 0x85, laut Tabelle oben ist das ein Fehler bei WRITE SINGLE COIL. Der Fehlercode ist 0x7, laut der zweiten Tabelle oben bedeutet dieser, daß der Zugriff auf das Gerät mit dem zuletzt gesendeten Befehl verweigert wurde. Das kann u. A. bedeuten, daß es bereits über eine andere Schnittstelle, z. B. eine analoge, ferngesteuert wird.

3.2.11 Beispiele für ModBus RTU-Telegramme

3.2.11.1 Einen Sollwert setzen



Sollwerte sind setzbare Stellwerte für die Regelung der physikalischen Größen Strom und Spannung. Sie können nur dann an das Gerät geschickt werden, wenn es vorher in Fernsteuerbetrieb versetzt wurde.

Beispiel: Den Strom-Sollwert auf 50% setzen. Laut Registerliste ist der „Sollwert Strom“ auf Registeradresse 501 (0x1F5) festgelegt und die zu verwendende ist Funktion WRITE SINGLE REGISTER. Vorausgesetzt das Gerät ist bereits in Fernsteuerung, müßte das Telegramm muß dann so aussehen:

| Sende- telegramm: | ID | FN | Start | Daten | CRC | Erwartete Antwort: | ID | FN | Start | Daten | CRC |
|----------------------|------|------|--------|--------|--------|-----------------------|------|------|--------|--------|--------|
| | 0x01 | 0x06 | 0x01F5 | 0x6666 | 0x338E | | 0x01 | 0x06 | 0x01F5 | 0x6666 | 0x338E |

Durch ein Echo des Telegramms meldet das Gerät, daß der Befehl akzeptiert wurde. Am Display könnte man nun den Stromsollwert überprüfen, bei einer Last bzw. Netzgerät mit 10 A Nennstrom müßte dieser dann als 5.00 A auf der Frontanzeige abzulesen sein bzw. bei einem Modell mit 5 A Nennstrom dann als 2.50 A.

3.2.11.2 Alle Istwerte auf einmal anfragen

Das Gerät bietet drei auslesbare Istwerte für Spannung, Strom und Leistung pro Ausgang. Diese Istwerte können einzeln oder zusammen angefragt werden. Der Vorteil bei der gleichzeitigen Anfrage aller Istwerte ist, daß man damit den Gesamtzustand der DC-Ausgangswerte zu einem bestimmten Zeitpunkt erhält. Bei Einzelabfrage könnten sich die Werte zwischen zwei Anfragen bereits geändert haben.

Laut Registerliste sind die Istwerte für den Ausgang 1 (Single- und Triple-Modelle) ab Register 507 bzw. für Ausgang 2 (Triple-Modell) ab Register 517 zu finden. Es sollen drei Register ab 507 gelesen werden:

| Anfrage- telegramm: | ID | FN | Start | Daten | CRC | ► |
|------------------------|------|------|--------|--------|--------|---|
| | 0x01 | 0x03 | 0x01FB | 0x0003 | 0x75C6 | |

Mögliche
Antwort:

| ID | FN | Länge | Daten | CRC |
|------|------|-------|----------------------|--------|
| 0x01 | 0x03 | 0x06 | 0x2620 0x0C9B 0x091B | 0x9350 |

3.2.11.3 Nennspannung auslesen

Die Gerätenennspannung, sowie auch die Nennwerte für Strom und Leistung sind wichtige Referenzwerte für die Umrechnung der Soll- und Istwerte und sollten nach dem Öffnen der Kommunikationsverbindung zu einem Gerät mit als Erstes ausgelesen werden. Die Nennspannung ist laut Registerliste ein Floatwert mit 4 Bytes Länge ab Adresse 121.

| Anfrage- telegramm: | ID | FN | Start | Anzahl | CRC | ► | Mögliche Antwort: | ID | FN | Länge | Daten | CRC |
|------------------------|------|------|--------|--------|--------|---|----------------------|------|------|-------|------------|--------|
| | 0x01 | 0x03 | 0x0079 | 0x0002 | 0x15D2 | | | 0x01 | 0x03 | 0x04 | 0x42A00000 | 0xEE69 |

Siehe auch 3.2.8. Die Beispielantwort enthält einen Floatwert im IEEE754-Format, der sich zu 80.0 umrechnet.

