BETRIEBSANLEITUNG

laser lieben

Universal-Positioniersteuerung

PS 90+

9013.0013 / 31.07.2020

OWIS GmbH
Im Gaisgraben 7
79219 Staufen i. Br. (Germany)
Tel. +49 (0) 76 33 / 95 04-0
Fax +49 (0) 76 33 / 95 04-440
info@owis.eu
www.owis.eu

Inhalt

1. Allgemeines 5
2. Ausführung und Lieferumfang 5
   2.1 Standard 5
   2.2 Zubehör 5
   2.3 Optionen 5
3. Sicherheit 6
      Ströme und Spannungen 6
      Erwärmung des Kühlkörpers bis max. 70° C 6
4. Normen und Richtlinien 6
5. Technische Übersicht 7
      Eingang 7
      Anschlüsse 7
6. Aufbau der Steuerung 8
      Ein-/Aus-Taster 8
      Reset-Taster 8
   6.1 Anschlüsse 9
      USB- und RS-232-Schnittstelle 9
      Ethernet-Schnittstelle 9
      Anybus®-Schnittstelle 9
      NOT-AUS-Funktion 9
      Stromversorgung 9
      Universal-Motoranschluss 9
      End- und Referenzschalter 10
      Encodereingang 10
   6.2 Eingänge und Ausgänge 10
7. Steuerungsarchitektur und Funktion 11
   7.1 Aufbau 11
   Hauptplatine 11
      Antriebsplatine 11
      Motorplatine 11
      Sicherungskonzept 11
   7.2 Betrieb unterschiedlicher Motortypen 11
      Schrittmotoren 11
      DC-Motoren 11
      BLDC-Motoren 11
   7.3 Konfiguration der Motorendstufe 11
      2-Phasen-Schrittmotor (Open Loop) 11
      DC-Servomotor 11
   7.4 Strombereichsumschaltung der Motorendstufe 12
      Vorwahl des Phasenstromes für 2-Phasen-Schrittmotoren . 12
      Strombereichseinstellung für DC-Servomotoren 12
8. Steuerungsfunktionen 12
   8.1 Trapezförmiges Punkt-zu-Punkt-Profil 12
   8.2 S-Kurven-Punkt-zu-Punkt-Profil 13
   8.3 Geschwindigkeitsmodus 13
   8.4 Referenzierung 14
   8.5 Linearinterpolation 14
      Begriffsbestimmung 14
      Funktionsprinzip 14
   8.6 Synchroner Start 14
   8.7 Funktionsweise der allgemeinen Bahnsteuerung 14
      Definition 14
      Realisierung des Vektormodus 15
      Kreisinterpolation 16
   8.8 Automatisches Reagieren auf externe
      Auslöser und Setzen von Ausgängen 17
   8.8.1 Automatisches Reagieren auf Eingänge 17
      Definition des Eingangs 17
      Mögliche Aktionen 17
      Zuweisung einer Aktion zu einem Auslöser 18
      Verhalten 18

Content

1. General information 48
2. Setup and Scope of Delivery 48
   2.1 Standard 48
   2.2 Accessories 48
   2.3 Options 48
3. Safety 49
   Currents and voltages 49
   Heat Sink Temperature up to 65° C max. 49
4. Standards and Directives 49
5. Technical Overview 50
   Input 50
   Connections 50
6. Setup of the Control Unit 51
   On/Off Switch 51
   Reset Button 51
   6.1 Connections 52
      USB and RS-232 Interfases 52
      Ethernet interface 52
      Anybus® Interface 52
      Emergency-Stop Function 52
      Power Supply 52
      Universal Motor Connector 52
      Limit and Reference Switches 53
      Encoder Input 53
   6.2 Inputs and Outputs 53
7. Control Unit Architecture and Function 54
   7.1 Assembly 54
      Main Board 54
      Drive Controller Board 54
      Motor Driver Board 54
      Safety Fuse Concept 54
   7.2 Operation of Different Motor Types 55
      Step Motors 55
      DC Motors 55
      BLDC Motors 55
   7.3 Settings of the Motor Output Stage 55
      2-Phase Step Motor (Open Loop) 55
      DC Servo and BLDC Motor 55
   7.4 Selection of the Current Range for the Motor Power Stage .. 55
      Phase Current Setting for 2-Phase Step Motors 55
      Current Range Setting for DC Servo and BLDC Motors 55
8. Control Functions 55
   8.1 Trapezoidal Point-to-Point Profile 55
   8.2 S-Curve Point-to-Point Profile 56
   8.3 Velocity Mode 57
   8.4 Reference run 57
   8.5 Operating Mode of Linear Interpolation 57
      Definition 57
      Functional Principle 57
   8.6 Synchronous Start 58
   8.7 Operation mode of the General Continuous Path Control .. 58
      Definition 58
      Realisation of Vector Mode 58
      Circular interpolation 60
   8.8 Automatic Reaction To External Triggers And
      Setting of Outputs 60
   8.8.1 Automatic Reaction To External Triggers 60
      Definition Of An Input 60
      Possible Actions 60
      Assigning An Action To A Trigger 61
      Behaviour 61

8.8.2 Automatisches Setzen von Ausgängen 18
Definition der Ereignisse 18
Zuordnung zu Ausgängen 18
Verhalten 18
8.8.3 Konfiguration 19
9. Wegerfassung 19
Encoder 19
Linearmesssystem 19
Funktionsweise der Nachlaufregelung 19
10. PID-Regelschleifenalgorithmus 21
11. Positioniergeschwindigkeit und -beschleunigung, Berechnung 21
11.1 2-Phasen-Schrittmotor (Open Loop) 21
Allgemeines 21
Periodendauer 21
Endgeschwindigkeit 21
Beschleunigung bei Trapezprofil 22
11.2 DC-Servomotor und 2-Phasen-Schrittmotor (Closed-Loop) 22
Allgemeines 22
Abtastzeit 22
Endgeschwindigkeit 22
Beschleunigung bei Trapezprofil 22
12. Nano-Hybrid-Ansteuerung 23
Allgemeines 23
Technische Übersicht und Aufbau der Steuerung 23
Sicherheit 23
Steuerungsarchitektur und Funktion 23
Anschluss 23
Positionierung im Nano-Hybrid-Betrieb 23
Allgemeine Beschreibung der Nachlaufregelung für Piezo-Antriebe 24
13. Inbetriebnahme der PS 90+ 25
13.1 Vorbereitung der Steuerung 25
Aufstellung 25
NOT-AUS-Funktion 25
13.2 Anschluss der Peripherie und Geräte 25
13.3 Systemstart 25
Initialisierung 25
Software 25
14. Fehlerüberwachung 26
14.1 Endschalter 26
Funktion der Endschalter-Überwachung 26
Konfiguration der End- und Referenzschalter 26
Wiederinbetriebnahme nach Achsenfehler 26
14.2 Endstufen-Fehlerüberwachung 26
14.3 Motion-Controller-Fehlerüberwachung 26
14.4 Time-Out-Überwachung 26
15. Joystick 27
16. Hinweise zum Aufbau einer eigenen Applikationssoftware 27
17. Befehlssatz der PS 90+ 28
Anhang 29
I Befehlstabelle 29
II Relevanz der Parameter für verschiedene Motortypen 42
III Belegungstabellen 43
TTL-Ein- /Ausgänge 43
Analog-Ein- /Ausgänge 43
SPS-Ein- /Ausgänge 43
RS-232 43
Universal-Motorstecker 44
Anschlusskabel 45
Kabelvorschlag für RS-232-Schnittstelle 45
Konformitätserklärung 89

8.8.2 Automatic Setting Of Outputs 61
Definition Of Events 61
Assigning Of Outputs 61
Behaviour 62
8.8.3 Configuration 62
9. Travel Measuring 62
Encoder 62
Linear Measuring System 62
Function of the follow up control 62
10. PID Servo Loop Algorithm 64
11. Positioning Velocity and Acceleration, Calculation 65
11.1 2-Phase Step Motor (Open Loop) 65
General Information 65
Cycle Time 65
Final Velocity 65
Acceleration for Trapezoidal Velocity Profiling 65
11.2 DC Servo Motor and 2-Phase Step Motor (Closed-Loop) 65
General Information 65
Servo Loop Cycle Time 65
Final Velocity 65
Acceleration for Trapezoidal Velocity Profiling 66
12. Nano-Hybrid Control 67
General Information 67
Technical Overview and Setup of the Control Unit 67
Safety 67
Control Architecture and Function 67
Connection 67
Positioning in nano-hybrid mode 67
General Description of Follow-up Control for piezo-drives 68
13. Initial Operation of the PS 90+ 69
13.1 Installation and Preparing 69
Installation 69
Emergency-Stop Function 69
13.2 Connection of Peripherals and Devices 69
13.3 Getting Started 69
Initialization 69
Software 69
14. Malfunction Monitoring 70
14.1 Limit Switches 70
Working Principle of the Limit Switch Monitoring 70
Configuration of Limit and Reference Switches 70
Reconnection after Axis Error 70
14.2 Output-Stage Error Monitoring 70
14.3 Motion-Controller Error Monitoring 70
14.4 Time-Out Monitoring 70
15. Joystick 71
16. Instructions Concerning the Setup of an Own Application Software 71
17. Command Set for the PS 90+ 72
Attachment 73
I Command Table 73
II Parameter Relevance for the different Motor Types 85
III Connecting Table 86
TTL In- / Outputs 86
Analog In- / Outputs 86
SPS In- / Outputs 86
RS-232 86
Universal Motor Connector 87
Connecting Cable 88
Recommendation for a RS-232 Interface Cable 88
UE Declaration of Conformity 89

1. Allgemeines

Die OWIS® Steuerung PS 90+ ist eine universelle Positioniersteuerung, die für anspruchsvolle Steuerungsaufgaben eingesetzt wird.

Sie ist modular aufgebaut und wird flexibel auf den jeweiligen Anwendungsbereich konfiguriert.

Die PS 90+ ist sehr leistungsstark und kann bis zu neun Achsen mit Schrittmotoren, DC-, oder BLDC- Servomotoren oder bis zu sechs Nano-Hybrid-Achsen betreiben.

Die in einem stabilen Metallgehäuse untergebrachte Steuerung kann eigenständig (Stand-Alone) oder mit einem Rechner betrieben werden.

Für die Kommunikation mit unterschiedlicher Peripherie sind zahlreiche Ein- und Ausgänge integriert, zum Beispiel: TTL/SPS/ Analog und PWM.

Ist eine erhöhte Präzision gefordert, kann an jeder Achse ein zusätzliches Inkremental- oder Wegmesssystem angeschlossen werden. Die Steuerung bietet außerdem die Möglichkeit, Schrittmotoren mit einem zusätzlichen Encoder im Closed-Loop-Modus zu betreiben.

Bei Applikationen, für die höchste Präzision gefordert ist, können mit der PS 90+ bis zu sechs Nano-Hybrid-Achsen betrieben werden. Die Hybrid-Technologie verbinden die Vorteile der spindelbetriebene Positionierung mit der Präzision von Piezo-Aktoren.

Die PS 90+ kann Punkt-zu-Punkt-Positionierbetrieb, Trapez- oder S-förmige Geschwindigkeitsprofile, sowie komplexe, mehrachsige Bahnsteuerungen, wie Linearinterpolation oder Kreisinterpolation, ausführen.

Die PS 90+ kann selbständig auf Zustandswechsel der digitalen Eingänge reagieren und zum Beispiel Positioniervorgänge starten oder stoppen. Sie ist außerdem in der Lage, in Abhängigkeit von bestimmten Ereignissen automatisch Ausgänge zu setzen.

Zum Lieferumfang der Steuerung gehört auch die Software OWISoft. Damit kann die PS 90+ komfortabel konfiguriert und betrieben werden. OWIS® Positioniereinheiten sind in OWISoft hinterlegt und müssen nur dem jeweiligen Antrieb zugeordnet werden.

Integration und Betrieb von Fremdmotoren ist ebenfalls möglich.

2. Ausführung und Lieferumfang

Die PS 90+ besteht aus einem Grundgerät für unterschiedliche Motorspannungen und wird entsprechend den Kundenanforderungen mit Achsmodulen, zusätzlichen Funktionen und Anschlüssen bestückt. Ein Nachrüsten mit Achsmodulen,
Funktionen und Anschlüssen ist ebenso möglich. Das Gerät wird bei OWIS® komplett aufgebaut, getestet und anschlussfertig geliefert. Die gültige Firmware für die Steuerung ist eingespielt. Sie kann gegebenenfalls über die USB- oder RS-232-Schnittstelle aktualisiert werden.

Zum Lieferumfang der Steuerung gehören:

- PS 90+ in der gewünschten Motorkonfiguration
- Netzkabel mit 2,5 m Länge
- USB-Kabel mit 2 m Länge
- CD mit OWISoft
- Betriebsanleitung als PDF Version in Deutsch und Englisch
- Kurzanleitung gedruckt und als PDF Version in Deutsch und Englisch
- Datenblatt gedruckte Version in Deutsch und Englisch

2.1 Standard

Die Steuerung verfügt über:

- USB-Anschluss
- RS-232-Anschluss
- Anschluss für externen NOT-AUS-Taster
- 4 Eingänge für Referenz- bzw. Endschalter je Achse
- 8 TTL- und Analogeingänge
- 8 TTL- und Analogausgänge
- 8 SPS-Ein- und Ausgänge
- Motoranschluss D-Sub 37-polig mit Anschluss für Motorhaltebremse (Option), End-/Referenzschalter und weitere Signale (siehe Pinbelegung, S.38) + je nach Version bis zu 3 Motorhaltebremsen Anschlüsse

2.2 Zubehör

Folgendes Zubehör ist erhältlich:

- Anschlusskabel mit Stecker für unterschiedliche Positioniersysteme
- Joystick für drei Achsen, analog, mit 3 m Kabel
- NOT-AUS-Taster mit 3 m Kabel
- bis zu 4 Ausgänge für Motor-Haltebremsen

2.3 Optionen

Es sind folgende Optionen verfügbar:

- Anybus®-Schnittstelle (Modbus/TCP)

3. Sicherheit

- Vor der Inbetriebnahme des Gerätes die Gerätebeschreibung lesen und diese für den späteren Gebrauch aufbewahren.
- Die Warnhinweise, Sicherheits- und Installationshinweise müssen beachten werden.
- Technische Daten und Anschlussbelegungen beachten.
- Das Gerät darf nur für den bestimmungsgemäßen Gebrauch verwendet werden.
- Das Gerät ist für Innenraumanwendungen konzipiert und darf nicht im Freien verwenden.
- Das Gerät muss vor zu hoher Luftfeuchtigkeit (80%), Erschütterungen sowie explosiven Gasen geschützt werden.
- Das Gerät darf nur von dafür qualifiziertem Fachpersonal in Betrieb genommen und verwendet werden.
- Die geltenden Installations-, Sicherheits- und Unfallverhütungsvorschriften sind einzuhalten.
- Das Gerät darf nur in seinem geschlossenen Metallgehäuse betrieben werden.
- Anschluss-, Montagearbeiten und Sicherungswechsel dürfen nur im spannungsfreien Zustand des Gerätes ausgeführt werden.
- Sicherungswechsel darf nur von einer autorisierten Fachkraft vorgenommen werden.
- Es darf nur die vorgeschriebene Sicherung verwendet werden.
- Nicht verwendete Slots müssen mit der vorgesehenen metallischen Slot-Blende abgedeckt sein.
- Vor dem Öffnen des Gerätes muss das Gerät spannungsfrei geschaltet und vom Stromnetz getrennt werden. Das Gerät ausschalten und das Kaltgerätekabel vom Gerät und der Netzversorgung entfernen!
- Das Gerät erzeugt Wärme (Netzteil/Endstufen). Die Lüftungsschlitze nicht abdecken, genügend Abstand zu anderen Gegenstände einhalten (mind. 15cm).
- Es dürfen nur dafür vorgesehene Komponenten und Betriebsmittel, so wie Kabel und Leitungen, welche den geltenden Bestimmungen/Normen entsprechen, angeschlossen werden.
- Es dürfen keine Leitungen mit Netzspannungs- oder gefährlichen Potentialen am Gerät angeschlossen werden (ausgenommen Netzanschluss).
- Schäden die durch Nichtbeachtung dieser Hinweise entstehen, sind von Ansprüchen jeglicher Art ausgeschlossen

Die Steuerung hat je nach Ausführung ein Gewicht von etwa 15 kg. An der Frontseite unten befindet sich eine Griffmulde und an der Rückseite oben ein Handgriff. Damit kann die PS 90 sicher transportiert werden.

Das Steuergerät ist für Betriebstemperaturen von + 10 bis + 40°C und Lagertemperaturen von - 10 bis + 50°C konzipiert.

Die PS 90 hat eine NOT-AUS-Schaltung, deren Funktion an die EN 418 angelehnt ist. Sie unterbricht die Leistungsversorgung der Motorendstufen auf der Sekundärseite (Kleinspannungsbereich 24 V bzw. 48 V). Ferner wird der an einer Motorendstufe angeschlossene Motortyp über einen Codierwiderstand erkannt. So wird verhindert, dass ein versehentlich falsch angeschlossener Motortyp (z.B. ein DC-Motor an einer Schrittmotor-Endstufe) unkontrolliert losläuft.

Die jeweiligen Achsmodule der Steuerungen dürfen nur mit den für sie konfigurierten Motortypen betrieben werden. Andere oder weiterführende Nutzungen entsprechen nicht dem vorgesehenen Verwendungszweck.

Ströme und Spannungen
Das Schaltnetzteil der PS 90+ besitzt einen Weitbereichseingang für eine Primärspannung von 100VAC bis 240VAC mit 50/60 Hz. Der Netzeingang ist über eine Feinsicherung 15AT (480W) abgesichert.

Ausgangsseitig sind keine besonderen Sicherheitsvorkehrungen erforderlich, da die PS 90+ ausschließlich mit Kleinspannung (PELV) bis 48VDC arbeitet. Falls die PS 90+ zur Ansteuerung von Nano-Hybrid-Achsen konfiguriert ist, wird der Piezozweig mit Spannungen im Bereich von -71V bis +71V betrieben. Besondere Sicherheitshinweise finden Sie im Kapitel „Nano-Hybrid-Ansteuerung".

Erwärmung des Kühlkörpers bis max. 70° C
Während des Betriebs der Steuerung wird die Abwärme der eingebauten Motorplatinen (Endstufen) über den seitlich angebrachten Kühlkörper an die Außenluft abgegeben.
Je nach Anzahl und Größe (Stromaufnahme) der angeschlossenen Motoren, sowie der Betriebsart (Kurzzeit-, Aussetz-, Dauerbetrieb) erwärmt sich der Kühlkörper und kann eine Temperatur von maximal 70° C erreichen. Wärmestau in der Steuerung oder am Kühlkörper ist zu vermeiden.
Es muss ein Mindestabstand von 15 cm zu geschlossenen Flächen und Wänden eingehalten werden.

Bei Nichtbeachtung der Sicherheitshinweise der Betriebsanleitung sind Sachschäden sowie Personenschäden möglich. Daher müssen diese jedem Nutzer zugänglich gemacht und eingehalten werden.

Die Universal-Positioniersteuerung PS 90+ ist nach den anerkannten sicherheitstechnischen Regeln gebaut und erfüllt die im folgenden Kapitel aufgeführten Normen und Richtlinien.

4. Normen und Richtlinien

Richtlinien:

2014/30/EU (EMV-Richtlinie)
Harmonisierte Normen

EN 55011:2016 + A1:2017
EN 61000-6-2:2005
EN 61000-3-2:2014
EN 61000-3-3:2013

2014/35/EU (Niederspannungs-Richtlinie)
Harmonisierte Norm

EN 61010-1:2010

2011/65/EU (RoHS-Richtlinie)
Harmonisierte Norm

EN 50581:2012

5. Technische Übersicht

Hersteller	OWIS GmbH
Name	Universal-Positioniersteuerung PS 90+

Eingang

Netzanschluss	Kaltgerätebuchse (C20), (Leitungslängen max. 3m, Schutzleiteranschluss)
Spannung:	100...240V AC 50/60Hz ±10%
Strom	max. 15 A
Sicherung	interne Sicherung 15 AT

Hinweis:
Die Stromaufnahme ist abhängig von der Ausstattung/Ausbaustufe und von den angeschlossenen Komponenten/Peripherie.

Betriebstemperatur	+10...40°C, 80% relative Feuchte, nicht kondensierend
Lagertemperatur	+10...50°C, 80% relative Feuchte, nicht kondensierend
Schutzart	IP20
Verwendungsort	Innenräume
Betriebshöhe	bis zu max. 2000m
Verschmutzungsgrad	2 (trocken nicht leitend)
Überspannungskategorie	II
Schutzklasse	I (Schutzleiter)
Gehäuse	Metallgehäuse
Maße	ca. 20x50x50cm BxHxT
Gewicht	ca. max. 15 kg, (abhängig von der Ausführung/Ausstattung)

Anschlüsse

PC/RS232	D-SUB Buchse 9pol. female; Kabellänge max. 3 m, geschirmt
PC/USB	USB 2.0, USB-Buchse Typ-B, female; Kabellänge max. 3 m, geschirmt
Anybus	Abhängig vom Modul
STOP	Rundstecker, female, max. 3 m. geschirmt
TTL I/O	D-SUB 37pol. male; Kabellänge max. 3 m, geschirmt
SPS I/O	D-SUB 37pol. female; Kabellänge max. 3 m, geschirmt
Analog I/O	D-SUB 37pol. male; Kabellänge max. 3 m, geschirmt
2x LAN	2x Ethernet, RJ45-Buchse LAN-Kabel, CAT5; geschirmt, Kabellänge max. 10m
M1 bis M9	Motoren/Antriebe, D-SUB 37 pol. female; Kabellänge Standard 3m geschirmt, längere Kabel auf Anfrage

Bedienelemente:

on/off

I/0

ACHTUNG:
Für den Netzanschluss darf nur das zugehörige Kaltgerätekabel verwendet werden!

VORSICHT:
Zur sichern Trennung von der Netzspannung, das Gerät ausschalten und das Kaltgerätekabel vom Gerät und vom Netz entfernen!

WARNUNG	Sicherheits- und Installationsvorschriften beachten! Anschlussbelegungen beachten, bei Fehlanschluss Funktionsausfall oder Zerstörung möglich!
Stromversorgung	Schaltnetzteil mit Strombegrenzung
Anzahl der Antriebe	bis zu 9 Achsen (SM, DC und BLDC), bis zu 6 Nano-Hybrid-Achsen (SM)
Antriebsart	2-Phasen Schrittmotoren Open Loop (OL), 2-Phasen Schrittmotoren Closed-Loop (CL), DC-Servomotoren, BLDC-Servomotoren, Nano-Hybrid-Achsen
Kommunikation	USB 2.0, RS-232, optional Anybus®-Schnittstelle (Modbus/TCP)
Aufbau	Tischgerät in Metallgehäuse Schutzart: IP 20
Encoder	Quadratur-Signale A/B und Index, RS-422- oder TTL-Pegel, mit 4-fach- Auswertung, max. Zählfrequenz 5 MHz (Signal) bzw. 20 MHz (Quadratur)
Funktionen	Parametrierbare Beschleunigungsrampe (/ Bremsrampe), Dreieckiges- bzw. trapezförmiges Geschwindigkeitsprofil oder S-Kurve
Bewegungsabläufe	Punkt-zu-Punkt Positionierbetrieb und Linear- und Kreisinterpolation

6. Aufbau der Steuerung

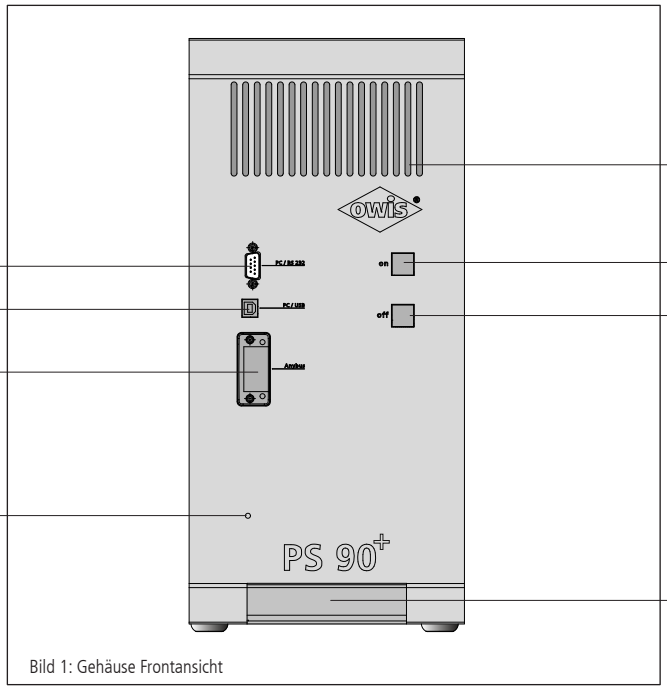

Bild 1: Gehäuse Frontansicht

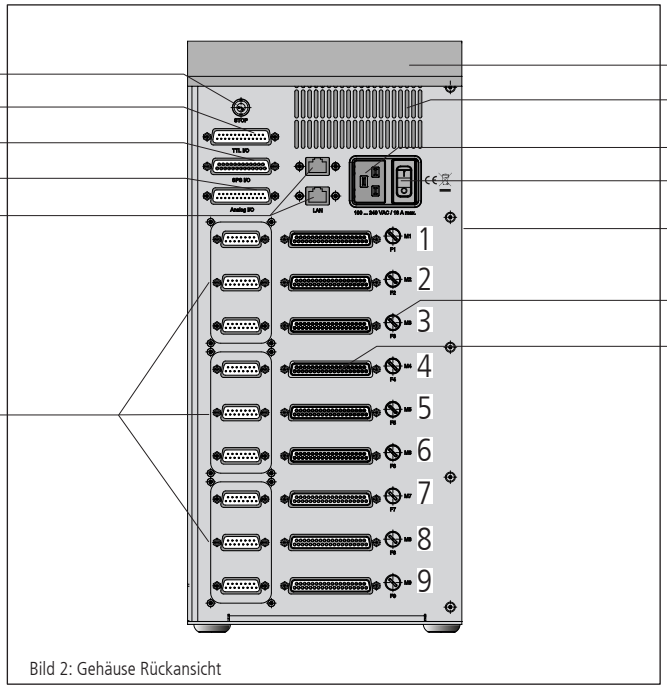

Bild 2: Gehäuse Rückansicht

Die PS 90+ ist in einem hochwertigen und stabilen Metallgehäuse untergebracht.

Zur internen Kühlung sind auf der Gehäusevorder- und Rückseite im oberen Bereich Lüftungsschlitze angebracht. Die Abwärme der eingebauten Motorplatinen (Endstufen) wird über den seitlich angebrachten Kühlkörper an die Außenluft abgegeben.

Ein-/Aus-Taster
Hauptschalter sowie Ein-/Aus-Taster der Steuerung sind beleuchtet. Die Beleuchtung des Hauptschalters zeigt die Aktivierung der Steuerung an. Die Taster werden nach dem Einschalten der Endstufen-Leistungsversorgung durch Betätigung des Ein-Tasters zur Betriebskontrolle mit maximaler Intensität beleuchtet. Betätigung des Aus-Tasters schaltet die Motorendstufen aus und reduziert die Hintergrundbeleuchtung der Taster auf ein Minimum.

Reset-Taster
Falls die PS 90+ nicht mehr reagiert oder unerwartete Fehler auftreten, kann der versenkte Reset-Taster z.B. mittels einer Kugelschreiberspitze betätigt werden. Alternativ ist es möglich, das Gerät aus- und wiedereinzuschalten.

6.1 Anschlüsse

Die Anschlüsse der PS 90+ befinden sich auf der Gehäusevorder- und auf der Gehäuserückseite. Dies sind Schnittstellen zur Kommunikation, Ein- und Ausgänge für Peripherie, sowie Anschlüsse für die Positionierer (siehe Bild 1 und 2).

Anschluss	Funktionen	Buchse
USB-Slave	Kommunikation mit einem PC	USB-Buchse Typ B
RS-232	Kommunikation mit einem PC	D-Sub 9-poliger Stecker
Ethernet-Schnittstelle	Verbindung mit einem PC über Ethernet / Integrierte Switch-Funktion	RJ 45
RS-485	Fernbedienung der Steuerung mit dem Handterminal	D-Sub 9-polige Buchse
Joystick	Manuelles Verfahren von maximal 3 Achsen	Analogeingänge 1, 2, 3
TTL-Ein-/Ausgänge	Interaktion mit externen Sensoren und Aktoren	D-Sub 25-poliger Stecker
Analog-Ein-/Ausgänge	Interaktion mit externen Sensoren und Aktoren, Joystick	D-Sub 25-poliger Stecker
SPS-Ein-/Ausgänge	Interaktion mit externer (SPS)-Steuerung	D-Sub 25-polige Buchse
Universal-Anschlussstecker	Motorversorgung mit Motor-Haltebremse und Encoder-/Endschalteranschluss	D-Sub 37-polige Buchse
Wegmesssystem/ Encoder	Anschluss von Wegmesssystemen	CONNEI 12-polige Buchse
Netzanschluss	Spannungsversorgung	Kaltgerätebuchse

Option:

Anybus-Modul Modbus/TCP	Kommunikation mit einem PC über Ethernet	RJ 45

USB- und RS-232-Schnittstelle

Die PS 90+ hat eine USB 2.0-Slave-Schnittstelle, der Anschluss befindet sich auf der Geräterückseite. Die Schnittstelle ist USB 1.1 und 2.0 kompatibel. Die USB-Schnittstelle der PS 90+ ist als sogenannte
COM-Brücke realisiert. Der Windows-Gerätetreiber erkennt die PS 90+
als „USB-Serial-Port" und weist ihr eine COM-Portnummer zu, die vom Anwender bei Bedarf verändert werden kann. Die USB-Schnittstelle wird nach erfolgreicher Installation als virtuelle RS-232-Schnittstelle angesprochen.

Alternativ zur USB-Schnittstelle kann die Steuerung über die RS-232 mit einem PC kommunizieren.

Die PS 90+ kann mit Übertragungsraten von 9 600, 19 200, 38 400, 57 600 oder 115 200 Baud arbeiten. Es ist unbedingt darauf zu achten, dass die Übertragungsrate der PS 90+ mit der im Gerätetreiber
eingestellten Übertragungsrate übereinstimmt, sonst ist keine Kommunikation möglich. Voreinstellung ist 9 600 Baud.
(Kann aus dem Abnahmeprotokoll entnommen werden.)

Ethernet-Schnittstelle

Die PS 90+ besitzt zwei Ethernet-Schnittstellen mit integrierter Switchfunktion. Über einen der beiden Anschlüsse kann die Steuerung mit dem lokalen Netzwerk verbunden werden. Der zweite Anschluss steht zum Verbinden weiterer Geräte zur Verfügung. Somit kann bei nur einem vorhandenen Netzwerkanschluss die Steuerung mit dem Netzwerk und ein PC über die Steuerung ebenfalls mit dem Netzwerk verbunden werden. Zum Verbinden muss der Port 8777 gewählt werden.

Anybus®-Schnittstelle

Die PS 90+ kann optional mit einem Anybus®-Modul „Modbus/TCP" geliefert werden. Mit diesem Modul ist es möglich, über Ethernet Kommandos zur PS 90+ zu schicken und entsprechende Rückmeldungen zu erhalten.

NOT-AUS-Funktion

An der Geräterückseite ist ein Anschluss für einen externen NOT-AUS-Taster vorgesehen, an welchem standardmäßig ein Kurzschlussstecker eingesteckt ist. Soll ein NOT-AUS-Taster angeschlossen werden, ist der Kurzschlussstecker zu entfernen.

> **Hinweis**:
> Wird der Kurzschlussstecker entfernt und kein NOT-AUS-Taster angeschlossen, ist die Funktion der Motorplatinen (Motorendstufen) blockiert

Die NOT-AUS-Schaltung der PS 90+ ist angelehnt an die EN 418 und unterbricht die Leistungsversorgung der Motorendstufen auf der
Sekundärseite (Kleinspannungsbereich 24 V oder 48 V). Die Funktion wird durch ein selbsthaltendes Relais mit zwangsgeführten Kontakten (2 Öffnerkontakte in Reihe) umgesetzt. Bei Abschaltung der Endstufen wird zusätzlich zur Endstufenversorgung die Endstufenfreigabe weggeschaltet (doppelte Sicherheit).

Stromversorgung

Die Stromversorgung der PS 90+ ist für eine Eingangsspannung von 100 VAC bis 240 VAC mit 50/60 Hz ausgelegt (Weitbereichseingang).Ein Schaltnetzteil generiert 24 VDC und versorgt die Ein- und
Ausgänge auf der Hauptplatine. Die Logikspannungen + 5 V, + 2,5 V und + 3,3 V für Haupt- und Antriebsplatine werden aus dieser 24 VDC-Versorgung erzeugt. Ein zweites Schaltnetzteil generiert die Zwischenkreisspannung für die Motorplatinen (wahlweise 24 oder 48 VDC). Diese Spannung speist die Leistungsendstufen der Motorplatinen.
Die Versorgungsspannungen für Logik und Leistung sind galvanisch getrennt.

Universal-Motoranschluss

Mit dem passenden OWIS® Anschlusskabel werden die OWIS® Positioniereinheiten angeschlossen. Über diesen Anschlussstecker wird der Motor mit Leistung versorgt, die Signale des Encoders und der Endschalter übertragen, sowie die Motor-Haltebremse, falls vorhanden, gesteuert.
Die Endstufe hat eine zusätzliche Schutzeinrichtung, die dafür sorgt, dass ein versehentlich falsch angeschlossener Motortyp (z.B. ein DC-Motor an einer Schrittmotor-Endstufe) nicht unkontrolliert startet. Am Motoranschlusskabel ist zwischen Pin 14 und Pin 15 ein Widerstand zur Codierung des Motortyps eingebaut.

Codierung:
- 0 Ohm: DC-Servomotor
- Widerstand unendlich: 2-Phasen-Schrittmotor
- 470 Ohm: BLDC

Beim Einschalten misst die Steuerung den Widerstandswert und signalisiert einen Fehler, wenn der gemessene Wert nicht zu der jeweiligen Steuerplatine passt. Die Fehlermeldung der Endstufe

wird über das Kommando „?ASTAT" und „?MPUNISTAT<n>" ausgelesen (siehe Befehlssatz ab S. 26).

Der Steckerbelegungsplan ist im Anhang aufgeführt. Die Belegung entspricht dem OWIS®-Standard.

End- und Referenzschalter
Pro Achse können maximal 4 Schalter angeschlossen werden. Dies können 24V-Induktivschalter, Mikroschalter, TTL-Hall-Effekt-Endschalter oder TTL-Lichtschranken sein. An die Eingänge können beliebige Pegel, ± 5V – ± 24V, Öffner oder Schließer, gegen $+U_b$ oder Masse schaltend, angeschlossen werden.

Einer der vier Schalter ist zusätzlich als Referenzschalter definiert.

Der aktive Pegel und die Zuordnung der Schalter werden per Software konfiguriert.

Encodereingang
Der Encodereingang ermöglicht sowohl den Anschluss von Encodern mit Leitungstreibern (antivalente Signale für CHA, CHB und optional Index I), als auch von Encodern mit TTL-/CMOS-Signalen.

Folgende Eingangssignale sind definiert:

Versorgung	V_{CC} (+ 5V); GND
Kanal	A (TTL oder CMOS)
Kanal	A invertiert
Kanal	B (TTL oder CMOS)
Kanal	B invertiert
Kanal	I (TTL oder CMOS)
Kanal	I invertiert

Die Umsetzung der antivalenten Signale auf TTL-Signale erfolgt mit RS-422-Leitungsempfängern. Schließt man einen Encoder mit TTL-/CMOS-Signalen an, so bleibt der Eingang für das invertierte Signal offen und wird intern mit einem hocho

6.2 Eingänge und Ausgänge

Zur Interaktion mit externen Sensoren und Aktoren sind entsprechende digitale und analoge Ein- und Ausgänge vorgesehen.

An die TTL-kompatiblen Eingänge können einfache Gabellicht-schranken etc. angeschlossen werden.

Mit den TTL-Ausgängen ist es möglich, digitale Hardware in der Anwendung direkt anzusteuern.

Die SPS-kompatiblen Eingänge ermöglichen die Verwendung der im Anlagenbau üblichen 24VDC-Induktiv-Sensoren in Zweidraht- und Dreidraht-Technik. Die Arbeitswiderstände der SPS-Eingänge können per Software gemeinsam als Pull-Up oder Pull-Down geschaltet werden.

Die SPS-Ausgänge steuern Magnetventile oder sonstige induktive und ohmsche Lasten direkt an (gegen + 24V schaltend).
Die Ausgänge sind kurzschlussfest.

Eigenschaften	Pegel	Strom	Sonstiges
TTL-Eingänge	0-5V		—
SPS-Eingänge	0-24VDC		2-Draht/3-Draht
Analogeingänge	0-4,096VDC		Auflösung 10 Bit
TTL-Ausgänge	0-5V	10mA	—
SPS-Ausgänge	0-24VDC	300mA	kurzschlußfest
Analogausgänge	0-4,096VDC	10mA	Auflösung 10 Bit
Leistungsausgänge	0-24VDC	1,0A	PWM

Die analogen Eingänge können Spannungen zwischen 0V und 4,096V direkt messen und mit 10-Bit-Auflösung wandeln (Referenzspannung: 4,096V). Die Ein- und Ausgänge sind nicht galvanisch getrennt.
Die Abfragebefehle „?ANIN<uv>" und „?INPUTS" beziehen sich auf dieselben Eingänge der PS 90 (siehe Befehlssatz ab S. 26).
Die Auswertung der Eingänge erfolgt entweder analog oder digital.

Die vier Leistungsausgänge sind pulsweitenmoduliert und nach Masse schaltend. Sie können induktive Lasten ansteuern, die kurzzeitig einen hohen Anzugsstrom und anschließend nur noch einen geringen Haltestrom brauchen, wie Haltebremsen oder Hubmagnete.

Die Leistungsausgänge können als Haltebremsenansteuerung konfiguriert werden.

Die NOT-AUS-Schaltung der PS 90 ist angelehnt an die EN 418 und unterbricht die Leistungsversorgung der Motorendstufen auf der Sekundärseite (Kleinspannungsbereich 24V oder 48V). Die Funktion wird durch ein selbsthaltendes Relais mit zwangsgeführten Kontakten (2 Öffnerkontakte in Reihe) umgesetzt. Bei Abschaltung der Endstufen wird zusätzlich zur Endstufenversorgung die Endstufenfreigabe weggeschaltet (doppelte Sicherheit).

7. Steuerungsarchitektur und Funktion

Die Steuerung besteht im Wesentlichen aus folgenden Komponenten:
1. ein eingebautes Netzteil

2. eine Hauptplatine

3. max. 9 Antriebsplatinen

4. max. 9 Motorplatinen (Endstufen)

7.1 Aufbau

Hauptplatine

Die Hauptplatine ist das Kernstück der PS 90+. Sie übernimmt die Steuerung des Hauptablaufs, kommuniziert mit dem PC und mit den Antriebsplatinen und verwaltet die digitalen und analogen Ein- und Ausgänge.

Die Hauptplatine hat einen USB-Anschluss für die Kommunikation mit einem PC. Eine weitere RS-232-Schnittstelle ist als alternative Kommando-Schnittstelle zum PC implementiert.

Mit dem optionalen Anybus®-Modul „Modbus/TCP" ist die Kommunikation mit einem PC über Ethernet möglich.

Antriebsplatine
Jede Antriebsplatine beinhaltet ferner einen Motion-Prozessor, der eine Achse steuern bzw. regeln kann. Der Motion-Prozessor verarbeitet die Befehle des Mikrocontrollers und generiert entsprechend die Ansteuersignale für die Endstufenmodule. Die Schnittstelle zu den Endstufen ist mittels Optokoppler galvanisch getrennt.

Motorplatine
Die PS 90+ kann mit maximal neun Motorplatinen bestückt werden.

Auf der Motorplatine befindet sich die Endstufe, die die Wicklung(en) des Motors mit Strom versorgt und damit das Drehmoment steuert.

Die Motorplatine ist mit dem Universal-Anschlussstecker verbunden, an dem der Motor mit seinen Wicklungen, der Encoder, evtl. die Hall-Effekt-Kommutierungssensoren und alle Schalter, die zu dieser Antriebsachse gehören, angeschlossen sind.

Sicherungskonzept
Für jede Motorplatine ist eine eigene Schmelzsicherung (5 x 20 mm) vorhanden, die entsprechend dem maximal auftretenden Strom ausgelegt ist. Sie soll helfen, im Falle eines Hardwaredefektes größeren Schaden zu vermeiden. Die Sicherung ist von der Geräterückseite zugänglich und kann von außen getauscht werden.

Standardmäßige Absicherung: 6,3 AT. Zusätzlich ist jede Motorplatine mit einer elektronischen Sicherung versehen. Wird der maximal zulässige Phasenstrom überschritten, so wir die Platine abgeschaltet. Zusätzlich wird die Freigabe entfernt.

7.2 Betrieb unterschiedlicher Motortypen

Schrittmotoren
Die PS 90+ ist für den Betrieb von 2-Phasen-Schrittmotoren ausgelegt, die sowohl gesteuert (Open Loop), als auch geregelt (Closed-Loop) betrieben werden können.

DC-Motoren
Die PS 90+ kann ebenfalls DC-Motoren (bürstenbehaftete Servomotoren) ansteuern.

Die Endstufe ist als H-Brücke mit Strombegrenzung ausgeführt, die mit einem PWM-Signal und einem Richtungssignal angesteuert wird. Es ist eine automatische Strombegrenzung eingebaut, die beim Überschreiten des maximalen Motorstroms anspricht.

BLDC-Motoren
Ein Betrieb von BLDC-Motoren (bürstenlose Servomotoren) mit drei Motorphasen ist auch möglich.

Die Endstufe steuert drei Motorwicklungen mit drei vom Motion-Controller generierten 50/50-PWM-Signalen an. In allen 3 Brückenzweigen wird der Summenstrom gemessen. Treten zu hohe Motorströme auf, so wird der Strom mittels Stromchopper begrenzt.

7.3 Konfiguration der Motorendstufe

Die Endstufen sind ab Werk auf einen Motortyp fest vorkonfiguriert. Diese Einstellung kann vom Anwender nicht geändert werden. Im Folgenden werden die Konfigurationsmöglichkeiten für die jeweiligen Typen beschrieben.

2-Phasen-Schrittmotor (Open Loop)
Für diesen Motortyp ist keine Strombegrenzung vorgesehen. Die Einstellung des Motorstromes ist im folgenden Abschnitt 7.4 beschrieben.

Die Stromregelung erfolgt über einen PID-Regler. Dieser darf nicht mit dem PID-Regler zur Positionslagebestimmung verwechselt werden, auch wenn die Begriffe identisch sind.

Über vier Parameter (P-, I-Schnell-, I-Langsam und D-Wert) wird die Reglercharakteristik bestimmt. Ungünstige, zu hoch eingestellte Werte können dazu führen, dass der Motor pfeift. Zu niedrige Werte reduzieren die maximal erreichbare Geschwindigkeit. Für jeden Motortyp müssen die optimalen Parameter individuell eingestellt werden. Die häuft bei Schrittmotoren auftretenden Geräusche im Betrieb können durch eine günstige PID-Einstellung stark reduziert werden. Insbesondere bei niedrigen Maximalgeschwindigkeiten ist so ein besonders geräuscharmer Betrieb möglich.

Die PS 90 wird in Kombination mit OWIS®-Positioniereinheiten bereits mit passenden Reglereinstellungen ausgeliefert. In OWISoft sind außerdem Parametersätze hinterlegt, die entweder für geräuschreduzierten, langsamen oder sehr dynamischen Betrieb optimiert sind.