3.2.11.4 Gerätestatus auslesen

Laut Registerliste liefert ein Gerät pro Ausgang einen 32 Bit-Wert mit mehreren Statusbits. Beispiel für Ausgang 1 und Register 505:

| Anfrage- telegramm: | ID | FN | Start | Anzahl | CRC | ► | Mögliche Antwort: | ID | FN | Länge | Daten | CRC |
|------------------------|------|------|--------|--------|--------|---|----------------------|------|------|-------|------------|--------|
| | 0x01 | 0x03 | 0x01F9 | 0x0002 | 0x15C6 | | | 0x01 | 0x03 | 0x04 | 0x00000483 | 0xB952 |

Siehe auch 3.2.8. Die Antwort 0x00000483 sagt nach Aufschlüsselung per Registerliste (Adresse 505) aus, daß das Gerät in Fernsteuerung über den USB-Port ist, der Ausgang eingeschaltet und die Regelungsart CC ist.

3.2.11.5 Fernsteuerung aktivieren oder beenden

Bevor Sie das Gerät fernsteuern können, ist Umschalten auf digitalen Fernsteuerbetrieb mittels eines entsprechenden Befehls erforderlich.



Das Gerät schaltet sich niemals automatisch in den Fernsteuerbetrieb und kann ohne diesen nicht ferngesteuert werden. Auslesen von allen lesbaren Registern ist jedoch auch ohne aktiven Fernsteuerbetrieb möglich.



Das Gerät verläßt die Fernsteuerung niemals automatisch, außer es wird während des Fernsteuerbetriebs ausgeschaltet oder es tritt ein Netzausfall auf. Die Fernsteuerung kann per Befehl beendet bzw. manuell am Gerät abgebrochen werden.

Das Umschalten auf Fernsteuerung kann durch folgende bestimmte Umstände blockiert sein und wird durch die Rückgabe eines Fehlertelegramms gekennzeichnet:

- Es ist Zustand „**Lock**“ aktiv (siehe Anzeige auf der Front oder auslesbaren Gerätestatus), der dann entsteht, wenn die zum adressierten Ausgang zugehörige Anzeige gerade im Menü-Modus ist

► So schalten Sie das Gerät in die Fernsteuerung:

1. Erstellen und senden Sie das Telegramm nach dem oben beschriebenen Aufbau, z. B. 01 05 01 92 FF 00 2C 2B für Ausgang 1.
2. Bei erfolgreicher Umschaltung des Gerätes in Fernsteuerung, also wenn das Gerät den Befehl akzeptiert hat, wird der Zustand im Allgemeinen in der Anzeige des Gerätes angezeigt, sowie der Befehl 1:1 als Bestätigung zurückgegeben (Echo).

Fernsteuerung kann auf zwei Arten beendet werden: per Befehl oder durch manuelle Interaktion am Bedienfeld des Gerätes. Da es hier um Programmierung geht, wird die erstgenannte Möglichkeit betrachtet.

► So beenden Sie die Fernsteuerung:

1. Erstellen und senden Sie das Telegramm nach dem oben beschriebenen Aufbau, z. B. 01 05 01 92 00 00 6D DB für Ausgang 1.

3.3 SCPI

SCPI ist ein internationaler Standard für eine klartextbasierte Befehlssprache. Näheres zum Standard selbst finden Sie im Internet.

3.3.1 Format der Soll- und Istwerte

Bei SCPI werden alle Werte immer als **reale Werte** dargestellt, mit oder ohne Einheit. Wenn Sie z. B. den Ausgangsstrom auf 17,5 A festlegen möchten, dann könnte das mit dem Befehl **CURR_17.5** oder mit **CURR_17.5A** geschehen. Eine genauere Erläuterung der Syntax finden Sie unten. Das Leerzeichen, was zwischen Befehl und Parameter erforderlich ist, wird unten zur Verdeutlichung durch das Symbol „_“ ersetzt.

3.3.2 Syntax

Spezifikation nach „1999 SCPI Command reference“. Folgende Formate für Werte und Parameter können in Befehlen bzw. Antworten auftreten:

| | | |
|---------|---|--|
| <value> | Der Zahlenwert entspricht dem Zahlenformat im Display des Gerätes und ist abhängig von den Nennwerten des Gerätes. Es gilt: - Der Wert wird vom voranstehenden Befehl immer mit einem Leerzeichen getrennt eingegeben - Anstatt eines Zahlenwertes können alternativ eingegeben werden: | |
| | MIN | Entspricht dem Minimalwert des Parameters, hier immer 0 |
| | MAX | Entspricht dem Maximalwert des Parameters, der unterschiedlich sein kann: MAX bei einem Sollwert (U, I) = zugehöriges Einstellungsgrenze (z. B. U-max) MAX bei einer Schutzgrenze = 110% Nennwert von U oder I |
| <NR1> | Zahlenformat ohne Dezimalpunkt | |
| <NR2> | Zahlenformat mit Dezimalpunkt (Fließkomma), inkludiert NR1 | |
| <NRf> | <NR1> oder <NR2> oder <NR3> | |
| Unit | V (Volt), A (Ampere), W (Watt) | |
| <CHAR> | 0..255: Dezimalzahl | |
| <+INT> | 0..32768: positive Integerzahl (Ausgabe) | |
| <B0> | 1 oder ON: Funktion wird eingeschaltet | |
| | 0 oder OFF: Funktion wird ausgeschaltet | |
| <B1> | NONE: lokaler Betrieb, eine Umschaltung auf Fernbedienung ist möglich | |
| | REMOte: Fernbedienung des Gerätes ist aktiviert | |
| <ERR> | Fehlernummer und Fehlerbeschreibung | |
| <SRD> | Stringdaten, verschiedene Formate | |
| ; | Das Semikolon wird verwendet, um innerhalb einer Message mehrere Befehle zu senden. | |
| : | Der Doppelpunkt trennt höherwertige Schlüsselwörter von niederwertigeren Schlüsselwörtern | |
| [] | Kleinbuchstaben und der Inhalt in rechteckigen Klammern sind optional. | |
| ? | Das Fragezeichen kennzeichnet eine Abfrage. Die Abfrage kann gleichzeitig mit einer Datensendung verknüpft werden. Hierbei ist darauf zu achten, daß, bevor eine neue Datensendung erfolgt, die Antwort des Systems abgewartet werden muß. | |
| -> | Antwort vom Gerät | |

3.3.3 Befehlsverkettung

Es ist möglich, bis zu fünf (5) Befehle auf einmal an das Gerät zu schicken. Zur Trennung ist das Semikolon (;) zu verwenden. Beispiel:

VOLT 20;CURR 10;MEAS:ARR?

Befehle werden in der gesendeten Reihenfolge verarbeitet und, sofern der Ausgang/Eingang des Gerätes eingeschaltet ist, auch sofort aktiv. Die Reihenfolge der Sollwerte kann daher von Bedeutung sein bzw. ob sie bei Ausgang = aus oder Ausgang = ein gesendet werden. Wenn mehrere Werte oder Parameter auf einmal angefragt werden, kommen diese auch zusammen in einer Antwort, in der angefragten Reihenfolge und auch dann durch Semikolons getrennt zurück.

3.3.4 Groß-/Kleinschreibung

Bei SCPI ist Großschreibung üblich. Die Geräte akzeptieren jedoch auch Kleinschreibung.