DC-Servomotor
Beim Betrieb von DC-Servomotoren oder BLDC-Motoren ist es üblich, eine Strombegrenzung einzustellen. Dies erfolgt über den Befehl DRICUR (siehe Befehsreferenz). Die Strombegrenzung wird nach dem Einschalten der Steuerung mit der ersten Initialisierung übernommen. Um die Begrenzung zu ändern ist ein Neustart der Steuerung notwendig. DRICUR<n>=100 entspricht dabei 100% von 12 A. Die Werte müssen entsprechend kleiner gewählt werden. Eine zu niedrig eingestellte Begrenzung reduziert die verfügbare Dynamik deutlich, da DC-Motoren und BLDC-Motoren beim Beschleunigen und Verzögern kurzzeitig größere Ströme benötigen. Dies stellt in der Regel keine Gefahr für den Motor dar.

7.4 Strombereichsumschaltung der Motorendstufe

Die PS 90+-Endstufe besitzt zwei umschaltbare Strombereiche, um möglichst hohe Auflösung der Stromeinstellung bzw. möglichst feinen Mikroschrittbetrieb zu ermöglichen.

Der gewählte Strombereich wird abgespeichert. Um den neuen Strombereich zu aktivieren, ist es erforderlich, die Achse <n> nach der Bereichsumschaltung neu zu initialisieren.

Vorwahl von Strombereich 2 (hoch) für Achse <n> erfolgt über folgende Kommandofolge:

AMPSHNT<n>=1

INIT<n>

Zurückschalten in Strombereich 1 (niedrig) kann mittels folgender Befehlssequenz vorgenommen werden:

AMPSHNT<n>=0

INIT<n>

Vorwahl des Phasenstromes für 2-Phasen-Schrittmotoren
Für 2-Phasen-Schrittmotoren können Fahrstrom und Haltestrom separat voreingestellt werden. Die Einstellung für Achse <n> kann wie nachfolgend beschrieben vorgenommen werden. Die Angabe <uv> erfolgt als ganzzahliger Prozentwert des Maximalstromes im vorgewählten Strombereich (1 oder 2).

Fahrstrom: DRICUR<n>=<uv>

Haltestrom: HOLCUR<n>=<uv>

Maximaler Phasenstrom Strombereich 1 (entsprechend 100%): 2,4A

Maximaler Phasenstrom Strombereich 2 (entsprechend 100%): 5,45A

Hinweis:
Alle Strombereich 2 darf maximal ein Phasenstrom von 3,6A, entsprechend 66% des Endwerts, eingestellt werden.

Es sollte generell der kleinstmögliche Strombereich gewählt werden, um eine optimale Mikroschrittauflösung zu erhalten.

Strombereichseinstellung für DC-Servomotoren
Für DC-Servomotoren ist der geeignete Strombereich unter Berücksichtigung des thermisch zulässigen Dauerstroms des jeweiligen Motortyps vorzuwählen. Eine Strombegrenzung kann durch das Setzen der jeweiligen Parameter eingestellt werden. (Weitere Hinweise sind im Kapitel „Einstellelemente der Motorendstufe" zu finden.)

8. Steuerungsfunktionen

8.1 Trapezförmiges Punkt-zu-Punkt-Profil

Die folgende Tabelle umfasst die spezifischen Profilparameter für den trapezförmigen Punkt-zu-Punkt-Modus:

Profilparameter	Format	Wort-länge	Bereich
Position	32.0	32 bit	-2.147.483.648…+2.147.483.647 Counts
Geschwindigkeit	16.16	32 bit	(1…2.147.483.647)/65.536 Counts/Cycle
Beschleunigung	16.16	32 bit	(1…2.147.483.647)/65.536 Counts/Cycle²
Verzögerung	16.16	32 bit	(1…2.147.483.647)/65.536 Counts/Cycle²

Für dieses Profil errechnet der Host eine Beschleunigung, eine Verzögerung, eine Geschwindigkeit und eine Endposition.
Das Profil ist nach der Kurvenform (Bild 3, 5) benannt: Die Achse beschleunigt linear (anhand des programmierten Beschleunigungswertes), bis sie die programmierte Geschwindigkeit erreicht. Die Achse bremst dann linear ab (den Verzögerungsgswert nutzend), bis sie an der vorgegebenen Position stehen bleibt. Falls die programmierte Fahrdistanz so kurz ist, dass die Verzögerung einsetzen muss, bevor die Achse die programmierte Geschwindigkeit erreicht, wird das Profil keinen konstanten Geschwindigkeitsbereich aufweisen, und das Trapez wird zum Dreieck (Bild 4).

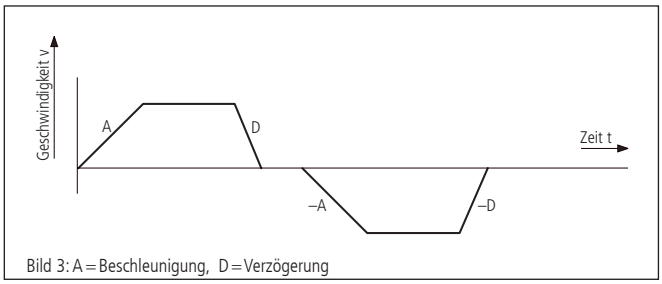

Bild 3: A = Beschleunigung, D = Verzögerung

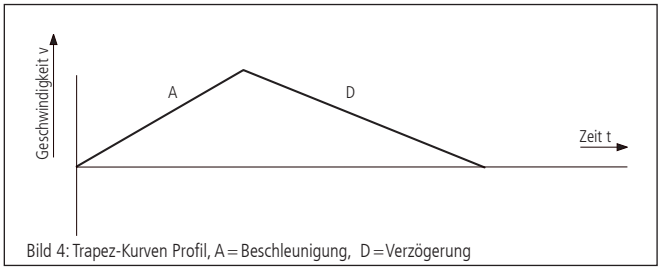

Bild 4: Trapez-Kurven Profil, A = Beschleunigung, D = Verzögerung

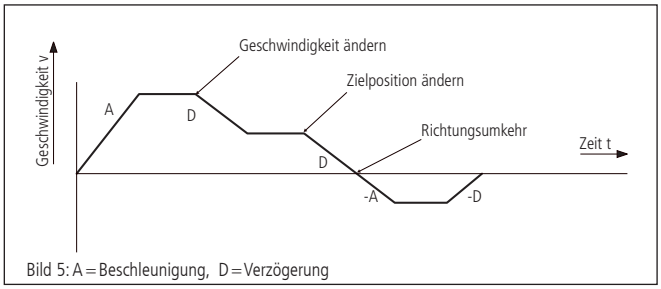

Bild 5: A = Beschleunigung, D = Verzögerung

Die Beschleunigungs- und Verzögerungsrampen können symmetrisch (wenn die Beschleunigung gleich der Verzögerung ist) oder asymmetrisch sein (wenn die Beschleunigung nicht gleich der Verzögerung ist).

Der Beschleunigungsparameter wird immer am Anfang der Bewegungssequenz benutzt. Danach wird der Wert für die Beschleunigung in dieselbe Richtung verwendet, und der Wert für die Verzögerung wird in entgegengesetzter Richtung eingesetzt. Falls keine Bewegungsparameter während der Bewegungssequenz verändert werden, wird der Beschleunigungswert verwendet bis die maximale Geschwindigkeit erreicht wurde. Der Verzögerungswert wird für die Abbremsrampe eingesetzt, bis die Geschwindigkeit auf Null sinkt.

Es ist möglich, einen der Profilparameter zu verändern, während die Achse sich in diesem Profilmodus befindet. Der Profilgenerator wird immer versuchen, die Bewegung innerhalb der durch die Parameter vorgegebenen gesetzten Bedingungen auszuführen. Wird während der Bewegung die Endposition in solch einer Weise verändert, dass die restliche Fahrdistanz das Vorzeichen wechselt, wird die PS 90+ mit Rampe bis zum Stopp abbremsen und dann in entgegengesetzte Richtung beschleunigen, um sich zu der neuen angegebenen Position zu bewegen.

8.2 S-Kurven-Punkt-zu-Punkt-Profil

Die folgende Tabelle fasst die Profilparameter für den S-Kurven-Punkt-zu-Punkt-Modus zusammen:

Profilparameter	Format	Wort-länge	Bereich
Position	32.0	32 bit	-2.147.483.648…+2.147.483.647 Counts
Geschwindigkeit	16.16	32 bit	(1…2.147.483.647)/65.536 Counts/Cycle
Beschleunigung	16.16	32 bit	(1…2.147.483.647)/65.536 Counts/Cycle²
Verzögerung	16.16	32 bit	(1…2.147.483.647)/65.536 Counts/Cycle²
Jerk	0.32	32 bit	(1…2.147.483.647)/4.294.967.296 Counts/Cycle³

Das S-Kurven-Punkt-zu-Punkt-Profil fügt im Vergleich zum Trapezprofil einen weiteren Parameter („Jerk" oder „Ruck") hinzu. Dieser gibt die Änderungsrate der Beschleunigung an. Wenn in diesem Profilmodus eine Positionierung durchgeführt wird, wird zunächst die Beschleunigung linear mit dem eingestellten Wert Jerk erhöht, bis sie den programmierten Wert erreicht. Der Übergang von konstanter Beschleunigung zu konstanter Geschwindigkeit erfolgt ebenfalls mit einem linearen Anwachsen der Verzögerung. Das Verhalten am Ende der Bewegung ist analog dazu.

Im S-Kurven-Profilmodus muss der gleiche Wert sowohl für die Beschleunigungs- als auch für die Verzögerunsrampe benutzt werden. Asymmetrische Profile sind nicht erlaubt. Dies ist nur im trapezförmigen Profilmodus möglich.

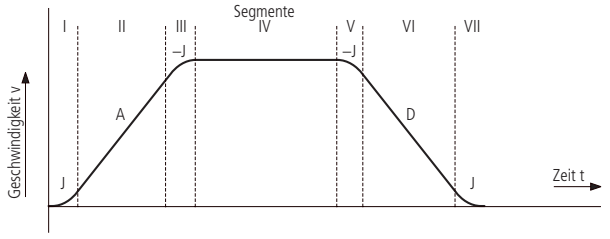

Bild 6: S-Kurven Profil, A = Beschleunigung, D = Verzögerung, J = Beschleunigungsänderung

Bild 7 zeigen ein typisches S-Kurven-Profil. In Segment I erhöht sich der Beschleunigungswert um den per Jerk gesetzten Wert, bis die maximale Beschleunigung erreicht wurde. Im nächsten Segment wird die Achse linear (Jerk = 0) beschleunigt. Das Profil wendet dann im Segment III den negativen Wert des Jerks an, um die Beschleunigung zu reduzieren. Im Segment IV verfährt die Achse jetzt mit maximaler (programmierter) Geschwindigkeit (V). Das Profil wird dann in einer dem Beschleunigungswert ähnlichen Weise abbremsen, indem in umgekehrter Richtung der negative Jerk verwendet wird, um zuerst die maximale Verzögerung zu erreichen (A), und dann die Achse zu einem Halt an der Endposition zu bringen.

Ein S-Kurven-Profil enthält u.U. nur einen Teil der in Bild 7 gezeigten Segmente. Dies kann z.B. der Fall sein, wenn nicht die maximale Beschleunigung vor dem „Halbweg" in Richtung Endgeschwindigkeit oder Endposition erreicht werden kann. Hier würde das Profil dann nicht die Segmente II und VI enthalten (siehe Bild 7.

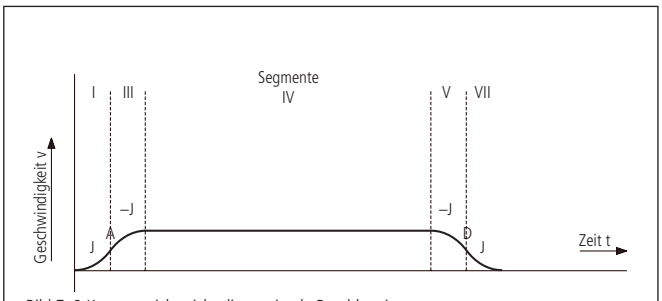

Bild 7: S-Kurve erreicht nicht die maximale Beschleunigung, A = Beschleunigung, D = Verzögerung, J = Beschleunigungsänderung

Falls eine Position derart angegeben wird, dass die Endgeschwindigkeit nicht erreicht werden kann, wird es kein Segment IV geben (siehe Bild 8).

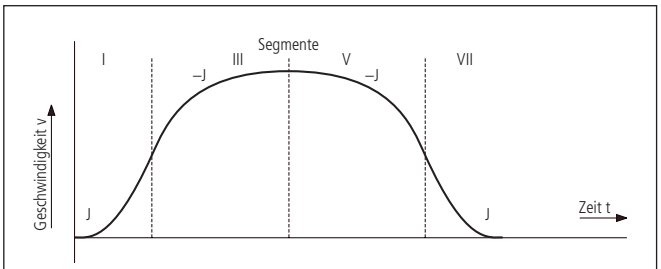

Bild 8: S-Kurve ohne maximales Geschwindigkeitssegment, A = Beschleunigung, D = Verzögerung, J = Beschleunigungsänderung

Im Gegensatz zum trapezförmigen Profilmodus erlaubt der S-Kurven-Profilmodus keine Änderungen an einem der Profilparameter, während die Achse in Bewegung ist. Ebenfalls darf die Achse nicht in den S-Kurven-Modus geschaltet werden, während die Achse in Bewegung ist. Es ist allerdings erlaubt, vom S-Kurven-Modus zu einem anderen Profilmodus während der Bewegung zu wechseln.

8.3 Geschwindigkeitsmodus

Die folgende Tabelle fasst die Profilparameter für den Geschwindigkeitsmodus zusammen:

Profilparameter	Format	Wort-länge	Bereich
Geschwindigkeit	16.16	32 bit	(-2.147.483.648…+2.147.483.647) /65.536 Counts/Cycle
Beschleunigung	16.16	32 bit	(1…2.147.483.647)/65.536 Counts/Cycle²
Verzögerung	16.16	32 bit	(1…2.147.483.647)/65.536 Counts/Cycle²

Im Gegensatz zu den trapezförmigen und S-Kurven-Profilmodi, bei denen die Endposition bestimmt, ob positive oder negative Geschwindigkeit vorgegeben wird, bestimmt das Vorzeichen des im Geschwindigkeitsmodus übergebenen Geschwindigkeitswerts, ob in positiver oder negativer Richtung gefahren werden soll. Deswegen kann der Geschwindigkeitswert der zur PS 90+ übermittelt wird, positive Werte (für positive Bewegungsrichtung) oder negative Werte (für entgegengesetzte Bewegungsrichtung) annehmen. Bei diesem Profil wird keine Endposition angegeben.

Die Bahn wird ausgeführt, indem die Achse mit dem angegebenen Wert kontinuierlich beschleunigt, bis die jeweilige Endgeschwindigkeit erreicht wird. Die Achse fängt an abzubremsen, wenn eine neue Geschwindigkeit angegeben wird, die einen kleineren Wert hat als die aktuelle Geschwindigkeit oder ein anderes Vorzeichen hat als die aktuelle Richtung vorgibt.

Ein einfaches Geschwindigkeitsprofil sieht aus wie ein einfaches trapezförmiges Punkt-zu-Punkt-Profil, wie in Bild 4 dargestellt. Bild 10 zeigt ein komplizierteres Profil, in dem beides, die Geschwindigkeit als auch die Bewegungsrichtung, zweimal wechseln.

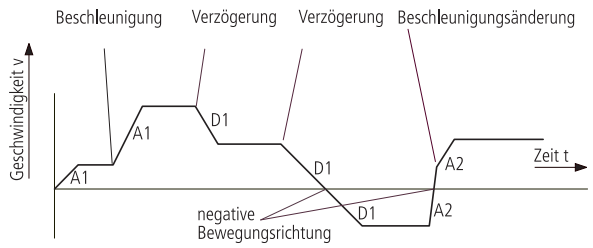

Bild 9: Geschwindigkeitsprofil

Hinweis:
Im Geschwindigkeitsmodus ist die Achsenbewegung nicht an eine Endposition gebunden. Es liegt in der Verantwortung des Anwenders, Geschwindigkeits- und Beschleunigungswerte zu verwenden, die einen sicheren Bewegungsablauf garantieren.

8.4 Referenzierung

Bei der Referenzfahrt wird einer der vier Endschalter angefahren. Die Position kann an dieser Stelle genullt werden. Dazu werden zwei Referenzfahrtgeschwindigkeiten mit Betrag und Vorzeichen und eine Referenzbeschleunigung parametriert. Der Endschalter wird mit großer Geschwindigkeit angefahren und mit kleiner Geschwindigkeit verlassen, dann wird gestoppt.

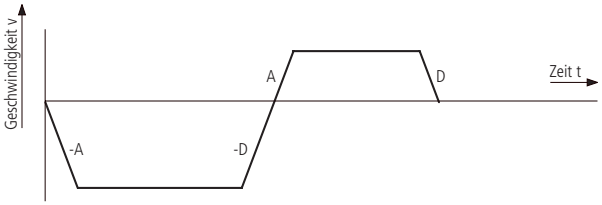

Bild 10: Referenzfahrt, A = Beschleunigung, D = Verzögerung

8.5 Linearinterpolation

Begriffsbestimmung
Linearinterpolation bezeichnet hier die Synchronisation der Bewegung aller beteiligten Achsen derart, dass die Achsen quasi-simultan starten und ihre Ziele praktisch gleichzeitig erreichen. Die Bewegung erfolgt hierbei mittels trapezförmiger Geschwindigkeitsprofile, wobei die Beschleunigungs- und Bremsrampen so angepasst werden, dass alle Achsen ebenfalls synchron beschleunigen bzw. bremsen. Die Bewegung eines aus Linearachsen bestehenden XYZ-Systems, das über Linearinterpolation angesteuert wird, beschreibt somit im kartesischen Koordinatensystem näherungsweise eine Gerade im Raum.

Die Achse mit der niedrigsten Achsnummer, welche den längsten Verfahrweg (umgerechnet in Inkremente) zurückzulegen hat, wird als Führungsachse f bezeichnet. Auf diese Achse werden die restlichen an der Linearinterpolation beteiligten Achsen steuerungsintern per Software synchronisiert.

Funktionsprinzip
Welche der maximal 9 Achsen an der Linearinterpolation beteiligt sind, wird über einen Binärcode beim Start der Achsen angegeben. Ein gesetztes Bit bedeutet hierbei, dass die entsprechende Achse aktiv ist.

Für jede Achse muss vor Verwendung der Linearinterpolation ein maximaler Geschwindigkeits- sowie ein maximaler Beschleunigungswert definiert werden, der während des Positioniervorganges nicht überschritten werden darf. Das Geschwindigkeit-Zeit-Profil eines linearinterpolierten Bewegungsablaufes ist symmetrisch.

Unter Berücksichtigung der digitalen Systemzeit (Abtastzeit bzw. Periodendauer des Profilgenerators) für jede Achse werden die Maximalwerte in der Weise umgerechnet, dass die Führungsachse f schnellstmöglich (mit maximal möglicher Geschwindigkeit v_{max} (f) und Beschleunigung a_{max} (f)) ihr Ziel erreicht. Die restlichen Achsen werden auf die Führungsachse synchronisiert, wobei die gegebenen Grenzwerte von der Steuerung einzuhalten sind.

Die Linearinterpolationsachsen seien nachfolgend mit (i) bezeichnet. Das folgende Diagramm zeigt den prinzipiellen Verlauf des Geschwindigkeitsprofils der Führungsachse v_f (t) und einer beliebigen Linearinterpolationsachse v_n (t) an einem Beispiel:

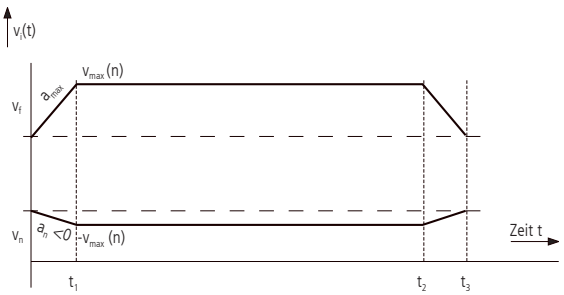

Bild 11: Linearinterpolation

Die Fahrdistanz der Achse (n) ist im Beispiel negativ, die Fahrdistanz der Führungsachse (f) positiv. Zum Zeitpunkt t_1 ist die Beschleunigungsphase beendet. Die Bremsung wird bei t_2 eingeleitet, und alle Achsen stoppen gemeinsam zum Zeitpunkt t3.

8.6 Synchroner Start

Ähnlich der Linearinterpolation ist es möglich, den Positioniervorgang oder den Geschwindigkeitsmodus mehrerer Achsen synchron starten zu lassen.

Mit Hilfe der dazu notwendigen Befehle (siehe Befehlsreferenz) werden alle Berechnungen zunächst intern durchgeführt. Anschließend erfolgt ein nahezu gleichzeitiger Start aller ausgewählter Achsen. Im Unterschied zur Linearinterpolation führt jede Achse ihre Bewegung so aus, wie sie es auch beim einzelnen Start geschehen würde.

8.7 Funktionsweise der allgemeinen Bahnsteuerung

Definition
Die PS 90+ ermöglicht, beliebige Bahnkurven über Ketten von Einzelvektoren zu approximieren, die in Form einer Vektortabelle an die Steuerung übergeben werden. Die allgemeine Bahnsteuerung wird somit über einen Vektormodus realisiert.

In die Vektortabelle werden relative Positionswerte eingetragen, die zu bestimmten, diskreten Zeitpunkten möglichst genau erreicht werden sollen. Bezugspunkt bzw. Startpunkt der Tabellenvektoren ist die jeweilige aktuelle Sollposition der Achsen.

Die approximierten Bahnkurven werden im Geschwindigkeitsmodus mit Trapezprofil gefahren.

Realisierung des Vektormodus

Vektortabelle

Jeder Tabelleneintrag n definiert ein komplettes Fahrsegment und enthält den relativen Fahrtvektor $\vec{\Delta x}$ für maximal neun Achsen (a bis i, entsprechend den Achsnummern 1 bis 9), das für die Fahrt des Vektors vorgegebene Zeitintervall Δt, einen 16-Bit-Funktionscode F, einen 8-Bit-Fehlercode E und einen 8-Bit-Achsfreigabecode T:

n	$\vec{\Delta x}$	Δt	F	E	T
1	$\Delta x_{a1},\ldots, \Delta x_{i1}$	t_1	t_1	e_1	t_1
...	...	...	...	...	...
N	$\Delta x_{aN},\ldots, \Delta x_{iN}$	Δt_N	f_N	e_N	t_N

Es können maximal 4000 Vektoren definiert werden ($N_{max} = 4000$).

Die Elemente des Fahrtvektors (Einzeldistanzen) werden als ganzzahlige Werte mit Vorzeichen (Integer 16-Bit) dargestellt. Die maximale Wegdistanz für ein Zeitintervall Δt_n beträgt 2147483648 Inkremente, d.h. als Wertebereich für einen Positionseintrag ist ein Zahlenwert von -2147483648 bis +2147483648 zulässig.

Segmentdauer

Das Zeitintervall Δt_n für Fahrsegment <n> wird als ganzzahliges Vielfaches von 1,024 ms angegeben. Der Wertebereich reicht von 20 bis 65535, woraus sich eine definierbare Segmentzeit von minimal 20,48 ms bis maximal 67,10784 s in Schritten von 1,024 ms ergibt:

$\Delta t_{n_{min}} = 20 \cdot 1{,}024\,\text{ms} = 20{,}48\,\text{ms}$

$\Delta t_{n_{max}} = 65635 \cdot 1{,}024\,\text{ms} = 67{,}10784\,\text{s}$

Steuercodes

Alle hier verwendeten Codes (F, E und T) sind prinzipiell Binärcodes, die grundsätzlich als positive ganzzahlige Werte (Integer) repräsentiert und an die Steuerung übergeben werden, unabhängig von dem mittels „TERM=..." vorgewählten Terminalmodus.

Der Funktionscode F wird als 16-Bit-Wert dargestellt. Bit 15 wird zur Vorwahl der Betriebsart, d.h. „konstante Geschwindigkeit" (v=const., Bit 15 gelöscht) oder „konstante Beschleunigung" (a=const., Bit 15 gesetzt), verwendet.

Die restlichen Bits werden benutzt, um bis zu drei Ausgänge pro Zeile jeweils entweder zu setzen oder zu löschen. Über Bits 0-3 wird binär der Ausgang gewählt. Bit 4 entscheidet dann, ob der Ausgang gesetzt oder gelöscht wird. Dieses Schema wiederholt sich für Bits 5 bis 9 und Bits 10 bis 14.

Werte für die Wahl des Ausgangs zwischen 1 und 8 entsprechen den TTL-Ausgangsnummern 1 bis 8. Für die SPS-Ausgänge 1 bis 7 muss entsprechend 9 bis 15 gewählt werden. Wird der Ausgangswert auf Null gesetzt, so wird keine Aktion ausgeführt.

Somit ist für die Standard-Betriebsart „konstante Geschwindigkeit" f = 0 zu setzen, wenn keine Ausgänge angesteuert werden sollen.

Der 8-Bit-Fehlercode E gibt an, ob und gegebenenfalls bei welcher der maximal 8 im Vektormodus aktiven Achsen während der Plausibilitätsüberprüfung der Vektortabelle ein Fehler aufgetreten ist. Hierbei zeigt ein gesetztes Bit 0 einen Fehler bei Achse 1 an, ein gesetztes Bit 1 einen Fehler bei Achse 2 usw. Der 8-Bit-Freigabecode T definiert, welche der Achsen 1 bis 8 im Vektormodus aktiv ist. Die Zuordnung der einzelnen Bits zur Achsnummer entspricht dem Fehlercode E, d.h. ein gesetztes Bit 0 bedeutet, dass Achse 1 aktiv ist usw.

Betriebsarten

In den nachfolgenden Diagrammen werden beide über den Funktionscode F vorwählbaren Betriebsarten anhand des Geschwindigkeit-Zeit-Verlaufes am Beispiel veranschaulicht. Die Zeitintervalle der fünf dargestellten Fahrsegmente werden mit „Δt_1" bis „Δt_5" bezeichnet, die Geschwindigkeitswerte am Ende des jeweiligen Segments mit „v_1" bis „v_5" und die Beschleunigungswerte mit „a_1" bis „a_5".

Geschwindigkeit-Zeit-Diagramm für Betriebsart v=const. (Beispiel):

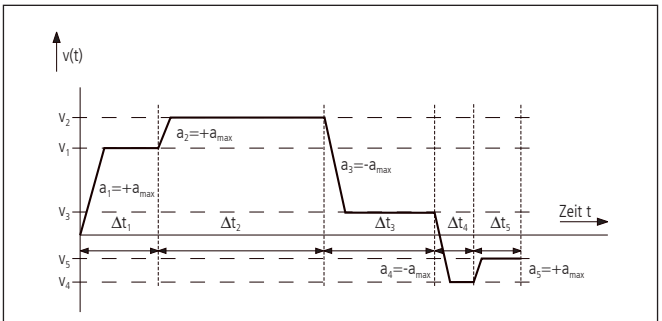

Bild 12: Geschwindigkeit-Zeit-Diagramm für Betriebsart v=const.

Die Fahrgeschwindigkeit wird im Konstantgeschwindigkeits-Modus mit der vorgegebenen Maximalbeschleunigung geändert und bleibt danach konstant. Sie wird während der Abarbeitung der Vektortabelle für jedes Segment zyklisch neu berechnet. Eine eventuell auftretende Lageabweichung am Ende eines Segments fließt im darauffolgenden Segment als Korrekturwert ein, um eine Akkumulation des Positionierfehlers zu vermeiden.

Geschwindigkeit-Zeit-Diagramm für Betriebsart a=const. (Beispiel):

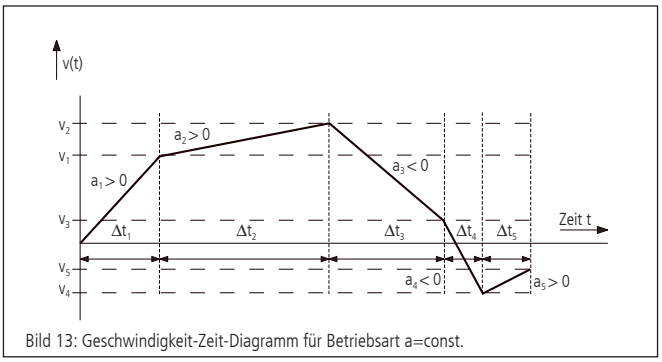

Bild 13: Geschwindigkeit-Zeit-Diagramm für Betriebsart a=const.

Die Fahrgeschwindigkeit ändert sich im Konstantbeschleunigungs-Modus stetig. Der Beschleunigungswert ist innerhalb eines Segments für jede Achse konstant. Endgeschwindigkeit und Beschleunigung innerhalb des Segments werden während der Abarbeitung der Vektortabelle für jedes Segment zyklisch neu berechnet. Eine eventuell auftretende Lageabweichung am Ende eines Segments fließt im darauffolgenden Segment, analog zur Betriebsart v=const., als Korrekturwert ein.

Maximalgeschwindigkeit und -beschleunigung

Die maximal zulässige Geschwindigkeit bzw. Beschleunigung im Vektormodus wird für jede Achse separat mittels der Befehle „IVEL" und „IACC" gesetzt. Diese Grenzwerte gelten gleichermaßen für den Vektormodus als auch für den Betrieb mit Linearinterpolation.

Plausibilitätsprüfung

Über das Kommando „PTABPLAUS" kann eine Vektortabelle auf Plausibilität geprüft werden. Falls die gegebene Zielposition einer Achse nur bei Überschreitung des vorgegebenen Geschwindigkeits- oder Beschleunigungslimits erreicht werden könnte, wird für die betreffende Achse das entsprechende Bit im Fehlercode E gesetzt.

Gesetzte Fehlerbits werden während des Positioniervorgangs ignoriert und dienen nur der Information des Anwenders. Der Tabelleneintrag kann auch dann ausgeführt werden, wenn E ungleich Null ist, jedoch ist dann mit einem sehr großen Positionierfehler zu rechnen.

Achsenfreigabe

Für jede innerhalb eines Fahrsegments aktive Achse muss im Freigabecode T ein Bit gesetzt werden. Achsen mit gelöschtem Bit werden im Fahrtvektor nicht berücksichtigt bzw. mit der programmierten Maximalbeschleunigung auf Geschwindigkeit Null abgebremst, falls die aktuelle Fahrgeschwindigkeit ungleich Null sein sollte.

Syntax

Der Tabelleneintrag <n> wird über den Befehl „POSTAB" generiert und zur Steuerung übertragen. Die Syntax ist wie folgt:

POSTAB <n> = Δx_{an},..., Δx_{in}, Δt_n,f_n, e_n, t_n

Als Wert für den Fehlercode E sollte immer Null übergeben werden, damit eventuell gesetzte Fehlerbits gelöscht werden.

Die Plausibilitätsprüfung für die Fahrsegmente <n> bis zum Ende der Tabelle wird mittels

PTABPLAUS <n>

vorgenommen. Hierbei werden für alle aktiven Achsen jedes Segments die Geschwindigkeits- bzw. Beschleunigungswerte berechnet und die Einhaltung der gesetzten Grenzwerte überprüft. Im Fehlerfall wird das der Achse entsprechende Bit im Fehlercode E gesetzt. Der berechnete Geschwindigkeits- und Beschleunigungswert (Vel_i and Acc_i) für Segment <n> der letzten aktiven Achse <i> (d.h. derjenigen aktiven Achse mit der höchsten Achsnummer <i>) wird zu Kontrollzwecken ebenfalls in der Tabelle gespeichert und kann mittels „?POSTAB" ausgelesen werden. Beide Kontrollwerte dienen insbesondere der Fehlersuche bzw. der erweiterten Plausibilitätskontrolle von Fahrsegmenten mit einer einzigen aktiven Achse.

?POSTAB <n>

liefert als Antwort:

Δx_{an},..., Δx_{in}, Δt_n,f_n, e_n, t_n Vel_i, Acc_i

Beispiel:

Das nachfolgende Beispiel soll die grundlegenden Funktionen zur Erstellung der Tabelleneinträge veranschaulichen. Gegeben seien:

Segmentnummer: 0 (erster Tabelleneintrag)

Segmentzeit: ca. 100 ms

aktive Achsen für die Bahnsteuerung: Achsen 1, 2, 3

Geschwindigkeitslimits Achse 1, 2, 3 : 800000, 500000, 300000

Beschleunigungslimits Achse 1, 2, 3 : 2000, 4000, 10000

Fahrdistanzen Achse 1, 2, 3 (relativ, in Inkrementen): 1000, -500, 2000

Betriebsart a=const.

Zu berechnen sind die normierte Segmentzeit Δt_0 und der Freigabecode t_0:

$$\Delta t_0 = \frac{100\,\text{ms}}{1{,}024\,\text{ms}} \sim 98$$

$$t_0 = 2^0 + 2^1 + 2^2 = 7$$

Folgende Befehle sind zu senden, um die Geschwindigkeits- und Beschleunigungslimits zu setzen sowie den ersten Tabelleneintrag zu definieren:

IVEL1=800000

IVEL2=500000

IVEL3=300000

IACC1=2000

IACC2=4000

IACC3=10000

POSTAB0=1000,-500,2000,0,0,0,0,0,0,98,32768,0,7

Plausibilitätskontrolle mittels

?PTABPLAUS0

und Auslesen des Tabellenelements über

?POSTAB0

ergibt als Antwort:

1000,-500,2000,0,0,0,0,0,0,98,32768,4,7,668734,1705,

Der Fehlercode „4" zeigt an, dass der Eintrag für die dritte (und letzte) Achse fehlerhaft ist. Es wurden ein Geschwindigkeitswert von 668734 und eine Beschleunigung von 1705 bei einer gegebenen Fahrstrecke von 2000 Inkrementen für diese Achse berechnet. Der Geschwindigkeitswert liegt über dem zulässigen Grenzwert 300000.

Fahrtende

Nach Abarbeitung des letzten Tabelleneintrags oder bei gelöschtem Freigabe-Bit bremsen die dann nicht mehr aktiven Achsen mit der jeweiligen Maximalbeschleunigung auf Geschwindigkeit Null ab. Danach wird der Geschwindigkeitsmodus deaktiviert und die Achsen werden von Bahnsteuerungskontrolle auf Positionshaltung umgeschaltet.

Daraus ergibt sich bei Beendung der Bahnkurve ein Nachlaufen um eine gewisse durch die Ausgangsgeschwindigkeit am Ende des letzten Segments und die Maximalbeschleunigung bestimmte Distanz.

Auswahl von Segmenten

Neben der Möglichkeit, über Freigabebits die gesamte Tabelle zu segmentieren, kann dies auch direkt über den Startbefehl PTABGO erfolgen. PTABGO<n> startet die Tabelle ab Zeile n. PTABGO<n><m> führt die Zeilen n bis m aus.

Kreisinterpolation

Die approximative Bahnkurvenerzeugung über tabellierte Segmente ermöglicht auch, mit zwei beliebigen Achsen X und Y eine kreisähnliche Figur bzw. einen Teil davon zu generieren. Hierbei wird der gewünschte Kreisbogen durch eine Sequenz von Kreissekanten angenähert.

Über einen speziellen Befehl kann die Vektortabelle ab einem bestimmten Index mit entsprechenden Kreisdaten gefüllt werden, sofern die entsprechenden Basisparameter vorher korrekt gesetzt worden sind.

Über einen Skalierungsfaktor, der die Weginkremente der beiden Achsen in eine bestimmte Beziehung zueinander setzt, ist es möglich, unterschiedliche Achsauflösungen zu kompensieren oder elliptische Konturen zu erzeugen.

Definitionen:

Nummer der Sekanten: $k \in (1,..., m)$; m = Gesamtanzahl Sekanten

Startwinkel (Winkeloffset) des Kreissegments: α

Vom Kreissegment abzudeckender Winkelbereich: $\Delta\alpha$

Radius des Kreissegments: r

Veranschaulichung am Diagramm:

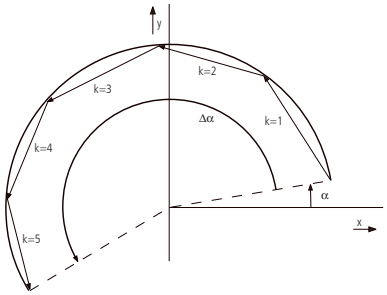

Bild 14: Geschwindigkeit-Zeit-Diagramm für Betriebsart

hier: Teilkreis mit Radius r, Winkeloffset $\alpha = 10°$,
Winkelbereich $\Delta\alpha = +190°$, m = 5 Sekanten

Berechnung

Das zu approximierende Kreissegment wird über den Radius, die Sekantenanzahl, Winkeloffset und Winkelbereich definiert.

Die Drehrichtung wird über das Vorzeichen der Winkelbereichsangabe festgelegt. Hierbei entspricht ein positiver Winkel einer Drehung im Gegenuhrzeigersinn bei entsprechender Anordnung der Achsen (siehe auch Lage des Koordinatensystems in oben genanntem Diagramm).

Der Startwinkel der einzelnen Sekantenvektoren k ergibt sich zu:

$$\alpha_k = \alpha + \Delta\alpha \cdot \frac{k-1}{m}$$

Die x- und y-Koordinaten der Sekantenvektoren sind dann:

$$\Delta x_k = -2r \cdot \sin\left(\frac{\Delta\alpha}{2m}\right) \cdot \sin\left(\alpha_k + \frac{\Delta\alpha}{2m}\right) \quad \text{und}$$

$$\Delta y_k = 2r \cdot \sin\left(\frac{\Delta\alpha}{2m}\right) \cdot \cos\left(\alpha_k + \frac{\Delta\alpha}{2m}\right)$$

Mit $\left|2r \cdot \sin\left(\frac{\Delta\alpha}{2m}\right)\right|$ wird die Länge eines Sekantenvektors bezeichnet.

Skalierungsfaktor

Der Skalierungsfaktor zum Ausgleich unterschiedlicher Auflösungswerte der beiden Kreisinterpolations-Achsen bzw. zur Realisierung von Ellipsen wird über Zähler und Nenner dargestellt, die über zwei separate Kommandos gesetzt werden können. Der Nenner sei mit N, der Zähler mit Z bezeichnet.

Falls N > Z, führt die Y-Achse und die Wegangaben für X werden durch (N / Z) dividiert. Falls Z > N, führt die X-Achse und die Wegangaben für Y werden durch (N / Z) dividiert. Der Standardwert ist Z = N = 1, falls seitens des Anwenders keine Angaben gemacht werden.

Syntax

Ab Tabellenelement <n> werden über den Befehl „PTABCIRCLE" Kreisdaten in Form von <m> Sekantenvektoren generiert und zur Steuerung übertragen. Hierbei bedeutet Angabe von Null für eine Achsnummer, dass die Achse nicht verwendet wird. Die Syntax ist wie folgt:

PTABCIRCLE <n> = <Achsnummer x>, <Achsnummer y>, Δt_n, f_n, m_n, r_n, α_n, $\Delta\alpha_n$, Z_n, N_n

Beispiel:

PTABCIRCLE0=1,2,326,0,5,1000,10,190,1,1

generiert einen Teilkreis ab Tabellenelement 0 mit Achse 1 als X- und Achse 2 als Y-Achse, Segmentzeit 1/3 Sekunde, Betriebsart v=const., 5 Sekanten, Radius 1000 Inkremente, Startwinkel 10°, Winkelbereich 190° und Skalierung 1 mit Trapezprofil gefahren.

8.8 Automatisches Reagieren auf externe Auslöser und Setzen von Ausgängen

Die PS 90+ besitzt 16 digitale Eingänge und Ausgänge. Die TTL-Eingänge und Ausgänge werden im Folgenden mit 1 bis 8, die SPS-Eingänge und Ausgänge mit 9 bis 16 durchnummeriert.

8.8.1 Automatisches Reagieren auf Eingänge

Ein automatisches Reagieren auf externe Auslöser (Trigger-Funktion) bedeutet, dass vordefinierte Aktionen mit einem Zustandswechsel eines Eingangs verknüpft werden können. Sobald dieser Zustandswechsel eintritt, wird die Aktion selbständig durch die Steuerung ausgeführt.

Definition des Eingangs
Jeder Eingang kann den logischen Zustand 0 oder 1 annehmen. Jeder Zustandswechsel wird als Flanke bezeichnet. Ein Wechsel von 0 auf 1 ist eine steigende, ein Wechsel von 1 auf 0 ist eine fallende Flanke. Somit gibt es insgesamt 32 Auslöser (16 Eingänge mit je zwei möglichen Flanken).

Mögliche Aktionen
Es gibt insgesamt acht Aktionen, die ausgelöst werden können. Jeder Aktion ist eine Aktions-ID zugewiesen. Außerdem kann jede Aktion mit bis zu zwei Parametern konfiguriert werden:

Aktion	Aktions-ID	Parameter	Verhalten
Geschwindigkeitsmodus starten	1	Achsen-Nr. (1…9)	VGO
Geschwindigkeitsmodus stoppen	2	Achsen-Nr. (1…9)	VSTP
Positioniervorgang starten	3	Achsennummer (1…9)	PGO
Mehrere Achsen im Geschwindigkeitsmodus starten	4	Bitmaske Bit 0 = Achse 1 Bit 8 = Achse 9	MVGO
Geschwindigkeitsmodus mehrerer Achsen stoppen	5	Bitmaske Bit 0 = Achse 1 Bit 8 = Achse 9	MSTOP
Achse stoppen	6	Achsennummer (1…9)	STOP
Achse anschalten	7	Achsennummer (1…9)	MON
Achse ausschalten	8	Achsennummer (1…9)	MOFF

Die Aktion beinhaltet bei allen Fahrbefehlen lediglich das Starten der betroffenen Achsen. Alle zu diesem Zeitpunkt gesetzten Parameter, insbesondere auch eine relative Distanz oder absolute Position besitzen Gültigkeit.

Falls ein Befehl wegen eines unzulässigen Zustands der Achse nicht ausgeführt werden kann, wird die Ausführung ignoriert. Bei dem nächsten Auftreten des Auslösers wird die Ausführung erneut versucht.

Zuweisung einer Aktion zu einem Auslöser

Es können Aktionen einem Auslöser zugewiesen und in einer Aktionstabelle gespeichert werden. Ein Tabelleneintrag beinhaltet die folgenden Informationen:

Eingang	Flanke	Aktions-ID	Parameter
1-16	0 (fallend) oder 1 (steigend)	1 bis 8	2 x 32-bit Signed decimal oder signed hex String mit Präfix 0x

Verhalten

Wenn ein Auslöser auftritt, wird die PS 90+ die zugeordneten Aktionen in kürzest möglicher Zeit ausführen. Die Aktionstabelle wird im Hintergrund sequentiell abgearbeitet.

Die Tabelle besitzt eine fixe Größe, daher kann der Anwender alle Einträge bearbeiten, löschen, abfragen und die Ausführungsreihenfolge festlegen.

Die Ausführung beginnt immer in Zeile 1. Wenn eine Bedingung erfüllt ist, wird die Aktion sofort ausgeführt und die nächste Zeile bearbeitet. Es wird nicht auf das Ende einer Aktion einer Zeile gewartet.

Durch die Programmierung der Tabelle muss durch den Anwender sichergestellt sein, dass kein widersprüchliches oder ungewolltes Verhalten auftritt. Es findet keine Prüfung statt.

Unter diesen Voraussetzungen ist es möglich, den gleichen Auslöser für mehrere Aktionen zu benutzen, das heißt dass ein Auslöser mehrfach in der Tabelle verwendet werden darf. Genauso ist es möglich, gleiche Aktionen mehrfach zu verwenden.

Die Tabelle besteht aus maximal 64 Einträgen.

Zur Erläuterung sollen die folgenden Beispiele dienen:

Starte und stoppe Achse 1 mit Eingang 14. Starte und stoppe Achse 2 mit Eingang 0. Starte Achse 3, wenn Achse 2 über Eingang 0 gestoppt wird.

! **Hinweis:** Es ist nicht definiert, ob erst Achse 2 stoppt oder Ache 3 startet. Beide Aktionen werden so schnell wie möglich gestartet.