3.3.5 Langform und Kurzform

SCPI-Befehle haben immer eine Lang- und eine Kurzform. Die Kurzform (z. B. SOUR) oder die Langform (z. B. SOURCE) sind beliebig verwendbar. Um die beiden Formen zu unterscheiden sind nachfolgend die Befehle teils mit Großbuchstaben (markiert die Kurzform) und Kleinbuchstaben geschrieben (markiert den zusätzlichen Teil der Langform).

3.3.6 Abschlußzeichen

Ein Abschlußzeichen ist bei bestimmten Schnittstellen nötig, bei USB nicht. Hier kann es optional gesendet werden, damit Steuerungssoftwares über verschiedene Schnittstellen hinweg kompatibel bleiben.

Unterstützt wird: **0xA** (LF, line feed)

3.3.7 Kommunikationsfehler

Gemäß IEEE-Standard melden Geräte bei Kommunikation über SCPI Fehler nicht automatisch zurück. Der oder die möglicherweise aufgetretenen Fehler müssen per Befehl abgefragt werden. Das kann direkt über Abfragebefehle (siehe 3.3.17) oder indirekt (Signal „err“ im Register Statusbyte, siehe „3.3.9. Statusregister“) geschehen.

Das Antwortformat ist im Standard vorgegeben und besteht immer aus einer Zahl (Fehlercode) und einem kurzen, erläuternden Text. Folgende SCPI-Fehler können generiert werden:

| Fehlercode / Fehlertext | Beschreibung |
|---------------------------------|--|
| 0, "No error" | Kein Fehler |
| -100, "Command error" | Befehl unbekannt oder unvollständig |
| -102, "Syntax error" | Befehl nicht richtig geschrieben, z. B. SYST:LOKC (teilweise richtig) |
| -108, "Parameter not allowed" | Der Befehl wurde mit einem Parameter gesendet, obwohl dieser Befehl keine verwendet |
| -200, "Execution error" | Befehl konnte nicht ausgeführt werden |
| -220, "Parameter error" | Falscher Parameter wurde verwendet |
| -221, "Settings conflict" | Befehl konnte wegen des gegenwärtigen Zustandes des Gerätes nicht ausgeführt werden (Gerät ist im MENU o. ä.) |
| -222, "Data out of range" | Parameter konnte nicht gesetzt werden, weil er eine Grenze überschritt |
| -223, "Too much data" | Zu viele Parameter pro Befehl oder zu viele Befehle auf einmal |
| -224, "Illegal parameter value" | Es wurde ein Parameter vorgegeben, der für den Befehl nicht definiert ist |
| -225, "Out of memory" | Die erwartete Antwort wurde nicht gesendet werden, weil ihre Länge die Größe des internen Puffers überschreiten würde (kann nur bei 5x *IDN? in einer Nachricht auftreten) |

3.3.8 Standard-IEEE-Befehle

3.3.8.1 *CLS

Löscht die Fehler-Queue und das Statusbyte (STB).

3.3.8.2 *IDN?

Fragt den Beschreibungsstring des Gerätes ab, der kommagetrennt folgendes enthält:

1. Hersteller
2. Gerätebezeichnung
3. Seriennummer
4. Firmwareversion(en) des Gerätes (falls mehrere, dann mit Leerzeichen getrennt)
5. Benutzertext (wie per SCPI-Befehl `sys:config:user:text` definierbar)

3.3.8.3 *RST

Setzt das Gerät auf folgenden definierten Zustand, falls Fernsteuerung nicht durch das Gerät blockiert ist:

1. Fernsteuerung einschalten (identisch zu `SYST:LOCK 1`)
2. DC-Eingang/Ausgang = aus
3. Alarmspeicher leeren
4. Zustandsregister löschen (QUESTIONABLE Event, OPERATION Event, STB)

| Befehl | Beschreibung |
|--|--|
| STATus:QUEStionable? STATus:QUEStionable:CONDition? | Liest das Questionable Statusregister und gibt einen Wert zurück, der dem Bitzustand an CONDITION entspricht. |
| STATus:QUEStionable:EVENT? | Liest das Event-Unterregister des Questionable Statusregister und gibt einen Wert zurück, der dem Bitzustand von EVENT entspricht. |
| STATus:QUEStionable:ENABle_<NR1> STATus:QUEStionable:ENABle | Definiert einen Filter für die Bits im CONDITION bevor sie in das Event-Unterregister des Questionable Statusregister übergeben werden oder liest den aktuellen Filterwert. Dies kann dazu dienen, bestimmte Ereignisse zu unterdrücken. Standardmäßig sind alle Bits aktiviert, für die ein Signal aktiviert wurde (siehe Übersichtsschema). Das bedeutet, der Setzwert darf die Wertigkeit der verwendeten Bits nicht überschreiten. |
| STATus:OPERation? STATus:OPERation:CONDition? | Liest das Operation Statusregister und gibt einen Wert zurück, der dem Bitzustand an CONDITION entspricht. |
| STATus:OPERation:EVENT? | Liest das Event-Unterregister des Operation Statusregister und gibt einen Wert zurück, der dem Bitzustand von EVENT entspricht. |
| STATus:OPERation:ENABle_<NR1> STATus:OPERation:ENABle? | Definiert einen Filter für die Bits im CONDITION bevor sie in das Event-Unterregister des Operation Statusregister übergeben werden oder liest den aktuellen Filterwert. Dies kann dazu dienen, bestimmte Ereignisse zu unterdrücken. Standardmäßig sind alle Bits aktiviert, für die ein Signal aktiviert wurde (siehe Übersichtsschema). Das bedeutet, der Setzwert darf die Wertigkeit der verwendeten Bits nicht überschreiten. |

3.3.10 Adressierung der Ausgänge

Single-Modelle haben einen Ausgang, genannt Ausgang 1. Triple-Modelle haben noch einen zweiten, separat steuer- und adressierbaren Ausgang, genannt Ausgang 2. Das ist der Grund für die Adressierung, die ansonsten nicht nötig wäre. Bei den Triple-Modellen muß man den Befehl einem Ausgang zuweisen. Das geschieht bei Setz- sowie Abfragebefehlen durch einen Befehlszusatz. Folgendes gilt:

- Der Zusatz **@1** gehört zu Ausgang 1 (vorhanden bei Single- und Triple-Modellen)
- Der Zusatz **@2** gehört zu Ausgang 2 (nur bei Triple-Modellen vorhanden)
- Beim einem Single-Modell ist die Verwendung von **@1** nicht erforderlich, wird aber unterstützt
- Beim einem Triple-Modell ist es möglich, die Adressierung von Ausgang 1 durch **@1** wegzulassen, der Befehl wird dann automatisch Ausgang 1 zugeordnet
- Beim einem Triple-Modell können beide Ausgänge mit einem Befehl gleichzeitig adressiert und somit quasi synchron gesetzt bzw. abgefragt werden

Format der Adressierung für Setzbefehle anhand eines Beispiels:

CURR 12, (@1,2) adressiert die Ausgänge 1 und 2 eine Triple-Modells und soll 12 A setzen. Dieser Befehl würde bei einem Single-Modell zu einem Fehler führen, da dieses keinen Ausgang 2 zum Adressieren hat. Das Komma und das Leerzeichen nach dem Komma sind erforderlich. Alternative Schreibweisen: **CURR 12, (@1-2)** oder **CURR 12, (@1:2)**

Format der Adressierung für Abfragebefehle anhand eines Beispiels:

MEAS:ARR? (@2) adressiert Ausgang 2 eine Triple-Modells zur Abfrage der Istwerte. Das Leerzeichen nach dem eigentlichen Befehl, der beim Fragezeichen endet, ist erforderlich. Erweitert auf beide Ausgänge wäre der Befehl **MEAS:ARR? (@1,2)** und würde die Istwerte von beiden Ausgängen zurückgeben, als kommasetrennte Stringgruppe. Z. B. als 19.80 V, 3.25 A, 64.4 W, 0.00 V, 0.00 A, 0.0 W, wobei die erste Gruppe dann zu Ausgang 1 gehört. Die Werte lassen erkennen, daß Ausgang 2 entweder auf 0 V gesetzt wurde oder ausgeschaltet ist.

3.3.11 Sollwertbefehle



Alle Werte, die per individuellem Befehl setzbar sind und während Fernsteuerung an das Gerät gesendet werden, sind nicht nur durch die Nennwerte des Gerätes begrenzt, sondern noch zusätzlich durch die an der Bedieneinheit oder per digitaler Fernsteuerung justierbaren Einstellungsgrenzen!



Auch bei Fernsteuerung gilt, daß sich die Sollwerte von Strom und Spannung gegenseitig beeinflussen, damit die max. Leistung nicht überschritten werden kann. Daher sollte man z. B. den Strom als zweites setzen, wenn man dessen Wert so gesetzt haben will wie mit dem CURR-Befehl vorgegeben. Wie sich der jeweils andere Sollwert dann ergibt, kann nur durch Auslesen mit VOLT? bzw. CURR? ermittelt werden.

| Befehl | Beschreibung |
|---|--|
| [SOURce:]VOLTage_<NRf>[Unit] [SOURce:]VOLTage? | Setzt oder liest den Sollwert der Spannung. Die Einheit ist beim Setzen optional. Abfragte Werte kommen immer mit Einheit. Beispiele: VOLT_10 -> Absolute Kurzform, setzt 10 V SOURCE:VOLTAGE_5.35V -> absolute Langform, setzt 5,35 V |
| [SOURce:]CURRent_<NRf>[Unit] [SOURce:]CURRent? | Setzt oder liest den Sollwert des Stromes. Die Einheit ist beim Setzen optional. Abfragte Werte kommen immer mit Einheit. Beispiele: CURR_8 -> Absolute Kurzform, setzt 8 A SOURCE:CURRENT_18.7A -> absolute Langform, setzt 18.7 A |

3.3.12 Meßbefehle

Meßbefehle geben die vom Gerät ermittelten (U, I) bzw. berechneten (P) Istwerte zurück, also die tatsächlichen Werte am (adressierten) DC-Ausgang. Die Werte werden asynchron ermittelt. Das heißt, die Abfrage löst keinen Meßvorgang aus. Istwerte müssen nicht zwangsweise mit den gewünschten (Soll)Werten identisch sein und geben zu jedem Abfragezeitpunkt den zuletzt ermittelten Zustand am DC-Anschluß des Gerätes wieder. Die Meßwerte werden periodisch erfaßt.

| Befehl | Beschreibung |
|--------------------------------|---|
| MEASure:[SCALar:]VOLTage[:DC]? | Liest den letzten ermittelten Istwert der Spannung aus. Dieser wird augenblicklich zurückgegeben. Beispiel: 3.45V |
| MEASure:[SCALar:]CURRent[:DC]? | Liest den letzten ermittelten Istwert des Stromes aus. Dieser wird augenblicklich zurückgegeben. Beispiel: 10.12A |
| MEASure:[SCALar:]POWer[:DC]? | Liest den letzten ermittelten Istwert der Leistung aus. Dieser wird augenblicklich zurückgegeben. Beispiel: 34.9W |
| MEASure:[SCALar:]ARRay? | Liest alle drei Istwerte auf einmal. Der Befehl kombiniert die obigen drei. Die Istwerte werden dann in der Reihenfolge U, I, P zurückgegeben: Beispiel: 3.45V, 10.12A, 34.9W |