Zeile	Eingang	Flanke	Aktions-ID	Parameter
1	14	1	1	1
2	14	0	2	1
3	0	1	1	2
4	0	0	2	3
5	0	0	1	3

Stoppe Achse 1 über Eingang 14 und starte Achse 1 über Eingang 3 und 12.

Zeile	Eingang	Flanke	Aktions-ID	Parameter
1	14	0	1	1
2	2	1	2	1
3	12	1	1	1

8.8.2 Automatisches Setzen von Ausgängen

Ein automatisches Setzen von Ausgängen (Event-Funktion) bedeutet, dass Ausgänge auf einen vordefinierten Zustand gesetzt werden, wenn bestimmte interne Ereignisse eintreten. Interne Ereignisse beziehen sich auf Achsenzustände.

Definition der Ereignisse

Es gibt sechs interne Ereignisse, die benutzt werden können. Jedem sind eine Ereignis-ID und gegebenenfalls mehrere Parameter zugeordnet.

Internes Ereignis	Ereignis-ID	Parameter	Anmerkung
Geschwindigkeit einer Achse ist größer als ein bestimmter Wert	1	Achsennummer (1…9) Geschwindigkeitswert: signed decimal	Aktuelle Geschwindigkeit > Wert
Geschwindigkeit einer Achse ist kleiner als ein bestimmter Wert	2	Achsennummer (1…9) Geschwindigkeitswert: signed decimal	Aktuelle Geschwindigkeit < Wert
Position einer Achse ist größer als ein bestimmter Wert	3	Achsennummer (1…9) Positionswert: signed decimal	Aktuelle Position > Wert
Position einer Achse ist kleiner als ein bestimmter Wert	4	Achsennummer (1…9) Positionswert: signed	Aktuelle Position < Wert
Achse wechselt in einen Bewegungszustand	5	Achsennummer (1…9)	Basierend auf Achsenzustand
Achse wechselt in einen ruhenden Zustand	6	Achsennummer (1…9)	Basierend auf Achsenzustand

Für die beiden letzten Einträge gilt:

Die Achsenzustände (siehe ?ASTAT) T, S, V, P, F, W, X, Y, C und N gelten als Bewegungszustand. Ein Wechsel von einem anderen in einen dieser Zustände wird als Wechsel in einen Bewegungszustand interpretiert. Ein Wechsel aus einem dieser Zustände in einen nicht aufgeführten wird als Wechsel in einen ruhenden Zustand interpretiert.

Zuordnung zu Ausgängen

Die Zuordnung zu Ausgängen erfolgt über eine Ereignistabelle. Ein Tabelleneintrag beinhaltet die folgenden Daten:

Ereignis-ID	2 Parameter	Ausgangsmaske	Ausgangswert
1-8	2 * 32 bit	16 bit	16 bit

Ein Ereignis beeinflusst nur die Ausgänge, die in der Ausgangsmaske mit 1 gewählt sind. Die Ausgänge der so ausgewählten Ausgänge werden auf die zugehörigen Ausgangswerte gesetzt.

Die Tabelle kann bis zu 64 Einträge enthalten.

Verhalten

Die Ausgänge sind nicht synchronisiert. Es gibt auch kein Clock-Signal. Dies heißt, dass bei einer gleichzeitigen Änderung mehrerer Ausgänge Übergangszustände entstehen können.

Die Liste wird zyklisch abgearbeitet. Falls ein Ereignis eintritt, werden die entsprechenden Ausgänge gesetzt. Es kann keine Aussage über die exakte Ausführungsreihenfolge getroffen werden.

Eine Ereignis-ID kann mehrfach verwendet werden. Das Verhalten soll an zwei Beispielen veranschaulicht werden:

Achse 2 startet und beschleunigt bis zu einer Geschwindigkeit von 20000 und wird dann nach einiger Zeit wieder gestoppt. Sobald die Geschwindigkeit 10000 überschritten ist, soll Ausgang 2 auf 0 gesetzt werden. Falls die Geschwindigkeit von einem höheren Wert als 9000 diesen unterschreitet, soll der Ausgang 2 auf 1 gesetzt werden. Der Ausgang ist zu Beginn der Sequenz nicht definiert.

Ereignis-ID	2 Parameter	Ausgangsmaske	Ausgangswert
1	Achse 2 10000	0x0002	0x0000
2	Achse 2 9000	0x0002	0x0002

Wenn Achse 1 in einen Bewegungszustand wechselt, soll Ausgang 15 auf 0 gesetzt werden. Wenn Achse 3 in einen Ruhezustand wechselt, soll Ausgang 15 auf 1 gesetzt werden. Wenn Achse 5 in einen Bewegungszustand wechselt, soll Ausgang 3 auf 1 gesetzt werden.

Ereignis-ID	2 Parameter	Ausgangsmaske	Ausgangswert
5	Achse 1	0x8000	0x0000
6	Achse 3	0x8000	0x0001
5	Achse 5	0x0004	0x0004

In diesem Beispiel wirken sowohl die erste als auch die dritte Zeile auf Ausgang 15. Falls beide Ereignisse gleichzeitig auftreten ist nicht definiert, welches zuerst bearbeitet wird. Somit ist auch der Wert des Ausgangs nicht eindeutig. Diese Konflikte können nur durch den Benutzer erkannt und verhindert werden.

8.8.3 Konfiguration

Jede Tabelle beinhaltet bis zu 64 Einträge.

Folgende Funktionen stehen über die Kommandoschnittstelle zur Verfügung:

- Die komplette Funktionalität „Reagieren auf externe Auslöser und Setzen von Ausgängen" kann über einen Befehl aktiviert oder deaktiviert werden. Da diese Funktionalität permanent Rechenleistung beansprucht, ist es ratsam, Sie nur bei tatsächlicher Benutzung zu aktivieren.
- Der Zustand aktiviert oder deaktiviert kann abgefragt werden.
- Parameter können in die beiden Tabellen geladen werden.
- Einzelne Zeilen der Tabellen können ausgelesen werden.
- Einzelne Zeilen der Tabelle können gelöscht werden.
- Befehle mit Zeilennummern größer als der zulässige Bereich werden ignoriert.
- Falls beim Eintragen von Zeilen in die Aktionstabelle ein Fehler auftritt, weil zum Beispiel ein unzulässiger Parameter gesetzt wird, so wird die Aktions-ID auf 0 gesetzt.
- Leere Zeilen in der Aktionstabelle sind durch die ID 0 gekennzeichnet.
- Falls beim Eintragen von Zeilen in die Ereignistabelle ein Fehler auftritt, weil zum Beispiel ein unzulässiger Parameter gesetzt wird, so wird die Ereignis-ID auf 0 gesetzt.
- Leere Zeilen in der Ereignistabelle sind durch die ID 0 gekennzeichnet.

9. Wegerfassung

Encoder

Der Encoder ist ein auch als Drehgeber bezeichnetes Wegerfassungssystem zur Positionsrückmeldung, das für den Motorcontroller im geregelten (Closed-Loop) Betrieb genutzt wird.

Ohne Encoder ist nur der gesteuerte Betrieb (Open Loop) mit Schrittmotoren möglich. Um BLDC- oder DC-Motoren betreiben zu können, muss ein Wegerfassungssystem angeschlossen sein. Dies kann ein Encoder sein. Üblicherweise besitzen sie 500, 1250 oder 2500 Linien pro Umdrehung. Über den Encoder erfasst der Motion-Controller die aktuelle Position der Achse und berechnet aus der zeitlichen Veränderung der Positionswerte die aktuelle Geschwindigkeit des Rotors.

Encoder sind fest am Motor angeflanscht und direkt mit dem Rotor verbunden. Die Signale des Encoders sind Kanal A und B (CHA und CHB), 90 Grad versetzt (sog. Quadratur-Signale), und ggf. ein Index-Impuls pro Umdrehung. Die PS 90+ kann als Encodersignale TTL-Pegel oder antivalente Signale (über Leitungstreiber) verarbeiten. Die Signale werden nach einer Pegelumwandlung und Filterung direkt an den Motion-Controller weitergegeben.

Linearmesssystem

Ein Messsystem, welches direkt an die Bewegung des Aktors gekoppelt ist, nennt man Linearmesssystem. Das Wegmesssystem kann entweder alternativ zum Encoder der Wegerfassung dienen oder zusätzlich zu einem vorhandenen Encoder zum Nachführen des Positioniersystems auf die Zielposition verwendet werden. Dieses Verfahren nennt sich Nachlaufregelung. Hierbei ist dann Korrektur systematischer Fehler (z.B. Spindelsteigungsfehler) möglich.

Die Zielposition wird bei Verwendung eines Wegmesssystems zur Nachlaufregelung separat (ebenfalls mit einer Auflösung von 32 Bit) angegeben. Der eigentliche Positioniervorgang wird dann vom Motion-Controller über Encoder durchgeführt. Meldet dieser „Position erreicht", dann führt der Hauptprozessor die Position solange nach, bis die vom Messsystem erfaßte exakte Zielposition innerhalb des definierten Zielfensters liegt.

Die Signale des Wegmesssystems entsprechen den vorher genannten Encodersignalen (Quadratur A und B, sowie Index). Die maximale Zählfrequenz beträgt 5 MHz (Signal) bzw. 20 MHz (Quadratur).

Funktionsweise der Nachlaufregelung

Um eine Nachlaufregelung für eine bestimmte Positioniereinheit realisieren zu können, ist es erforderlich, die Positioniereinheit mit einem zusätzlichen inkrementalen Linearmesssystem auszustatten, welches die reale Absolutposition des Schlittens unter Zuhilfenahme einer eindeutigen Referenzmarke erfasst. Die aus Motor und Antriebsspindel bestehende Antriebseinheit (nachfolgend als "Aktor" bezeichnet) wird über die Steuerung auf die reale Absolutposition nachgeführt (nachgeregelt). Dies kann durch iterative Korrekturbewegungen oder Korrekturfahrt mit konstanter Geschwindigkeit erfolgen. Eine Kombination beider Verfahren ist ebenfalls möglich. Die Auswahl wird über die Betriebsartenvorwahl der Nachlaufregelung vorgenommen. Die Werte für die rechnerische Auflösung von Linearmesssystem und Positioniereinheit sind in der Regel unterschiedlich.

Vor Verwendung der Nachlaufregelung ist eine Referenzierung in Referenzfahrmodus 6 oder 7 durchzuführen. Hierbei wird der insgesamt zur Verfügung stehende Hub in Inkrementen des Linearmesssystems gemessen und der Absolutpositionszähler automatisch bei Überfahren der Referenzmarke des Linearmesssystems auf Null gesetzt.

Die Zielposition einer nachlaufgeregelten Positioniereinheit wird über die nach erfolgreicher Referenzfahrt definierte Absolutposition des Linearmesssystems angegeben, d.h. eine Zielposition wird als Absolut- oder Relativdistanz angegeben, bezogen auf ein ganzzahliges Vielfaches des Weginkrements des Linearmesssystems, den Referenzpunkt und ggf. die aktuelle Position.

Zur steuerungsinternen Berechnung der Wegstrecke des Aktors wird das Verhältnis zwischen Weginkrement des Aktors und Weginkrement des Linearmesssystems über einen Umrechnungsfaktor F := Z / N definiert, der sich aus dem Quotienten beider Auflösungswerte ergibt.

Ein Positioniervorgang mit Nachlaufregelung entspricht folgendem 3-Phasen-Schema:

- Mittels des gegebenen Umrechnungsfaktors (Z / N) wird aus den gegebenen Positionsdaten die zu verfahrende Relativdistanz des Aktors berechnet.
- Die so berechnete Distanz wird verfahren (Phase 1, Grobpositionierung) und die Abweichung zur Sollposition berechnet.
- Liegt die Istposition außerhalb des definierten Zielfensters, erfolgt, falls gewünscht, eine iterative Annäherung, d.h. es wird zyklisch eine Relativdistanz des Aktors berechnet und an den Motor ausgegeben usw. (Phase 2, Iteration).
- Hierbei gilt als Konvergenzkriterium, dass sich der Betrag der Lageabweichung bei jedem Iterationsschritt verringern muss, bis die Istposition schließlich innerhalb des Zielfensters liegt. Daraus folgt als Divergenzkriterium für die Iteration, dass der Abbruch der Iteration dann erfolgt, wenn der Betrag der Lageabweichung nach Korrekturfahrt (n) größer oder gleich dem Betrag der Lageabweichung nach Korrekturfahrt (n-1) ist.
- Nach erfolgreichem Abschluß (Konvergenz, Istposition liegt innerhalb Zielfenster) oder Abbruch (Divergenz) der Iteration folgt optional eine Korrekturphase im Geschwindigkeitsmodus (Phase 3). Ob Phase 3 aktiv ist oder nicht, ist wählbar, d.h. sie wird über einen Parameter vorgegeben.
- In der anschließenden Korrekturphase wird die Istposition des Linearmesssystems abgefragt. Liegt die Istposition außerhalb des Zielfensters, wird der Geschwindigkeitsmodus mit der vorher definierten Nachlaufgeschwindigkeit als Parameter aufgerufen. Sobald die Istposition innerhalb des Zielfensters liegt, stoppt der Nachführvorgang, d.h. es wird eine Bremsrampe ausgelöst. Fährt der Aktor über das Ziel hinaus, erfolgt eine Drehrichtungsumkehr usw.
- Über einen weiteren Parameter kann vorgegeben werden, ob die Nachführung im Geschwindigkeitsmodus ständig aktiv sein soll oder beim ersten Erreichen des Zielfensters abschaltet.

Berechnung des Umrechnungsfaktors F:

Bei nachlaufgeregeltem Betrieb werden Fahrdistanzen grundsätzlich in Vielfachen der Messsystemauflösung (Weginkrement des Linearmesssystems) angegeben. Die Auflösung des Aktors ist bestimmt durch die Motorauflösung (z.B. Mikroschrittfaktor, Encoderinkrement) und die mechanischen Parameter (z.B. Spindelsteigung).

Aus der gegebenen Fahrdistanz muss die zurückzulegende Relativdistanz des Aktors vor jeder Fahrt berechnet werden.

Nachfolgend soll die Berechnung beispielhaft für einen Lineartisch mit Spindel-Direktantrieb und 2-Phasen-Schrittmotor (ungeregelt) durchgeführt werden.

$$F = \frac{Z}{N} = \frac{r_s}{r_m} = \frac{\text{Auflösung des Aktors}}{\text{Auflösung des Messsystems}}$$

Berechnung von r_s:

$$r_s = \frac{h}{n \cdot m}$$

wobei:

h = Spindelsteigung (Verstellweg pro Motorumdrehung),
n = Motorschrittzahl (Vollschritte pro Motorumdrehung),
m = Mikroschrittfaktor (Mikroschritte pro Vollschritt)

Beispiel:

h = 5 mm,
n = 200,
m = 50

$$r_s = \frac{5\,\text{mm}}{200 \cdot 50} = 0{,}5\,\mu\text{m}$$

Die Auflösung des Messsystems r_m ist gegeben, z.B.:

$r_m = 0{,}1\ \mu\text{m}$

Somit ist im Beispiel

$$F = \frac{r_s}{r_m} = \frac{0{,}5\,\mu\text{m}}{0{,}1\,\mu\text{m}} = \frac{5}{1} =: \frac{Z}{N}$$

und damit:

$Z = 5$

$N = 1$.

10. PID-Regelschleifenalgorithmus

Das in der PS 90+ benutzte Servofilter arbeitet nach einem PID-Algorithmus. Ein Integrationslimit sichert nach oben gegen einen akkumulierten Fehler ab.

Die PID-Formel lautet wie folgt:

$$Output_n = K_p E_n + K_d(E_n - E_{(n-1)}) + \sum_{j=0}^{n} E_j \frac{K_i}{256}$$

Hierbei ist:

E_n Regelabweichung zum diskreten Zeitpunkt n
K_i Integralanteil des Lagereglers
K_d Differentialanteil des Lagereglers
K_P Proportionalanteil des Lagereglers

Alle Filterparameter und die Drehmomentsignalbegrenzung sind programmierbar, so dass der Filter durch den Anwender fein abgestimmt werden kann. Wertebereiche und Formate werden in der folgenden Tabelle aufgelistet:

Terminus	Name	Bereich
I_{lim}	Begrenzung Integralanteil	32 bit unsigned (0…2.124.483.647)
K_i	Integralanteil des Lagereglers	16 bit unsigned (0…32.767)
K_d	Differentialanteil des Lagereglers	16 bit unsigned (0…32.767)
K_p	Proportionalanteil des Lagereglers	16 bit unsigned (0…32.767)

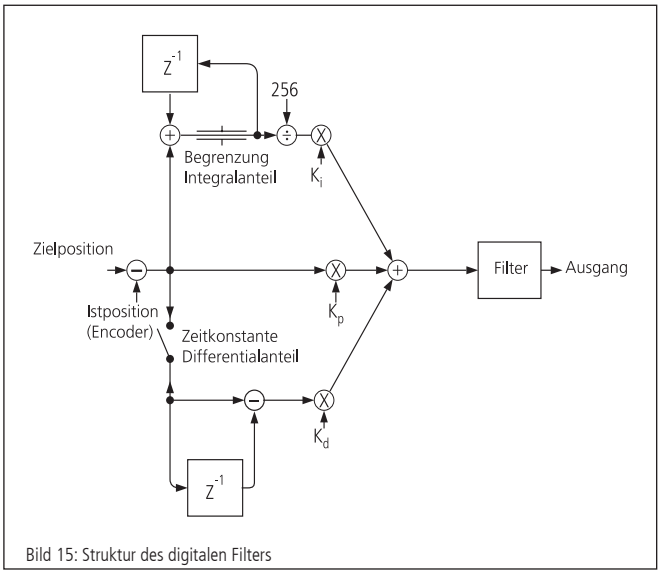

Bild 15: Struktur des digitalen Filters

11. Positioniergeschwindigkeit und -beschleunigung, Berechnung

11.1 2-Phasen-Schrittmotor (Open Loop)

Allgemeines

Jede schrittmotorgetriebene Mechanik besitzt eine insbesondere von Motortyp, Systemreibung und Last abhängige sog. Start-Stop-Frequenz. Die Start-Stop-Frequenz bezeichnet die maximale Fahrfrequenz des betreffenden Schrittmotors, mit welcher dieser noch aus dem Stillstand ohne Beschleunigungsphase loslaufen kann. Es ist üblich, diese und andere Kennfrequenzen von Schrittmotoren in Hertz Vollschritt („HzVS"), d.h. Vollschritte pro Sekunde, anzugeben. Die Welle eines Schrittmotors mit Schrittwinkel 1,8°, d.h. R = 200 Vollschritte pro Motorumdrehung, der z.B. mit 400 HzVS läuft, dreht mit einer Geschwindigkeit von zwei Umdrehungen pro Sekunde oder 120 Umdrehungen pro Minute.

Um höhere Geschwindigkeiten als die Start-Stop-Frequenz zu erreichen, muss der Schrittmotor über diese Frequenz hinaus mittels geeigneter Beschleunigungsrampe beschleunigt, bzw. unter diese Frequenz mittels geeigneter Bremsrampe abgebremst werden. Diese Beschleunigung bzw. Bremsung erfolgt mittels trapezförmigem oder S-förmigem Geschwindigkeit-Zeit-Profil. Gegebenenfalls ist eine Dämpfung (Viskosedämpfer, am zweiten Wellenende des Motors montiert) erforderlich, um überhaupt höhere Drehzahlen erreichen zu können.

Fast alle Standard-Schrittmotoren, die bei OWIS® eingesetzt werden, sind in der Lage, einer Frequenz von 400 HzVS im Start-Stop-Betrieb zu folgen.

Die PS 90+ besitzt einen digitalen Profilgenerator. Die Geschwindigkeitsprofile werden periodisch berechnet und an den 2-Phasen-Schrittmotor ausgegeben.

Periodendauer

Die Periodendauer des digitalen Profilgenerators ist durch die Hardware festgelegt.

$T_P = 256\,\mu s$

Endgeschwindigkeit

Die Positionierung der Achsen wird im Punkt-zu-Punkt-Verfahren vorgenommen. Hierbei beschleunigt jede Achse wahlweise mit trapezförmigem oder S-förmigem Geschwindigkeits-Profil.

Die Endgeschwindigkeit V nach der Beschleunigungsrampe wird als 32-Bit-Wort angegeben. Ihr Wertebereich reicht von 1 bis 2147483647.

> **! Hinweis:**
> Keinesfalls darf eine höhere Geschwindigkeit vorgegeben werden, als die Mechanik in der Lage ist, zu fahren, da sonst die angeschlossene Mechanik beschädigt oder zerstört werden kann.

Bei gegebener Geschwindigkeit V und gegebenem Mikroschrittfaktor Mcstp errechnet sich die Schrittfrequenz f wie folgt:

$$f_{Mcstp} = \frac{1}{T_P} \cdot \frac{V}{65536}$$ (Schrittfrequenz im Mikroschrittmodus)

bzw.

$$f_{VS} = \frac{1}{Mcstp \cdot T_P} \cdot \frac{V}{65536}$$ (auf Vollschrittmodus normierte Schrittfrequenz)

Hieraus ergibt sich die Motordrehzahl n_{RPM} (ohne Berücksichtigung eines evtl. vorhandenen Getriebes) bei einem Schrittmotor mit R Vollschritten pro Motorumdrehung:

$$n_{RPM} = \frac{60}{min} \cdot \frac{1}{Mcstp \cdot R \cdot T_P} \cdot \frac{V}{65536} \quad \text{(Umdrehungen/Minute)}$$

bzw.

$$n_{RPS} = \frac{1}{s} \cdot \frac{1}{Mcstp \cdot R \cdot T_P} \cdot \frac{V}{65536} \quad \text{(Umdrehungen/Sekunde)}$$

Für die Umrechnung von der Motordrehzahl in eine Positioniergeschwindigkeit der Mechanik sind zusätzlich die mechanischen Daten, wie z.B. Spindelsteigung und ggf. die Getriebeübersetzung, zu berücksichtigen.

Beschleunigung bei Trapezprofil
Als Beschleunigung („ACC") ist ein 32-Bit-Wort anzugeben, der Wertebereich reicht von 1 bis 2147483647.

Dauer der Trapezprofil-Beschleunigungsrampe bei gegebener Geschwindigkeit V und Beschleunigung ACC:

$$\Delta t = 1\,s \cdot \frac{V \cdot T_P}{ACC} \quad \text{(Anlauf-/Nachlaufdauer in Sekunden)}$$

Zurückgelegte Distanz während der Trapezprofil-Beschleunigungsrampe:

$$\Delta s = 1\,\text{Mikroschritt} \cdot \frac{V^2}{131072 \cdot ACC} \quad \text{(Nachlaufweg in Mikroschritten)}$$

11.2 DC-Servomotor und 2-Phasen-Schrittmotor (Closed-Loop)

Allgemeines
Die PS 90+ hat einen digitalen Lage-/Geschwindigkeits-Regler. Stell- und Regelgröße werden periodisch berechnet. Die Erfassung des Positions-Istwertes geschieht im einfachsten Fall mittels eines Drehgebers (auch „Encoder" genannt), der am 2. Wellenende des Motors angeflanscht ist. Wichtigste Kenngröße des Encoders ist die Encoder-Strichzahl R. Sie gibt die Anzahl der sog. Linien, d.h. Hell-Dunkel-Perioden je Motorwellenumdrehung, an. Die Signale durchlaufen eine Vierfach-Auswertung, woraus sich generell eine 4-fach höhere Auflösung als die Encoder-Strichzahl ergibt.

Abtastzeit
Die Periodendauer des digitalen Reglers wird auch als Abtastzeit bezeichnet und ist durch die Hardware festgelegt. Die minimale Abtastzeit beträgt 51,2 µs. Sie kann bei Bedarf um ganzzahlige Vielfache von 51,2 µs erhöht werden:

$T_S = 51{,}2\,\mu s + n \cdot 51{,}2\,\mu s;\ n \in [0, 1, \ldots, 386]$

entsprechend einer Abtastzeit von

$T_S = [51{,}2\,\mu s, 102{,}4\,\mu s, 153{,}6\,\mu s, 204{,}8\,\mu s, 256\,\mu s, \ldots, 19986\,\mu s]$

Als Abtastzeit können nur ganzzahlige Werte an die PS 90+ übergeben werden. Der Wert wird intern auf den nächsten gültigen Wert gerundet.

Standardwert (Voreinstellung): $T_S = 256\,\mu s$

Endgeschwindigkeit
Die Positionierung der Achsen wird im Punkt-zu-Punkt-Verfahren vorgenommen. Hierbei beschleunigt jede Achse wahlweise mit trapezförmigem oder S-förmigem Geschwindigkeits-Profil.

Die Endgeschwindigkeit V nach der Beschleunigungsrampe wird als 32-Bit-Wort angegeben. Ihr Wertebereich reicht von 1 bis 2147483647.

> **! Hinweis:**
> Keinesfalls darf eine höhere Geschwindigkeit vorgegeben werden, als die Mechanik in der Lage ist, zu fahren, da sonst die angeschlossene Mechanik beschädigt oder zerstört werden kann.

Bei gegebener Geschwindigkeit V und der Encoder-Linienzahl R errechnet sich die Motordrehzahl (ohne Berücksichtigung eines evtl. vorhandenen Getriebes) wie folgt:

$$n = \frac{60}{min} \cdot \frac{1}{T_S} \cdot \frac{1}{4R} \cdot \frac{V}{65536} \quad \text{(Umdrehungen pro Minute)}$$

bzw.

$$n = \frac{1}{s} \cdot \frac{1}{T_S} \cdot \frac{1}{4R} \cdot \frac{V}{65536} \quad \text{(Umdrehungen pro Sekunde)}$$

bzw.

$$n = \frac{1\,\text{Inkrement}}{s} \cdot \frac{1}{T_S} \cdot \frac{V}{65536} \quad \text{(Inkremente pro Sekunde)}$$

Die letzte Formel kann auch wie folgt verstanden werden:

Der Controller verfährt V / 65536 Inkremente je Abtastintervall T_S.

Für die Umrechnung von der Motordrehzahl in eine Positioniergeschwindigkeit der Mechanik sind zusätzlich die mechanischen Daten, wie z.B. Spindelsteigung und ggf. die Getriebeübersetzung, zu berücksichtigen.

Beispiel:

Es ist eine Positionierung mit einer Nenndrehzahl n = 1800 U/min auszuführen. Es wird ein Encoder mit R = 500 Linien (entspr. 2000 Impulsen/Umdrehung) am Motor eingesetzt.

Wie ist V zu wählen?

Lösung:

Es ergibt sich allgemein nach Umstellen der Drehzahlgleichung für die Geschwindigkeit:

$$V = \frac{n}{60} \cdot 4 \cdot R \cdot 65536 \cdot T_S$$

Damit wird V = 1006633 für n = 1800 U/min bei Einsatz eines 500-Linien-Encoders. Unter Verwendung einer direktgetriebenen Spindel mit 1 mm Steigung entspricht dies einer Verstellgeschwindigkeit von genau 1,8 m/min. bzw. 30 mm/s.

Beschleunigung bei Trapezprofil
Als Beschleunigung („ACC") ist ein 32-Bit-Wort anzugeben, der Wertebereich reicht von 1 bis 2147483647.

Dauer der Trapezprofil-Beschleunigungsrampe bei gegebener Geschwindigkeit V und Beschleunigung ACC:

$$\Delta t = 1\,s \cdot \frac{V \cdot T_S}{ACC} \quad \text{(Anlauf-/Nachlaufdauer in Sekunden)}$$

Zurückgelegte Distanz während der Trapezprofil-Beschleunigungsrampe:

$$\Delta s = 1\,\text{Inkrement} \cdot \frac{V^2}{131072 \cdot ACC} \quad \text{(Nachlaufweg in Inkrementen)}$$

12. Nano-Hybrid-Ansteuerung

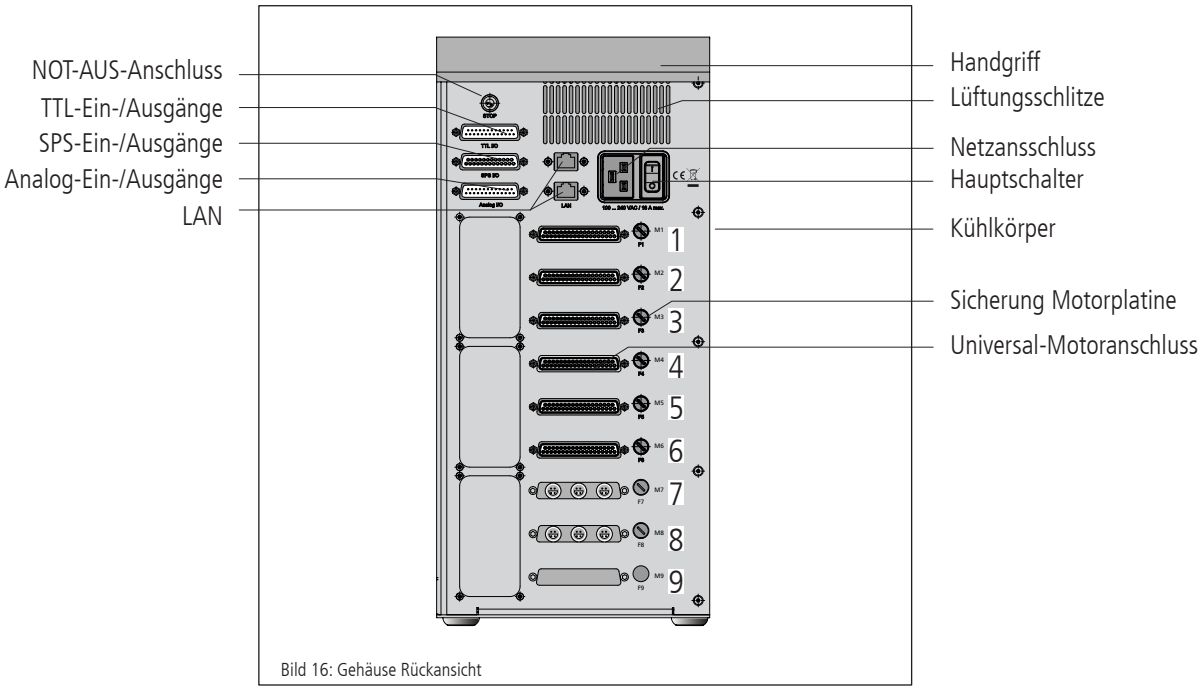

Bild 16: Gehäuse Rückansicht

Allgemeines
Um OWIS® Positioniereinheiten mit Nano-Hybrid-Technik ansteuern zu können, ist eine entsprechend ausgestattete PS 90 notwendig. In diesem Kapitel werden die Unterschiede und Besonderheiten dieser Ausstattungsvariante sowie die Betriebsmodi erklärt. Alle allgemeinen Eigenschaften, insbesondere die Bedien- und Sicherheitshinweise gelten uneingeschränkt. Vor der Benutzung der PS 90 zur Ansteuerung von OWIS® Nano-Hybrid-Positioniereinheiten ist die Kenntnis der vorherigen Kapitel notwendig.

Technische Übersicht und Aufbau der Steuerung
Nano-Hybrid Positioniereinheiten von OWIS® besitzen einen hybriden Antrieb. Die Grobpositionierung erfolgt über einen hochauflösenden Schrittmotor. Die Feinpositionierung erfolgt durch einen Piezoaktor.

Eine PS 90, die zur Ansteuerung dieser Nano-Hybrid-Positioniereinheiten ausgerüstet ist, kann nur noch für sechs ausgerüstet sein. Statt der Achsen 7-9 ist die Ansteuerelektronik für die Piezoaktoren vorhanden.

Die Positioniereinheiten werden über den Universal-Motorstecker angeschlossen. Zusätzlich wird der Piezozweig über ein weiteres Kabel verbunden. Die Anschlüsse dafür befinden sich für die ersten drei Nano-Hybrid-Positioniereinheiten auf Position 7. Falls vier bis sechs solcher Einheiten angesteuert werden sollen, ist auch Position 8 belegt. Position 9 ist immer unbelegt.

Sicherheit
Die Ansteuerung der Piezozweige erfolgt mit einer Spannung im Bereich zwischen -71 V und +71 V und kann schwerste Verletzungen verursachen. Die Benutzung darf nur durch Personal erfolgen, welches im Umgang mit solchen Spannungen unterwiesen ist. Die allgemeinen Unfallverhütungsvorschriften müssen beachtet werden.

Steuerungsarchitektur und Funktion
Eine Nano-Hybrid-PS 90 besteht im Wesentlichen aus folgenden Komponenten:

1. Ein eingebautes Netzteil
2. Eine Hauptplatine
3. Maximal zwei Antriebsplatinen
4. Maximal sechs Motorplatinen (Endstufen)
5. Maximal zwei Wegmessplatinen
6. Eine D/A-Wandlerkarte für die Piezoaktoren
7. Maximal zwei Steuermodule für die Piezoaktoren

Anstelle der dritten Antriebsplatine wird eine D/A-Wandlerkarte für die Piezoaktoren eingesetzt. Diese kann drei oder sechs Achsen ansteuern. Besteht nur der erste Antriebsverband aus Nano-Hybrid-Positioniereinheiten, so wird zusätzlich ein Steuermodul für Piezoaktoren benötigt. Sind beide Antriebsverbände in Nano-Hybrid-Technik, so werden davon zwei benötigt. Das Steuermodul für den Piezozweig besitzt eine Schutzeinrichtung, die prüft, ob auch eine entsprechende OWIS®-Nano-Hybrid-Einheit angeschlossen ist. Die grundsätzliche Ansteuerung der Schrittmotoren bleibt unverändert.

Anschluss
Der Anschluss erfolgt über zwei Kabel. Zum einen wird ein Motorkabel an den 37-poligen D-Sub-Stecker der Steuerung angeschlossen, welches am Tisch an der 18-poligen Lemo-Buchse befestigt wird. Zum anderen wird das Piezo-Kabel sowohl am Tisch als auch an der Steuerung am 4-poligen Lemo-Anschluss befestigt.

Positionierung im Nano-Hybrid-Betrieb
Zur Positionierung mit einer Nano-Hybrid-Positioniereinheit stehen drei Möglichkeiten zur Verfügung. Normale Positionierungen können wie mit jeder Einheit mit Schrittmotor erfolgen. Da jede Nano-Hybrid-Positioniereinheit mit einem integrierten Messsystem ausgestattet ist, kann auch die Nachlaufregelung (Modi 0 bis 5) als Betriebsart gewählt werden.

Um die Möglichkeiten der hochgenauen Positionierung mit Hilfe des Piezo-Antriebs nutzen zu können, sind spezielle Modi (6 bis 9) der Nachlaufregelung vorhanden, die im Folgenden erklärt werden:

Allgemeine Beschreibung der Nachlaufregelung für Piezo-Antriebe

Bei der Nachlaufregelung erfolgt die Positionierung in mehreren, aufeinander folgenden Schritten. Der erste Schritt ist dabei immer eine Grobpositionierung über den Motor. Anschließend erfolgt eine Korrektur des Positionierfehlers durch den Schrittmotor und dann durch den Piezo-Antrieb. Folgende Werte sind für die Positionierung mit Hilfe des Piezo-Antriebs relevant:

- PWMSSET

Dieser Wert gibt die Sollposition für den zu startenden Positioniervorgang an. Im absoluten Positioniermodus wird der Wert als Absolutposition betrachtet. Im relativen Positioniermodus entscheidet das Vorzeichen die Fahrtrichtung.

- PWMSPWIN

Dieser Wert beschreibt ein zulässiges Zielfenster, welches vor und hinter der Zielposition PWMSSET liegt. Er wird in Inkrementen des Messsystems angegeben. Die Positionierung wird erfolgreich beendet, sobald die Istposition zwischen PWMSSET-PWMSPWIN und PWMSSET+PWMSPWIN liegt. PWMSPWIN muss immer positiv sein und sollte im Bereich zwischen 2 und 10 Inkremente liegen. Dieser Wert kann die Geschwindigkeit, mit der eine Positionierung erfolgt, beeinflussen. Wird er zu groß gewählt, so ist die Positionier- und Wiederholgenauigkeit zu grob. Wird er zu klein gewählt, so kann die Positionierung möglicherweise nicht zu einer stabilen Endposition führen.

- WMSOFFS

Der Schrittmotor muss den Schieber mit einem Versatz zur Sollposition positionieren, damit die Sollposition durch den Piezo-Antrieb angefahren werden kann. Dieser Versatz wird durch WMSOFFS in Messsysteminkrementen angegeben. WMSOFFS muss immer negativ sein und sollte im Bereich zwischen -20 und -100 liegen. Wird der Wert unpassend gewählt, so muss möglichweise eine zusätzliche Korrektur durch den Motor erfolgen und der Positioniervorgang dauert länger.

- PWMSWIN

Dieser Wert beschreibt ein zulässiges Zielfenster, welches vor und hinter dem Ziels der Positionierung mit dem Schrittmotor liegt. Er wird in Inkrementen des Messsystems angegeben. Die Positionierung mit dem Schrittmotor wird erfolgreich beendet, wenn zu diesem Zeitpunkt die Istposition in einem Bereich zwischen (PWMSSET+WMSOFFS)-PWMSWIN und (PWMSSET+WMSOFFS)+PWMSWIN liegt. PWMSWIN muss immer positiv sein und sollte im Bereich zwischen 10 und 50 Inkrementen liegen. Dieser Wert kann die Dauer der Positionierung beeinflussen. Ist er ungünstig gewählt, so muss gegebenenfalls eine zusätzliche Korrektur durch den Schrittmotor oder den Piezo-Antrieb erfolgen.

- WMSVEL

Dieser Wert legt die Geschwindigkeit fest, mit der in den Modi 7 und 9 die Korrekturfahrt in Phase 2 durchgeführt wird. WMSVEL muss immer positiv sein.

Es wird dabei je nach Modus das folgende Schema abgearbeitet:

- Mittels des gegebenen Umrechnungsfaktors (Z/N) wird aus den gegebenen Positionsdaten die zu verfahrende Relativdistanz des Aktors berechnet.
- Die so berechnete Distanz wird verfahren (Phase 1) und die Abweichung zur Sollposition berechnet.
- Liegt die Istposition außerhalb des Fensters (PWMSSET+WMSOFFS)+/-PWMSWIN, so wird eine Korrekturphase (Phase 2) gestartet. Liegt die Istposition innerhalb des Fensters, so wird die Positionierung mit dem Piezo-Antrieb gestartet (Phase 3).
- Falls Phase 2 notwendig ist, so hängt die Art und Weise der Korrektur vom gewählten Modus ab. In den Modi 6 und 7 wird eine erneute Grobpositionierung (Phase 1) durchgeführt. Das heißt, dass von der aktuellen Position ausgehend eine neue Zielposition berechnet wird, die anschließend vom Schrittmotor mit den eingestellten Werten durchgeführt wird. Phase 2 wird beendet, falls am Ende die Istposition innerhalb des Fensters liegt oder falls sich der Fehler im Vergleich zum vorherigen Schritt vergrößert hat. In den Modi 8 und 9 wird die Korrektur durch eine Bewegung im Geschwindigkeitsmodus durchgeführt. Die Geschwindigkeitsfahrt wird beendet, sobald das Fenster erreicht ist. Sollte das Fenster überfahren werden, so wird eine erneute Fahrt im Geschwindigkeitsmodus in die andere Richtung gestartet.
- Nachdem die Positionierung mit dem Schrittmotor abgeschlossen ist, wird die Feinpositionierung mit dem Piezo-Antrieb gestartet. Die Positionierung endet, sobald die Istposition im Bereich PWMSSET+/-PWMSPWIN liegt. Sollte der Piezo-Antrieb das Zielfenster nicht erreichen, wird in Phase 2 gesprungen und eine erneute Korrektur mit dem Schrittmotor gestartet.
- In den Modi 7 und 9 bleibt die Phase 3 auch nach erfolgreicher Positionierung aktiv. Das heißt, dass Änderungen der Istposition zum Beispiel aufgrund von externen Kräften auf den Schieber fortlaufend durch den Piezo-Antrieb korrigiert werden. Sollte der Piezo-Antrieb alleine nicht in der Lage sein, das Zielfenster zu erreichen, wird automatisch Phase 2 gestartet.

13. Inbetriebnahme der PS 90+

13.1 Vorbereitung der Steuerung

Aufstellung
Die Steuerung ist für den Einsatz in Forschung, Entwicklung sowie für industrielle Anwendungen konzipiert. Sie darf nur in trockener, staubarmer Umgebung betrieben werden. Grundsätzlich wird sie freistehend betrieben.

Zur internen Kühlung sind auf der Gehäusevorder- und Rückseite im oberen Bereich Lüftungsschlitze angebracht. Die Abwärme der Motorplatinen (Endstufen) wird über den seitlich angebrachten Kühlkörper an die Außenluft abgegeben. Die Steuerung darf nicht in ein Gehäuse oder einen Schrank ohne ausreichende Luftzirkulation eingebaut werden.

Hinweis:
Wärmestau in der Steuerung oder am Kühlkörper ist zu vermeiden. Es soll ein Mindestabstand von 15 cm zu geschlossenen Flächen und Wänden eingehalten werden.

NOT-AUS-Funktion

An der Geräterückseite ist ein Anschluss für einen externen NOT-AUS-Taster vorgesehen, an welchem standardmäßig ein Kurzschlussstecker eingesteckt ist. Soll ein NOT-AUS-Taster angeschlossen werden, ist der Kurzschlussstecker zu entfernen.

Hinweis:
Wird der Kurzschlussstecker entfernt und kein NOT-AUS-Taster angeschlossen, ist die Funktion der Motorplatinen (Motorendstufen) blockiert.

13.2 Anschluss der Peripherie und Geräte

Vor dem Einschalten der Steuerung müssen sämtliche Anschlussstecker für Geräte und Peripherie angeschlossen sein, damit sie von der Steuerung erkannt und initialisiert werden.

Zunächst sind die Positioniereinheiten an die entsprechenden Achsen anzuschließen (siehe Abnahmeprotokoll). Es müssen:

- die Positioniereinheit
- die Stromversorgung
- der Computer

angeschlossen werden.

Die Verbindung zum Computer erfolgt über die USB, RS-232- oder Ethernet- Schnittstelle.
Mit dem optionalen Anybus®-Modul „Modbus/TCP" ist die Kommunikation mit einem PC über Ethernet möglich.

Für USB-Schnittstelle ist eine Treiberinstallation notwendig.
Der Treiber befindet sich auf der mitgelieferten CD.
Für die Installation starten Sie bitte „setup. exe".

Hinweis:
Jegliche Geräte und Peripherie müssen vor dem Systemstart angeschlossen sein, da sie sonst von der Steuerung nicht initialisiert und somit erkannt werden.

13.3 Systemstart

Durch das Betätigen des Hauptschalters wird die Steuerung aktiviert. Der Mikrocontroller startet alle vorhandenen Programme und Parameter und initialisiert sich und seine Peripherie. Der Initialisierungsvorgang dauert ca. 10 Sekunden. Danach ist die Steuerung bereit, Kommandos
vom PC zu empfangen und zu bearbeiten. Beim ersten Windows-Start mit angeschlossener PS 90+ sollte das Betriebssystem die neue Hardware erkennen. Die Treiber können nun installiert werden. Hierzu sind ggf. Administratorrechte erforderlich.

Initialisierung
Nachdem die Stromversorgung eingeschaltet und das Gerät aktiviert wurde, muss jede Achse, die verwendet werden soll, zunächst per INIT-Befehl initialisiert werden.

Achsenparameter, die verändert wurden, werden ebenfalls mit der Initialisierung übernommen.

Software
Für die Inbetriebnahme gehören zum Lieferumfang der Steuerung das Softwaretool OWISoft, der USB-Treiber und die Software-Schnittstelle (SDK/API) für C, C++, C#, LabView (ab V 8.2) und zusätzliche Programmiersprachen (32/64-Bit). Damit kann die PS 90+ komfortabel konfiguriert und betrieben werden.