3.3.13 Zustandsbefehle

Zustandsbefehle verändern den Zustand des Gerätes im Sinne von Fernsteuerung ein/aus oder DC-Ausgang ein/aus bzw. fragen den Zustand ab.

| Befehl | Beschreibung |
|------------------------|---|
| SYSTem:LOCK_<B0> | Aktiviert die Fernsteuerung mit ON , sofern der gegenwärtige Zustand des Gerätes es zuläßt, bzw. verläßt diese mit OFF . Der Zustand kann vor oder nach diesem Befehl mit SYST:LOCK:OWN? abgefragt werden. |
| SYSTem:LOCK:OWNer? | Frägt den Zustand der Fernsteuerung ab. Folgende mögliche Rückgaben: REMOTE = Der (adressierte) Ausgang ist in Fernsteuerung NONE = Der (adressierte) Ausgang ist nicht in Fernsteuerung |
| OUTPut_<B0> OUTPut? | Schaltet den (adressierten) DC-Ausgang mit ON ein, sofern sich das der (adressierte) Ausgang in Fernsteuerung befindet, bzw. schaltet ihn mit B aus. Die Abfrageversion gibt in diesem Fall den Status des (adressierten) DC-Ausgangs zurück, wie er im Questionable Statusregister über die Bits 11 bzw. 14 abfragbar wäre. |

| Befehl | Beschreibung |
|---|---|
| SYSTem:ERRor? SYSTem:ERRor:NEXT? | Fragt den zuletzt aufgetretenen Kommunikationsfehler ab. Dieser wird dann aus dem Fehlerspeicher gelöscht und eventuelle weitere rutschen, so daß erneute Abfrage mit diesem Befehl alle Fehler nach und nach abfragt. Rückgabebeispiel: -223, "Too much data" . |
| SYSTem:ERRor:ALL? | Fragt alle momentan im Fehlerspeicher vorhandenen Fehler ab und gibt sie in einem kombinierten String zurück, in dem der letzte Fehler zuerst kommt. Rückgabebeispiel: -221, "Settings conflict;@1", -100, "Command error", -223, "Too much data" Die Verwendung von @1 kennzeichnet, daß der im Zusammenhang mit einer Aktion an einem bestimmten Ausgang auftrat. Der Fehlercode -221 deutet darauf hin, daß versucht wurde für Ausgang 1 etwas zu setzen während er nicht in Fernsteuerung war. |

3.3.14 Befehle für Schutzfunktionen

Das Gerät bietet zwei Gerätealarme (OVP, OCP) mit einstellbaren Schwellen, die zum Schutz des angeschlossenen Verbrauchers dienen. Dieselben Schwellen können natürlich auch am Gerät konfiguriert werden. Weitere Alarme wie OT können nicht parametrisiert werden.

| Befehl | Beschreibung |
|--|---|
| [SOURce:]VOLTage:PROTection[:LEVel]_<NRf>[Unit] | Setzt die sogenannte OVP-Schwelle für den (adressierten) Ausgang. Beispiel: VOLT:PROT 46 -> absolute Kurzform, setzt die Schwelle auf 46 V, was auch für ein 42 V Modell paßt. |
| [SOURce:]CURRent:PROTection[:LEVel]_<NRf>[Unit] | Setzt die sogenannte OCP-Schwelle für den (adressierten) Ausgang. Beispiel: CURR:PROT 11 -> absolute Kurzform, setzt die Schwelle auf 11 A, was nur für Modelle ab 10 A Nennstrom passen würde. |

3.3.15 Befehle für Einstellgrenzen

Einstellgrenzen sind zusätzliche, global wirkende, einstellbare Sollwertgrenzen, die verhindern sollen, daß ein bestimmter Sollwert aus Versehen zu hoch eingestellt werden kann. Zum Schutz gegen z. B. Überspannung am Verbraucher gibt es noch den Überspannungsschutz (OVP), aber generell ist es besser, einen falschen Sollwert gar nicht erst zuzulassen. Diese Limits sind grundsätzlich bis 100% des Nennwerts einstellbar.