Unterstützte Betriebssysteme: Windows XP, Windows Vista (32/64-Bit), Windows 7 (32/64-Bit), Windows 8 (32/64-Bit), Windows 8.1 (32/64-Bit) und Windows 10 (32/64-Bit).

Die Software-Schnittstelle enthält Beispielprogramme mit dem Quellcode und Hilfedateien.

Für die Inbetriebnahme mit OWISoft sind die jeweiligen Parameter der Positionierer für die Achsen hinterlegt, die nur noch angewählt werden müssen.

Hinweis:
Die hinterlegten Parameter sind für unbelastete Positionierer voreingestellt. Für optimalen Lauf müssen die Reglerparameter der konkreten Belastungen angepasst werden.

Lesen Sie hierfür bitte die Bedienungsanleitung OWISoft.

Für die Inbetriebnahme mittels eigener Applikationssoftware lesen Sie bitte das Kapitel „Hinweise zum Aufbau einer eigenen Applikationssoftware". Dort ist im Anschluss auch eine Tabelle mit den Befehlssätzen der PS 90+ angefügt.

14. Fehlerüberwachung

14.1 Endschalter

Die PS 90+ besitzt vier Endschaltereingänge, davon jeweils zwei Eingänge für Limit-Schalter (MINSTOP, MAXSTOP) und zwei für Bremsschalter (MINDEC, MAXDEC), sowie Auswertemöglichkeit für einen Referenzschalter je Achse.
Einer der vier Schalter ist als Referenzschalter definiert.

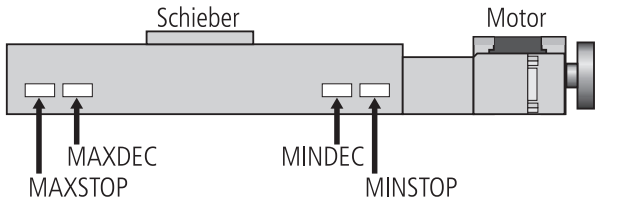

Bild 17: Endschalterkennzeichnung

Die Endschalter in negativer Fahrrichtung (Bewegung des Schiebers zum Motor hin) werden mit MINDEC und MINSTOP bezeichnet. Die Endschalter in positiver Fahrrichtung (Bewegung des Schiebers vom Motor weg) werden mit MAXDEC und MAXSTOP bezeichnet.

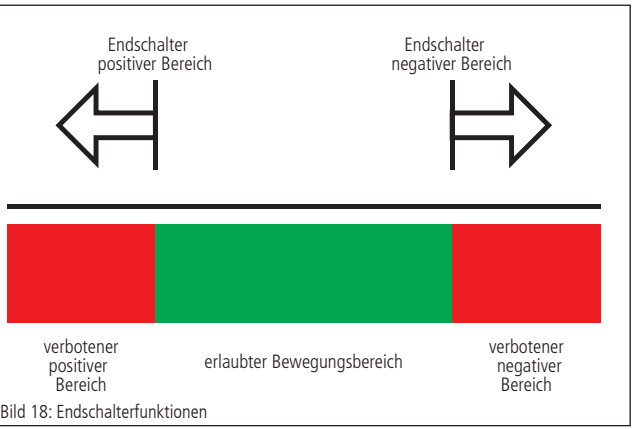

Bild 18: Endschalterfunktionen

Funktion der Endschalter-Überwachung

1. MINSTOP: Auslösen dieses Schalters bei Fahrt in negative Richtung bewirkt nach einer gewissen Reaktionszeit, die einige Millisekunden betragen kann, einen sofortigen, abrupten Motorstop. Der Motor wird hierbei stromlos geschaltet.

   DC-Servomotor: Der Motor wird stromlos geschaltet, jedoch führt die vorhandene kinetische Energie zu einer Restbewegung, bis sie durch Reibung oder mechanische Anschläge verbraucht wurde.

   Schrittmotor (Open Loop): Fals die aktuelle Fahrfrequenz, von der aus gestoppt wurde, höher gewesen ist als die Start-Stop-Frequenz des Systems, führt dies auf Grund der kinetischen Energie im System dazu, dass der Motor noch eine Bewegung ausführt. Dies kann von der Steuerung nicht erfasst werden, so dass der angezeigte Positionswert falsch ist. Eine Referenzfahrt ist nötig, um die Motorschritte wieder mit der angezeigten Position übereinstimmen zu lassen.

2. MINDEC: Dieser Endschalter löst bei Betätigung während negativer Fahrt eine Bremsrampe mit programmierbarer Verzögerung aus. Der Motor wird nach ausgeführter Bremsrampe nicht abgeschaltet, sondern bleibt weiterhin aktiv. Falls der Nachlaufweg der Bremsrampe zu groß gewesen sein sollte, und die Positioniereinheit anschließend den MINSTOP-Endschalter erreicht, siehe 1.

3. MAXDEC: Die Reaktion ist äquivalent zum MINDEC-Endschalter, jedoch wirkt dieser Endschalter nur bei Fahrt in positiver Richtung.

4. MAXSTOP: Die Reaktion ist äquivalent zum MINSTOP-Endschalter, jedoch wirkt dieser Endschalter nur bei Fahrt in positiver Richtung.

Konfiguration der End- und Referenzschalter
Welche Endschalter an der jeweils angeschlossenen Positioniereinheit vorhanden sind, kann mit dem Befehl „SMK…" definiert werden. Ein gesetztes Bit (=1) bedeutet, dass der jeweilige Schalter ausgewertet wird.

Die Endschalterpolarität wird mit dem Kommando „SPL…" vorgewählt. Der übergebene Wert definiert, ob Endschalter bzw. Referenzschalter „low" oder „high" aktiv sein sollen.
Ein gelöschtes Bit bedeutet, dass der jeweilige Schalter „low" aktiv ist (z.B. Schließerkontakt nach Masse, d.h. offen in nicht betätigtem Zustand). Ein gesetztes Bit (Standardkonfiguration) bedeutet, dass der jeweilige Schalter „high" aktiv ist (z.B. Öffnerkontakt nach Masse, d.h. geschlossen in nicht betätigtem Zustand).

Die Endschaltereingänge arbeiten standardmäßig mit 5V-CMOS-Pegel, wobei Open-Collector-NPN- oder Push-Pull-Ausgänge gleichermaßen angeschlossen werden können, da hochohmige Pullup-Widerstände (4,7 kOhm) nach + 5V bereits geräteintern vorgesehen sind.
Die Endschaltereingänge sind tolerant gegen eine Fremdspannung von bis zu + 24V.

Wiederinbetriebnahme nach Achsenfehler
Nachdem ein Achsenfehler durch Betätigung eines Limit-Schalters (MINSTOP oder MAXSTOP) aufgetreten ist, wird die Achse <n> wie folgt wieder in Betrieb genommen:

1. Initialisierung mittels Befehl INIT<n>
2. Freifahren des Limit-Schalters mittels Befehl EFREE<n>

14.2 Endstufen-Fehlerüberwachung

Jedes Endstufe meldet mit einer digitalen Leitung ihren Status an den Mikrocontroller zurück. Dieses Signal wird zyklisch kontrolliert. Meldet eine Endstufe einen Fehler, so wird der Antrieb stromlos geschaltet, d.h. die Regelschleife wird geöffnet und das Endstufen-Freigabe-Signal wird inaktiv gesetzt.

14.3 Motion-Controller-Fehlerüberwachung

Die Kommunikation mit den Motion-Controllern wird ebenfalls überwacht. Treten dabei Fehler oder Unplausibilitäten auf, so wird der Antrieb stromlos geschaltet, d.h. die Regelschleife wird geöffnet und das Endstufen-Freigabe-Signal wird inaktiv gesetzt.

14.4 Time-Out-Überwachung

Für jede Achse kann zusätzlich als Parameter eine Timeout-Zeit (in ms, Wertebereich 32 Bit) definiert werden. Die Überwachung kann durch die Einstellung Timeout-Zeit = 0 abgeschaltet werden. Während eine Bewegung (PGO, REF, EFREE, PWMSGO, LIGO) durchgeführt wird, wird zyklisch diese Timeout-Zeit überwacht. Dauert die Bewegung länger als diese Zeit, so wird der Antrieb stromlos geschaltet (?ASTAT → „Z", siehe Befehlssatz ab S.25), d.h. die Regelschleife wird geöffnet und das Endstufen-Freigabe-Signal wird inaktiv gesetzt. Diese Funktion ist nützlich, wenn z.B. bei der Referenzfahrt der Referenzschalter nicht gefunden wird.

15. Joystick

Zusätzlich zum Handterminal besteht die Möglichkeit, einen Joystick an die Steuerung anzuschließen, der als Zubehör erhältlich ist. Mit ihm können maximal drei Achsen manuell verfahren werden. Der XYZ-Joystick wird an den Analogeingänge (1, 2, 3) der PS 90+ angeschlossen.

Bild 19: Joystick

16. Hinweise zum Aufbau einer eigenen Applikationssoftware

Eine PS 90+-Applikation besteht allgemein aus einem Initialisierungsteil, welcher die erforderlichen Achsparameter für alle zu verwendenden Achsen <n> setzt und die Achsen einschaltet, einer Schleife, die eine Referenzfahrt für alle Achsen durchführt, und dem eigentlichen Anwenderprogramm, welches die vom Anwender gewünschte Funktionalität beinhaltet.

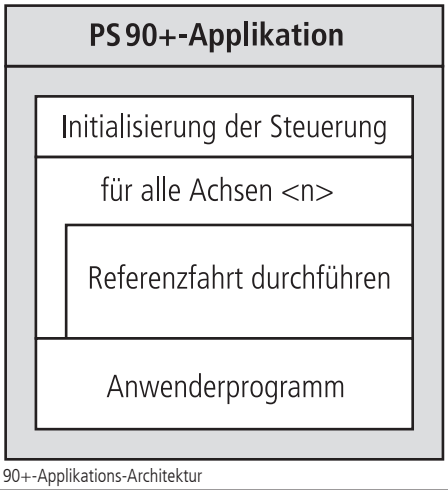

Bild 20: PS 90+-Applikations-Architektur

Die Initialisierung der gewünschten Achsen geschieht im einfachsten Fall über das INIT-Kommando, falls die im statischen RAM gespeicherten Parameter übernommen werden sollen. Andernfalls ist es erforderlich, die gewünschten Parameter vor Senden des INIT-Kommandos zu übertragen.

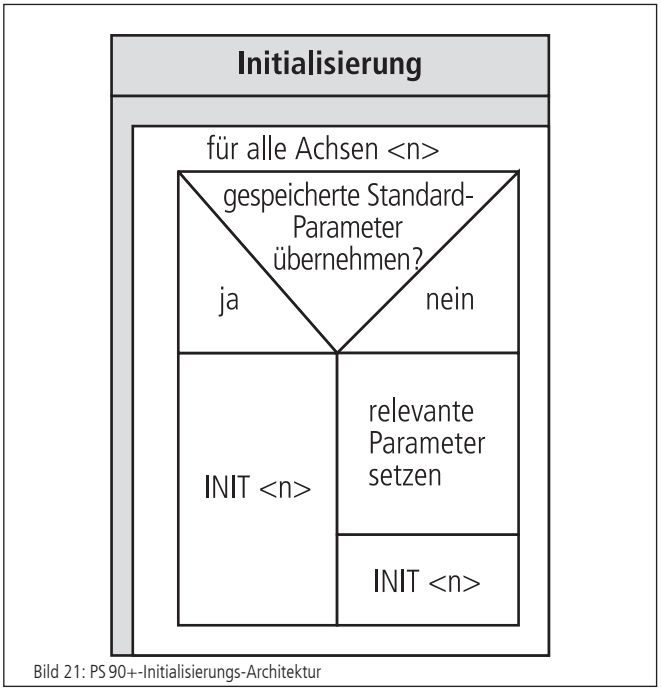

Bild 21: PS 90+-Initialisierungs-Architektur

Soll eine Referenzfahrt für eine Achse durchgeführt werden, sind Referenzmaske und Referenzpolarität vorher zu setzen, falls dies nicht bereits erfolgt ist oder entsprechende Werte in den Standard-einstellungen hinterlegt worden sind. Danach wird die Referenz-fahrt gestartet.

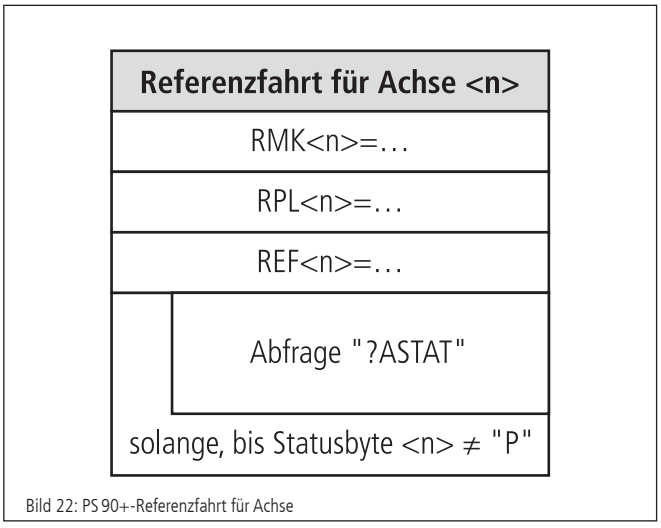

Bild 22: PS 90+-Referenzfahrt für Achse

Zwischen zwei einzelnen Befehlen, die zur PS 90+ gesendet werden, ist eine Verarbeitungszeit (Interpreterzeit) von ca. 20 bis 40 Millisekunden zu berücksichtigen. Empfangene Gerätemeldungen können z.B. Zeichen für Zeichen im Millisekunden-Takt abgeholt werden, bis die definierte Stringende-Kennung empfangen wird.

Eine Verwendung des mitgelieferten Softwarepakets OWISoft (inklusive SDK und DLL) erleichtert die Inbetriebnahme wesentlich, da häufig verwendete Befehlsfolgen bereits als Funktionen bzw. Prozeduren zusammengefasst sind, und der erforderliche Laufzeit-abgleich ebenfalls implementiert ist.

17. Befehlssatz der PS 90+

Generelles zum Format der Befehle:

Jeder Befehl wird über die Schnittstelle (RS-232, USB oder Ethernet) in Form von ASCII-Zeichen übertragen. Die einzelnen Zeichen eines Befehls werden automatisch in Großbuchstaben umgewandelt. Jeder Befehl wird mit CR oder CR+LF oder LF (einstellbar) abgeschlossen.
Weiterhin ist der Antwortmodus einstellbar (TERM). Dazu gibt es drei Einstellungen:

1) Beim Auslesen des Message-Ausgangs-Buffers wird nur eine zweistellige Zahl zurückgegeben (Fehlercode). Diese Einstellung wird vorzugsweise bei Ansteuerung über Software gewählt, da die Gerätemeldungen hier am kürzesten sind, womit der Befehlsdurchsatz optimiert wird.
2) Beim Auslesen des Message-Ausgangs-Buffers wird eine zweistellige Zahl mit Klartext ausgegeben.
3) Wie 2) und zusätzlich wird jeder ausgeführte Befehle, der keinen Wert zurückmeldet, mit „OK" quittiert.

Rückmeldungen werden auch entweder mit CR oder CR + LF oder LF zurückgesendet (einstellbar).

Im ersten Antwortmodus (TERM=0) werden die binären Informationen (z.B. Endschalterkonfiguration, Endschalterstatus, digitale/analoge Eingänge/Ausgänge usw.) als Bits einer Dezimalzahl angegeben. In den anderen Modi (TERM=1, TERM=2) werden diese Werte als binäre Zahl angegeben. Dies gilt sowohl für die Abfrage als auch für die Einstellung eines Wertes.

Alle Parameter werden resident abgespeichert und mit einer Checksumme versehen. Nach dem Aus- und erneutem Einschalten des Gerätes ist der letzte Stand der Parameter wieder gültig. Sollte die Checksumme nicht mehr stimmen, so werden beim Einschalten automatisch die Werte aus dem FRAM geladen und eine Fehlermeldung in den Fehlerspeicher eingetragen.

Bei Befehlen mit einer Rückantwort (z.B. Abfragen von Parametern) wird die Antwort sofort zum PC zurückgeschickt.

<n> = Achsennummer 1...9 (bzw. höchste Achsennummer)
<uv> = Zahlenwert ohne Vorzeichen
<sv> = Zahlenwert mit Vorzeichen
<v> = vorzeichenbehaftete Wegangabe

Anhang

I Befehlstabelle

Befehls-gruppe	Kommando	Funktionsbeschreibung	Beispiel	Antwort
Allgemeine Statusabfragen	?ASTAT	Statusabfrage der Achsen, pro Achse wird ein Zeichen zurückgeschickt, das den aktuellen Zustand der Achse beschreibt: „I" = Achse nicht initialisiert „O" = Achse stromlos in Ruhe „R" = Achse bestromt in Ruhe „T" = Achse positioniert im Trapez-Profil „S" = Achse positioniert im S-Kurven-Profil „V" = Achse arbeitet im Geschwindigkeitsmodus „P" = Achse fährt auf Referenzposition „F" = Achse fährt einen Endschalter frei „J" = Achse arbeitet im Joystick-Betrieb „L" = Achse stromlos, nachdem sie auf Limitschalter (MINSTOP, MAXSTOP) gefahren ist „B" = Achse wird gestoppt, nachdem sie auf einen Bremsschalter (MINDEC, MAXDEC) gefahren ist „A" = Achse stromlos nach Endstufen-Fehler „M" = Achse stromlos nach Motion-Controller-Fehler „Z" = Achse stromlos nach Timeout-Fehler „H" = Phaseninitialisierung aktiv (Schrittmotor-Achse) „U" = Achse nicht freigegeben „E" = Achse stromlos nach Bewegungsfehler „W" = Achse positioniert im Trapez-Profil mit WMS „X" = Achse positioniert im S-Kurven-Profil mit WMS „Y" = Achse arbeitet im Geschwindigkeitsmodus mit WMS „C" = Achse arbeitet im Bahnsteuerungsgeschwindigkeitsmodus „N" = Achse arbeitet im Piezo-Nachführmodus mit WMS „?" = Fehler, unbekannter Achsenstatus	?ASTAT	IIOURTTJV
	?MSG	Liest den Message-Ausgangs-Buffer aus, der Message-Ausgangs-Buffer wird nur für Fehlermeldungen, die die Kommando-Schnittstelle betreffen (falscher Befehl, fehlende Parameter, ungültiger Wert) verwendet. Folgende Meldungen sind möglich: „00 NO MESSAGE AVAILABLE" (wird ausgegeben, wenn der Meldungspuffer ausgelesen wird, obwohl keine Meldung verfügbar ist) „01 PARAMETER BEFORE EQUAL WRONG" (wird in den Meldungspuffer geschrieben, wenn der Befehlsinterpreter den Parameter vor dem Gleichheitszeichen nicht korrekt in einen Zahlenwert umwandeln konnte) „02 AXIS NUMBER WRONG" (wird in den Meldungspuffer geschrieben, wenn der Befehlsinterpreter die übergebene Achsennummer nicht auswerten konnte; zulässig z.B. 1 bis 9) „03 PARAMETER AFTER EQUAL WRONG" (wird in den Meldungspuffer geschrieben, wenn der Befehlsinterpreter den Parameter nach dem Gleichheitszeichen nicht korrekt in einen Zahlenwert umwandeln konnte) „04 PARAMETER AFTER EQUAL RANGE" (wird in den Meldungspuffer geschrieben, wenn der Befehlsinterpreter erkannt hat, dass der Parameter hinter dem Gleichheitszeichen außerhalb des zulässigen Wertebereichs liegt) „05 WRONG COMMAND ERROR" (wird in den Meldungspuffer geschrieben, wenn der gesendete Befehl syntaktisch nicht korrekt war, d.h. vom Befehlsinterpreter nicht erkannt wurde) „06 REPLY IMPOSSIBLE" (wird ausgegeben, wenn die Antwort nicht gesendet werden konnte, z.B., weil der Sendepuffer nocht nicht leer ist) „07 AXIS IS IN WRONG STATE" (wird in den Meldungspuffer geschrieben, wenn ein Fahr- oder Konfigurierungsbefehl gesendet wurde, der nicht ausgeführt werden konnte, da sich die Achse momentan in einem anderen Fahrzustand befindet) „08 AXIS NOT RELEASED" (wird ausgegeben, wenn eine nicht freigegebene Achse initialisiert wird) „09 ERROR IN POSITION TABLE" (wird ausgegeben, wenn es ein Problem in der Tabelle für Bahnsteuerung gibt) „10 MPUNI CAN ERROR" (wird in den Melduingspuffer geschrieben, wenn es einen Fehler in der internen Kommunikation gibt)	?MSG	00 NO MESSAGE...

Befehls-gruppe	Kommando	Funktionsbeschreibung	Beispiel	Antwort
Allgemeine Statusabfragen	?ERR	Abfrage eines Fehlers aus dem Fehlerspeicher mit einer Speichertiefe von 20. Die Fehlernummer wird immer als 4-stellige Zahl zurückgegeben. Anhand des Fehlercodes kann die Ursache ermittelt werden. Wird 0 zurückgegeben, so sind keine weiteren Fehler mehr gespeichert.	?ERR	1211
	ERRCLEAR	Fehlerspeicher löschen.	ERRCLEAR	
	?ESTAT<n>	Auslesen des aktuellen logischen Zustandes der Endschalter und Endstufen-rückmeldung einer Achse. Bit 0 = MINSTOP, Bit 1 = MINDEC, Bit 2 = MAXDEC, Bit 3 = MAXSTOP, Bit 4 = Rückmeldung der Endstufe.	?ESTAT	10101
	?AXSIGNALS<n>	Hardware-Achsen-Signale einer Achse abfragen. Bit 0 = Encoder CHA, Bit 1 = Encoder CHB, Bit 2 = Encoder Index, Bit 3 = Encoder Home, Bit 4 = MAXSTOP, Bit 5 = MINSTOP, Bit 6 = AxisIn-Pin, Bit 7 = Hall A, Bit 8 = Hall B, Bit 9 = Hall C, Bit 10 = AxisOut-Pin, Bit 11-15 = reserviert.	?AXSIGNALS1	0000011101101001
	?MPUNISTAT<n>	Statusinformationen über eine Achse abfragen: 1. Achsennummer 2. Fehlercode 3. Typ 4. AD-Digits 5. Version 6. Signale		
	?READOWID<n>=<uv>	Auslesen des Speicherinhaltes des One-Wire-Chips in der Positioniereinheit bis zur 0x00 Endkennung und Übertragen der Daten an den PC. Als Parameter wird die Anfangsadresse 0x00 bis 0x70 im One-Wire-Chip übergeben, ab dieser Adresse werden dann max. 16 Bytes gelesen oder es wird bis zur Enderkennung gelesen.	?READOWID1=0	INFO1|INFO2|....
	?READOWUB<n>	Auslesen des Speicherinhaltes des One-Wire-Chips aus Adresse 0x86 und 0x87 (=UserBytes) in der Positioniereinheit und Übertragen der Daten an den PC.	?READOWUB1	90
Basis-Konfiguration	AXIS<n>=<uv>	Eine Achse freigeben bzw. sperren. Mit diesem Befehl kann eine Achse freigegeben (1) oder gesperrt (0) werden.	AXIS5=1	
	?AXIS<n>	Freigabezustand einer Achse auslesen. Ist die Achse freigegeben, so wird eine 1 angezeigt, ansonsten eine 0.	?AXIS5	1
	MOTYPE<n>=<uv>	0 = DC-Brush 2 = Schrittmotor Open Loop 3 = Schrittmotor Closed-Loop 4 = BLDC	MOTYPE1=0	
	?MOTYPE<n>	Motortyp für eine Achse auslesen.	?MOTYPE1	0
	AMPSHNT<n>=<uv>	Stombereich für eine Achse einstellen: 0 = Strombereich 1 (niedrig) 1 = Strombereich 2 (hoch)	AMPSHNT1=0	
	?AMPSHNT<n>	Vorgewählten Strombereich für eine Achse auslesen.	?AMPSHNT1	1
	TERM=<uv>	Terminalmodus einstellen: Modus 0 = kurze Antwort Modus 1 = Antwort mit Klartext Modus 2 = Antwort mit Klartext und OK nach jedem Befehl ohne Rückmeldung	TERM=2	
	?TERM	Terminalmodus abfragen.	?TERM	2
	BAUDRATE	Baudrate der seriellen Schnittstelle einstellen, erlaubte Werte sind: 9 600, 19 200, 38 400, 57 600, 115 200. Diese Einstellung wird erst nach dem nächsten Reset aktiv.	BAUDRATE=9600	
	?BAUDRATE	Aktuelle Baudrate der seriellen Schnittstelle abfragen.	?BAUDRATE	9600
	COMEND	Befehlsendekennung einstellen: 0 = CR, 1 = CR+LF, 2 = LF.	COMEND=0	
	?COMEND	Befehlsendekennung abfragen.	?COMEND	0
	SAVEGLOB	Globale Parameter im seriellen FRAM abspeichern.	SAVEGLOB	
	LOADGLOB	Globale Parameter aus dem seriellen FRAM abrufen.	LOADGLOB	
	SAVEAXPA<n>	Achsen-Parameter einer Achse im seriellen FRAM abspeichern.	SAVEAXPA1	
	LOADAXPA<n>	Achsen-Parameter einer Achse aus dem seriellen FRAM abrufen.	LOADAXPA1	
	?SERNUM	Serien-Nummer der Steuerung abfragen.	?SERNUM	09080145
	?VERSION	Software-Version HP-Firmware auslesen.	?VERSION	PS90-V8.0-xxxxxxxxx

Befehls-gruppe	Kommando	Funktionsbeschreibung	Beispiel	Antwort
Basis-Konfiguration	?MCTRVER	Versionsdaten des Motion-Controller-Chips zurückgeben.		
	?PCHECK	Checksumme über den Programmspeicher berechnen und auslesen.	?PCHECK	12227
	JZONE=<uv>	Inaktive Zone des Joysticks einstellen (0-256).	JZONE=25	
	?JZONE	Inaktive Zone des Joysticks auslesen.	?JZONE	25
	JZEROX=<uv>	Nullpunkt des X-Joysticks setzen.	JZEROX=505	
	?JZEROX	Nullpunkt des X-Joysticks auslesen.	?JZEROX	505
	JZEROY=<uv>	Nullpunkt des Y-Joysticks setzen.	JZEROY=515	
	?JZEROY	Nullpunkt des Y-Joysticks auslesen.	?JZEROY	515
	JZEROZ=<uv>	Nullpunkt des Z-Joysticks setzen.	JZEROZ=508	
	?JZEROZ	Nullpunkt des Z-Joysticks auslesen.	?JZEROZ	508
	JBUTTON=<uv>	Auswertung des Joystickbuttons ein-/ ausschalten.	JBUTTON=1	
	?JBUTTON	Auslesen, ob der Joystickbutton ausgewertet wird oder nicht.	?JBUTTON	1
	IPADDR=<uv>	Feste IPv4-Adresse setzen als Dezimalzahl Aktiv nach globalem Speichern und Neustart	IPADDR= 168428041 (für 10.10.2.9) IPADDR= 3232235628 (für 192.168.0.108)	
	IPADDR1=<uv>	Feste IPv4-Adresse setzen. Notation mit Punkten Aktiv nach globalem Speichern und Neustart	IPADDR1= 10.10.2.9 IPADDR1= 192.168.0.108	
	?IPADDR	IPv4-Adresse abfragen als Dezimalzahl. Gibt die zuletzt gespeicherte Adresse zurück.	?IPADDR	168428041
	?IPADDR1	IPv4-Adresse abfragen. Notation mit Punkten. Gibt die zuletzt gespeicherte Adresse zurück.	?IPADDR1	10.10.2.9
	IPGWADDR=<uv>	Feste IPv4-Gateway-Adresse setzen als Dezimalzahl Aktiv nach globalem Speichern und Neustart	IPGWADDR = 168248287 (für 10.10.2.255)	
	IPGWADDR1=<uv>	Feste IPv4-Gateway-Adresse setzen. Notation mit Punkten Aktiv nach globalem Speichern und Neustart	IPGADDR1= 10.10.2.255	
	?IPGWADDR	IPv4-Gateway-Adresse abfragen als Dezimalzahl Gibt die zuletzt gespeicherte Adresse zurück.	?IPGWADDR	168248287
	?IPGWADDR1	IPv4-Gateway-Adresse abfragen. Notation mit Punkten Gibt die zuletzt gespeicherte Adresse zurück.	?IPGWADDR1	10.10.2.255
	IPNETMASK=<uv>	Feste IPv4-Netzmaske-Adresse setzen als Dezimalzahl Aktiv nach globalem Speichern und Neustart.	IPGWADDR = 168428032 (für 10.10.2.0)	
	IPNETMASK1=<uv>	Feste IPv4-Netzmaske-Adresse setzen. Notation mit Punkten Aktiv nach globalem Speichern und Neustart.	IPGWADDR1= 10.10.2.0	
	?IPNETMASK	IPv4-Netzmaske abfragen als Dezimalzahl Gibt die zuletzt gespeicherte Adresse zurück.	?IPGWADDR	168428032
	?IPNETMASK1	IPv4-Netzmaske abfragen. Notation mit Punkten Gibt die zuletzt gespeicherte Adresse zurück.	?IPGWADDR1	10.10.2.0
	DHCP=<uv>	DHCP aktivieren (=1) oder deaktivieren (=0) Aktiv nach globalem Speichern und Neustart.	DHCP=0 DHCP=1	
	?DHCP	Status des DHCP abfragen. Gibt den zuletzt gesetzten Status zurück.	?DHCP	0
	?IPADDRACTIVE	Gibt die aktuell verwendete IPv4-Adresse als Dezimalzahl zurück.	?IPADDRACTIVE	168428041
	?IPADDRACTIVE1	Gibt die aktuell verwendete IPv4-Adresse zurück. Notation mit Punkten.	?IPADDRAC- TIVE1	10.10.2.9
	?MACADDR	Gibt die MAC-Adresse der Ethernet-Schnittstelle zurück.	?MACADDR	ab:bc:cd:01.02.03

Befehls-gruppe	Kommando	Funktionsbeschreibung	Beispiel	Antwort
Positionierbetrieb	INIT<n>	Endstufe Freigabe einschalten und Positionsregler aktivieren. Mit diesem Befehl wird die Achse komplett initialisiert und befindet sich anschließend im bestromten Zustand mit aktivem Positionsregler. **Dieser Befehl muss nach dem Einschalten der Steuerung übermittelt werden, damit die Achse anschließend mit den Befehlen REF, PGO, VGO etc. bewegt werden kann**. Vorher müssen folgende Parameter eingestellt worden sein: Motortyp, Limitschalter-Maske und -Polarität, Achsparameter/ Regelparameter, Strombereich der Motorendstufe.	INIT1	
	PSET<n>=<sv>	Zielposition bzw. Relativweg (ABSOL/RELAT) für eine Achse setzen. Ist absolutes Positionsformat eingeschaltet, so wird der Parameter als absolute Position mit Vorzeichen interpretiert, ist relative Positionsangabe gewählt, so wird der Parameter als Weg mit Vorzeichen interpretiert. Die neue absolute Zielposition berechnet sich dann aus der Summe von letzter absoluter Zielposition und übergebenem Weg.	PSET2=100000	
	?PSET<n>	Zielposition bzw. Relativweg für eine Achse auslesen.	?PVEL1	10000
	PCHANGE<n>=<sv>	arbeitet wie PSET, aber ändert zusätzlich bei laufender Trapez-Positionierung die Zielposition "on the fly".	PCHANGE2= 50000	
	?CMDPOS	aktuelle Kommando-Position für eine Achse auslesen. Dieser Befehl liefert die aktuelle Zielposition für den Lagerregler zurück.	?CMDPOS1	5000
	VVEL<n>=<sv>	Sollgeschwindigkeit für Geschwindigkeitsmodus einer Achse setzen. Mit diesem Befehl wird die Startgeschwindigkeit und auch evtl. eine neue Geschwindigkeit, während die Achse im Geschwindigkeitsmodus fährt, übergeben.	VVEL1=-20000	
	?VVEL<n>	Sollgeschwindigkeit für Geschwindigkeitsmodus auslesen.	?VVEL1	-20000
	PGO<n>	Positionierung einer Achse starten. Die Achse fährt die neue Zielposition entweder im Trapez- oder S-Kurven-Profil an (siehe „PMOD").	PGO2	
	VGO<n>	Geschwindigkeitsmodus einer Achse starten.	VGO2	
	MPGO=<uv>	Positionieren mit mehreren Achsen starten. Übergeben wird ein Zahlenwert, der angibt, welche Achsen gestartet werden sollen. Bit 0 = Achse 1... Bit 8 = Achse 9	MPGO= 000000011	
	MVGO=<uv>	Geschwindigkeitsmodus mit mehreren Achsen starten. Übergeben wird ein Zahlenwert, der angibt, welche Achsen gestartet werden sollen. Bit 0 = Achse 1... Bit 8 = Achse 9	MVGO= 000000011	
	STOP<n>	Bewegung einer Achse stoppen. Jegliche aktive Bewegung einer Achse wird abgebrochen. Der Antrieb stoppt mit der programmierten Bremsrampe und bleibt stehen.		
	MSTOP=<uv>	Mehrere Achsen stoppen. Übergeben wird ein Zahlenwert, der angibt, welche Achsen gestoppt werden sollen. Bit 0 = Achse 1... Bit 8 = Achse 9	MSTOP = 000000011	
	VSTP<n>	Geschwindigkeitsmodus einer Achse stoppen. Arbeitet eine Achse im Geschwindigkeitsmodus, so wird dieser mit diesem Befehl beendet und die Achse gestoppt.	VSTP2	
	EFREE<n>	Endschalter einer Achse freifahren. Nachdem ein Antrieb in einen Limit-Schalter (MINSTOP, MAXSTOP) oder Bremsschalter (MINDEC, MAXDEC) gefahren ist, kann mit diesem Befehl der Antrieb aus dem Schalter herausgefahren werden. Die Richtung der Bewegung wird dabei selbsttätig entschieden, je nach dem, ob ein positiver oder negativer Endschalter aktiviert ist.		
	MON<n>	Endstufe Freigabe einschalten und Positionsregler aktivieren. Mit diesem Befehl wird die Achse, nachdem der Motor stromlos geschaltet war, wieder eingeschaltet und befindet sich anschließend im bestromten Zustand mit aktivem Positionsregler.	MON1	
	MOFF<n>	Endstufe Freigabe ausschalten und Positionsregler deaktivieren. Mit diesem Befehl wird der Positionsregler deaktiviert und die Freigabe-Leitung für die Endstufe deaktiviert.	MOFF1	
	JOYON	Startet das Verfahren von 1 bis 3 Achsen mit dem Joystick. Anschließend bewegen sich diese Achsen im Geschwindigkeitsmodus. Die Geschwindigkeit und das Vorzeichen werden mit dem Joystick vorgegeben.	JOYON	
	JOYOFF	Stoppt das Verfahren von 1 bis 3 Achsen mit Joystick.	JOYOFF	
	CNT<n>=<sv>	Aktuellen Positionszähler für eine Achse setzen.	CNT1=5000	
	?CNT<n>	Aktuellen Positionszähler für eine Achse auslesen.	?CNT1	5000
	CRES<n>	Aktuellen Positionszähler für eine Achse nullen.	CRES1	
	?POSERR<n>	Auslesen des aktuellen Positionsfehlers einer Achse. Zurückgegeben wird die Differenz zwischen Encoder-Position und Sollposition. Kann auch „on the fly" benutzt werden um den Schleppfehler auszulesen.	?POSERR1	-15

Befehls-gruppe	Kommando	Funktionsbeschreibung	Beispiel	Antwort
Positionierbetrieb	?VACT<n>	Aktuelle Geschwindigkeit einer Achse auslesen. Die aktuelle Geschwindigkeit wird im Format 16.16 mit Vorzeichen zurückgegeben, der fraktionale Anteil ist jedoch 0, weil die aktuelle Geschwindigkeit aus der Positionsdifferenz zwischen zwei Sample-Zeiten berechnet wird.	?VACT2	1000
	?ENCPOS<n>	Aktuellen Positionszähler des Encoders für eine Achse auslesen. Dieser Befehl liefert bei Open-Loop-Schrittmotorachsen, die aber mit Encoder betrieben werden, die aktuelle Encoder-Position zurück (umgerechnet in Schritte).	?ENCPOS1	5000
	?MXSTROKE<n>	Gemessenen Tischhub auslesen. Bei der Referenzierung in den Modi 6 und 7 wird der Tischhub ermittelt und kann mit diesem Kommando ausgelesen werden.	?MXSTROKE1	340000
Positionierparameter	RELAT<n>	Positionsangaben für eine Achse auf „relativ" umschalten (= Angabe des Weges mit Vorzeichen).	RELAT1	
	ABSOL<n>	Positionsangaben für eine Achse auf „absolut" umschalten (= Angabe der Zielposition mit Vorzeichen).	ABSOL2	
	?MODE<n>	Abfrage des aktuell eingestellten Positionsformates für eine Achse.	?MODE2	ABSOL
	PMOD<n>=<uv>	Positioniermodus Trapez/S-Kurve für eine Achse setzen. (0 = Trapez-Profil, 1 = S-Kurven-Profil).	PMOD1=0	
	?PMOD<n>	Positioniermodus Trapez/S-Kurve für eine Achse auslesen.	?PMOD1	1
	PVEL<n>=<uv>	Max. Positioniergeschwindigkeit für eine Achse setzen, wird für das Trapez- und S-Kurven-Profil verwendet.	PVEL1=10000	
	?PVEL<n>	Max. Positioniergeschwindigkeit für eine Achse auslesen.	?PVEL1	10000
	FVEL<n>=<uv>	Endschalterfreifahrtgeschwindigkeit für eine Achse setzen (ohne Vorzeichen).	FVEL1=1000	
	?FVEL<n>	Endschalterfreifahrtgeschwindigkeit für eine Achse auslesen.	?FVEL1	1000
	ACC<n>=<uv>	Beschleunigung (= Anfahrrampe) für eine Achse setzen, wird für alle Modi verwendet (Trapez, S-Kurve, Geschwindigkeitsmodus etc).	ACC1=100	
	?ACC<n>	Beschleunigung für eine Achse auslesen.	?ACC1	100
	DACC<n>=<uv>	Verzögerung (= Bremsrampe) für eine Achse setzen, wird für alle Modi ausser S-Kurve verwendet.	DACC2=68	
	?DACC<n>	Verzögerung für eine Achse auslesen.	?DACC2	68
	JACC<n>=<uv>	Maximalen Jerk („Ruck") für eine Achse setzen, wird nur beim S-Kurven-Profil verwendet.	JACC9=5	
	?JACC<n>	Maximalen Jerk („Ruck") für eine Achse auslesen.	?JACC9	5
	EDACC<n>=<uv>	NOT-AUS-Verzögerung (Bremsbeschleunigung) für eine Achse einstellen. Diese Verzögerung wird benutzt, wenn ein Bremsschalter angesprochen hat.	EDACC1=1000	
	?EDACC<n>	NOT-AUS-Verzögerung einer Achse auslesen.	?EDACC1	1000
	JVEL<n>=<sv>	Max. Geschwindigkeit der Achse bei „Joystickfahrt" einstellen. Mit diesem Befehl wird die maximale Geschwindigkeit bei voller Auslenkung des Joysticks definiert.	JVEL3=1000	
	?JVEL<n>	Max. Geschwindigkeit der Achse bei Fahren mit Joystick auslesen.	?JVEL3	1000
	JOYACC<n>=<uv>	Beschleunigung und Verzögerung der Achse bei Fahren mit Joystick einstellen.		
	?JOYACC<n>	Beschleunigung/Verzögerung der Achse bei Fahren mit Joystick auslesen.	?JOYACC3	100
	JAUTOMOFF<n>=<uv>	Nur bei DC-Betrieb: Motor im Joystickmodus automatisch stromlos schalten, wenn die Zielposition erreicht ist.	JAUTOMOFF1=0	
	?JAUTOMOFF<n>	Abfrage, ob im Joystickmous auomatisch stromlos geschaltet wird.	?JAUTOMOFF1	0
	JPLAX=<n>	Joystickebenenzuordnung X-Ebene Achsen-Nummer setzen. Die übergebene Achsen-Nummer ist danach dem X-Joystick zugeordnet. Übergibt man „0", so ist anschließend keine Achse dem X-Joystick zugeordnet.	JPLAX=2	
	?JPLAX	Joystickebenenzuordnung X-Ebene Achsen-Nummer auslesen.	?PLAX	2
	JPLAY=<n>	Joystickebenenzuordnung Y-Ebene Achsen-Nummer setzen. Die übergebene Achsen-Nummer ist danach dem Y-Joystick zugeordnet. Übergibt man „0", so ist anschließend keine Achse dem Y-Joystick zugeordnet.	JPLAY=3	
	?JPLAY	Joystickebenenzuordnung Y-Ebene Achsen-Nummer auslesen.	?JPLAY	3

Befehls-gruppe	Kommando	Funktionsbeschreibung	Beispiel	Antwort
Positionierparameter	JPLAZ=<n>	Joystickebenenzuordnung Z-Ebene Achsen-Nummer setzen. Die übergebene Achsen-Nummer ist danach dem Z-Joystick zugeordnet. Übergibt man „0", so ist anschließend keine Achse dem Z-Joystick zugeordnet.	JPLAZ=3	
	?JPLAZ	Joystickebenenzuordnung Z-Ebene Achsen-Nummer auslesen.	?JPLAZ	
	LIGO=<uv>	Positionierung mit Linearinterpolation für eine Achsengruppe (binäre Definitionsmaske) starten. Bit-Reihenfolge :<Achse 9, Achse 8, Achse 7, Achse 6,..., Achse 2, Achse 1>.	LIGO=000000111	
	IVEL<n>=<uv>	Maximalgeschwindigkeit <uv> für Linearinterpolationsachse <n> einstellen.	IVEL1=50000	
	?IVEL<n>	Maximalgeschwindigkeit für Linearinterpolationsachse <n> auslesen.	?IVEL1	50000
	IACC<n>=<uv>	Maximalbeschleunigung <uv> für Linearinterpolationsachse <n> einstellen.	IACC3=2000	
	?IACC<n>	Maximalbeschleunigung für Linearinterpolationsachse <n> auslesen.	?IACC3	2000
	POSTAB<uv>=<v>, <v>,...	Eine Tabellenzeile in die Bahntabelle herunterladen; vor dem „="-Zeichen steht die Tabellen-Zeilennummer (0 bis ...), hinter dem „="-Zeichen folgen durch Komma getrennt die Werte für die Tabellen-Spalten. Parameter-Liste: 1. Weg mit Vorzeichen Achse 1 (-2147483648 bis 2147483648) 2. Weg mit Vorzeichen Achse 2 (-2147483648 bis 2147483648) 3. Weg mit Vorzeichen Achse 3 (-2147483648 bis 2147483648) 4. Weg mit Vorzeichen Achse 4 (-2147483648 bis 2147483648) 5. Weg mit Vorzeichen Achse 5 (-2147483648 bis 2147483648) 6. Weg mit Vorzeichen Achse 6 (-2147483648 bis 2147483648) 7. Weg mit Vorzeichen Achse 7 (-2147483648 bis 2147483648) 8. Weg mit Vorzeichen Achse 8 (-2147483648 bis 2147483648) 9. Weg mit Vorzeichen Achse 9 (-2147483648 bis 2147483648) 10. Funktionscode Bit 3 bis 0: Ausgangsnummer 1 Bit 4: Ausgangspegel 1 Bit 8 bis 5: Ausgangsnummer 2 Bit 9: Ausgangspegel 2 Bit 13 bis 10: Ausgangsnummer 3 Bit 14: Ausgangspegel 3 Bit 15 im Funktionscode gesetzt: Verfahren mit a=const. Bit 15 im Funktionscode gelöscht: Verfahren mit v=const. 11. Fehlerbyte 12. Freigabe-Byte (immer als Zahlenwert)	POSTAB0=1000, 2000, 0, 0, 0, 0, 0, 0, 0, 100, 0, 0, 192	
	?POSTAB<uv>	Eine Tabellenzeile aus der Bahntabelle auslesen. Als Parameter wird die Tabellen-Zeilennummer (0 bis ...) übergeben.	?POSTAB0	1000, 2000, 0, 0, 0, 0, 0, 0, 100, 0, 0, 192, 2100, 50,
	PTABPLAUS<uv>	Plausibilitätskontrolle der Bahntabelle durchführen. Die Grenzwerte für Geschwindigkeit und Beschleunigung werden geprüft und die Fehlerbytes entsprechend gesetzt. Die Geschwindigkeiten und Beschleunigungen in den einzelnen Tabellenzeilen werden berechnet und deren Werte für die aktive Achse mit der höchsten Achsnummer eingetragen.	PTABPLAUS0	
	PTABGO<uv>, <uv>	Mit einem Parameter: Bahnsteuerung ab einem bestimmten Tabelleneintrag starten. Mit zwei Parametern: Bahnsteuerung ab einem bestimmten Tabelleneintrag starten und vor einem anderen Tabelleneintrag stoppen.	PTABGO0, 15	
	PTABSTP	Eine laufende Bahnsteuerung abbrechen; die beteiligten Achsen verhalten sich wie am Ende der Bahntabelle.	PTABSTP	
	PTABCLR	Tabelle für Bahnsteuerung löschen.		
	PTABCIRCLE <uv>=<uv>, ...	Kreisinterpolation berechnen und in die Positionstabelle ab der angegebenen Startzeile eintragen. Die Parameter werden als Liste durch Komma getrennt übergeben.Die Achsen-Freigabebits werden mit den evtl. bereits bestehenden Einträgen der aktuellen Tabelle bitweise ODER-verknüpft. 1. Achsennummer für X (0 für keine X-Achse) 2. Achsennummer für Y (0 für keine Y-Achse) 3. Segmentzeit in ms (16 Bit) 4. Funktionscode (OR-Maske) für die Segmente (16 Bit) 5. Anzahl der Kreissegmente (16 Bit) 6. Kreisradius mit Vorzeichen (32 Bit) 7. Anfangswinkel in Grad mit Vorzeichen (16 Bit) 8. Winkelbereich in Grad mit Vorzeichen (16 Bit) 9. Optional Skalierung Zähler mit Vorzeichen (16 Bit) 10. Optional Skalierung Nenner mit Vorzeichen (16 Bit)		

Befehls-gruppe	Kommando	Funktionsbeschreibung	Beispiel	Antwort
Positionierparameter	PTABCPY <uv>=<uv>, <uv>	Einen Bereich der Tabelle für Bahnsteuerung kopieren. Der Wert vor dem "="-Zeichen gibt den Zielindex in der Positionstabelle an, der Wert hinter dem "="-Zeichen gibt den Quellindex an und der Wert hinter dem Komma die Anzahl Zeilen die kopiert werden sollen.	PTABCPY 50=10, 20	
	PTABDEL<uv>=<uv>	Einen Bereich der Tabelle für Bahnsteuerung löschen. Der Wert vor dem „="-Zeichen gibt den Zeilenindex an, ab dem gelöscht werden soll; der Wert hinter dem „="-Zeichen gibt die Anzahl Zeilen an, die gelöscht werden sollen.	PTABDEL50=20	
	PTABSAVE	Tabelle in Flash speichern.	PTABSAVE	
	PTABSAVE<uv,uv>	Teile der Tabelle im Flash speichern. Der erste Wert gibt die Startzeile an, der zweite Wert die Anzahl der Zeilen. Alle anderen Zeilen im Flash werden nicht geändert.	PTABSAVE3,10	
	PTABLOAD	Komplette Tabelle aus dem Flash in den Arbeitsspeicher laden.	PTABLOAD	
	PTABLOAD<uv,uv>	Teile der Tabelle aus dem Flash in den Arbeitsspeicher laden. Der erste Wert gibt die Startzeile an, der zweite Wert die Anzahl der Zeilen.		
	?PTABFLASHBUSY	Prüfen, ob eine Speicheraktion in den Flash aktiv ist (=1). Falls ja, sollte die Steuerung nicht ausgeschaltet oder neugestartet werden.	?PTABFLASHBUSY	0
PositionAutomatisches Reagieren und Setzen von Ausgängen	TRIG<uv>=<a>, <b>,<c>,[<d>[,<e>]]	Schreibe einen Eintrag in die Aktionstabelle <uv>=Zeilennummer <a>=Aktions-ID <b>=Eingang <c>=Flanke (0=fallend, 1=steigend) <d>=Parameter 1, signed 32 bit decimal oder Hex-String, z.B. Achsennummer oder Achsenmaske <e>=ggf. Parameter 2 Keine Leerzeichen! Vorzeichen sind möglich. Hex-Strings müssen mit 0x oder 0X beginnen. Parameter d und e hängen von der gewählten Aktion ab.	TRIG2=10,0,3,5 Zeile 2: Fallende Flanke auf Eingang 10 startet Positio-niervorgang von Achse 5	
	?TRIG<uv>	Zeile <uv> aus der Aktionstabelle ausgeben. Ausgaben immer als Dezimalzahl.	?TRIG2	10,0,3,5
	EVENT<uv>=<a>, <b>,<c>,[<d>[,<e>]]	Schreibe einen Eintrag in die Ereignistabelle <uv>=Zeilennummer <a>=Ereignis-ID <b>=Ausgangsmaske 16 bit decimal oder Hex-String <c>=Ausgangswertemaske 16 bit decimal oder Hex-String <d>=Parameter 1, signed 32 bit decimal oder Hex-String, z.B. Achsennummer oder Achsenmaske <e>=ggf. Parameter 2 Keine Leerzeichen! Vorzeichen sind möglich. Hex-Strings müssen mit 0x oder 0X beginnen. Parameter d und e hängen von der gewählten Aktion ab.	EVENT6=2,0x24, 0x4,8,10000 EVENT6=2,36, 4,8, 10000 Zeile 6: Wenn die Geschwindigkeit von Achse 8 unter 10000 fällt, Ausgang 3 auf 1 setzen und Ausgang 6 auf 0 setzen	
	?EVENT<uv>	Zeile <uv> aus der Ereignistabelle ausgeben. Ausgaben immer als Dezimalzahl.	?EVENT6	2,36,4,8,10000
	TRIGDEL[<a>[,<b>]]	Ohne Parameter: Gesamte Aktionstabelle löschen Ein Parameter: Zeile a löschen Zwei Parameter: Zeilen a bis a+b löschen.	TRIGDEL3,10	
	EVENTDEL[<a>[,<b>]]	Ohne Parameter: Gesamte Ereignistabelle löschen Ein Parameter: Zeile a löschen Zwei Parameter: Zeilen a bis a+b löschen.	EVENTDEL3,10	
	TRIGENABLE=<uv>	Ereignis- und Aktionsfunktion aktivieren (<uv>=1) oder deaktivieren (<uv>=0).	TRIGENABLE=1	
	?TRIGENABLE	Zustand der Funktion abfragen.	?TRIGENABLE	1
	TRIGTABSAVE	Aktionstabelle in Flash speichern.	TRIGTABSAVE	
	EVENTTABSAVE	Ereignistabelle in Flash speichern.	EVENTTABSAVE	
	?TRIGEVENTBUSY	Abfragen, ob aktuell Speichervorgänge in den Flash laufen 0: keine Speicherung 1: Speicherung Aktionstabelle 2: Speicherung Ereignistabelle 3. Speicherung beide Tabellen Während einer Speicherung die Steuerung nicht ausschalten oder neu starten.	?TRIGEVENTBUSY	0
	TRIGTABLOAD	Aktionstabelle aus Flash in Arbeitsspeicher laden.	TRIGTABLOAD	
	EVENTTABLOAD	Ereignistabelle aus Flash in Arbeitsspeicher laden.	EVENTTABLOAD	

Befehls-gruppe	Kommando	Funktionsbeschreibung	Beispiel	Antwort
Achsparameter	MCSTP<n>=<uv>	Mikroschrittauflösung bei Schrittmotorachsen einstellen.	MCSTP1=50	
	?MCSTP<n>	Mikroschrittauflösung bei Schrittmotorachsen auslesen.	?MCSTP1	50
	DRICUR<n>=<uv>	Fahrstrom bei Schrittmotorachsen als ganzzahliger Prozentwert des Maximalstromes im vorgewählten Strombereich (1 oder 2) in Prozent einstellen. Strombegrenzung bei DC-Achsen einstellen. 100 % entsprechen 12 A. Diese Einstellung wird nur nach einem Neustart der Steuerung mit der ersten Initialisierung übernommen.	DRICUR1=50	
	?DRICUR<n>	Fahrstrom bei Schrittmotorachsen in Prozent auslesen.	?DRICUR1	50
	HOLCUR<n>=<uv>	Haltestrom bei Schrittmotorachsen in Prozent einstellen.	HOLCUR1=30	
	?HOLCUR<n>	Haltestrom bei Schrittmotorachsen in Prozent auslesen.	?HOLCUR1	30
	ATOT<n>=<uv>	Achsen-Timeout-Zeit einstellen in Millisekunden, 0 schaltet die Timeout-Überwachung ab.	ATOT1=20000	
	?ATOT<n>	Achsen-Timeout-Zeit abfragen.	?ATOT1	20000
	FKP<n>=<uv>	Regelparameter KP für eine Achse einstellen.	FKP1=25	
	?FKP<n>	Regelparameter KP für eine Achse abfragen.	?FKP1	25
	FKD<n>=<uv>	Regelparameter KD für eine Achse einstellen.	FKD1=5	
	?FKD<n>	Regelparameter KD für eine Achse abfragen.	?FKD1	5
	FKI<n>=<uv>	Regelparameter KI für eine Achse einstellen.	FKI1=10	
	?FKI<n>	Regelparameter KI für eine Achse abfragen.	?FKI1	10
	FIL<n>=<uv>	Regelparameter Integrationslimit für eine Achse einstellen.	FIL1=100000	
	?FIL<n>	Regelparameter Integrationslimit für eine Achse abfragen.	?FIL1	100000
	FST<n>=<uv>	Sample-Zeit für eine Achse einstellen (in Mikrosekunden).	FST1=500	
	?FST<n>	Sample-Zeit für eine Achse abfragen (in Mikrosekunden).	?FST1	500
	FDT<n>=<uv>	Verzögerungszeit des D-Anteils für eine Achse einstellen (in Sample-Zeit-Zyklen).	FDT1=5	
	?FDT<n>	Verzögerungszeit des D-Anteils für eine Achse abfragen (in Sample-Zeit-Zyklen).	?FDT1	5
	MXPOSERR<n>=<uv>	Maximalen Positionierfehler für eine Servo-Achse setzen. Wird dieser Wert überschritten, so schaltet die Achse ab. Diese Abschaltung gilt nur für die Motortypen DC-Brush, Schrittmotor Closed Loop und BLDC.	MXPOSERR1=50	
	?MXPOSERR<n>	Maximalen Positionierfehler einer Achse abfragen.	?MXPOSERR1	50
	MAXOUT<n>=<uv>	Maximalen Ausgabewert der Servoregelschleife in Prozent einstellen. Mit diesem Befehl kann der maximale Wert für eine Achse, der an den Servo-Verstärker ausgegeben wird, eingestellt werden. **Max. zulässiger Wert: 99 %**	MAXOUT1=95	
	?MAXOUT<n>	Maximalen Ausgabewert in Prozent auslesen.	?MAXOUT1	95
	MPUNIPID<n>=<uv1>, <uv2>, <uv3>, <uv4>	PID-Stromregler für Schrittmotorbetrieb einstellen. uv1 = KP uv2 = KI fast uv3 = KI slow uv4 = KD	MPUNIPID1=900,0,40,1500	
	?MPUNIPID<n>	PID-Stromreglerwerte abfragen	?MPUNIPID1	900|0|40|1500
	INPOSMOD<n>=<uv>	Bewegungsfertigmeldemodus einstellen: 0 = Zielposition erreicht 1 = für eine gewisse Zeit im Fenster um die Zielposition	INPOSMOD1=0	
	?INPOSMOD<n>	Bewegungsfertigmeldemodus abfragen.	?INPOSMOD1	0
	INPOSTIM<n>=<uv>	Bewegungsfertigmeldezeit einstellen in Sample-Zeit-Zyklen.	INPOSTIM1=1000	
	?INPOSTIM<n>	Bewegungsfertigmeldezeit abfragen.	?INPOSTIM1	1000
	INPOSWND<n>=<uv>	Bewegungsfertigmeldefenster einstellen in Encoder-Counts.	INPOSWND1=50	
	?INPOSWND<n>	Bewegungsfertigmeldefenster abfragen.	?INPOSWND1	50
	AMPPWMF<n>=<uv>	PWM-Frequenz für Endstufe einstellen, 20000 oder 80000 ist möglich.	AMPPWMF1=20000	
	?AMPPWMF<n>	PWM-Frequenz für Endstufe abfragen.	?AMPPWMF1	20000
	ENCLINES<n>=<uv>	Linien-Anzahl des Encoders für eine Achse einstellen.	ENCLINES1=500	
	?ENCLINES<n>	Linien-Anzahl des Encoders für eine Achse abfragen.	?ENCLINES1	500
	MOTPOLES<n>=<uv>	Polanzahl des Motors für eine Achse einstellen.	MOTPOLES1=25	
	?MOTPOLES<n>	Polanzahl des Motors für eine Achse auslesen.	?MOTPOLES1	25
	BLDCCT<n>=<uv>	Kommutierungsmodus bei BLDC einstellen: 0 = Blockkommutierung mit Hallsensoren 1 = Sinuskommutierung mit Encoder	BLDCCT1=0	
	?BLDCCT<n>	Kommutierungsmodus bei BLDC abfragen.	?BLDCCT1	0

Befehls-gruppe	Kommando	Funktionsbeschreibung	Beispiel	Antwort
Achsparameter	ELCYCNT<n>=<uv>	Encoder-Counts für einen elektrischen Kommutierungszyklus einstellen.	ELCYCNT1=128	
	?ELCYCNT<n>	Encoder-Counts für einen elektrischen Kommutierungszyklus abfragen.	?ELCYCNT1	128
	PHINTIM<n>=<uv>	Phasen-Initialisierungszeit in Sample-Zeit-Zyklen einstellen.	PHINTIM1=10	
	?PHINTIM<n>	Phasen-Initialisierungszeit in Sample-Zeit-Zyklen abfragen.	?PHINTIM1	10
	PHINAMP<n>=<uv>	Phasen-Initialisierungsamplitude in % einstellen.	PHINAMP1=50	
	?PHINAMP<n>	Phasen-Initialisierungsamplitude in % abfragen.	?PHINAMP1	50
Endschalterkonfiguration und Referenzfahrt	REF<n>=<uv>	Referenzfahrt mit Angabe des Referenzfahrtmodus für eine Achse starten: Modus 0 = nächsten Index-Impuls suchen und stehenbleiben Modus 1 = Referenzschalter anfahren und stehenbleiben Modus 2 = Referenzschalter anfahren, nächsten Index-Impuls suchen und stehenbleiben Modus 3 = Modus 0, zusätzlich akt. Positon auf 0 setzen Modus 4 = Modus 1, zusätzlich akt. Positon auf 0 setzen Modus 5 = Modus 2, zusätzlich akt. Positon auf 0 setzen Modus 6 = Maximalen Referenzschalter anfahren, minimalen Referenzschalter anfahren, aktuelle Position auf 0 setzen Modus 7 = Minimalen Referenzschalter anfahren, maximalen Referenzschalter anfahren, aktuelle Positon auf 0 setzen		
	RVELS<n>=<sv>	Referenzfahrtgeschwindigkeit „langsam" für eine Achse setzen. Mit dieser Geschwindigkeit wird der Index gesucht bzw. aus dem Referenzschalter herausgefahren (vorzeichenbehaftet).	RVELS2=2000	
	?RVELS<n>	Referenzfahrtgeschwindigkeit „langsam" für eine Achse auslesen.	?RVELS2	2000
	RVELF<n>=<sv>	Referenzfahrtgeschwindigkeit „schnell" für eine Achse setzen. Mit dieser Geschwindigkeit fährt der Antrieb auf den Referenzschalter (vorzeichenbehaftet).	RVELF2=-20000	
Endschalterkonfiguration und Referenzfahrt	?RVELF<n>	Referenzfahrtgeschwindigkeit „schnell" für eine Achse auslesen.	?RVELF2	-20000
	RDACC<n>=<uv>	Referenzfahrt-Verzögerung für eine Achse einstellen. Diese Verzögerung wird benutzt, wenn der Referenzpunkt angefahren wird.	RDACC1=1000	
	?RDACC<n>	Referenzfahrt-Verzögerung einer Achse auslesen.	?RDACC1	1000
	SMK<n>=<uv>	Endschaltermaske für eine Achse setzen. Mit diesem Befehl werden die Endschalter und die Bremsschalter aktiv bzw. inaktiv gesetzt. Wird auf einen Endschalter gefahren, so wird die Bewegung abrupt gestoppt und der Motor danach stromlos geschaltet. Bit-Reihenfolge: <MAXSTOP, MAXDEC, MINDEC, MINSTOP>.	SMK3=0110	
	?SMK<n>	Endschaltermaske für eine Achse auslesen.	?SMK3	0110
	SPL<n>=<uv>	Endschalterpolarität für eine Achse setzen. Mit diesem Befehl wird der aktive Pegel für die Endschalter und Bremsschalter festgelegt. Bit-Reihenfolge: <MAXSTOP, MAXDEC, MINDEC, MINSTOP>.	SPL3=1111	
	?SPL<n>	Endschalterpolarität für eine Achse auslesen.	?SPL3	1111
	RMK<n>=<uv>	Referenzschaltermaske für eine Achse setzen. Mit dem Befehl wird definiert, welcher der 4 Endschalter einer Achse als Referenzschalter interpretiert werden soll. Es muss eine Maske mit genau einer „1" übergeben werden. Bit-Reihenfolge: <MAXSTOP, MAXDEC, MINDEC, MINSTOP>.	RMK3=0001	
	?RMK<n>	Referenzschaltermaske für eine Achse auslesen.	?RMK3	1
	RPL<n>=<uv>	Referenzschalterpolarität für eine Achse setzen. Dieser Befehl definiert den aktiven Pegel des Referenzschalters. Bit-Reihenfolge: <MAXSTOP, MAXDEC, MINDEC, MINSTOP>.	RPL3=1110	
	?RPL<n>	Referenzschalterpolarität für eine Achse auslesen.	?RPL3	1110
	?HYST<n>	Referenzschalterhysterese einer Achse auslesen. Nach erfolgter Referenzfahrt kann mit dem Kommando die Hysterese des Schalters ausgelesen werden.	?HYST1	28
	?REFST<n>	Abfrage der Gültigkeit der Referenzfahrt. Nach erfolgter Referenzfahrt wird der Status auf 1 = „Gültig" gesetzt. Schaltet man einen Antrieb ohne Encoder (z.B. Schrittmotor Open-Loop) stromlos, so wird die Gültigkeit auf 0 zurückgesetzt.	?REFST	1

Befehls-gruppe	Kommando	Funktionsbeschreibung	Beispiel	Antwort
Endschalterkonfiguration und Referenzfahrt	LMK<n>=<uv>	Limit-Positionsüberwachungsmaske für die Achse setzen. Mit diesem Befehl wird die Limit-Positionsüberwachung für die untere und/oder die obere Grenzposition aktiv bzw. inaktiv geschaltet. Die Limit-Positionsüberwachung verhält sich beim Überschreiten der Grenzen wie der entsprechende DEC-Schalter. Bit-Reihenfolge: <MAXDEC, MINDEC>	LMK1=01	
	?LMK<n>	Limit-Positionsüberwachungsmaske für die Achse auslesen.	?LMK1	01
	?LSTAT<n>	Aktuellen, logischen Zustand der Limit-Positionsüberwachung der Achse auslesen. Bit 0 = MINDEC untere Grenze überschritten Bit 1 = MAXDEC obere Grenze überschritten	?LSTAT1	01
	SLMIN<n>=<uv>	Negative Limit-Position für die Achse einstellen.	SLMIN1=100	
	?SLMIN<n>	Negative Limit-Position für die Achse auslesen.	?SLMIN1	100
	SLMAX<n>=<uv>	Positive Limit-Position für die Achse einstellen.	SLMAX1= 100000	
	?SLMAX<n>	Positive Limit-Position für die Achse auslesen.	?SLMAX1	100000
Ein-/Ausgänge	ETTLOUTS<n>=<bin>	TTL-Ausgänge zur Endstufe einer Achse setzen. Übergeben wird die Achsennummer und eine binäre Setzmaske.	ETTLOUTS1=10	
	ETTLOUTC<n>=<bin>	TTL-Ausgänge zur Endstufe einer Achse rücksetzen. Übergeben wird die Achsennummer und eine binäre Löschmaske.	ETTLOUTC1=01	
	?INPUTS	Aktuellen Zustand der Eingänge auslesen (16-Bit-Binärzahl).	?INPUTS	0010100100101101
	?INPTTL	aktuellen Zustand der TTL-Eingänge auslesen (8-Bit Binärzahl).	?INPTTL	00110011
	?INPSPS	aktuellen Zustand der SPS-Eingänge auslesen (8-Bit Binärzahl).	?INPSPS	00110011
	OUTPUT<uv>=<uv>	Aktuellen Zustand eines Ausgangs ändern.	OUTPUT1=0	
	?OUTPUTS	Aktuellen Zustand aller Ausgänge auslesen.	?OUTPUTS	0010100100101101
	OUTTTL<uv>=<uv>	Aktuellen Zustand eines TTL-Ausgangs ändern.	OUTTTL1=0	
	?OUTTTL	Aktuellen Zustand aller TTL-Ausgänge auslesen.	?OUTTTL	00101001
	OUTSPS<uv>=<uv>	Aktuellen Zustand eines SPS-Ausgangs ändern.	OUTSPS1=0	
	?OUTSPS	Aktuellen Zustand aller SPS-Ausgänge auslesen.	?OUTSPS	00101001
	INMODE=<uv>	Eingangspegel TTL/SPS umschalten (0 = TTL, 1 = SPS).	INMODEO	
	?INMODE	Aktuellen eingestellten Eingangspegel TTL/SPS abfragen.	?INMODE	0
	?ANIN<uv>	Analog-Eingang abfragen, angegeben wird die Kanal-Nummer von 1 bis 8, zurückgegeben wird der gewandelte 10-Bit Wert.	?ANIN3	234
	DAOUT<uv>=<uv>	Analog-Ausgang setzen, angegeben wird die Kanal-Nummer von 1 bis 8 und der Ausgabewert für den DA-Wandler.	DAOUT2=250	
	?DAOUT<uv>	Analog-Ausgang abfragen, angegeben wird die Kanal-Nummer von 1 bis 8, zurückgegeben wird der zuletzt eingestellte Digital-Wert.	?DAOUT2	250
	OPWM<uv>=<uv>	PWM-Ausgang setzen, angegeben wird die Kanal-Nummer von 1 bis 4 und der Aussteuerungswert von 0 bis 100%.	OPWM1=55	
	?OPWM<uv>	PWM-Ausgang abfragen, angegeben wird die Kanal-Nummer von 1 bis 4 und zurückgegeben wird der zuletzt eingestellte Aussteuerungswert von 0 bis 100%.	?OPWM1	55
	AXOUTPUT<n>=<uv>	AxisOut-Pin einer Achsen auf High/Low setzen.		
	?IOCONFIG	aktuell eingestellte IO-Konfiguration auslesen.	?IOCONFIG	15
Nachlaufregelung	?APWMS<n>	Aktuelle Position des Wegmesssystems einer Achse auslesen.	APWMS4	3000
	WMSRES<n>	Aktuelle Position des Wegmesssystems einer Achse auf 0 setzen (wird nicht benötigt bzw. darf nach erfolgter Referenzierung nicht mehr benutzt werden, da sonst die Position verloren geht).	WMSRES4	
	?MXWMSSTRK<n>	Den nach Referenzfahrt mit Modus 6 oder 7 ermittelten maximalen Gesamthub in Inkrementen des Wegmesssystems abfragen.	?MXWMSSTRK2	
	WMSFAKZ<n>=<uv>	Faktor für die Positionierung mit Nachlaufregelung Zähler setzen.	WMSFAKZ1=1	
	?WMSFAKZ<n>	Faktor für die Positionierung mit Nachlaufregelung Zähler abfragen.	?WMSFAKZ1	1
	WMSFAKN<n>=<uv>	Faktor für die Positionierung mit Nachlaufregelung Nenner setzen.	WMSFAKN1=5	
	?WMSFAKN<n>	Faktor für die Positionierung mit Nachlaufregelung Nenner abfragen.	?WMSFAKN1	5
	PWMSSET<n>=<sv>	Zielposition bzw. Relativweg (Vorwahl erfolgt, analog zur normalen Positionierung ohne Nachlaufregelung, über die Kommandos ABSOL bzw. RELAT) für eine Achse setzen; ist die absolute Positionsangabe eingeschaltet, so wird der Parameter als absolute Position mit Vorzeichen interpretiert, ist relative Positionsangabe gewählt, so wird der Parameter als Weg mit Vorzeichen interpretiert. Die neue absolute Zielposition berechnet sich dann aus der letzten absoluten Zielposition plus Weg.	PWMSSET2= 100000	
	?PWMSSET<n>	Zielposition bzw. Relativweg für eine Achse auslesen.	?PWMSSET2	100000
	PWMSGO<n>	Positionierung mit WMS bei einer Achse starten, die Achse fährt die neue Zielposition entweder im Trapez- oder S-Kurven-Profil an (siehe PMOD).	PWMSGO2	

Befehls-gruppe	Kommando	Funktionsbeschreibung	Beispiel	Antwort
Nachlaufregelung	PWMSWIN<n>=<uv>	Halbe Zielfensterbreite für die Positionierung mit WMS einstellen (gesamte Breite des Zielfensters = ±PWMSWIN).	PWMSWIN1=10	
	?PWMSWIN<n>	Halbe Zielfensterbreite für die Positionierung mit WMS abfragen.	?PWMSWIN1	10
	PWMSMODE<n>=<uv>	Positioniermodus für die Positionierung mit WMS einstellen. Modus 0 : Nur Grobpositionierung mit Phase 1, Iteration durch mehrmaligen Aufruf. Modus 1: Grobpositionierung Phase 1 und Iteration mit Phase 2 Modus 2: Grobpositionierung Phase 1 und Iteration mit Phase 2 und Korrekturfahrt mit Phase 3, Phase 3 bleibt aktiv und muss mit PWMSSTP vor nächster Positionierung beendet werden. Modus 3: Grobpositionierung Phase 1 und Korrekturfahrt mit Phase 3, Phase 3 bleibt aktiv und muss mit PWMSSTP vor nächster Positionierung beendet werden. Modus 4: Grobpositionierung Phase 1 und Iteration mit Phase 2 und Korrekturfahrt mit Phase 3, Phase 3 wird bei Erreichen des Zielfensters beendet. Modus 5: Grobpositionierung Phase 1 und Korrekturfahrt mit Phase 3, Phase 3 wird bei Erreichen des Zielfensters beendet. Modus 6: Grobpositionierung (Phase 1), Iteration (Phase 2), Phase 3 mit Hybrid-Nachführung, Phase 3 wird im Zielfenster beendet. Modus 7: Grobpositionierung (Phase 1), Iteration (Phase 2), Phase 3 mit Hybrid-Nachführung, Phase 3 bleibt aktiv. Modus 8: Grobpositionierung (Phase 1), Korrekturfahrt (Phase 2), Phase 3 mit Hybrid-Nachführung, Phase 3 wird im Zielfenster beendet. Modus 9: Grobpositionierung (Phase 1), Korrekturfahrt (Phase 2), Phase 3 mit Hybrid-Nachführung, Phase 3 bleibt aktiv.	PWMSMODE1=6	
	?PWMSMODE<n>	Positioniermodus für die Positionierung mit WMS abfragen.	?PWMSMODE1	6
Nachlaufregelung	WMSVEL<n>=<uv>	Nachfahrgeschwindigkeit beim Positionieren mit WMS einstellen (ohne Vorzeichen).	WMSVEL1=100	
	?WMSVEL<n>	Nachfahrgeschwindigkeit beim Positionieren mit WMS auslesen.	?WMSVEL1	100
	PWMSSTP<n>	Positionierung mit WMS bei einer Achse stoppen, befindet sich die Achse beim Positionieren mit WMS in Phase 3, so muss vor dem Verfahren der Achse mit einem neuen Befehl diese Betriebsart mit diesem Befehl beendet werden.	PWMSSTP1	
	?PWMSSTATE<n>	Zustand beim Positionieren einer Achse mit WMS auslesen Bit 0: Axse positioniert mit WMS Bit 1: Achse positioniert mit WMS und ist in Phase 1 Bit 2: Achse positioniert mit WMS und ist in Phase 2 Bit 3: Achse positioniert mit WMS und ist in Phase 3 Bit 4: Achse ist im vorgegebenen Zielfenster	?PWMSSTATE1	16
	?PWMSERR<n>	Auslesen des aktuellen Positionsfehlers einer Achse beim Positionieren mit WMS.	?PWMSERR1	0
	WMSINV<n>=<uv>	Zählrichtung des WMS invertieren (1=ja / 0=nein).	WMSINV1=0	
	?WMSINV<n>	Auslesen, ob Zählrichtung WMS invertieren (ja/nein).	?WMSINV1	0
	WMSOFFS<n>=<sv>	Positionsoffset mit Vorzeichen für die Grobposition bei Piezo-Positionierung mit WMS einstellen.	WMSOFFS1=-80	
	?WMSOFFS<n>	Positionsoffset mit Vorzeichen für die Grobposition bei Piezo-Positionierung mit WMS abfragen.	?WMSOFFS1	80
	PWMSPWIN<n>=<uv>	Zielfensterbreite für die Feinpositionierung mit WMS und Piezo einstellen (Phase 3).	PWMSPWIN1=0	
	?PWMSPWIN<n>	Zielfensterbreite für die Feinpositionierung mit WMS und Piezo abfragen (Phase 3).	?PWMSPWIN1	0
	?PVOLTG<n>	Anfragen des aktuellen Piezo-Ausgangswertes bei WMS-Positionierung mit Piezo.	?PVOLTG1	487

Befehls-gruppe	Kommando	Funktionsbeschreibung	Beispiel	Antwort
Nachlaufregelung	?DACINPUTS	Hybrid-Fehlerstatus abfragen (als Bitmuster). Bit 0: Fehler, Hybrid-Achse 1 Bit 1: Fehler, Hybrid-Achse 2 Bit 2: Fehler, Hybrid-Achse 3 Bit 3: Fehler, Hybrid-Betriebsspannung 1 Bit 4: Fehler, Hybrid-Achse 4 Bit 5: Fehler, Hybrid-Achse 5 Bit 6: Fehler, Hybrid-Achse 6 Bit 7: Fehler, Hybrid-Betriebsspannung 2	?DACINPUTS	11110001
	PWMSPTIM<n>=<uv>	Zykluszeit der Hybrid-Positionierung einer Achse mit Nachlaufregelung einstellen.	PWMSPTIM1=1	
	?PWMSPTIM<n>	Zykluszeit der Hybrid-Positionierung einer Achse mit Nachlaufregelung abfragen.	?PWMSPTIM1	1
	PWMSPMXO<n>=<uv>	Maximalen Hybrid-Ausgabewert einer Achse setzen.	PWMSPMXO1=4095	
	?PWMSPMXO<n>	Maximalen Hybrid-Ausgabewert einer Achse abfragen.	?PWMSPMXO1	4095
Haltebremsenansteuerung	HBCH<n>=<uv>	PWM-Ausgang für Haltebremse einer Achse zuordnen: <Achsennummer> = <PWM-Kanal> PWM-Kanal = 0 für Haltebremsenfunktion aus.	HBCH8=3	
	?HBCH<n>	Zuordnung Haltebremse PWM-Kanal einer Achse abfragen.	?HBCH8	3
	HBFV<n>=<uv>	Ersten PWM-Wert (zum Anziehen) bei der Ansteuerung der Haltebremse einstellen: <Achsennummer> = <Prozentwert>.	HBFV8=50	
	?HBFV<n>	Ersten PWM-Wert (zum Anziehen) bei der Ansteuerung der Haltebremse abfragen	?HBFV8	50
	HBSV<n>=<uv>	Zweiten PWM-Wert (zum Halten) bei der Ansteuerung der Haltebremse einstellen: <Achsennummer> = <Prozentwert>.	HBSV8=20	
	?HBSV<n>	Zweiten PWM-Wert (zum Halten) bei der Ansteuerung der Haltebremse abfragen.	?HBSV8	20
	HBTI<n>=<uv>	Zeit für ersten PWM-Wert bei der Ansteuerung der Haltebremse einstellen: <Achsennummer> = <Zeit für ersten PWM-Wert in ms>	HBTI8=300	
	?HBTI<n>	Zeit für ersten PWM-Wert bei der Ansteuerung der Haltebremse abfragen	?HBTI8	300
Reset	RESETAC	Reset Antriebsplatinen auslösen.	RESETAC	
	RESETMB	Reset Hauptplatine auslösen.	RESETMB	
Stand-Alone-Programmierung	SAMEM	Merker-Wert setzen.	SAMEM38=50	
	?SAMEM	Marker-Wert abfragen.	?SAMEM38	50
	SAEXEC	Stand-Alone-Programm-Ausführung starten (1) /stoppen (0).	SAEXEC0	
	SASTEP	Eine Stand-Alone-Programm-Zeile ausführen, der Zeilen-Index wird übergeben und zurückgegeben wird der Zeilen-Index der nächsten Zeile.	SASTEP1	2
	SALOAD	Eine Stand-Alone-Programm-Zeile laden, übergeben wird der Zeilen-Index und der Inhalt der Programm-Zeile (16 Byte) als Hex-Dump im ASCII-Format.	SALOAD11=04000015F900...	
	SACHKS	Checksumme über das Stand-Alone-Programm aktualisieren, nachdem ein neues Stand-Alone-Programm geladen wurde.		
Anybus®-Modul	ABNETADR=<uv>	Feste Anybus-IPv4-Adresse setzen als Dezimalzahl Aktiv nach globalem Speichern und Neustart.	ABNETADR=168428041 (für 10.10.2.9) ABNETADR=3232235628 (für 192.168.0.108)	
	ABNETADR1=<uv>	Feste Anybus-IPv4-Adresse setzen. Notation mit Punkten Aktiv nach globalem Speichern und Neustart.	ABNETADR1=10.10.2.9 ABNETADR1=192.168.0.108	

Befehls-gruppe	Kommando	Funktionsbeschreibung	Beispiel	Antwort
Anybus®-Modul	?ABNETADR	Anybus-IPv4-Adresse abfragen als Dezimalzahl. Gibt die zuletzt gespeicherte Adresse zurück.	?ABNETADR	168428041
	?ABNETADR1	Anybus-IPv4-Adresse abfragen. Notation mit Punkten. Gibt die zuletzt gespeicherte Adresse zurück.	?ABNETADR1	10.10.2.9
	ABNETSUB=<uv>	Anybus-IPv4-Netzmaske-Adresse setzen als Dezimalzahl Aktiv nach globalem Speichern und Neustart.	ABNETSUB= 168428032 (für 10.10.2.0)	
	ABNETSUB1=<uv>	Anybus IPv4-Netzmaske-Adresse setzen. Notation mit Punkten Aktiv nach globalem Speichern und Neustart.	ABNET- SUB1=10.10.2.0	
	?ABNETSUB	Anybus-IPv4-Netzmaske abfragen als Dezimalzahl Gibt die zuletzt gespeicherte Adresse zurück.	?ABNETSUB	168428032
	?ABNETSUB1	Anybus-IPv4-Netzmaske abfragen. Notation mit Punkten Gibt die zuletzt gespeicherte Adresse zurück.	?ABNETSUB1	10.10.2.0
	ABNETGW=<uv>	Anybus-IPv4-Gateway-Adresse setzen als Dezimalzahl Aktiv nach globalem Speichern und Neustart.	ABNETGW= 168248287 (für 10.10.2.255)	
	ABNETGW1=<uv>	Anybus-IPv4-Gateway-Adresse setzen. Notation mit Punkten Aktiv nach globalem Speichern und Neustart	ABNETGW1= 10.10.2.255	
	?ABNETGW	Anybus-IPv4-Gateway-Adresse abfragen als Dezimalzahl Gibt die zuletzt gespeicherte Adresse zurück.	?ABNETGW	168248287
	?ABNETGW1	Anybus-IPv4-Gateway-Adresse abfragen. Notation mit Punkten Gibt die zuletzt gespeicherte Adresse zurück.	?ABNETGW1	10.10.2.255
	ABNETDHCP=<uv>	Anybus-DHCP aktivieren (=1) oder deaktivieren (=0) Aktiv nach globalem Speichern und Neustart.	DHCP=0 DHCP=1	
	?ABNETDHCP	Status des DHCP abfragen.Gibt den zuletzt gesetzten Status zurück.	?DHCP	0
	ABNETCOM=<uv>	Anybus-Netzwerk-Kommunikationsparameter einstellen, z.B. die Bitrate.	ABNETCOM=10	
	?ABNETCOM	Anybus-Netzwerk-Kommunikationsparameter abfragen.	?ABNETCOM	10

II Relevanz der Parameter für verschiedene Motortypen

Parameter	DC-Brush	2-Phasen-Schrittmotor Open Loop	2-Phasen-Schrittmotor Closed-Loop	BLDC
MOTYPE	+	+	+	+
AXIS	+	+	+	+
FKP	+	–	+	+
FKD	+	–	+	+
FDT	+	–	+	+
FKI	+	–	+	+
FIL	+	–	+	+
FST	+	–	+	+
MAXOUT	+	+[1)]	+	+
MXPOSERR	+	–	+	+
SMK	+	+	+	+
SPL	+	+	+	+
RMK	+	+	+	+
RPL	+	+	+	+
RVELF	+	+	+	+
RVELS	+	+	+	+
ACC	+	+	+	+
DACC	+	+	+	+
JACC	+	+	+	+
PVEL	+	+	+	+
EDACC	+	+	+	+
FVEL	+	+	+	+
ABSOL	+	+	+	+
RELAT	+	+	+	+
PMOD	+	+	+	+
AMPPWMF	+	+	+	+
MCSTP	–	+	–	–
DRICUR	–	+	+	–
HOLCUR	–	+	+	–
AMPSHNT	+	+	+	+

Parameter	DC-Brush	2-Phasen-Schrittmotor Open Loop	2-Phasen-Schrittmotor Closed-Loop	BLDC
MOTPOLES	–	–	+	+
ENCLINES	–	–	+	+
ELCYCNT	–	–	+	+
BLDCCT	–	–	+	+
PHINTIM	–	–	+	+
PHINAMP	–	–	+	+
ATOT	+	+	+	+
INPOSTIM	+	–	+	+
INPOSWND	+	–	+	+
INPOSMOD	+	–	+	+
HBCH	(+)	(+)	(+)	(+)
HBFV	(+)	(+)	(+)	(+)
HBTI	(+)	(+)	(+)	(+)
HBSV	(+)	(+)	(+)	(+)
JPLAX	(+)	(+)	(+)	(+)
JPLAY	(+)	(+)	(+)	(+)
JPLAZ	(+)	(+)	(+)	(+)
JOYACC	(+)	(+)	(+)	(+)
JVEL	(+)	(+)	(+)	(+)
JZONE	(+)	(+)	(+)	(+)
JZEROX	(+)	(+)	(+)	(+)
JZEROY	(+)	(+)	(+)	(+)
JBUTTON	(+)	(+)	(+)	(+)

\+ benötigt
– nicht benötigt
(+) optional

1) Verwendung ist möglich, jedoch ist darauf zu achten, dass der hier gesetzte Wert größer oder gleich dem maximalen PWM-Wert für DRICUR bzw. HOLCUR ist. Der Ausgang wird auf jeden Fall auf den per MAXOUT definierten Wert begrenzt. Wird ein zu kleiner Wert gewählt, funktioniert der Mikroschrittbetrieb nicht mehr ordnungsgemäß.

III Belegungstabellen

TTL-Ein-/Ausgänge
Pinbelegung des 25-poligen D-Sub (male)

TTL-I/O	Pin
Input 1	16
Input 2	17
Input 3	18
Input 4	19
Input 5	20
Input 6	21
Input 7	22
Input 8	23
Output 1	3
Output 2	4
Output 3	5
Output 4	6
Output 5	7
Output 6	8
Output 7	9
Output 8	10
+5V, max. 300 mA Gesamtstrom	1, 2, 14, 15
GND	11, 12, 24, 25
n. c.	13

SPS-Ein-/Ausgänge
Pinbelegung des 25-poligen D-Sub (female)

SPS-I/O	Pin
Input 1	16
Input 2	17
Input 3	18
Input 4	19
Input 5	20
Input 6	21
Input 7	22
Input 8	23
Output 1	3
Output 2	4
Output 3	5
Output 4	6
Output 5	7
Output 6	8
Output 7	9
Output 8	10
+24V, max. 1000 mA Gesamtstrom	1, 2, 14, 15
GND	11, 12, 24, 25
n. c.	13

Analog-Ein-/Ausgänge
Pinbelegung des 25-poligen D-Sub (male)

Analog-I/O	Pin
Input 1	6
Input 2	5
Input 3	4
Input 4	3
Input 5	10
Input 6	9
Input 7	8
Input 8	7
Output 1	23
Output 2	22
Output 3	21
Output 4	20
Output 5	19
Output 6	18
Output 7	17
Output 8	16
+5V, max. 300 mA Gesamt-strom	1, 2, 14, 15
GND	11, 12, 24, 25
U_{ref} Output 4,096V	13

RS-232
Pinbelegung des 9-poligen D-Sub (female)

RS-232	Pin
CD	1
RX	2
TX	3
DTR	4
GND	5
DSR	6
RTS	7
CTS	8
RI	9

Universal-Motorstecker

Mit dem passenden OWIS® Anschlusskabel werden die OWIS® Positioniereinheiten angeschlossen. Über diesen Anschlussstecker wird der Motor mit Leistung versorgt, die Signale des Encoders, evtl. der Hall-Effekt-Sensoren und der Schalter übertragen, sowie die Motor-Haltebremse, falls vorhanden, gesteuert.

Pinbelegung des 37-poligen D-Sub (female):

	Pin	DC-Motor	Schrittmotor OL	BLDC
Leistung	19	Motor +	Phase 1 +	U
	18	Motor -	Phase 1 -	V
	17	Motor +	Phase 2 +	W
	16	Motor -	Phase 2 -	–

Signale	15	Motorcodierung		
	14	Motorcodierung		
	13	GND		
	12	+ 5V		
	11	Encoder A		
	10	Encoder $\overline{A}$		
	9	Encoder B		
	8	Encoder $\overline{B}$		
	7	Encoder Index		
	6	Encoder $\overline{Index}$		

Schalter + Signale	5	MINSTOP		
	4	MINDEC		
	3	MAXDEC		
	2	MAXSTOP		
	1	GND		
	37	Motorhaltebremse +24V		
	36	Motorhaltebremse -		
	35	(reserviert)		
	34	(reserviert)		
	33	(reserviert)		
	32	(reserviert)		
	31	GND		
	30	+ 5V		
	29	(reserviert)		
	28			Hallsensor A+
	27			Hallsensor A–
	26			Hallsensor B+
	25			Hallsensor B–
	24			Hallsensor C+
	23			Hallsensor C–
	22	+ 5V		
	21	GND		
	20	+24V		

Anschlusskabel

1. Signalkabel mit Gesamtschirm Twisted Pair 8 x 2 x 0,15 mm² und Sternvierer innen, geschirmt, 4 x 0,25 mm²

Paar Nr.	Farbe Ader 1	Farbe Ader 2	Querschnitt
1	rot	blau	0,15 mm²
2	weiß	braun	0,15 mm²
3	grün	gelb	0,15 mm²
4	grau	rosa	0,15 mm²
5	schwarz	violett	0,15 mm²
6	grau/rosa	rot/blau	0,15 mm²
7	orange	orange/schwarz	0,15 mm²
8	transparent	transparent/rot	0,15 mm²
9 a	grün/weiß	grün/braun	0,25 mm²
9 b	gelb/weiß	gelb/braun	0,25 mm²

2. Motorkabel mit Gesamtschirm

Ader Nr.	Farbe	Querschnitt
1	rot	0,6 mm²
2	blau	0,6 mm²
3	weiß	0,6 mm²
4	schwarz	0,6 mm²
5	braun	0,6 mm²
6	rosa	0,5 mm²
7	grau	0,5 mm²

Kabelvorschlag für RS-232-Schnittstelle

Zur Herstellung einer Kommunikationsverbindung mit einem PC wird ein Standardkabel mit 1:1 Verdrahtung verwendet.