Wird ein Sollwert an das Gerät gesendet, der höher als die Einstellgrenze ist, wird er ignoriert und ein Fehler in die Fehlerqueue geschrieben. Die Einstellgrenze kann dabei nicht niedriger gesetzt werden als der jeweilige Sollwert aktuell eingestellt ist. Die Befehle sind verknüpft mit den Einstellgrenzen („Limits“) wie sie im Menü des Gerätes gesetzt werden können. Siehe dazu auch die Gerätehandbücher.

| Befehl | Beschreibung |
|--|--|
| [SOURce:]VOLTage:LIMit:HIGH[?]<NRf>[Unit] | Legt die obere Einstellgrenze für den Spannungssollwert fest. Dies ist nur möglich, wenn das Limit größer oder gleich dem aktuellen Sollwert ist. Beispiel: VOLT:LIM:HIGH 30 -> absolute Kurzform, setzt die Einstellgrenze auf 30 V, sofern der aktuelle Spannungssollwert gleich oder kleiner 30 V ist. Das könnte man vorher mit VOLT? überprüfen. |
| [SOURce:]CURRent:LIMit:HIGH[?]<NRf>[Unit] | Legt die obere Einstellgrenze für den Stromsollwert fest. Dies ist nur möglich, wenn das Limit größer oder gleich dem aktuellen Sollwert ist. Beispiel: CURR:LIM:HIGH 6 -> absolute Kurzform, setzt die Einstellgrenze auf 6 A, sofern der aktuelle Stromsollwert gleich oder kleiner 6 A ist. Das könnte man vorher mit CURR? überprüfen. |

3.3.16 Befehle zum Tracking-Modus

Der sog. Tracking-Modus ist nur in den Triple-Modellen vorhanden. Er koppelt die beiden adressierbaren Ausgänge, so daß Ausgang 2 dem Ausgang 1 folgt, auch was das Setzen von Sollwerten, Limits und Schutzgrenzen angeht. Sobald Tracking aktiviert wurde, ist Ausgang 2 nicht mehr separat adressierbar. Tracking wird als Status „Tracking“ in den Anzeigen eingeblendet, sowie als Modus über das Ausschalten des Gerätes hinweg gespeichert.

| Befehl | Beschreibung |
|--|--|
| SYSTem:CONFig:TRACking_{ON OFF} SYSTem:CONFig:TRACking? | Aktiviert den Tracking-Modus mit ON bzw. deaktiviert ihn mit OFF , oder liest den Zustand zurück. Dieser zeigt denselben aktuellen wie das Bit im Questionable Statusregister. |

3.3.17 Weitere Befehle

Hier werden weitere Befehle zum Abfragen von Informationen vom Gerät gelistet.

| Befehl | Beschreibung |
|--------------------------------|---|
| SYSTem:NOMinal:VOLTage? | Liest den Nennwert der Spannung des Gerätes |
| SYSTem:NOMinal:CURREnt? | Liest den Nennwert des Stromes des Gerätes |
| SYSTem:NOMinal:POWer? | Liest den Nennwert der Leistung des Gerätes |
| SYSTem:DEVice:CLASs? | Liest die Geräteklasse, ein Zahlenwert, der zur Unterscheidung von verschiedenen Geräteserien dient. Bei Single-Modellen der Serie PS 2000 B ist dieser Wert immer 16, bei den Triple-Modellen ist er 24. |



Elektro-Automatik

EA Elektro-Automatik GmbH & Co. KG

Entwicklung - Produktion - Vertrieb

Helmholtzstraße 31-37

41747 Viersen

Telefon: 02162 / 37 85-0

Telefax: 02162 / 16 230

E-Mail: ea1974@elektroautomatik.de

Internet: www.elektroautomatik.de