Universal Position Control Unit

PS 90+

9013.0013 / 31.07.2020

OWIS GmbH
Im Gaisgraben 7
79219 Staufen i. Br. (Germany)
Tel. +49 (0) 76 33 / 95 04-0
Fax +49 (0) 76 33 / 95 04-440
info@owis.eu
www.owis.eu
mobile.owis.eu

1. General information

The PS 90+ is an universal position control unit, to be used for complex positioning tasks.

It is modularly designed and flexibly configurable according to the corresponding range of applications.

It is a powerful device for control of nine axes maximum, which is able to drive step motors as well as DC or BLDC motors. When driving nano-hybrid stages the maximum is six axes.

The control is mounted in a stable metal housing and can be operated independently (stand-alone), or with a computer.

Several inputs and outputs are integrated, e. g.TTL/SPS/analog and PWM, for the communication with different periphery.

If an increasing precision is required, there is a further input for incremental encoder or linear measuring system available for each axis. Step motors having an additional encoder can be operated in closed-loop mode as well.

For positioning tasks which require highest accuracy up to six nano-hybrid axes can be operated with PS 90+. Hybrid technology combines the advantages of positioning with spindle drive and precision of piezoelectric actuators.

Point-to-point positioning mode with different velocity profiles (triangle/trapezoidal or S-curve) as well as complex continuous path control, like linear or circular interpolation, are possible.

The PS 90+ can autonomously react on a change of states of its inputs and start a motion for instance. It can also set outputs when certain events are happening.

The software OWISoft is included in delivery, too. Thus, the PS 90+ can be configured and operated comfortably. Configurations for OWIS® standard positioning units are stored in OWISoft and can be assigned to the corresponding motor easily. Third-party motors can also be actuated.

2. Setup and Scope of Delivery

The PS 90+ consists of a basic unit for different motor voltages. It is equipped with axes modules, additional functions and connections according to customer's requirements. Upgrading with other axes modules, functions and connections is also possible. The unit is completely assembled and tested by OWIS® and will be supplied ready for installation. The valid firmware for operation is installed. It can be updated, if necessary, through the USB or RS-232 interface.

Following parts are included in delivery:

- PS 90+ in the required motor configuration
- mains cable, 2.5 m length
- USB cable, 2 m length
- CD with software OWISoft tool
- documentation in English/German as PDF version
- short instruction in English/German as print and PDF version
- data sheet in English/German

2.1 Standard

The basic version of the PS 90+ comes with:

- USB port
- RS-232 port
- connection for external emergency-stop button
- 4 inputs for reference and limit switches per axis
- 8 TTL and analog inputs
- 8 TTL and analog outputs
- 8 SPS inputs and outputs
- motor plug D-Sub-37 with additional connections for motor holding brake (option), limit / reference switches and other signals (see pin asigment, p. 75) + up to 3 outputs for motor holding brake, depending on the version

2.2 Accessories

The following accessories are available:

- connecting cable with plug for different positioning systems
- joystick for 3 axes, analog, with 3 m cable
- emergency-stop button with 3 m cable
- up to 4 outputs for motor holding brakes

2.3 Options

The following options can be provided:

- Anybus® interface (Modbus/TCP)

3. Safety

- Read user's manual before using the control unit and keep it available for later use.
- Warnings, safety, and installation instructions must be obeyed.
- Technical specificatios and pin-outs must not be ignored.
- This device must be operated for specified normal use only.
- This device is for indoor use only and must not be used in the open.
- The device must be protected against excess humidiy (80%), shock, and explosive gases.
- This device must only be used by authorised qualified personnel.
- Applicable installation, safety, and accident-prevention regulations must be met.
- This device must be used with its locked metal enclosure.
- Connection and disconnection of periphery, mouting of this device, and change of fuses must only be performed in a de-energiesed state and with disconnected mains.
- Changing of fuses must only be performed by qualified personnell.
- Only specified fuses may be used.
- Non-used slots must be closed-off with their respective covers.
- Prior to opening this device it must be de-energised and disconnected from mains. It must be powered off. The power cord must be disconnected!
- This device produces excess heat (power supply, power amplifiers). Do not cover the heat openings. Keep distance to other objects (min 15 cm).
- Only components and periphery intended for usage with this device as well als cables which are in conformity with applicable regulations and norms may be connected.
- Connections to mains or other harmful potentials must not be attached to this device (with exception of the mains plug).
- Any kind of liability for damage which is the result of inobservance of those safety remarks is excluded.

The control unit weighs about 15 kg depending on configuration. At the bottom of the front side there is a recessed grip and at the top of the back a handle to carry the control unit.

The control unit is designed for an operating temperature range from + 10 up to + 40 °C, and storage temperature from - 10 up to + 50 °C.

The PS 90 has connection for an emergency-stop button. Its function follows the EN 418. This button interrupts the power supply of the motor output stages on the secondary side (safety-low voltage range 24 V or 48 V, respectively).

Furthermore, the motor type attached to a motor power stage is recognized over a coding resistor. Thus, it helps to avoid motor damage if a wrong motor type has been connected (e.g., a DC motor to a step motor output stage). The respective control axis modules are only intended to be operated

as they were preconfigured with the motor power stage. Other or related uses are not the intended purpose.

Currents and voltages

The switch-mode power supply of the PS 90+ has a wide range input for a primary stress from 100 VAC to 240 VAC with 50/60 Hz. The power input is protected by a 15 AT (480 W) microfuse. No special safety precautions are necessary for the outputs, as the PS 90 only works with safety-low voltage (PELV) to 48 VDC. If the PS 90+ is equipped to control nano-hybrid axes, voltages between -71 V and +71 V are being used. For special safety notes see chapter "Nano-Hybrid Control".

Heat Sink Temperature up to 65° C max.

The heat generated by the motor driver boards during the operation of the control is dissipated by the laterally attached heat sink.

Depending upon number and size (power input) of the connected motors as well as upon the mode of operation (short time, intermittent or continuous operation), the heat sink might reach a temperature of 65° C max.

Heat accumulation in the control or at the heat sink should be avoided. A minimum distance of 15 cm has to be kept to closed surfaces and walls.

Nonobservance of the safety instructions of this manual may result in material damage as well as damage to persons. Therefore, the manual has to be available and complied with for each user.

The position control unit PS 90 is built in accordance with accepted safety rules and satisfies the following standards and directives.

4. Standards and Directives

Directives:

2014/30/EU (EMC Directive)

Harmonised norms

EN 55011:2016 + A1:2017

EN 61000-6-2:2005

EN 61000-3-2:2014

EN 61000-3-3:2013

2014/35/EU (Low-voltage Directive)

Harmonised norm

EN 61010-1:2010

2011/65/EU (RoHS Directive)

Harmonised norm

EN 50581:2012

5. Technical Overview

Manufacturer	OWIS GmbH
Name	Universal-Positioniersteuerung PS 90+

Input

Mains	IEC power jacket (C20), (max. 3 m cable length, PE connection)
Voltage:	100...240V AC 50/60Hz ±10%
Current	max. 15A
Fuse	internal fuse 15A T

Remark:
Current consumption is dependent on configuration and connected components and periphery.

Operating temperature	+10...40°C, 80% relative humidity, non-condensing
Storage temepra-ture	+10...50°C, 80% relative humidity, non-condensing
IP code	IP20
Place of use	indoors
Max. elevation	Up to max. 2000m
Pollution degree	2 (dry, non-conductive)
Overvoltage category	II
Protection class	I (PE)
Enclosure	Metal housing
Size	Approx. 20x50x50cm WxHxD
Weight	Approx. max. 15 kg, (depending on configuration)

Connections

PC/RS232	D-SUB jack 9 pin. female; max. 3 m cable length, shielded
PC/USB	USB 2.0, USB jack Type-B, female; max. 3 m cable length, shielded
Anybus	Depending on module
STOP	Circular connector, female, max. 3 m cable length, shielded
TTL I/O	D-SUB 37 pin. male; max. 3 m cable length, shielded
SPS I/O	D-SUB 37 pin. female; max. 3 m cable length, shielded
Analog I/O	D-SUB 37 pin. male; max. 3 m cable length, shielded
2x LAN	2x Ethernet, RJ45 jack, LAN cable, CAT5; 10 m cable length, shielded
M1 to M9	Motors, D-SUB 37 pol. female; max. 3 m cable length, shielded, longer cables on request

Operating elements:

on/off

I/0

ATTENTION:
Only use the provided cable for connection to mains.

SAFETY:
For safe disconnection from mains power-off the device and disconnect from mains.

WARNING	Observe safety and installation instructions! Observe pin-outs! Nonobservance may lead to malfunction or permanent damage!
Power supply	Switching power supply with current limit.
Number of drives	Up to 9 axes (SM, DC, andBLDC), up to 6 nano-hybrid axes (SM)
Drive types	2 phase stepper motor open loop (OL), 2 phase stepper motor closed loop (CL), DC servo motor, BLDC servo motor, nano-hybrid axes
communication	USB 2.0, RS-232, optionally Anybus interface (Modbus/TCP)
Setup	Desktop device in metal enclosure, protection level: IP 20
Encoder	Quadrature signals A/B and Index, RS-422 or TTL level, with 4-times interpolation, max. counting frequency 5 MHz (signal) or 8 MHz (quadrature)
Functions	Acceleration/deceleration ramp, triangular or trapeze velocity profile or s-cure
Movement	Point-to-point positioning, linear and circular interpolation

6. Setup of the Control Unit

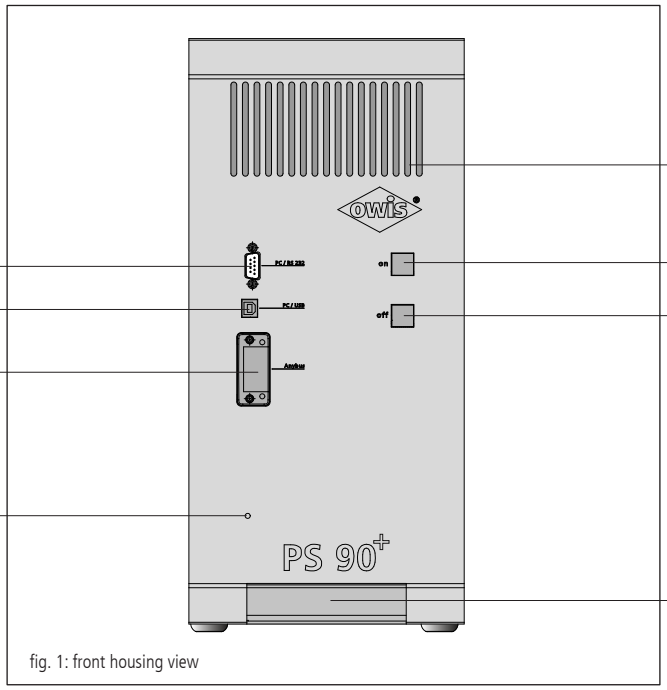

fig. 1: front housing view

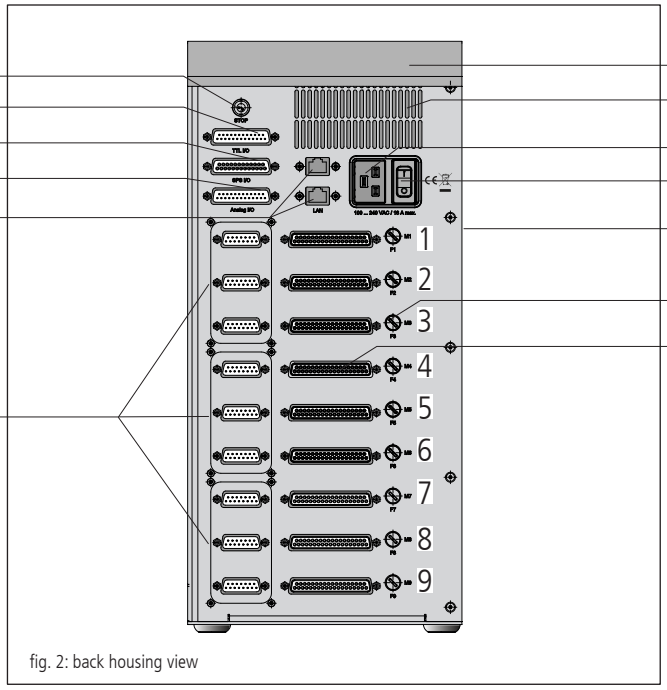

fig. 2: back housing view

The PS 90+ is placed in a top-quality, stable metal housing.

For the cooling of the internal components there are several ventilation slots at the top of the front and at the back side of the housing. The heat generated by the motor power stages during the operation of the control is dissipated by the laterally attached heat sink.

On/Off Switch
The mains switch and the on/off push-buttons are lighted. If the mains switch is lighted, the PS 90+ is activated.

After switching on the current supply of the motor power stages using the "ON" push-button, the buttons are lighted with maximum intensity. When pressing the "OFF" push-button, the motor power stages will be switched off and the backlight of the buttons will be reduced to a minimum.

Reset Button
Should the communication with the PS 90+ fail or should any unexpected errors (e.g., message "M" for motion controller errors) occur, the reset button can be pressed by using a ball-point pen, for example. Alternatively, it is possible to switch the control unit off and then again on.

6.1 Connections

The connections of the PS 90+ are located on the front and rear side
of the housing. These are communication interfaces, in- and outputs for the periphery as well as connections for the positioning units (see figure 1 and 2).

connection	function	socket
USB slave	communication with a PC	USB port type B
RS-232	communication with a PC	D-Sub 9-pin male connector
RS-485	remote operation of the controller with the hand-held terminal	D-Sub 9-pin female connector
Ethernet interface	communication with a PC with ethernet / integrated switch	RJ 45
joystick	manual positioning of max. 3 axes	analog inputs 1, 2, 3
TTL in-/outputs	interaction with external sensors and actuators	D-Sub 25-pin male connector
analog in-/outputs	interaction with external sensors and actuators, joystick	D-Sub 25-pin male connector
SPS in-/outputs	interaction with external (SPS) control	D-Sub 25-pin female connector
universal motor connector	motor supply with motor holding brake, encoder and limit-switch connection	D-Sub 37-pin female connector
linear measuring system/ encoder	connection of the travel measuring system	CONNEI 12-pin circular socket
power supply	AC inlet	IEC appliance connector (for cold condition)

Option:

Anybus®-module Modbus/TCP	communication with a PC via Ethernet	RJ 45

USB and RS-232 Interfases

The PS 90+ has an USB 2.0 slave-interface. Its connector is placed on the rear side of the equipment. The interface is compatible with USB 1.1 and 2.0. The USB interface of the PS 90+ is implemented as a so-called COM bridge. The Windows device driver recognizes the PS 90+ as "USB serial port" and assigns a COM port number to it. This number can be changed by the user, if necessary. After successful installation, the USB interface is addressed as virtual RS-232 interface.

As alternative to the USB interface, the control can communicate with a PC via RS-232.

Both interfaces work with a transfer rate of 9 600, 19 200, 38 400, 57 600 or 115 200 baud. Please make sure that the transfer rate of the PS 90+ corresponds to the transfer rate defined in the device driver, otherwise no communication is possible. Preset is 9 600 baud (can be seen in the acceptance certificate).

Ethernet interface

The PS 90+ hast wo ethernet connections with an integrated switch. Either of those two connections can be used to connect the control unit to the local network. The other is available to connection further devices. Thus, with only one network access point the control unit as well as a PC can be connected. Port 8777 must be used.

Anybus® Interface

Optionally, the PS 90+ can be equipped with an Anybus® module "Modbus/TCP". Using this Anybus® module, it is possible to send commands and to readout answers via Ethernet.

Emergency-Stop Function

On the rear panel, one can find the connector for an external emergency-stop button. If no emergency-stop button is used, a jump plug has to be inserted. If an emergency-stop button shall be connected, the jump plug has to be removed and the button (n.c. contact) has to be connected instead.

> **Note**:
> If the jump plug is removed and no emergency-stop button is connected, the operation of the motor output stages is blocked.

The emergency-stop functionality of the PS 90+ is based on EN 418 and interrupts the supply circuit of the motor output stages on the secondary side (safety-low voltage 24 V or 48 V). The function is implemented by means of a self-holding relay with forcibly actuated contacts (two n.c. contacts in series). When switching off the output stages, their supply is switched off and additionally, the output stages are disabled (dual security).

Power Supply

The switch-mode power supply of the PS 90+ has been designed for an input voltage of 100 VAC to 240 VAC at 50/60 Hz (wide-range input). A switch-mode power supply generates 24 VDC and supplies the outputs and the inputs on the main board. The logic voltages + 5 V, + 2.5 V and + 3.3 V for main board and motor driver boards are generated out of this 24 VDC supply. A second power supply generates the intermediate circuit voltage for the motor driver boards (alternatively 24 or 48 VDC). This voltage supplies the power output stages of the motor driver boards.
The supply voltages for logic and motor power are galvanically separated.

Universal Motor Connector

The positioning units are connected using the suitable OWIS® connecting cable. The universal motor connector enables the current supply of the motor, control of the motor holding brake, where applicable, and the transfer of the encoder or limit-switch signals.

The motor power stage contains an additional protection device which helps to avoid motor damage if a wrong motor type has been connected (e.g., a DC motor to a stepper motor output stage). For detection of the motor type, a coding resistor is provided in the 37-pin D-Sub connector of the motor connecting cable between pin 14 and 15.

Coding:
- 0 Ohm: DC servo motor (brush type)
- infinite resistance (no resistor): 2-phase step motor
- 470 Ohm: BLDC

When being switched on, the PS 90+ measures the resistance value and reports an error message if the measured value does not match the type of the motor power stage. The error message of the output stage can be read out using the command "?ASTAT" and "?MPUNISTAT<n>" (see command set, page 62).

The pin assignment can be seen in attachment. The pin assignment matches the OWIS® standard.

Limit and Reference Switches
Maximum four switches can be connected for each axis. They can be micro switches, TTL Hall switches or TTL light barriers with + 5V voltage. Various n.c. or n.o. contacts, switching towards $+U_b$ or GND, can be attached to the inputs.

Additionally, one of the four switches can be defined as reference switch, if necessary.

The active level and the switch assignment are configured by software.

Encoder Input
The encoder input enables both the connection of encoder with line drivers (antivalent signals for CHA, CHB and optionally index I), and of encoders with TTL/CMOS signals.

The following input signals are defined:

supply voltage	V_{CC} (+ 5V); GND
channel	A (TTL or CMOS)
channel	A inverted
channel	B (TTL or CMOS)
channel	B inverted
channel	I (TTL or CMOS)
channel	I inverted

The conversion of the antivalent signals to TTL signals takes place with RS-422 recievers. If an encoder with TTL/CMOS signals is connected, then the input for the inverted signal remains open and is internally pulled to 1.4V by a high-impedance voltage divider. The conductor paths of the inverted signals have cut-off points on the drive controller boards inside the PS 90+ with soldering jumper pads, in order to allow interruption and reconnection of the inverted signals, if necessary. A pull-up resistor is provided towards + 5V at the non inverted inputs.6.2 Inputs and Outputs

6.2 Inputs and Outputs

For the interaction with external sensors and actuators, corresponding digital and analog inputs and outputs are provided.

Forked light barriers, etc. can be connected to the TTL-compatible inputs.

Using the TTL outputs it is possible to control digital hardware directly in the application setup.

The SPS compatible inputs enable the use of the 24VDC inductive sensors in two-wire and three-wire technology, typically used in mechanical engineering. The load-resistor array of the SPS inputs can be configured as Pull-Up or Pull-Down, using the software.

The SPS outputs control single solenoid valves or other inductive and resistive loads directly (switching towards + 24V). The outputs are short-circuit proof.

features	level	current	others
TTL inputs	0-5V		
SPS inputs	0-24VDC		2-wire/3-wire
analog inputs	0-4.096VDC		resolution 10 bit
TTL outputs	0-5V	10mA	
SPS outputs	0-24VDC	300mA	short-circuit proof
analog outputs	0-4.096VDC	10mA	resolution 10 bit
power outputs	0-24VDC	1.0A	PWM

The analog inputs can measure voltages between 0V and 4.096V directly and convert them into digital values with a resolution of 10 bits (reference voltage: 4.096V). The in- and outputs are not galvanically separated.
The query commands "?ANIN<uv>" and "?INPUTS" correspond to the same inputs of the PS 90+ (see command set, page 62). The evaluation of the inputs takes place either analog or digital.
The four power outputs are PWM-type and switching towards GND. They are designed to drive inductive loads which need a high actuating current for a short time and a low stand-by current afterwards, as holding brakes or solenoids, e.g.

The power outputs can be configured by software especially for driving a motor holding brake.

The emergency-stop functionality of the PS 90+ is based on EN 418 and interrupts the supply circuit of the motor output stages on the secondary side (safety-low voltage 24V or 48V). The function is implemented by means of a self-holding relay with forcibly actuated contacts (two n.c. contacts in series). When switching off the output stages, their supply is switched off and additionally, the output stages are disabled (dual security).

7. Control Unit Architecture and Function

The control unit consists primarily of the following components:
1. an integrated power supply
2. a main board
3. max. 9 drive controller boards
4. max. 9 motor driver boards

7.1 Assembly

Main Board The main board is the core of the PS 90+. It takes over the control of the main process flow, communicates with the PC and with the motor driver boards, governs the digital and analog in- and outputs.
The main board has an USB connection for the communication with a PC. A further serial interface with RS-232 is implemented as alternative command interface to the PC.
With the Anybus®-module "Modbus/TCP" it is possible to communicate with a PC via Ethernet.

Drive Controller Board
Each drive controller board contains a motion processor which can control respectively actuate one axis. The motion processor executes the commands received from the microcontroller and generates the corresponding control signals for the output stage modules. The interface to the output stage modules is galvanically separated by optoelectronic couplers.

Motor Driver Board
The PS 90+ can be equipped with up to nine motor driver boards.

On the motor driver board itself, the H-bridge output stages are implemented. They supply appropriate current levels to the motor coils and control its torque that way.

On the motor driver board the universal motor connector is fitted. On this connector, all the necessary signals, such as motor current, limit switches, encoder, holding brake and Hall-commutation sensors (if any), can be found.

Safety Fuse Concept
There is a separate fuse (5 x 20 mm) for each motor board, rated according to the maximum possible current. It aims at avoiding serious damage or fire hazard in case of a hardware defect. The protection fuse is accessible on the back side of the unit next to each motor connector and can be exchanged easily, if necessary.

Standard fuse protection: 6.3A slow-blow.

Additionally, each motor driver board is equipped with an electronic overcurrent protection. If the phase current exceeds the maximum allowed current the driver board is switched off. Axis release will be removed as well.

7.2 Operation of Different Motor Types

Step Motors
The PS 90+ is designed for the use with 2-phase step motors, which can be operated in open-loop as well as in closed-loop mode.

DC Motors
The PS 90+ can also control brush-type DC servo motors.

The output stage is implemented as an H-bridge with current limiting, addressed with a PWM and a direction signal. An automatic current limiting is built-in, which is activated before the motor maximum current is being exceeded.

BLDC Motors
Three-phase BLDC motors (brushless servo motors) can also be controlled.

The output stage controls three motor coils by means of three 50/50-PWM signals, generated by the motion processor. Each current value of the three half-bridge sections is monitored.
To avoid overcurrent, the current is limited by means of a chopper.

7.3 Settings of the Motor Output Stage

The output stages are set to a fixed motor type as a factory setting. This cannot be changed by the user. The possible configurations are described as follows:

2-Phase Step Motor (Open Loop)
For this motor type no current limit can be set. The phase current settings are described in the following section 7.4

Current control is achieved trough a PID control loop. This controller must not be confused with the positioning control even though the used terms are identical.

Four parameters (coefficients P, I quick, I slow, and D) define the controller characteristics. Badly chosen values, usually too lagre, might lead to the motor being very noisy. If the values are chosen too low, the maximum velocity might not be acheived. Each motor type must be configured individually. Typical step motor noises might be greatly reduced by choosing optimal values. Especially at low velocities an exremely silent movement is possbile.

When being delivered together with OWIS stages the PS 90 is preconfigured with appropriate controller settings. Additionally, OWISoft contains pre-defined controller settings which are optimised for either low-noise or high-dynamic operation of the stage.

DC Servo and BLDC Motor
The operation of DC servo and BLDC motors usually uses a motor current limitiation. This is done by using DRICUR (see command set). The current limit is valid after a cold start and initialisiation of the controller. To re-set the limit a cold start is necessary.
CRICUR<n>=100 relates to 100% of 12 A. Values need to be set lower accordingly.
If this limit is set too low the resulting dynamic of the stage is limited. This is due to short peak currents during the acceleration or deceleration of DC motors. Normally, those peak currents do not pose any harm to the motor.

7.4 Selection of the Current Range for the Motor Power Stage

The PS 90+ motor power stage has two configurable current ranges in order to obtain a high precision in the current setting respectively a micro step resolution at its best.

The current range selected will be stored. In order to activate the new selected current range, it is necessary to re-initialize the axis <n> after the preset has been done.

Preselection of the current range 2 (high current) for axis <n> takes place after e.g. following command sequence:

AMPSHNT<n>=1

INIT<n>

In order to switch back to current range 1 (low current) one may use e.g. following command sequence:

AMPSHNT<n>=0

INIT<n>

Phase Current Setting for 2-Phase Step Motors

Driving and holding current can be separately preset with 2-phase step motors. The selection for axis <n> can be done as in the following description. The value <uv> is defined as integer percentage of the maximum current in the pre-selected current range (1 or 2).

driving current: DRICUR<n>=<uv>

holding current: HOLCUR<n>=<uv>

maximum phase current, current range 1

(corresponding to 100%): 2.4 A

maximum phase current, current range 2

(corresponding to 100%): 5.45 A

Note:
A phase current of 3.6 A max. in current range 2, corresponding to 66 % of the maximum value that can be preset, should not be exceeded.

In general, the lowest-possible current range should be selected, in order to obtain the optimal precision in high-resolution micro step operation.

Current Range Setting for DC Servo and BLDC Motors

The suitable current range for the DC servo and BLDC motors has to be set in accordance with the thermally admissible continuous current of the corresponding motor type. A current limiting can be configured by setting the according parameters. (For further information please see chapter "Settings of the Motor Output Stage".)

8. Control Functions

8.1 Trapezoidal Point-to-Point Profile

The following table contains the specific profile parameters for the trapezoidal point-to-point mode:

profile parameters	format	word length	range
position	32.0	32 bit	-2.147.483.648…+2.147.483.647 counts
velocity	16.16	32 bit	(1…2.147.483.647)/65.536 counts/cycle
acceleration	16.16	32 bit	(1…2.147.483.647)/65.536 counts/cycle²
deceleration	16.16	32 bit	(1…2.147.483.647)/65.536 counts/cycle²

For this profile, them host specifies an initial acceleration and deceleration, a velocity and a destination position. The profile is named after the curve shape (fig. 9, 11): the axis accelerates linearly (on the basis of the programmed acceleration value), until it reaches the programmed speed. Afterwards, the axis slows down linearly (using the deceleration value), until it stops at the defined position. If the programmed travelling distance is so short that deceleration must begin before the axis reaches the programmed velocity, the profile will not have a constant-velocity range, and the trapeze becomes a triangle (fig. 10).

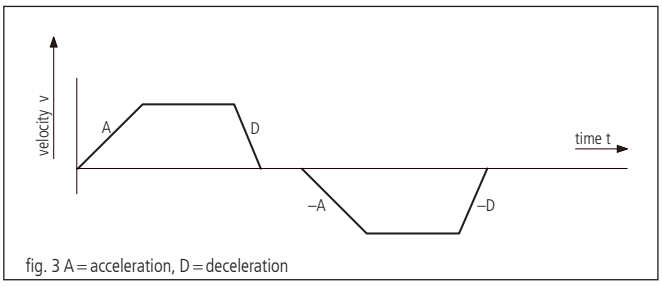

fig. 3 A = acceleration, D = deceleration

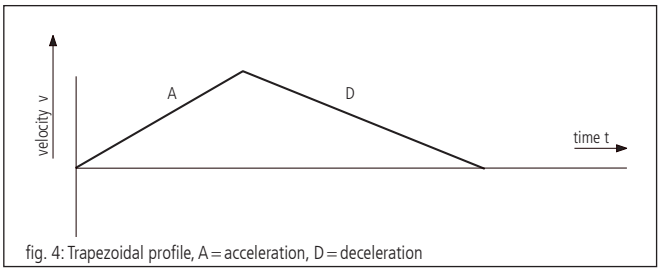

fig. 4: Trapezoidal profile, A = acceleration, D = deceleration

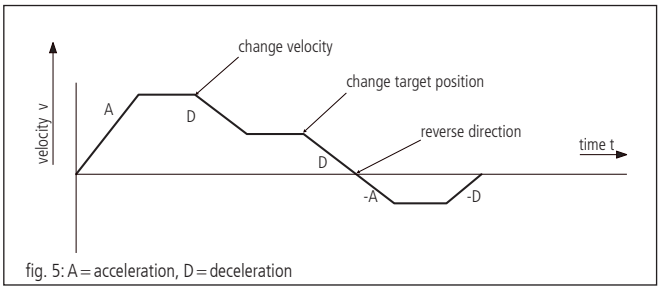

fig. 5: A = acceleration, D = deceleration

The acceleration and deceleration ramps can be symmetrical (if acceleration equals deceleration) or asymmetrical (if acceleration does not equal deceleration).

The acceleration parameter is always used at the beginning of the movement sequence. Afterwards, the value for acceleration is used in the same direction, and the value for deceleration is used in opposite direction. If no motion parameters are changed during the motion sequence, then the acceleration value is used, until the maximum velocity was reached. The deceleration value is used, until the velocity drops to zero.

It is possible to change one of the profile parameters while the axis is in this profile mode. The profile generator will always try to execute the movement within the set conditions given by the parameters. If the end position is changed during the movement so that the remaining travel distance changes sign, the PS 90+ will decelerate to stop and then accelerate in reverse direction to move to the specified target position.

8.2 S-Curve Point-to-Point Profile

The following table presents all the profile parameters for the S-curve point-to-point mode:

profile parameters	format	word length	range
position	32.0	32 bit	-2.147.483.648…+2.147.483.647 counts
velocity	16.16	32 bit	(1…2.147.483.647)/65.536 counts/cycle
acceleration	16.16	32 bit	(1…2.147.483.647)/65.536 counts/cycle²
deceleration	16.16	32 bit	(1…2.147.483.647)/65.536 counts/cycle²
jerk	0.32	32 bit	(1…2.147.483.647)/4.294.967.296 counts/cycle³

The S-curve point-to-point profile adds a limit to the relation of the acceleration change, in comparison with the basic trapezoidal profile A new parameter ("Jerk") specifies the maximum acceleration change within an update cycle.

In this profile mode, the acceleration increases gradually from "0"

to the programmed value, then the acceleration decreases proportionally, until it reaches "0" with the programmed end speed. The same sequence will be implemented reversely, in order to reach the end position.

Within the S-curve profile mode, the same value must be used for both the acceleration and the deceleration ramp. Asymmetrical profiles are not allowed. This is only possible in trapezoidal profiling mode.

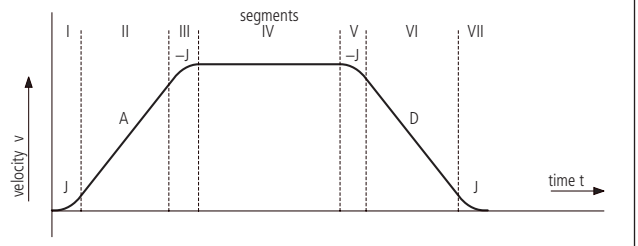

fig. 6: S-curve profile, A = acceleration, D = deceleration J = jerk

Fig. 6 shows a typical S-curve profile. In segment I, the acceleration value increases by the value set by the jerk, until the maximum acceleration is reached. The axis continues accelerating linearly (jerk = 0) within segment II. The profile uses then the negative value of the jerk in segment III in order to reduce acceleration. In segment IV the axis moves with maximum (programmed) speed (V). Then, the profile slows down similarly to the acceleration value, by using the negative jerk in opposite direction, in order to first reach the maximum acceleration (A) and then to halt the axis at the end position.

It is possible that a S-curve profile only contains some of the segments shown in fig. 12. This can e.g. be the case, if the maximum acceleration cannot be reached before "half a way" in direction end velocity or end position. This profile does not contain segments II and VI (see fig. 7).

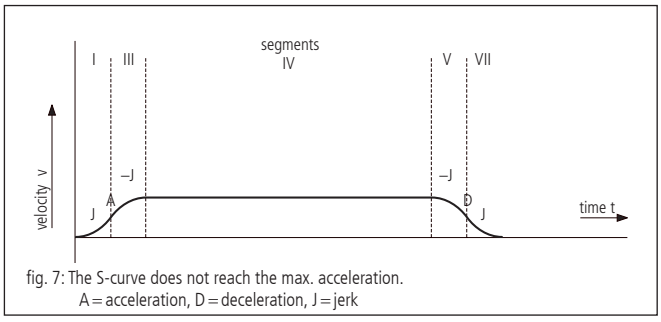

fig. 7: The S-curve does not reach the max. acceleration.
A = acceleration, D = deceleration, J = jerk

If a position is defined in such a way that the end acceleration cannot be reached, then there is no segment IV (see fig. 8).

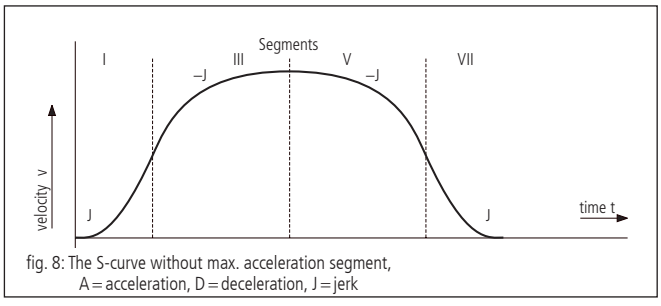

fig. 8: The S-curve without max. acceleration segment,
A = acceleration, D = deceleration, J = jerk

Contrary to the trapezoidal profiling mode, the S-curve profiling mode does not permit changes of any profiling parameters while the axis is in motion. Similarly, the axis may not be switched into the S-curve mode while it is in motion. However, it is allowed to switch from the S-curve mode to another profiling mode during the motion.

8.3 Velocity Mode

The following table presents the profile parameters for the velocity mode:

profile parameters	format	word length	range
velocity	16.16	32 bit	(-2.147.483.648…+2.147.483.647) /65.536 counts/cycle
acceleration	16.16	32 bit	(1…2.147.483.647)/65.536 counts/cycle²
deceleration	16.16	32 bit	(1…2.147.483.647)/65.536 counts/cycle²

Unlike in trapezoidal and S-curve profiling modes, where the final position determines whether positive or negative speed is defined, it is the sign of the velocity value transmitted within the velocity mode that determines whether the axis moves in positive or negative direction. Therefore, the velocity value sent to the PS 90+ can take positive values (for positive direction of motion) or negative values (for reverse direction of motion). For this profile no destination position is specified.

The trajectory is executed by continuously accelerating the axis at the specified rate until the corresponding end velocity is reached. The axis begins to slow down, if a new velocity is defined which value is smaller than the current velocity or if it has another sign than indicated by the current direction.

A simple velocity profile looks like a simple trapezoidal point-to-point profile as shown in fig. 3.

Fig. 9 shows a more complicated profile, in which both speed and direction of motion change twice.

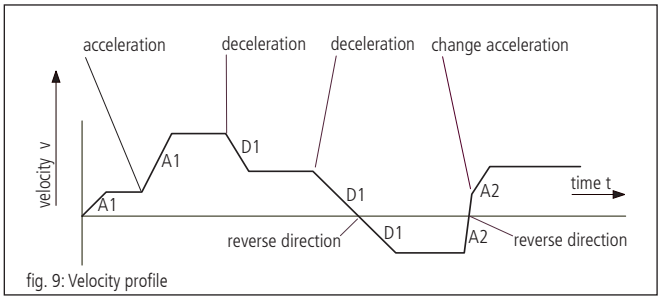

fig. 9: Velocity profile

! Note:
In the velocity mode, the axis movement is not bound to a final position. It is up to the user to select such velocity and acceleration values which guarantee a safe course of motion.

8.4 Reference run

The reference move drives onto one of the four limit switches. The position can be zeroized at this point. Therefor, two reference driving speeds with amount and sign and a reference acceleration are parameterised. The limit switch is approached with high speed and left with a low, then it is stopped.

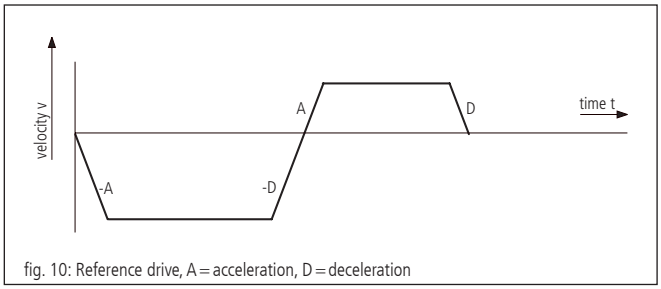

fig. 10: Reference drive, A = acceleration, D = deceleration

8.5 Operating Mode of Linear Interpolation

Definition

Linear interpolation designates here the synchronisation of the movement of all axes involved in such a manner that the axes start quasi-simultaneously and reach their target positions practically at the same time. The motion takes place here by means of trapezoidal velocity profiles, whereby acceleration and brake ramps are modulated in such a way that all axes accelerate and/or brake likewise synchronously. The motion of a XYZ linear axis actuated by linear interpolation, describes thus in the cartesian coordinate system approximately a straight line in space.

The axis with the lowest axis number, which has to pass the longest traverse path (converted into increments), is called guiding axis f. On this axis the remaining axes taken part in the linear interpolation are synchronized within the control by software.

Functional Principle

Which of the maximally 9 axes are involved in the linear interpolation, is indicated by a binary code at the start of the axes. A set bit means here that the appropriate axis is active.

For each axis a maximum speed as well as a maximum acceleration value must be defined before use of the linear interpolation, which should not be exceeded during the positioning procedure. The velocity-time profile of an linear-interpolated motion sequence is symmetrical.

With consideration of the digital system time (sample time and/or cycle time of the profile generator) for each axis the maximum values are converted so that the guiding axis f reaches the target position as fast as possible (with maximum possible speed v_{max} (f) and acceleration a_{max} (f)). The remaining axes are synchronised to the guiding axis, whereby the given limit values of the control should be complied with.

The linear interpolation axes are designated in the following by (i). The following diagram shows the process in principle of the speed profile of the guiding axis v_f (t) and any linear interpolation axis v_n (t) by an example:

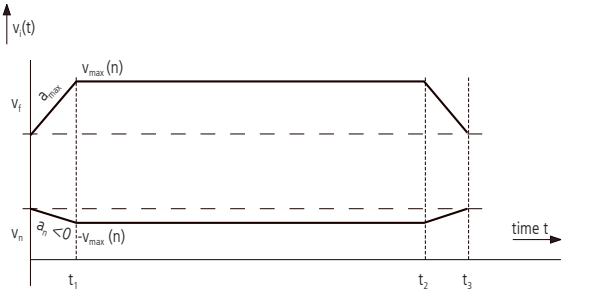

fig. 11: Linear interpolation

In the example the driving distance of the axis (n) is negative, the driving distance of the guiding axis (f) is positive. At the time t_1 the acceleration phase is completed. The brake actuation starts with t_2, and all axes stop together at the time t3.

8.6 Synchronous Start

Similar to linear interpolation it is possible to start either a positioning or velocity mode of multiple axes in a synchronous manner.

By using the appropriate commands (see command set) alle neccessary computations are done internally proior to starting the axes. Then, all included axes start virtually at the same time. In contrast to linear interpolation each axis is then performing its movement just as it would do after a single start.

8.7 Operation mode of the General Continuous Path Control

Definition

The PS 90+ enables the approximation of any paths by chains of single
vectors which are passed to the control in a vector table. Therefore, the general continuous path control is realised by a vector mode.

Relative positioning values which should be reached as accurately as possible at determined, discrete points in time are registered in the vector table. Point of reference and/or starting point of the table vectors is the respective current target position of the axes.

The approximated paths are driven in velocity mode with trapezoid profile.

Realisation of Vector Mode

Vector table

Each table entry n defines a complete driving segment and contains the relative path vector for maximum nine axes (a to i, according to the axis numbers 1 to 9), the time interval $\vec{\Delta x}$ given for the path vector contains a 16-bit function code F, an 8-bit error code E and an 8-bit enable axis code T:

n	$\vec{\Delta x}$	Δt	F	E	T
1	$\Delta x_{a1},\ldots,\Delta x_{i1}$	t_1	t_1	e_1	t_1
...	...	...	...	...	...
N	$\Delta x_{aN},\ldots,\Delta x_{hiN}$	Δt_N	f_N	e_N	t_N

Maximum 4000 vectors can be defined ($N_{max} = 4000$).

The elements of the motion vector (single distances) are represented as integral signed values (integer 32 bit). The maximum path distance for a time interval Δt_n is 2147483648 increments, i.e. for the range of values of a position entry numerical values from - 2147483648 to + 2147483648 are permissible.

Segment duration

The time interval Δt_n for the driving segment <n> is indicated as integral multiple of 1.024 ms. The values facet range from 20 to 65535, out of this a definable segment time of minimum 20.48 ms to maximum 67.10784 s in steps of 1.024 ms results:

$\Delta t_{n_{min}} = 20 \cdot 1.024\,ms = 20.48\,ms$

$\Delta t_{n_{max}} = 65635 \cdot 1.024\,ms = 67.10784\,s$

Control codes

All codes used here (F, E and T) are in principle binary codes, which are basically represented as positive integral values (Integer) and transferred to the control, independent of the terminal mode preselected by "TERM=… ".

The function code F is represented as a 16 bit value. Bit 15 is used to select the mode of operation, i.e.

"constant velocity" (v=const., bit 15 deleted) or

"constant acceleration" (a=const., bit 15 set).

The remaining bits are used to either set or delete up to three outputs per line. Bits 0 to 3 binary select the output. Bit 4 decides whether or not the respective output is set or deleted. This selection scheme repeats itself for bits 5 to 9 and 10 to 14.

Output selections between 1 and 8 correspond to TTL outputs 1 to 8. SPS outputs 1 to 7 are selected by setting 9 to 15. If the output is set to zero no action will occur.

The standard function code is 0, corresponding to "constant velocity" and no output actions.

The 8-bit error code E indicates whether and if at which of the maximum eight axes active in the vector mode an error occured during the plausibility check of the vector table. Here, a set bit 0 indicates an error at axis 1, a set bit 1 an error at axis 2 etc.

The 8-bit enable code T defines which of the axes 1 to 8 is active in the vector mode. The allocation of the single bits to the axis number corresponds to the error code E, i.e. a set bit 0 stands for axis 1 is active etc.

Operating methods

The following diagrams illustrate both modes of operation, preselectable by the function code F, on the basis of the velocity-time shape. The time intervals of the five represented path segments are marked with "Δt_1" to "Δt_5", the velocity values at the end of the respective segment with "v_1" to "v_5"and the acceleration values with "a_1"to "a_5".

Velocity-time-diagram for mode of operation v=const. (example):

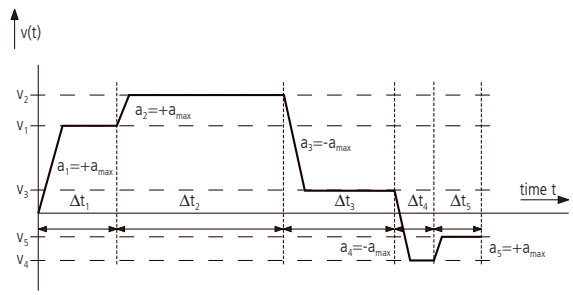

fig. 12: Velocity-time-diagram for mode of operation v=const.

The motion velocity is changed in the constant speed mode with the given max. acceleration and remains constant thereafter. It is cyclically recalculated for each segment during the processing of the vector table. A possibly occuring position deviation at the end of a segment is considered in the following segment as correction value, in order to avoid an accumulation of the positioning error.

Velocity-time-diagram for mode of operation a=const. (example):

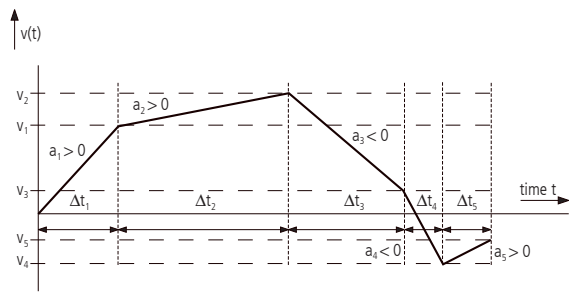

fig. 13: Velocity-time-diagram for mode of operation a=const.

The driving velocity continously changes in the constant acceleration mode. The acceleration value is constant within a segment for each axis. End velocity and acceleration within the segment are cyclically recalculated for each segment during the processing of the vector table. A possibly occuring position deviation at the end of a segment is considered in the following segment as correction value, similarly to the mode of operation v=const.

Maximum velocity and -acceleration

The maximum permitted velocity and/or acceleration in the vector mode is set for each axis separately using the commands "IVEL" and "IACC". These limits are valid likewise for the vector mode and for the operation with linear interpolation.

Plausibility check

By the command "PTABPLAUS" a vector table can be checked for plausibility. If the given target position of an axis could only be reached by exceeding the given velocity or acceleration limits, the appropriate bit in the error code E is set for the concerned axis.

Set error bits are ignored during positioning procedure and only serve for the information of the user. The table entry can also be executed if E is not equal to zero, however, it causes a very high positioning error.

Axis enabling

For each axis active within a motion segment a bit must be set in the enable code T. Axes with deleted bit are not considered in the motion vector and/or are not braked with the programmed max. acceleration to zero velocity, if the current motion velocity should not be equal zero.

Syntax

The table entry <n> is generated by the command "POSTAB"and transferred to the control. The syntax is as follows:

POSTAB <n> = Δx_{an},..., Δx_{in}, Δt_n,f_n, e_n, t_n

Zero should always be passed as value for the error code E, so that possibly set error bits are deleted.

The plausibility check for the motion segments <n> up to the end of the table is done by

PTABPLAUS <n>.

Here, for all active axes of each segment the velocity and / or acceleration values are calculated and the adherence to the set limit values is checked. In case of an error the appropriate bit for the axis is set in the error code E. The calculated velocity and acceleration value (Vel_i and Acc_i) for the segment <n> of the last active axis <i> (i.e. active axis with the highest axis number <i>) are stored to control purposes in the table as well and can be read out by using "? POSTAB". Both control values serve for debugging in particular and/or extended plausibility check of motion segments with a single active axis.

?POSTAB <n>

returns as answer:

Δx_{an},..., Δx_{in}, Δt_n,f_n, e_n, t_n Vel_i, Acc_i

Example

The following example is to illustrate the fundamental functions for the creation of the table entries. It´s given:

Segment time about 100 ms

Active axis for path control: axes 1, 2, 3

Velocity limits axis 1, 2, 3: 800000, 500000, 300000

Acceleration limits axis 1, 2, 3: 2000, 4000, 10000

Driving distances axis 1, 2, 3 (relative, in increments): 1000, -500, 2000

Operation mode a=const.

Calculation of the standardised segment time Δt_0 and the enable code t_0:

$$\Delta t_0 = \frac{100\,ms}{1.024\,ms} \sim 98$$

$$t_0 = 2^0 + 2^1 + 2^2 = 7$$

Following commands are to be sent, in order to set velocity and acceleration limits as well as to define the first table entry:

IVEL1=800000

IVEL2=500000

IVEL3=300000

IACC1=2000

IACC2=4000

IACC3=10000

POSTAB0=1000,-500,2000,0,0,0,0,0,0,98,32768,0,7

Plausibility check using

?PTABPLAUS0

and read out the table elements by

?POSTAB0

returns the answer:

1000,-500,2000,0,0,0,0,0,0,98,32768,4,7,668734,1705

The error code "4" indicates that the entry for the third (and last) axis is incorrect. A velocity value of 668734 and an acceleration of 1705 are calculated at a given motion distance of 2000 increments for the axis. The velocity value exceeds the permitted limit value of 300000.

End of motion

After processing the last table entry or at deleted enable bit the no longer active axes brake to velocity zero using the respective maximum acceleration. Afterwards, the velocity mode will be deactivated and the axes be changed from path control check to position holding.

The outcome of this is a follow at ending the path shape curve by a certain distance depending on the initial velocity at the end of the final segment and the maximum acceleration.

Selection of segments

Besides using enable bits to segment the table using the starting command PTABGO can also directly select only a part of the table for execution. PTABGO<n> starts the table with line n. PTABGO<n><m> executes lines n to m.

Circular interpolation

The approximate generation of path curve over tabulated segments makes it possible to generate a circle-similar figure with two arbitrary axes x and y or a part of it, too. Here, the desired circular arc is approximated by a sequence of circle secants.

The vector table can be filled starting from a certain index with appropriate district data over a special instruction, if the appropriate basis parameters are set correctly before.

It is possible to compensate different resolutions of axis or to produce elliptical contours by a scaling factor, which sets the path increments of the two axes into a certain relationship to each other.

Definitions

Number of secants: $k \in (1,\ldots, m)$; m = total number of secants

Starting angle (angle offset) of the segments of a circle: α

Angle range which can be covered of the segments of a circle: $\Delta\alpha$

Radius of the segments of a circle: r

Illustration by the diagram:

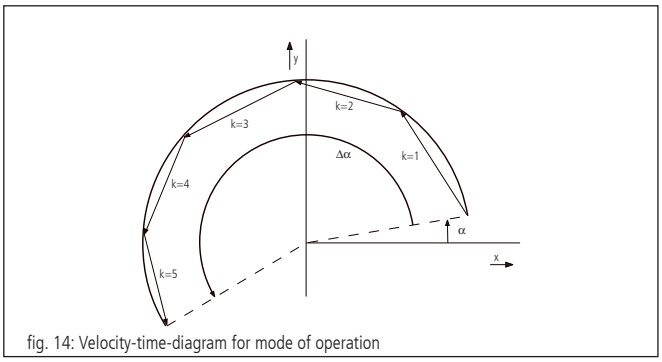

fig. 14: Velocity-time-diagram for mode of operation

with: part of circle with radius r, angle offset $\alpha = 10°$, angle range $\Delta\alpha = +190°$, m = 5 secants

Calculation

The approximated segment of a circle is defined by the radius, the number of secants, the angle offset and the angle range.

The direction of rotation is fixed by the sign of the angle range specification. Here a positive angle corresponds to a counterclockwise turn during appropriate arrangement of the axes (see also the position of the coordinate system in aforementioned diagram).

The starting angle for the single secant vectors k results to:

$$\alpha_k = \alpha + \Delta\alpha \cdot \frac{k-1}{m}$$

Then the x- and y-coordinates of the secant vectors are:

$$\Delta x_k = -2r \cdot \sin\left(\frac{\Delta\alpha}{2m}\right) \cdot \sin\left(\alpha_k + \frac{\Delta\alpha}{2m}\right) \quad \text{and}$$

$$\Delta y_k = 2r \cdot \sin\left(\frac{\Delta\alpha}{2m}\right) \cdot \cos\left(\alpha_k + \frac{\Delta\alpha}{2m}\right)$$

$\left|2r \cdot \sin\left(\frac{\Delta\alpha}{2m}\right)\right|$ designates the length of a secant vector.

Scaling factor

The scaling factor represented by counters and denominators is meant to adapt different resolution values of the two circle interpolation axes and to realize ellipses respectively. They can be set by two separate commands. The denominator is designated with N, the counter with Z.

If N > Z, the y-axis leads and the path data of x are divided by (N/Z). If Z > N, the x axis leads and the path data of y are divided by (Z/N). The default value is Z = N = 1, if no data are given by the user.

Syntax

Circular data are generated as secant vectors <m> starting from table element <n> with the command "PTABCIRCLE" and are transferred to the control. Here, the declaration of zero for an axis number means that the axis is not used. The syntax is as follows:

PTABCIRCLE <n> = <axis number x>, <axis number y>, Δt_n, f_n, m_n, r_n, α_n, $\Delta\alpha_n$, Z_n, N_n

Example

PTABCIRCLE0=1,2,326,0,5,1000,10,190,1,1

generates a pitch circle starting from table element 0 with axis 1 as x and axis 2 as y-axis, segment time 1/3 second, mode of operation v=const., 5 secants, radius 1000 increments, starting angle 10°, angle range 190° and scaling 1/1.

8.8 Automatic Reaction To External Triggers And Setting of Outputs

The PS 90+ offers 16 digital inputs and outputs. The TTL inputs and outputs will be numbered from 1 to 8, the SPS inputs will be numbered from 9 to 16.

8.8.1 Automatic Reaction To External Triggers

An automatic reaction to an external Trigger (trigger function) means that pre-defined actions are linked to a change of state of an input. As soon as this change of state is registered the action will be performed autonomously.

Definition Of An Input
Each input may assume the logical state of 0 or 1. Each change of state is called an edge. A change from 0 to 1 is a rising a change from 1 to 0 is a falling edge. Thus, there is a total of 32 triggers (16 inputs with two possible edges).

Possible Actions
There is a total of eight actions which can be triggered. Each action is disgnated by an action ID. Furthermore, each action can be defined by up to two parameters.

Action	Action ID	Parameter	Behaviour
Start velocity mode	1	Axis number (1…9)	VGO
Stop velocity mode	2	Axis number (1…9)	VSTP
Start positioning movement	3	Axis number (1…9)	PGO
Start velocity mode for multiple axes	4	Bit mask Bit 0 = axis 1 Bit 8 = axis 9	MVGO
Stop velocity mode for multiple axes	5	Bit mask Bit 0 = axis 1 Bit 8 = axis 9	MSTOP
Stop axis	6	Axis number (1…9)	STOP
Switch off axis	7	Axis number (1…9)	MON
Switch off axis	7	Axis number (1…9)	MON
Switch on axis	8	Axis number (1…9)	MOFF

All actions which start a movement only concern the actual starting command. All relevant parameters at the time of triggering, especially relative or absolute distance, are being used.

If a command cannot be executed because of an illegal state of the axis it will be ignored. At the next instance of the trigger execution will be tried again.

Assigning An Action To A Trigger
Actions can be assigned to a trigger and saved in an action table. A table entry consists of the following:

Input	Edge	Action ID	Parameter
1-16	0 (falling) or 1 (rising)	1 to 8	2 x 32-bit Signed decimal or signed hex string with prefix 0x

Behaviour
If a trigger occurs the PS 90+ executes the assigned action as fast as possible. The action table is processed sequentially.

The table has a fixed size. The user can therefore edit, delete, and query all entries and define the execution order.

Execution starts at the first line. If a condition is met the assigned action is executed and the next line will be processed. It will not be waited for the ending of an action.

It is the responsibility of the user to prevent unwanted behaviour. There is no mechanism which checks for contradicting programming.

Considering this, it is possible to assign the same trigger to multiple actions. That is, the same trigger may be appering multiple times in the action table. Likewise, it is possible to use the same action multiple times.

The table consists of a maximum of 64 entries.

The behaviour shall be explaing with the following exmaples:

Start and stop axis 1 unsing input 14. Start and stop axis 2 using input 0. Start axis 3 if axis 2 is stopped through input 0.

> **!** **Remark:** It is not defined if axis 2 stops first or axis 3 starts first. Both actions are starts as fast as possible.

Line	Input	Edge	Action ID	Parameter
1	14	1	1	1
2	14	0	2	1
3	0	1	1	2
4	0	0	2	3
5	0	0	1	3

Stop axis 1 using input 14 and start axis 1 using input 3 and 12.

Line	Input	Edge	Action ID	Parameter
1	14	0	1	1
2	2	1	2	1
3	12	1	1	1

8.8.2 Automatic Setting Of Outputs

A pre-defined setting of outputs (event function) means that outputs are set to a defined level if certain internal events happen. Internal events are dependent on axis states.

Definition Of Events
There are six internal events which can be used. Each one has its event ID and parameters where applicable.

Internal Event	Event ID	Parameter	Remark
Velocity of an axis is above a certain value.	1	Axis number (1…9) Velocity value: signed decimal	Current velocity > value

Velocity of an axis is below a certain value.	2	Axis number (1…9) Velocity value: signed decimal	Current velocity < value
Position of an axis is above a certain value.	3	Axis number (1…9) Position value: signed decimal	Current position > value
Position of an axis is below a certain value.	4	Axis number (1…9) Position value: signed decimal	Current position < value
Axis changes into a state of motion	5	Axis number (1…9)	Based on axis state
Axis changes into a state of no-motion	6	Axis number (1…9)	Based on axis state

The last two entries refer to:

Axis states (see ?ASTAT) T, S, V, P, F, W, X, Y, C , and N are regarded as states of motion. A change from any other state into one of those states is considered a change into a state of motion. A change from any of those states into any other state is considered a change into a state of no-motion.

Assigning Of Outputs
The assignment to outputs is done using an event table. A table entry consists of the following data:

Event ID	2 parameters	Output mask	Output value
1-8	2 * 32 bit	16 bit	16 bit

An event is only affecting outputs which are selected with a 1 in the output mask. Selected outputs are set to the values given in the output value parameter.

The table has up to 64 entries.

Behaviour
Outputs are not synchronised. There is no clock signal. Thus, intermediate states can be observed when several outputs are changed at the same time.

The table is executed sequentially. If an event occurs all assigned outputs are set. There is no information about the exact order of execution.

An event ID may be used multiple times. This shall be explained with the following examples:

Axis 2 starts and accelerates until a velocity of 20000. After a time it is then stopped. Once the velocity is above 10000 output 2 shall be set to 0. If the velocity falls below 9000 output 2 shall be set to 1. The output level at the beginning is not defined.

Event ID	2 parameters	Output mask	Output value
1	axis 2 10000	0x0002	0x0000
2	axis 2 9000	0x0002	0x0002

If axis 1 changes into a state of motion output 15 shall be set to 0. If axis 3 changes into a state of no-motion output 15 shall be set to 1. If axis 5 changes into a state of motion output 3 shall be set to 1.

Event ID	2 parameters	Output mask	Output value
5	Achse 1	0x8000	0x0000
6	Achse 3	0x8000	0x0001
5	Achse 5	0x0004	0x0004

In this example both the first and the second line affect output 15. If both events happen at the same time it is not defined which is executed first. Thus, the value of output 15 is not defined. Such conflicts can only be detected and prevented by the user.

8.8.3 Configuration

Each table consists of up to 64 entries.

The following functions are available on the command interface:

- The complete trigger and event functionality can be enabled or disabled with a command. Since these feature are permanently using processing power it is advised to only enable them when actually used.
- Whether the functionality is enables or disabled can be queried.
- Parameters can be loaded into both tables.
- Single table entries can be read out.
- Single table entries can be deleted.
- Commands with entry numbers lager than the permissible range will be ignored.
- If there is an error when writing entries into the action table, for instance an illegal parameter, the action ID will be set to 0.
- Empty entries in the action table are marked by an action ID of 0.
- If there is an error when writing entries into the event table, for instance an illegal parameter, the event ID will be set to 0.
- Empty entries in the event table are marked by an event ID of 0.

9. Travel Measuring

Encoder
The travel measuring system, also known as "rotary encoder", for the position feedback signals is evaluated only in the so-called closed-loop operation mode.

Without encoder, only open-loop operation with 2-phase step motors is possible. In order to be able to operate BLDC or DC motors, each axis must be equipped with a travel measuring system. This can be an encoder. Usually, encoders with 500, 1250 or 2500 lines per revolution are used. The motion processor measures the current axis position via encoder and calculates the appropriate rotational speed of the motor, considering the temporal change of the position parameters.

Encoders are mounted stationary on the motor and directly connected with the rotor. The encoder output signals are named A and B (CHA and CHB) with a phase-shift of 90 degrees (so-called quadrature signals), and, if necessary, one Index pulse I per revolution. The PS 90+
can process TTL-level or antivalent signals (line-driver outputs). After level transformation and filtering, the signals are transmitted directly to the motion processor.

Linear Measuring System
A position sensor, directly coupled to the actuator motion, is called linear measuring system. The linear measuring system can be used both instead of the encoder for position measuring or together with an existing encoder for adjusting the positioning system onto the target position. Hereby, a correction of systematic errors (e.g., spindle-pitch error) is possible.

By using a linear measuring system for the follow-up control, the target position is indicated separately (32-bit resolution). The actual positioning task is then accomplished by the motion processor in closed-loop mode via encoder. If it signals that the target position has been reached, the main processor will be going to adjust the position until the accurate target position, taken by the linear measuring system, moves into the predefined target window.

The signals of the linear measuring system correspond to the encoder signals specified before (quadrature A and B as well as Index I). The maximum counting frequency is 5 MHz (signal) or 20 MHz (quadrature), respectively.

Function of the follow up control

To realize a follow up controller for a certain positioning unit it is necessary to equip the positioning unit with an additional incremental linear measuring system, which detects the real absolute position of the slide using a clear reference mark. The drive unit, consisting of the motor and drive spindle (referred to in the following as the "actuator"), will be corrected to the real absolute position by the control. This can be done by iterative correction movements or with a correction run at a constant speed. A combination of both procedures is also possible. The selection is done via the operating modes of the follow up control. The values for the resolution of the linear measuring system and the positioning unit are usually different.

Before using the follow-up controller a reference scan has to be made in reference motion mode 6 or 7. Thereby the total available travel is measured in increments of the linear measuring system and the absolute position counter is automatically set to zero when the reference mark of the linear measuring system is attained.

The target position of a follow-up controlled positioning unit is indicated by the defined absolute position of the linear measuring system after a successful reference run, i.e. a target position is the absolute or relative distance indicated, related to an integer multiple of the increment of travel of the linear measuring system, the reference point and, if applicable, the current position.

For the control to internally calculate the position of the actuator, the relationship between the increment of position by the actuator and the increment of position by the linear measuring system is characterised by a conversion factor F = Z / N , which is the ratio of both resolutions.

A positioning run with a follow up control corresponds to the following 3-phase scheme:

- Using the given conversion factor (Z / N) the relative distance for travel of the actuator is calculated from the given position.
- The actuator is moved the calculated distance (phase 1, rough positioning), and the deviation from the nominal position is calculated.
- If the actual position is outside the defined target window, an iterative approach can be used, if desired, i.e. the relative distance of the actuator is calculated cyclically and output to the motor output, etc. (phase 2, iteration).
- In order for the motion to converge it is necessary that the amount moved in each iteration step is less that the previous step, until the current position is within the target window. It follows that a divergence criterion for the iteration phase is the situation when the amount of deviation after correction move (n) is greater or equal than the amount of deviation after the correction move (n-1).
- After successful completion (convergence, current position is within target window) or failure (divergence) of iteration a correction phase in speed mode (Phase 3) may follow . Whether phase 3 is active or not is selectable, ie is set by a parameter.
- In the subsequent correction phase, the actual position of linear measuring system is queried. If the actual position is outside the target window, speed mode is called with the previously defined follow up velocity as a parameter. Once the current position is within the target window, the follow up procedure stops, i.e. a break ramp will be triggered. If the actuator runs beyond the limit, the direction will be reversed, etc.
- It can also be set by a system parameter whether or not the follow up control in the velocity mode should always be active or switched off upon reaching the target window.

Calculation of the conversion factor F:

In follow up controlled operation driving distances are in principle multiples of the measurement resolution (the minimum increment of position of the linear measurement system). The resolution of the actuator is determined by the engine resolution (eg micro-step factor, increment of encoder) and mechanical parameters (for example, spindle pitch).

From the given movement distance, the relative distance to be passed to the actuator has to be calculated before each motion.

Below is an example calculation of a linear stage with direct drive spindle and 2-phase stepper motor (unregulated).

$$F = \frac{Z}{N} = \frac{r_s}{r_m} = \frac{\text{resolution of the actuator}}{\text{resolution of the measuring system}}$$

calculation of r_s:

$$r_s = \frac{h}{n \cdot m}$$

with:

h = spindle pitch (travel per motor revolution),
n = steps of motor (full steps per motor revolution),
m = micro step factor (microsteps per fullstep)

Example:

h = 5 mm,
n = 200,
m = 50

it follows:

$$r_s = \frac{5\,\text{mm}}{200 \cdot 50} = 0{,}5\,\mu\text{m}$$

The resolution of the measuring system r_m is given (in this example) by:

$r_m = 0{,}1\,\mu\text{m}$

Thus, we have

$$F = \frac{r_s}{r_m} = \frac{0{,}5\,\mu\text{m}}{0{,}1\,\mu\text{m}} = \frac{5}{1} = \; : \frac{Z}{N}$$

and therefore:

$Z = 5$

$N = 1$.

10. PID Servo Loop Algorithm

The servo filter used in the PS 90+ operates according to a PID algorithm. An integration limit provides an upper bound for the accumulated error.

The PID formula is as follows:

$$Output_n = K_p E_n + K_d(E_n - E_{(n-1)}) + \sum_{j=0}^{n} E_j \frac{K_i}{256}$$

Meaning of following abbreviations:

E_n accumulated error terms from the main encoder
K_i integral gain of feedback control loop
K_d differential gain of feedback control loop
K_P proportional gain of feedback control loop

All filter parameters and the torque signal limit are programmable, so that the user is able to fine-tune the filter. The ranges of values and formats are listed in the following table:

terminus	name	range
I_{lim}	integration limit	32 bit unsigned (0…2.124.483.647)
K_i	integral gain	16 bit unsigned (0…32.767)
K_d	derivative gain	16 bit unsigned (0…32.767)
K_p	proportional gain	16 bit unsigned (0…32.767)

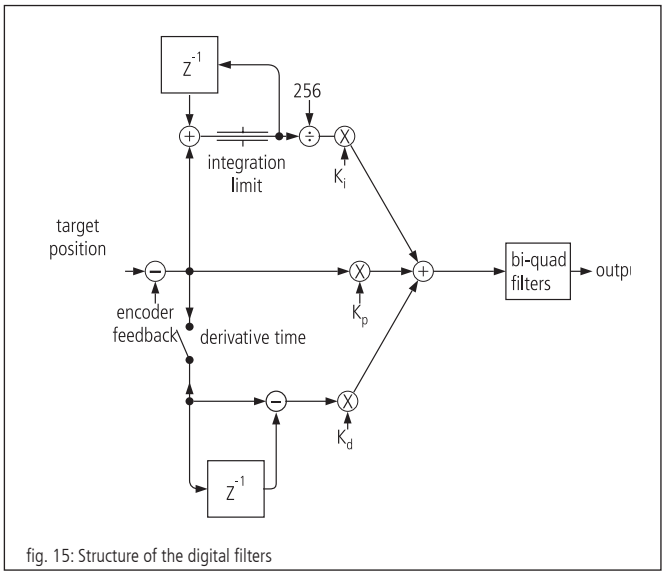

fig. 15: Structure of the digital filters

11. Positioning Velocity and Acceleration, Calculation

11.1 2-Phase Step Motor (Open Loop)

General Information

Each step motor-driven mechanics has a so-called start-stop frequency which is especially dependent from motor type, system friction and load. The start-stop frequency defines the maximum travel frequency of the step motor concerned, with which it starts directly from standstill without acceleration phase. It is usual to indicate these and other characteristic frequencies of step motors in Hertz full-step ("HzFS"), i.e. full steps per second. The shaft of a step motor with a step angle of 1.8°, i.e. R = 200 full steps per motor revolution, which runs with e.g. 400 HzFS, rotates with a speed of 2 revolutions per second or 120 revolutions per minute.

In order to reach speeds higher than the start-stop frequency, the step motor must be accelerated beyond this frequency with a suitable acceleration ramp, or be slowed down to a lower frequency with a suitable brake ramp. This acceleration or deceleration takes place by means of trapezoidal or S-curve velocity-time profile. If necessary, a damping (clean damper, installed at the second motor shaft end) is used in order to reach a higher rotary speed.

Nearly all standard step motors used by OWIS® are able to comply with a frequency of 400 HzFS in start-stop operation mode.

The PS 90+ has a digital profile generator. The speed profiles are periodically calculated and sent to the 2-phase step motor.

Cycle Time

The cycle duration of the digital profile generator is defined by hardware.

$T_P = 256\,\mu s$

Final Velocity

The positioning of the axes is done by means of the "point-to-point" method. Each axis follows a trapezoidal or S-shaped velocity profile.

The final velocity V after the acceleration ramp is specified by one 32-bit word. The value of V ranges from 1 to 2147483647.

! Note:
It must be ensured that no higher velocity is entered than the equipment is able to withstand, since otherwise the mechanism may be damaged or destroyed.

When the speed V and the encoder line number R is given, the motor speed is calculated as follows:

$$f_{Mcstp} = \frac{1}{T_P} \cdot \frac{V}{65536}$$ (step frequency in micro step mode)

resp.

$$f_{VS} = \frac{1}{Mcstp \cdot T_P} \cdot \frac{V}{65536}$$ (step frequency normed for full step mode)

The speed of rotation for a step motor with R full steps each motor revolution can be calculated as:

$$n_{RPM} = \frac{60}{min} \cdot \frac{1}{Mcstp \cdot R \cdot T_P} \cdot \frac{V}{65536}$$ (revolutions/minute)

resp.

$$n_{RPS} = \frac{1}{s} \cdot \frac{1}{Mcstp \cdot R \cdot T_P} \cdot \frac{V}{65536} \qquad \text{(revolutions/second)}$$

For the conversion of the motor rotary speed to the positioning velocity of mechanism, mechanical data, such as spindle pitch, and, where appropriate, the influence of a gearbox, must also be taken into consideration.

Acceleration for Trapezoidal Velocity Profiling
The acceleration ("ACC") is specified by a 12-bit word. The values of "ACC" range from 1 to 2147483647.

When the velocity V and the acceleration ACC are given, the duration of trapezoidal profile acceleration ramp is calculated as follows:

$$\Delta t = 1\,s \cdot \frac{V \cdot T_P}{ACC} \qquad \text{(acceleration/deceleration duration in seconds)}$$

Travelled distance during the trapezoidal profile acceleration/ deceleration:

$$\Delta s = 1\,microstep \cdot \frac{V^2}{131072 \cdot ACC} \qquad \text{(deceleration in microsteps)}$$

11.2 DC Servo Motor and 2-Phase Step Motor (Closed-Loop)

General Information
The PS 90+ has a digital position/speed controller. Output and control
variables are periodically calculated. The acquisition of the actual position value is done in the simplest case by means of a rotary encoder (also called "encoder"), which is attached to the 2nd shaft extension of the motor. The most important parameter of the encoder is the number of encoder lines R. This is the number of the light/dark periods on the encoder disc for each motor shaft revolution. The signals go through a quad evaluation, which results in a generally 4-fold higher resolution than the number of encoder lines.

Servo Loop Cycle Time
The cycle duration of the digital controller is also called cycle time. It is defined by hardware. The minimum cycle time is 51,2 µs. If necessary, it can be increased by an integer multiple of 51.2 µs:

$T_S = 51{,}2\,\mu s + n \cdot 51.2\,\mu s;\ n \in [0, 1, ..., 386]$

corresponding to a cycle time of

$T_S = [51.2\,\mu s, 102.4\,\mu s, 153.6\,\mu s, ..., 204.8\,\mu s, 256\,\mu s, ..., 19986\,\mu s]$

Only integer values can be handed over to the PS 90+. The value is rounded internally to the next valid value.
Default value (presetting): $T_S = 256\,\mu s$.

Final Velocity
The positioning of the axes is done by means of the "point-to-point" method. Each axis follows alternatively a trapezoidal or S-shaped velocity profile.

The final speed V after acceleration ramp is specified by a a 32-bit word. Its values range from 1 to 2147483647.

! Note:
It must be ensured that no higher velocity is entered than the equipment is able to withstand, since otherwise the mechanism may be damaged or destroyed.

At a given speed V and an encoder line number R, the motor speed (without consideration of a possibly existing gearbox) is calculated as follows:

$$n = \frac{60}{min} \cdot \frac{1}{T_S} \cdot \frac{1}{4R} \cdot \frac{V}{65536} \qquad \text{(revolutions per minute)}$$

resp.

$$n = \frac{1}{s} \cdot \frac{1}{T_S} \cdot \frac{1}{4R} \cdot \frac{V}{65536} \qquad \text{(revolutions per second)}$$

resp.

$$n = \frac{1\,increment}{s} \cdot \frac{1}{T_S} \cdot \frac{V}{65536} \qquad \text{(increments per second)}$$

The last formula can also be understood as follows: The controller travels V / 65536 increments for each sampling interval T_S.

For the conversion of motor rotary speed into positioning velocity of mechanics, mechanical data such as spindle pitch and, if appropriate, the influence of a gearbox have to be considered.

Example:

Positioning is to be effected at a rated speed of n = 1800 rev./min. An encoder with R = 500 lines (correspond to 2000 impulses/rev.) is to be used. What value of V should be selected?

Solution:

It results after resolving the equation for the speed of rotation:

$$V = \frac{n[min^{-1}]}{60} \cdot 4 \cdot R \cdot 65536 \cdot T_S$$

Thus, V = 1006633 for n = 1800 rev./min. when using a 500 lines encoder. A spindle pitch of 1 mm gives a speed of 1.8 m/min. or 30 mm/sec. then.

Acceleration for Trapezoidal Velocity Profiling
A 32-bit word is to be entered as acceleration ("ACC"), the values range from 1 to 2147483647.

Duration of the trapezoidal profile acceleration ramp at given speed V and acceleration ACC:

$$\Delta t = 1\,s \cdot \frac{V \cdot T_S}{ACC} \qquad \text{(acceleration/deceleration duration in seconds)}$$

Travelled distance during the trapezoidal acceleration/ deceleration ramp:

$$\Delta s = 1\,increment \cdot \frac{V^2}{131072 \cdot ACC} \qquad \text{(deceleration in increments)}$$

12. Nano-Hybrid Control

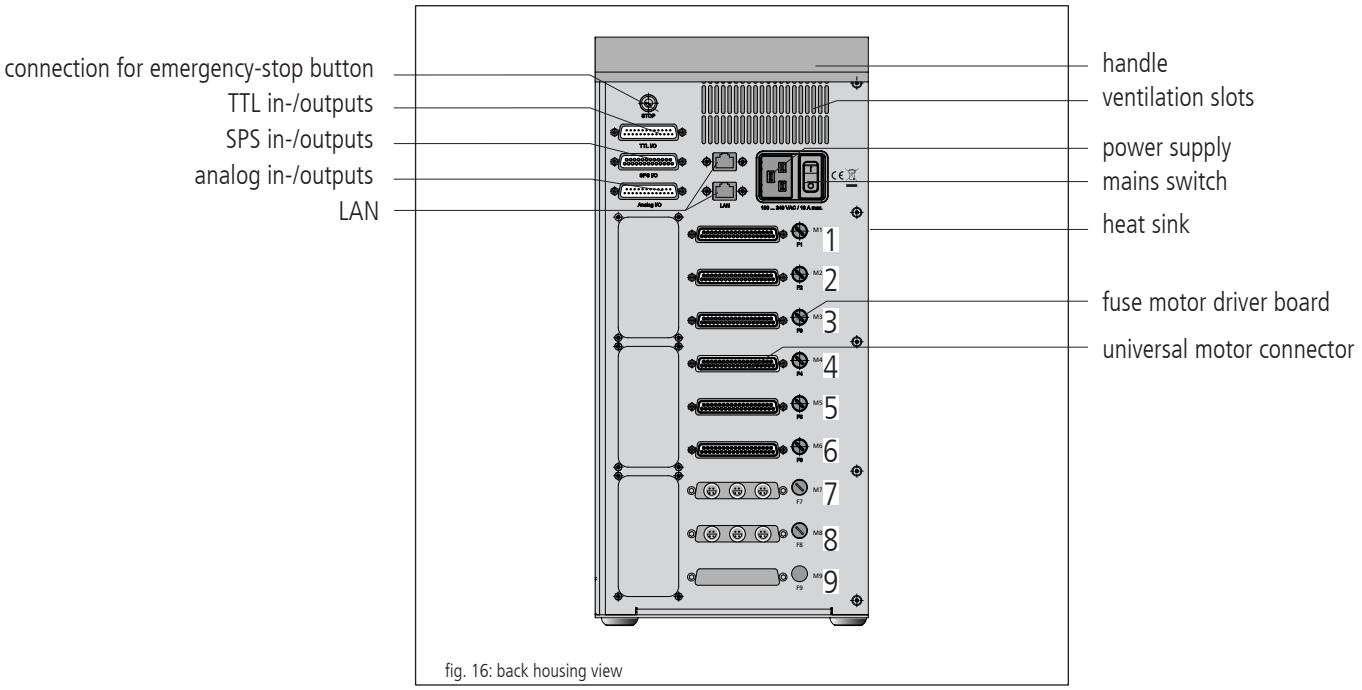

fig. 16: back housing view

General Information
In order to control OWIS® positioning stages with nano-hybrid technology a suitably equipped PS 90 control unit is necessary. This chapter explains the differences and special properties of such a unit as well as the corresponding control modes. All general information, especially those concerning safety and handling of the control unit, is valid without limitation. Prior to using OWIS® nano-hybrid positioning stages knowledge of all former chapters is mandatory.Technical Overview and Setup of the Control Unit

Technical Overview and Setup of the Control Unit
Nano-hybrid positioning stages by OWIS® possess a hybrid drive mechanism. Coarse positioning is performed with a high-resolution stepper motor. Fine positioning is achieved by a piezo actuator.

A PS 90 control unit which is equipped for nano-hybrid positioning stages is able to control six axes. Instead of the axes 7-9 control circuitry for the piezo actuators is installed.

The positioning stages are connected via the universal motor connector. Additionally, the piezo actuators are connected with a different cable. The corresponding connectors are located at position 7 for the first three nano-hybrid axes. If four to six axes shall be controlled, position 8 is used as well. Position 9 is always unused.

Safety
The control of the piezo actuators uses voltages between -71 V to +71 V. Those voltages may cause serious injuries. Personnel operating this device must be instructed on the proper handling of such voltages. The general accident prevention regulations must be followed.

Control Architecture and Function
A nano-hybrid PS 90 control unit consists primarily of the following components:

1. An integrated power supply
2. A main board
3. Max. 2 drive controller boards
4. Max. 6 motor driver boards
5. Max. 2 quadrature encoder boards
6. 1 D/A converter card for piezo actuators
7. Max. 2 control modules for piezo actuators

Instead of a third drive controller board a special D/A converter card for the piezo drives is installed. This can operate three or six axes. If only the first group of motors is configured for nano-hybrid positioning stages a single control module for piezo actuators is necessary. If both groups of motors are configured for nano-hybrid technology two of such control modules are needed. The control module for the piezo part of the positioning stage contains a protective circuit which ensures that only compatible OWIS® nano-hybrid units are connected. The basic concept of controlling the stepper motors is unchanged.

Connection
The connection is done with two cables. On the one hand a motor cable is plugged with its 37-pin connector to the control unit and with its 18-pin Lemo-plug to the stage. On the other hand the piezo-drive is connected via 4-pin Lemo-plugs on both control unit and stage.

Positioning in nano-hybrid mode
In order to achieve positioning with a nano-hybrid stage there are three possibilities. Normal positioning can be done as with any regular unit with a stepper motor. Since all nano-hybrid positioning stages are equipped with an integrated measurement system follow-up control (modes 0 to 5) can be chosen as well.

To make use of the ultra-high positioning with the piezo drive special modes (6 to 9) within the follow-up control are available. Those modes are being described as follows:

General Description of Follow-up Control for piezo-drives
When using follow-up control positioning is performed in separate, consecutive steps. The first step always is a coarse positioning done by the motor. Next, a correction of any positioning error is done by the motor and finally by the piezo drive. The following parameters are relevant when performing positioning with a piezo drive:

- PWMSSET

This value sets the desired position for the following positioning movement. When using absolute positions this value is interpreted as such. In relative mode the sign decides on the direction of the movement.

- PWMSPWIN

This parameter defines an acceptable target window in front and behind the set position PWMSSET. It is given in increments of the measurement system. The positioning movement ends successfully once the actual position is within PWMSSET-PWMSPWIN and PWMSSET+PWMSPWIN. PWMSPWIN must always be a positive value and should be in the range of 2 to 10 increments. Its value affects the time to achieve the set position. Choosing too large a value results in a to coarse positioning and repeatability accuracy. Choosing too small a value might lead to not achieving a stable end position.

- WMSOFFS

In order to make the usage of the piezo drive possible the stepper motor must move the stage to a different position than the actual desired position. This offset is defined by WMSOFFS and given in increments of the measurement system. WMSOFFS must always be negative and should be in the range of -20 to -100. If the value is set improperly addition correction movements by the motor might be necessary resulting in an increased time to finish the positioning.

- PWMSWIN

This value defines an acceptable target window in front and behind the position which the stepper motor has as target. It is given in increments of the measuring system. Positioning with the stepper motor is finished successfully as soon as the actual position is within (PWMSSET+WMSOFFS)-PWMSWIN and (PWMSSET+WMSOFFS)+PWMSWIN. PWMSWIN must always be positive and should be between 10 to 50 increments. Its value might affect the total time of the positioning phase. Improper setting of this value might lead to an addition correction phase of the stepper motor or the piezo drive.

- WMSVEL

This value sets the speed with which in modes 7 and 9 a correction movement in phase 2 is done. WMSVEL must always be positive.

Depending on the chosen mode the following procedure is performed:

- Using the given conversion factor (Z/N) the relative distance for travel of the actuator is calculated from the given position.
- The actuator is moved the calculated distance (phase 1) and the deviation from the nominal position is calculated.
- If the actual position is outside the defined window (PWMSSET+WMSOFFS)+/-PWMSWIN a correction phase (phase 2) is started. If the actual position is inside this window, fine positioning with piezo drive is started.
- If phase 2 is needed depending on the chosen mode one of two possible correction movements will be performed. In mode 6 and 7 a coarse positioning phase (phase 1) will be started again. This means that based on the current position the new target position is calculated and then the stepper motor will move the stage accordingly. Phase 2 ends if either the end position is within the set target window or the error between consecutive correction movements increases. In modes 8 and 9 correction is being done with a velocity movement. This movement is stopped once the target window is reached. If for any reason the movement ends behind the window another movement in the opposite direction starts.
- Once positioning with the stepper motor is completed fine positioning with the piezo drive starts (phase 3). This phase ends once the actual position is within PWMSSET+/-PWMSPWIN. Should the piezo be unable to reach the target window phase 2 is started again with a correction movement by the stepper motor.
- In modes 7 and 9 phase 3 remains active after the final position has been reached. This means that any changes in the actual position, which might be for example a result of external forces, will be corrected continually by the piezo drive. If the piezo drive itself cannot correct this, phase 2 will be initiated automatically.

13. Initial Operation of the PS 90+

13.1 Installation and Preparing

Installation
The control is designed for the use in research and development as well as for industrial applications. It may only be operated in dry, dust-free environment (normal ambient conditions). Normally, it is operated as a tabletop unit.

For internal cooling, ventilation slots are attached in the upper part of the housing front and back side. The waste heat of the motor power stages is dissipated by the laterally attached heat sink. The control should not be built into an additional housing or a cabinet without sufficient air circulation.

> **Note:**
> Heat accumulation in the control or at the heat sink should be avoided. A minimum distance of 15 cm has to be kept to closed surfaces and walls.

Emergency-Stop Function
On the rear panel, one can find a socket for connecting an external emergency-stop button. If no emergency-stop button is used, a jump plug has to be inserted. If an emergency-stop button shall be connected, the jump plug has to be removed and the button (n.c. contact) has to be connected instead.

> **Note**:
> If the jump plug is removed and no emergency-stop button is connected, the operation of the motor output stages is blocked.

13.2 Connection of Peripherals and Devices

Before switching on the control, all connecting plugs for devices and peripherals have to be connected, so that they are recognized and initialized by the control during start-up.

This is:
- the positioning unit
- the power supply
- the computer

The controller is connected via the USB, RS-232 or ethernet interface to the computer.
With additional Anybus®-module "Modbus/TCP" it is possible to communicate with a PC via Ethernet.

For this, a driver installation is required. The driver is on the included CD.

For the installation please start "setup.exe"

> **Note:**
> Before switching on and starting the control all peripheral devices have to be connected. Otherwise they will not be initialized and therefore not identified by the control unit.

13.3 Getting Started

The control is activated by pressing the mains switch. The microcontroller initializes itself and its periphery. The initialization procedure takes approx. 10 seconds. Afterwards, the PS 90+ is ready to receive and execute commands from the PC. When Windows is first started after the PS 90+ has been connected, the operating system should recognize the new hardware. The driver can then be installed. In order to do this, administrator rights are necessary.

Initialization
After having switched on the power supply and activated the unit, each axis that shall be used has to be initialized first by the INIT command.

Axes parameters having been changed will also be taken over during the initialization.

Software
Following tools are included with purchase: the software tool OWISoft, the USB driver and the software interface (SDK/API) for C, C++, C#, LabView (V 8.2 and higher) and additional programming languages (32/64 bit). Thus, the PS 90+ can be configured and operated comfortably.

Supported operating systems: Windows XP, Windows Vista (32/64 bit), Windows 7 (32/64 bit), Windows 8 (32/64 bit), Windows 8.1 (32/64 bit) and Windows 10 (32/64 bit).

The software interface includes example programs with source code and help files.

For start up with OWISoft the standard values of the respective OWIS® positioning units are stored and can be adjusted.

> **Note:**
> The default parameters stored in OWISoft apply for the idle operation (no load). For optimal positioning the standard parameters of the PID control must be adjusted for the specific application (specific load).

For adjusting, please read the manual OWISoft.

If the control shall be used by a user´s own software, please read the chapter "Instructions Concerning the Setup of User Application Software". There you will also find the command table for the PS 90+
as well.

14. Malfunction Monitoring

14.1 Limit Switches

The PS 90+ has four limit switch inputs, two for limit switches (MINSTOP, MAXSTOP) and two for brake switches (MINDEC, MAXDEC), as well as capability for a reference switch for each axis. One of the four limit switches is defined as reference switch.

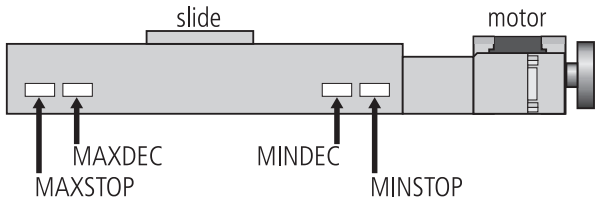

fig. 17: Identification of limit switches

OWIS® positioning units are provided with maximum of four limit switches. The limit switches working in negative direction (motion of the slide towards the motor) are named MINDEC and MINSTOP. The limit switches working in positive direction (motion of the slide away from the motor) are similarly named MAXDEC and MAXSTOP.

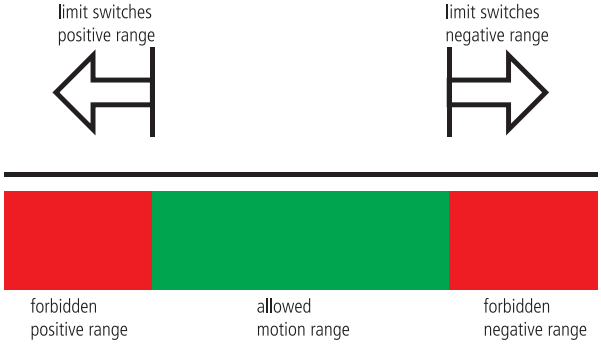

fig. 18: Function of limit switches

Working Principle of the Limit Switch Monitoring

1. MINSTOP: Actuation of this limit switch with motion in negative direction results in immediate disable of the motor power, after a certain reaction time which can be some milliseconds.
   DC servo motor: The motor is disabled. However, the residual kinetic energy leads to some remaining movement until it is used up by friction or stoppers.
   Step motor open loop: If the current travel frequency with which it is stopped was higher than the system start-stop frequency, the kinetic energy in the system leads to a remaining motion. This motion cannot be detected by the control unit, thus resulting in a wrongly indicated potision. A reference travel is necessary to match the current position with the motor steps.
2. MINDEC: Actuation of this limit switch results in execution of a deceleration ramp, using a programmable deceleration value. After execution of the braking ramp, the motor will not be switched off but is still under control. If the follow up path of the deceleration ramp has been too big and the slide reached the MINSTOP limit switch afterwards, please note point 1.
3. MAXDEC: The reaction is similar to the MINDEC limit switch, but the effect is in positive direction.
4. MAXSTOP: The reaction is similar to the MINSTOP limit switch, but the effect is in positive direction.

Configuration of Limit and Reference Switches

The command "SMK…" defines which end switches should be used with the corresponding positioning units connected. If one bit is set (=1), the corresponding limit switch will be recognized.

The limit switch polarity is preset with the command "SPL…". The value handed over defines whether the limit or reference switches should be set to "low"or "high". A cleared bit means that the respective switch is "low" active (e.g., normally-open contact towards GND, which means "not connected" in inactive mode). If one bit is set (standard configuration), then the corresponding switch must be "high" active (e.g., normally-open contact towards GND, which means "connected" in inactive mode).

The limit switch inputs work normally with 5V-CMOS level, while NPN open-collector or push-pull outputs can be equally connected, as high-impedance pull-up resistors (4.7 kOhm) towards + 5V are already built-in. The limit switch inputs accept external voltages of up to + 24V.

Reconnection after Axis Error

When an axis error occurs after activating a limit switch (MINSTOP oder MAXSTOP), the axis <n> should be reconnected as follows:

1. initialize via command INIT<n>
2. release limit switch via command EFREE<n>

14.2 Output-Stage Error Monitoring

The status of each motor power stage is transferred to the PS 90+ main microcontroller via digital line. This signal is periodically monitored. If a power stage detects an error, then the motor is shut off, i.e. the control loop is opened and the power stage is disabled.

14.3 Motion-Controller Error Monitoring

The communication with the motion processors is monitored in similar way. If error or implausibility occur, then the motor is shut off, that means that the control loop is opened and the power stage is disabled.

14.4 Time-Out Monitoring

Additionally, a timeout value (in ms, 32-bit range) can be defined as parameter for each axis. The monitoring can be switched off by setting the timeout to 0. This timeout is monitored periodically, while a motion is executed (PGO, REF, EFREE, PWMSGO, LIGO). If the motion lasts longer than this time, then the motor is shut off (?ASTAT → "Z", see comand table p. 62), that means that the control loop is opened and the power stage is disabled. This function is useful, if, for instance, during the reference motion one of the reference switches cannot be found.

15. Joystick

Additionally to the manual control, it is possible to connect a joystick which is available as accessory. It enables the manual operation of 3 axes. The joystick is attached at the analog input of PS 90+.

fig. 19: Joystick

16. Instructions Concerning the Setup of an Own Application Software

Generally, a PS 90+ application consists of an initialization part which sets the necessary axis parameters for all for axes <n> to be used and which switches on the axes, too. Furthermore, it consists of a loop which executes a reference motion for all axes and of the actual user program with all the functions required.

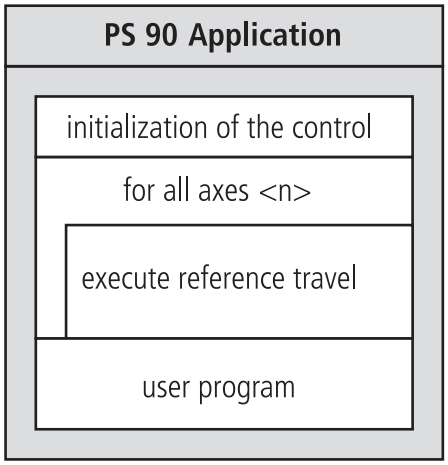

fig. 20: PS 90+ application architecture

The initialization of the axes required is with over the INIT command at the simplest, if the parameters stored in the static RAM are to be taken over. Otherwise, it is necessary to transfer the required parameters first, before sending the INIT command.

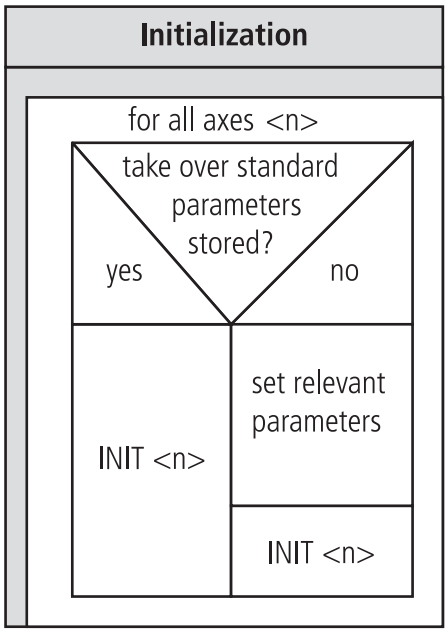

fig. 21: PS 90+ initialization architecture

A command processing time (interpretation time) of about 20 to 40 milliseconds has to be considered between two individual commands sent to the PS 90+. The control unit signals received can be e.g. retrieved character by character every millisecond, until the defined end-of-string identifier is received.

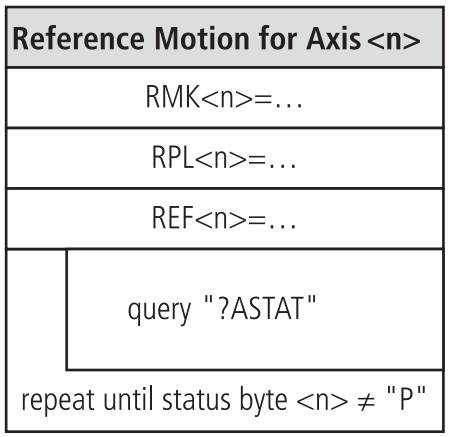

fig.22: PS 90+ reference motion for axis

If a reference motion for an axis is to be executed, reference mask and reference polarity are to be set before. This is necessary only if it has not already been done before or if no appropriate values have been set for the standard settings. Afterwards, the reference motion is started.

The use of the provided software tool OWISoft (including SDK and DLL) facilitates the setup considerably, since frequently used command sequences are already predefined as functions or routines. Furthermore, the necessary running time check is implemented, too.

17. Command Set for the PS 90+

General information concerning the command format:

Each command is transferred over the interface (RS-232, USB or ethernet) in ASCII format. The individual characters of a command are converted
automatically into capital letters. Each command ends with CR or CR+LF or LF (adjustable).

Furthermore, the response mode can be preset (TERM). For this purpose, there are three settings available:

1) When reading out the message buffer, only a two digit number is returned (error code). This setting is especially selected when a control takes place via software through a host PC, since the message strings are here at the shortest, and therefore the command throughput is optimized.
2) Reading the message buffer returns a two digit number (error code) and an additional plain text string explaining the error code.
3) Similar to 2) and, additionally, each executed command giving no return value, will be acknowledged by "OK".

Acknowledgment is returned with CR or CR + LF or LF (adjustable).

In the first response mode (TERM=0), the binary information (e.g., limit switch configuration, limit switch status, digital/analog inputs/outputs, etc.) is represented as bits of a decimal number. In the other modes (TERM=1, TERM=2), these values are indicated as a binary number (one bit is represented by one ASCII character, "0" or "1"). This applies both for the query and for the setting of a value.

All parameters are stored resident and provided with a check sum. After having switched the device off and then on, the last parameter setting is again valid. If a check sum error should arise, then, after switching on, default values are loaded automatically and an error message is written into the error output buffer.

For commands that give a response (e.g., parameter queries) the answer is sent back to the PC, immediately.

<n> = axis number 1…9 (respectively, highest possible axis number)
<uv> = unsigned integer value
<sv> = signed integer value
<v> = signed way indication

Attachment

I Command Table

command group	command	function description	example	response
general status request	?ASTAT	Axis status inquiry, a character each axis is returned to describe the current axis mode: "I" = axis is initialized "O" = axis is disabled "R" = axis initialised and ready "T" = axis is positioning in trapezoidal profile "S" = axis is positioning in S-curve profile "V" = axis is operating in velocity mode "P" = reference motion is in progress "F" = axis is releasing a limit switch "J" = axis is operating in joystick mode "L" = axis has been disabled after approaching a hardware limit switch (MINSTOP, MAXSTOP) "B" = axis has been stopped after approaching a brake switch (MINDEC, MAXDEC) "A" = axis has been disabled after limit switch error "M" = axis has been disabled after motion controller error "Z" = axis has been disabled after timeout error "H" = phase initialization activ (step motor axis) "U" = axis is not released. "E" = axis has been disabled after motion error "W" = axis is positioning in trapezoidal profile with WMS "X" = axis is positioning in S-curve profile with WMS "Y" = axis is operating in velocity mode with WMS "C" = axis is operating in velocity mode with continuous-path control "N" = axis is operating in piezo follow-up mode with WMS "?" = error, unknown state of axis	?ASTAT	IIOURTTJV
	?MSG	Read out the message exit buffer, the message exit buffer is used only for error messages, which concern the command interface (wrong command, missing parameters, invalid value). Possible messages are: "00 NO MESSAGE AVAILABLE" (will be returned after the attempt to read out the message buffer, even though no message is currently present) "01 PARAMETER BEFORE EQUAL WRONG" (will be written into the message buffer, if the command interpreter has failed to convert the parameter before the equals sign into a number correctly) "02 AXIS NUMBER WRONG" (will be written into the message buffer, if the command interpreter has failed to evaluate the given axis number correctly; valid: 1...9, e.g.) "03 PARAMETER AFTER EQUAL WRONG" (will be written into the message buffer, if the command interpreter has failed to convert the parameter after the equals sign into a number correctly) "04 PARAMETER AFTER EQUAL RANGE" (will be written into the message buffer, if the command interpreter has recognized that the parameter after the equals sign is beyond its valid range) "05 WRONG COMMAND ERROR" (will be written into the message buffer, if a syntax error has occurred, i.e., the command interpreter has not been able to recognize the given command) "06 REPLY IMPOSSIBLE"(will be returned, if the reply could not be transferred to the host, because the output buffer is not yet empty, e.g.) "07 AXIS IS IN WRONG STATE" (will be written into the message buffer, if a positioning command or a configuration parameter has been sent that could not be recognized because the axis is currently in a different motion state) "08 AXIS NOT RELEASED" (will be written into the message buffer, if a non-released axis is initialised) "09 ERROR IN POSITION TABLE" (will be written into the message buffer, if a problem with the positioning table occurs) "10 MPUNI CAN ERROR" (will be written into the message buffer, if there is an internal communications error)	?MSG	00 NO MESSAGE...

command group	command	function description	example	response
general status request	?ERR	Error query from the error memory with a memory depth of 20. The error number is always returned as number with four digits. Based on the error code its cause can be determined. If the returned value is 0, there are no further error messages stored.	?ERR	1211
	ERRCLEAR	Clear error memory.	ERRCLEAR	
	?ESTAT<n>	Read out current logical state of the limit switches and power stage feedback for an axis: bit 0 = MINSTOP, bit 1 = MINDEC, bit 2 = MAXDEC, bit 3 = MAXSTOP, bit 4 = motor power-stage error.	?ESTAT	10101
	?AXSIGNALS<n>	Query hardware axis signals for an axis: bit 0 = encoder CHA, bit 1 = encoder CHB, bit 2 = encoder Index, bit 3 = encoder Home, bit 4 = MAXSTOP, bit 5 = MINSTOP, bit 6 = AxisIn-Pin, bit 7 = Hall A, bit 8 = Hall B, bit 9 = Hall C, bit 10 = AxisOut-Pin, bit 11-15 = reserved.	?AXSIGNALS1	0000011101101001
	?MPUNISTAT<n>	Read status information about an axis: 1. axis number 2. error code 3. type 4. AD digits 5. version 6. signals		
	?READOWID<n>=<uv>	Read out memory contents of ONE-Wire-Chip from positioning unit until 0x00 limit detection and data transfer to the PC. As parameter the initial address 0x00 up to 0x70 will be passed in the ONE-Wire-Chip. From this Address max. 16 bytes will be detected or it will be read until the end recognition.	?READOWID1=0	INFO1|INFO2|....
	?READOWUB<n>	Read out memory contents of ONE-Wire-Chip from address 0x86 and 0x87 (=UserBytes) in the positioning unit and transmission of data to the PC.	?READOWUB1	90
base configuration	AXIS<n>=<uv>	Release or unreleased a certain axis. With this command one can release (1) or unrelease (0) an axis.	AXIS5=1	
	?AXIS<n>	Read out release status for an axis. If an axis is released, a "1" is sent back to the host, otherwise a "0" is returned.	?AXIS5	1
	MOTYPE<n>=<uv>	0 = DC brush 2 = Step motor Open Loop 3 = Step motor Closed-Loop 4 = BLDC	MOTYPE1=0	
	?MOTYPE<n>	Read out motor type for an axis.	?MOTYPE1	0
	AMPSHNT<n>=<uv>	Set current range for one axis: 0 = current range 1 (low) 1 = current range 2 (high)	AMPSHNT1=0	
	?AMPSHNT<n>	Read out preselected current range for one axis.	?AMPSHNT1	1
	TERM=<uv>	Set terminal mode: mode 0 = short response mode 1 = response with plain text mode 2 = response with plain text and OK handshake after each command without feedback	TERM=2	
	?TERM	Query terminal mode.	?TERM	2
	BAUDRATE	Set baud rate for the serial interface, allowed values : 9 600, 19 200, 38 400, 57 600, 115 200. This setting becomes active only after the next reset or power-on.	BAUDRATE=9600	
	?BAUDRATE	Read out current baud rate for the serial interface.	?BAUDRATE	9600
	COMEND	Set command end identification: 0 = CR, 1 = CR+LF, 2 = LF.	COMEND=0	
	?COMEND	Read out command end identification.	?COMEND	0
	SAVEGLOB	Store global parameters in serial FRAM.	SAVEGLOB	
	LOADGLOB	Recall global parameters out of the serial FRAM.	LOADGLOB	
	SAVEAXPA<n>	Store axis parameters for an axis in the serial FRAM.	SAVEAXPA1	
	LOADAXPA<n>	Recall axis parameters for an axis in the serial FRAM.	LOADAXPA1	
	?SERNUM	Query serial number of the PS 90+.	?SERNUM	09080145

command group	command	function description	example	response
base configuration	?VERSION	Read out software version of the firmware installed on the main board.	?VERSION	PS90-V8.0-xxxxxxxxx
	?MCTRVER	Return version data for the motion controller chips.		
	?PCHECK	Calculate and read out checksum of the program memory.	?PCHECK	12227
	JZONE=<uv>	Set inactive joystick zone (0-256).	JZONE=25	
	?JZONE	Read out inactive joystick zone.	?JZONE	25
	JZEROX=<uv>	Set center point for the X joystick.	JZEROX=505	
	?JZEROX	Read out center point for the X joystick.	?JZEROX	505
	JZEROY=<uv>	Set center point for the Y joystick.	JZEROY=515	
	?JZEROY	Read out center point for the Y joystick.	?JZEROY	515
	JZEROZ=<uv>	Set center point for the Z joystick.	JZEROZ=508	
	?JZEROZ	Read out center point for the Z joystick.	?JZEROZ	508
	JBUTTON=<uv>	Switch on/off the evaluation of joystick button.	JBUTTON=1	
	?JBUTTON	Read out whether the joystick button will be evaluated or not.	?JBUTTON	1
positioning operation	INIT<n>	Enable the motor power stage, release and activate position control loop. With this command, the axis is initialized completely, the motor is powered and the position control feedback loop is active. **This command must be transferred after switching-on the PS 90+, so that the axis can be taken into operation, using the commands REF, PGO, VGO, etc.** Before this, the following parameters must have been preset: motor type, limit switch mask and polarity, feedback control loop parameters, current range of the motor output stage.	INIT1	
	PSET<n>=<sv>	Set target position respectively relative travel distance (ABSOL/RELAT) for an axis. If absolute position format is switched on, then the parameter is interpreted as signed absolute position; if relative position indication is chosen, then the parameter is interpreted as signed travel distance. The new absolute target position is the sum of last absolute target position and transferred travel.	PSET2=100000	
	?PSET<n>	Read out target position resp. relative travel distance for an axis.	?PVEL1	10000
	PCHANGE<n>=<sv>	Change target position or distance of an axis, while this axis goes in the trapezoidal profile.	PCHANGE2= 50000	
	?CMDPOS	Read out current target position of the PID regulator.	?CMDPOS1	5000
	VVEL<n>=<sv>	Set target speed for velocity mode for an axis. With this command, the start speed and maybe also a new speed are transmitted, while the axis moves in the velocity mode.	VVEL1=-20000	
	?VVEL<n>	Read out target speed for velocity mode.	?VVEL1	-20000
	PGO<n>	Start positioning for an axis. The axis approaches the new target position in trapezoidal or S-curve profiling mode (see "PMOD").	PGO2	
	VGO<n>	Start velocity mode for an axis.	VGO2	
	MPGO=<uv>	Start positioning for mutiple axes. A numeric value is given which defines the affected axes. Bit 0 = axis 1... Bit 8 = axis 9	MPGO = 000000011	
	MVGO=<uv>	Start velocity mode for mutiple axes. A numeric value is given which defines the affected axes. Bit 0 = axis 1... Bit 8 = axis 9	MVGO = 000000011	
	STOP<n>	Stop motion of an axis. Any active motion command for an axis is interrupted. The drive decelerates with the preset ramp parameters and halts.		
	MSTOP=<uv>	Stop multiple axes. A numeric value is given which defines the affected axes. Bit 0 = axis 1... Bit 8 = axis 9	MSTOP = 000000011	
	VSTP<n>	Stop velocity mode for an axis. If an axis is in the velocity mode, this command will terminate this mode and stop the axis.	VSTP2	
	EFREE<n>	Release limit switch(es) of an axis. After a drive has moved onto a limit switch (MINSTOP, MAXSTOP) or brake switch (MINDEC, MAXDEC), the active switch(es) can be released using this command. The direction of the movement is automatically decided according to whether a positive or negative limit or break switch is activated.		

command group	command	function description	example	response
positioning operation	MON<n>	Enable the motor power stage and activate position control feedback loop. With this command, an axis that has been switched off previously (by means of the "MOFF" command) can be switched on again. Position control loop and the enable input for the power stage are activated.	MON1	
	MOFF<n>	Disable the motor power stage and deactivate position control feedback loop. With this command, position control loop and the enable input for the power stage are deactivated. The motor is switched off.	MOFF1	
	JOYON	Activate joystick mode for the predefined joystick axes. Thereafter, one up to three axes move in velocity mode. The velocity and the direction are given by joystick.	JOYON	
	JOYOFF	Terminate the joystick mode.	JOYOFF	
	CNT<n>=<sv>	Set current position counter for an axis.	CNT1=5000	
	?CNT<n>	Read out current position counter for an axis.	?CNT1	5000
	CRES<n>	Reset current position counter for an axis.	CRES1	
	?POSERR<n>	Read out the current position error for an axis. The difference between encoder position and default position is returned. May be used "on the fly" as well for read out of the contouring error.	?POSERR1	-15
	?VACT<n>	Read out current speed for an axis. This is returned in 16.16 format signed. The fractional part is "0" however, because the current speed is calculated on the basis of the position deviation between two samples.	?VACT2	1000
	?ENCPOS<n>	Read out current encoder position counter for an axis. Do not use this command with open-loop step-motor axes.	?ENCPOS1	5000
	?MXSTROKE<n>	Read out measured travel. This command reads out the travel ascertained by referencing in mode 6 and 7.	?MXSTROKE1	340000
positioning parameters	RELAT<n>	Set entry mode of coordinates for an axis to "relative" (= indication of the signed travel distance).	RELAT1	
	ABSOL<n>	Set entry mode of coordinates for an axis to "absolute" (= indication of the signed target position).	ABSOL2	
	?MODE<n>	Query the active/current entry mode of coordinates for one axis.	?MODE2	ABSOL
	PMOD<n>=<uv>	Set positioning mode trapezoidal/S-curve for an axis (0 = trapezoidal profiling mode, 1 = S-curve profiling mode).	PMOD1=0	
	?PMOD<n>	Read out positioning mode trapezoidal/S-curve for an axis.	?PMOD1	1
	PVEL<n>=<uv>	Set max. positioning velocity for an axis; used for the trapezoidal and S-curve profile.	PVEL1=10000	
	?PVEL<n>	Read out max. positioning velocity for an axis.	?PVEL1	10000
	FVEL<n>=<uv>	Set limit switch release speed for an axis (unsigned value).	FVEL1=1000	
	?FVEL<n>	Read out limit switch release speed for an axis.	?FVEL1	1000
	ACC<n>=<uv>	Set acceleration (= run-up ramp) for an axis, is used for all modi (trapezoidal, S-curve, velocity mode, etc.).	ACC1=100	
	?ACC<n>	Read out acceleration for an axis.	?ACC1	100
	DACC<n>=<uv>	Set deceleration (= slow-down ramp) for an axis, is used for all modi except S-curve.	DACC2=68	
	?DACC<n>	Read out deceleration for an axis.	?DACC2	68
	JACC<n>=<uv>	Set maximum "jerk" for an axis, is used only with S-curve profile.	JACC9=5	
	?JACC<n>	Read out maximum "jerk" for an axis.	?JACC9	5
	EDACC<n>=<uv>	Set emergency-stop deceleration for an axis. This is used when a brake switch has responded.	EDACC1=1000	
	?EDACC<n>	Read out emergency-stop deceleration for an axis.	?EDACC1	1000
	JVEL<n>=<sv>	Set maximum axis velocity for "joystick travel". With this command, the maximum velocity at maximum joystick deflection is defined.	JVEL3=1000	
	?JVEL<n>	Read out maximum axis velocity for "joystick travel".	?JVEL3	1000
	JOYACC<n>=<uv>	Set axis acceleration and deceleration for "joystick travel".		
	?JOYACC<n>	Read out axis acceleration and deceleration for "joystick travel".	?JOYACC3	100
	JAUTOMOFF<n>=<uv>	DC mode only: During joystick mode, automatically switch off motor when in target position.	JAUTOMOFF1=0	

command group	command	function description	example	response
positioning parameters	?JAUTOMOFF<n>	Read out automatic switch off setting in joystick mode.	?JAUTOMOFF1	0
	JPLAX=<n>	The joystick X axis (first axis of the joystick plane) is assigned to a certain axis number. If "0" is transferred, the X axis of the joystick is disabled.	JPLAX=2	
	?JPLAX	Read out joystick plane X axis assignment.	?PLAX	2
	JPLAY=<n>	The joystick Y axis (second axis of the joystick plane) is assigned to a certain axis number. If "0" is transferred, the Y axis of the joystick is disabled.	JPLAY=3	
	?JPLAY	Read out joystick plane Y axis assignment.	?JPLAY	3
	JPLAZ=<n>	The joystick Z axis (third axis of the joystick plane) is assigned to a certain axis number. If "0" is transferred, the Z axis of the joystick is disabled.	JPLAZ=3	
	?JPLAZ	Read out joystick plane Z axis assignment.	?JPLAZ	
	LIGO=<uv>	Start positioning with linear interpolation for a group of axis (binary definition mask). Bit-order: <axis 9, axis 8, axis 7, axis 6, ..., axis 2, axis 1>.	LIGO=000000111	
	IVEL<n>=<uv>	Set maximum velocity <uv> for axis<n> used for linear interpolation (signed).	IVEL1=50000	
	?IVEL<n>	Read out maximum velocity <uv> for axis<n> used for linear interpolation (signed).	?IVEL1	50000
	IACC<n>=<uv>	Set maximum acceleration <uv> for axis<n> used for linear interpolation (signed).	IACC3=2000	
	?IACC<n>	Read out maximum acceleration <uv> for axis<n> used for linear interpolation (signed).	?IACC3	2000
	POSTAB<uv>=<v>, <v>,...	Load table line in the path table; the value before the "="-sign indicates the table line number (0 to …), the value behind the "=" sign is for the table column follow separated by commas. Parameter list: 1. travel (signed) for axis 1 (-2147483648 to 2147483648) 2. travel (signed) for axis 2 (-2147483648 to 2147483648) 3. travel (signed) for axis 3 (-2147483648 to 2147483648) 4. travel (signed) for axis 4 (-2147483648 to 2147483648) 5. travel (signed) for axis 5 (-2147483648 to 2147483648) 6. travel (signed) for axis 6 (-2147483648 to 2147483648) 7. travel (signed) for axis 7 (-2147483648 to 2147483648) 8. travel (signed) for axis 8 (-2147483648 to 2147483648) 9. travel (signed) for axis 9 (-2147483648 to 2147483648) 10. function code Bit 3 to 0: output number 1 Bit 4: output level 1 Bit 8 to 5: output number 2 Bit 9: output level 2 Bit 13 to 10: output number 3 Bit 14: output level 3 Bit 15 of function code is set: move with a=const. Bit 15 of function code is deleted: move with v=const. 11. error byte 12. enable byte (always as numerical value)	POSTAB0=1000, 2000, 0, 0, 0, 0, 0, 0, 100, 0, 0, 3	
	?POSTAB<uv>	Load a table line out of the path table. The table line number (0 to …) is transferred as parameter. The table values are listed separated by commas. Parameter-list: 1. travel (signed) for axis 1 (-32760 to 32760) 2. travel (signed) for axis 2 (-32760 to 32760) 3. travel (signed) for axis 3 (-32760 to 32760) 4. travel (signed) for axis 4 (-32760 to 32760) 5. travel (signed) for axis 5 (-32760 to 32760) 6. travel (signed) for axis 6 (-32760 to 32760) 7. travel (signed) for axis 7 (-32760 to 32760) 8. travel (signed) for axis 8 (-32760 to 32760) 9. segment time in ms (16 Bit) 10. function code (16 Bit) 11. error byte (8 Bit) 12. enable byte (8 Bit) 13. velocity (32 bit) calculated by the plausibility check 14. acceleration (32 bit) calculated by the plausibility check	?POSTAB0	1000, 2000, 0, 0, 0, 0, 0, 0, 100, 0, 0, 3, 2100, 50,
	PTABPLAUS<uv>	Execute plausibility check for path table. The limits for velocity and acceleration are checked and the error bytes set, accordingly. Velocity and acceleration are calculated for each single table lines and ist values registered for the aktive axis with the highes axis number.	PTABPLAUS0	

command group	command	function description	example	response
positioning parameters	PTABGO<uv>,<uv>	With one parameter: Start path control at a certain table entry. With two parameters: Start path control a a certain table entry and stop it before a second table entry.	PTABGO0,15	
	PTABSTP	Cancel a current path control; the participating axes behave like at the end of the path table.	PTABSTP	
	PTABCLR	Delete table for path control.		
	PTABCIRCLE <uv>=<uv>, ...	Calculates circular interpolation and registers in the positioning table starting at the denoted start line. The parameters are passed as a list, separated by comma. The enable-bits of the axes are disjuncted bit by bit with the possibly already existing entries of the current table. 1. number of axis for X (0 for no X axis) 2. number of axis for Y (0 for no Y axis) 3. segment time in ms (16 bit) 4. function code (OR-template) for the segments (16 bit) 5. number of segment of the circle (16 bit) 6. signed radius (32 bit) 7. start angle in degree with signed value (16 bit) 8. angle range in degree with signed value (16 bit) 9. optionally scaling counter with signed value (16 bit) 10. optionally scaling denominator with signed value (16 bit)		
	PTABCPY <uv>=<uv>,<uv>	Copy an area of table for path control. The value before the "=" sign specifies the target index in the positioning table, the value behind the "=" sign specifies the source index and the value behind the comma the amount of lines that are to be copied.	PTABCPY 50=10, 20	
	PTABDEL<uv>=<uv>	Delete an area of the table for path control. The value before the "=" sign indicates the index of lines wherefrom it is to be deleted, the value behind the "=" sign indicates the amount of lines which are to be deleted.	PTABDEL50=20	
	PTABSAVE	Save table to flash.	PTABSAVE	
	PTABSAVE<uv,uv>	Save parts of the table to flash. The first parameter sets the starting line. The second the number of lines. Other lines in flash are not affected.	PTABSAVE3,10	
	PTABLOAD	Load complete table from flash to active memory.	PTABLOAD	
	PTABLOAD<uv,uv>	Load parts of the table to active memory. The first parameter sets the starting line the second the number of lines.		
	?PTABFLASHBUSY	Check if a saving operation is active (=1). If so, the control unit should not be switched off or restarted.	?PTABFLASH-BUSY	0
Automatic Reaction To External Triggers And Setting of Outputs	TRIG<uv>=<a>,<b>, <c>,[<d>[,<e>]]	Write entry to action table. <uv>=line number <a>=action ID <b>=input <c>=edge (0=falling, 1=rising) <d>=parameter 1, signed 32 bit decimal or hex string, e.g. axis number or axis mask <e>=opt. parameter 2 No empty characters! Prefixes are possible. Hex strings must start with 0x or 0X. Parameters d and e dependent on chosen action.	TRIG2=10,0,3,5 Line 2: Falling edge of input 10 starts positioning movement of axis 5.	
	?TRIG<uv>	Request line <uv> from action table. Output as decimal number..	?TRIG2	10,0,3,5
	EVENT<uv>=<a>, <b>,<c>,[<d>[,<e>]]	Write entry to event table. <uv>=line number <a>=event ID <b>=output mask 16 bit decimal or hex string <c>=output value mask 16 bit decimal or hex string <d>=parameter 1, signed 32 bit decimal or hex string, e.g. axis number or axis mask <e>=opt. parameter 2 No empty characters! Prefixes are possible. Hex strings must start with 0x or 0X. Parameters d and e dependent on chosen event.	EVENT6=2,0x24, 0x4,8,10000 EVENT6=2,36, 4,8,10000 Line 6: If velocity of axis 6 falls below 10000 set output 3 to 1 and output 6 to 0.	
	?EVENT<uv>	Request line <uv> from event table. Output as decimal number.	?EVENT6	2,36,4,8,10000

command group	command	function description	example	response
Automatic Reaction To External Triggers And Setting of Outputs	TRIGDEL[<a>[,<b>]]	No parameters: delete complete action table One paramter: Delete line a Two paramters: Delete lines a to a+b	TRIGDEL3,10	
	EVENTDEL[<a>[,<b>]]	No parameters: delete complete event table One paramter: Delete line a Two paramters: Delete lines a to a+b	EVENTDEL3,10	
	TRIGENABLE=<uv>	Activate (<uv>=1) or deactivate (<uv>=0) event and trigger functionality.	TRIGENABLE=1	
	?TRIGENABLE	Request state of event and trigger funtionality.	?TRIGENABLE	1
	TRIGTABSAVE	Save action table in flash.	TRIGTABSAVE	
	EVENTTABSAVE	Save event table in flash.	EVENTTABSAVE	
	?TRIGEVENTBUSY	Check if a saving operation is performed on flash. 0: no operation 1: saving to action table 2: saving to event table 3. saving to both tables During a saving operation the control unit should not be switched off or restarted.	?TRIGEVENT-BUSY	0
	TRIGTABLOAD	Load action table from flash to active memory.	TRIGTABLOAD	
	EVENTTABLOAD	Load event table from flash to active memory.	EVENTTABLOAD	
axis parameters	MCSTP<n>=<uv>	Set micro step resolution with the step motor axes.	MCSTP1=50	
	?MCSTP<n>	Read out micro step resolution with the step motor axes.	?MCSTP1	50
	DRICUR<n>=<uv>	Set driving current with step motors in percent of the maximum output value defined by the selected current range of the motor power stage. Set current limit for DC axes. 100 % refers to 12 A. This value will only be set after a cold start of the control unit with the first initialisiation.	DRICUR1=50	
	?DRICUR<n>	Read out driving current with step motors in percent.	?DRICUR1	50
	HOLCUR<n>=<uv>	Set holding current with step motor axes in percent.	HOLCUR1=30	
	?HOLCUR<n>	Read out holding current with step motor axes in percent.	?HOLCUR1	30
	ATOT<n>=<uv>	Set timeout in milliseconds, "0" switches off the timeout monitoring.	ATOT1=20000	
	?ATOT<n>	Query time-out for the axes.	?ATOT1	20000
	FKP<n>=<uv>	Set control parameter KP for an axis.	FKP1=25	
	?FKP<n>	Query control parameter KP for an axis.	?FKP1	25
	FKD<n>=<uv>	Set control parameter KD for an axis.	FKD1=5	
	?FKD<n>	Query control parameter KD for an axis.	?FKD1	5
	FKI<n>=<uv>	Set control parameter KI for an axis.	FKI1=10	
	?FKI<n>	Query control parameter KI for an axis.	?FKI1	10
	FIL<n>=<uv>	Set control parameter integration limit for an axis.	FIL1=100000	
	?FIL<n>	Query control parameter integration limit for an axis.	?FIL1	100000
	FST<n>=<uv>	Set sample time for an axis (in micro seconds).	FST1=500	
	?FST<n>	Query sample time for an axis (in micro seconds).	?FST1	500
	FDT<n>=<uv>	Set delay time of the differential term (KD) for an axis (in sample time cycles).	FDT1=5	
	?FDT<n>	Query delay time of the differential term (KD) for an axis (in sample time cycles).	?FDT1	5
	MXPOSERR<n>=<uv>	Set maximum position error for a servo axis. If the value is exceeded, the axis is deactivated. This works only for the motor types DC Brush, step motor Closed-Loop and BLDC.	MXPOSERR1=50	
	?MXPOSERR<n>	Read out maximum position error for an axis.	?MXPOSERR1	50
	MAXOUT<n>=<uv>	Set maximum output value for the servo loop in percent. With this command, the maximum value for an axis to be returned at the servo amplifier can be set. **Maximum value allowed: 99%.**	MAXOUT1=95	
	?MAXOUT<n>	Read out maximum output value in percent.	?MAXOUT1	95
	MPUNIPID<n>=<uv1>, <uv2>, <uv3>, <uv4>	Set PID current control values for step motor mode. uv1 = KP, uv2 = KI fast, uv3 = KI slow, uv4 = KD	MPUNIPID1=900,0,40,1500	
	?MPUNIPID<n>	Read out PID current control values.	?MPUNIPID1	900|0|40|1500

command group	command	function description	example	response
axis parameters	INPOSMOD<n>=<uv>	Set end-of-motion reporting mode: 0 = target position reached, 1 = actual position is within the settling window for a time defined by "INPOSTIM" command.	INPOSMOD1=0	
	?INPOSMOD<n>	Read out end-of-motion reporting mode.	?INPOSMOD1	0
	INPOSTIM<n>=<uv>	Set end-of-motion reporting time in multiples of the cycle time.	INPOSTIM1=1000	
	?INPOSTIM<n>	Read out end-of-motion reporting time.	?INPOSTIM1	1000
	INPOSWND<n>=<uv>	Set end-of-motion settling window in encoder counts.	INPOSWND1=50	
	?INPOSWND<n>	Read out end-of-motion settling window.	?INPOSWND1	50
	AMPPWMF<n>=<uv>	Set PWM output frequency of drive controller board, 20000 or 80000 is possible.	AMPPWMF1= 20000	
	?AMPPWMF<n>	Read out PWM frequency of drive controller board.	?AMPPWMF1	20000
	ENCLINES<n>=<uv>	Set encoder line number for an axis.	ENCLINES1=500	
	?ENCLINES<n>	Read out encoder line number for an axis.	?ENCLINES1	500
	MOTPOLES<n>=<uv>	Set motor pole number for an axis.	MOTPOLES1=25	
	?MOTPOLES<n>	Read out motor pole number for an axis.	?MOTPOLES1	25
	BLDCCT<n>=<uv>	Set commutation mode with BLDC: 0 = block commutation with Hall sensors 1 = sine commutation with encoder	BLDCCT1=0	
	?BLDCCT<n>	Query commutation mode with BLDC.	?BLDCCT1	0
	ELCYCNT<n>=<uv>	Set encoder counts for one electrical commutation cycle.	ELCYCNT1=128	
	?ELCYCNT<n>	Read out encoder counts for one electrical commutation cycle.	?ELCYCNT1	128
	PHINTIM<n>=<uv>	Set phase initialization time in multiples of the cycle time.	PHINTIM1=10	
	?PHINTIM<n>	Read out phase initialization time in multiples of the cycle time.	?PHINTIM1	10
	PHINAMP<n>=<uv>	Set phase initialisation amplitude in %.	PHINAMP1=50	
	?PHINAMP<n>	Read out phase initialisation amplitude in %.	?PHINAMP1	50
limit switch configuration / reference motion	REF<n>=<uv>	Start reference travel while indicating the reference mode for one axis: mode 0 = search next index impulse and stop mode 1 = approach reference switch and stop mode 2 = approach reference switch, search next index impulse and stop mode 3 = mode 0, additionally set act. position to "0" mode 4 = mode 1, additionally set act. position to "0" mode 5 = mode 2, additionally set act. position to "0" mode 6 = approach maximum reference switch, approach minimum reference switch, set current position to 0 mode 7 = approach minimum reference switch, approach maximum reference switch, set current position to 0		
	RVELS<n>=<sv>	Set reference travel speed "slow" for one axis. Using this speed, the index pulse will be searched or the reference switch will be released, respectively (signed value).	RVELS2=2000	
	?RVELS<n>	Read out reference travel speed "slow" for an axis.	?RVELS2	2000
	RVELF<n>=<sv>	Set reference travel speed "fast" for an axis. The drive moves with this speed towards the limit switch (signed value).	RVELF2=-20000	
	?RVELF<n>	Read out reference travel speed "fast" for an axis.	?RVELF2	-20000
	RDACC<n>=<uv>	Set reference travel deceleration for an axis. This value is used when the reference point is approached.	RDACC1=1000	
	?RDACC<n>	Read out reference travel deceleration for an axis.	?RDACC1	1000
	SMK<n>=<uv>	Set limit switch mask for an axis. This command activates or deactivates the limit and break switches. If a limit switch is approached, the movement is stopped abruptly and the motor is shut off. Bit sequence : <MAXSTOP, MAXDEC, MINDEC, MINSTOP>.	SMK3=0110	
	?SMK<n>	Read out limit switch mask for an axis.	SMK3	0110
	SPL<n>=<uv>	Set limit switch polarity for an axis. With this command, the active level for the limit and brake switches is defined. Bit sequence: <MAXSTOP, MAXDEC, MINDEC, MINSTOP>.	SPL3=1111	
	?SPL<n>	Read out limit switch polarity for an axis.	SPL3	1111

command group	command	function description	example	response
limit switch configuration / reference motion	RMK<n>=<uv>	Set reference switch mask for an axis. With this command, it can be defined which of the four limit switches for an axis should be interpreted as reference switch. A mask with exactly one character "1" has to be transferred. Bit sequence: <MAXSTOP, MAXDEC, MINDEC, MINSTOP>.	RMK3=0001	
	?RMK<n>	Read out reference switch mask for one axis.	?RMK3	1
	RPL<n>=<uv>	Set reference switch polarity for one axis. This command defines the active level of the reference switch. Bit sequence: <MAXSTOP, MAXDEC, MINDEC, MINSTOP>.	RPL3=1110	
	?RPL<n>	Read out reference switch polarity for one axis.	?RPL3	1110
	?HYST<n>	Read out reference switch hysteresis for an axis. After a reference motion has been terminated successfully, the hysteresis of the switch can be read out with this command. (The value is correct only, if none of the reference/limit switches is active any more).	?HYST1	28
	?REFST<n>	Inquiry of reference motion validity. When reference motion successfully completed, the status is set on 1 = "valid". If a motor without encoder is switched off (e.g. open-loop stepper), then the validity is reset to "0".	?REFST	1
	LMK<n>=<uv>	Set mask for limit-positioning monitoring for the axis. With this command the limit-positioning monitoring for tho lower limit and/or upper limit for position can be set activ and inactive, respectively. The limit-positioning monitoring behaves like the according DEC-switch. Bit sequence: <MAXDEC, MINDEC>.	LMK1=01	
	?LMK<n>	Read out limit-positioning monitoring mask for axis	?LMK1	01
	?LSTAT<n>	Read current, logical state of limit-positioning monitoring for the axis. Bit 0 = MINDEC lower limit is transcanded Bit 1 = MAXDEC upper limit is transcanded	?LSTAT1	01
	SLMIN<n>=<uv>	Set negative limit position for the axis.	SLMIN1=100	
	?SLMIN<n>	Read out negative limit position for the axis.	?SLMIN1	100
	SLMAX<n>=<uv>	Set positve limit position for the axis.	SLMAX1=100000	
	?SLMAX<n>	Read out positive limit position for the axis.	?SLMAX1	100000
in-/outputs	ETTLOUTS<n>=<bin>	Set TTL outputs for the motor power stage of an axis. The axis number and a binary set mask are transferred.	ETTLOUTS1=10	
	ETTLOUTC<n>=<bin>	Reset TTL outputs for the motor power stage of an axis. The axis number and a binary delete mask are transferred.	ETTLOUTC1=01	
	?INPUTS	Read out current state of the inputs (16-bit binary digit).	?INPUTS	0010100100101101
	?INPTTL	Read out current state of all TTL inputs (8-bit binary digit).	?INPTTL	10100100
	?INPSPS	Read out current state of all SPS inputs (8-bit binary digit).	?INPSPS	00110011
	OUTPUT<uv>=<uv>	Change current state of an output.	OUTPUT1=0	
	?OUTPUTS	Read out current state of all outputs.	?OUTPUTS	0010100100101101
	OUTTTL<uv>=<uv>	Change current state of a TTL output.	OUTTTL1=0	
	?OUTTTL	Read out current state of all TTL outputs.	?OUTTTL	00101001
	OUTSPS<uv>=<uv>	Change current state of a SPS output.	OUTSPS1=0	
	?OUTSPS	Read out current state of all SPS outputs.	?OUTSPS	00101001
	INMODE=<uv>	Select input level TTL/SPS (0 = TTL, 1 = SPS).	INMODEO	
	?INMODE	Query currently set input level TTL/SPS.	?INMODE	0
	?ANIN<uv>	Query analog input, the port number from 1 to 8 will be set, and the converted 10-bit value will be returned.	?ANIN3	234
	DAOUT<uv>=<uv>	Set analog output, the port number from 1 to 8 and the output value for the DA converter are set.	DAOUT2=250	
	?DAOUT<uv>	Query analog output, the port number from 1 to 8 is entered, the current digital value that has been set is returned.	?DAOUT2	250
	OPWM<uv>=<uv>	Set PWM output, the port number from 1 to 8 and the level control value are set from 0 to 100 %.	OPWM1=55	
	?OPWM<uv>	Query PWM output, the port number from 1 to 4 is entered and the level control value that has been set is returned from 0 to 100 %.	?OPWM1	55
	AXOUTPUT<n>=<uv>	Set axis out-pin for an axis to high/low.		
	?IOCONFIG	Read out current I/O configuration.	?IOCONFIG	15

command group	command	function description	example	response
follow up control	?APWMS<n>	Read out current position of linear measuring system.	APWMS4	3000
	WMSRES<n>	Set current position of measuring system of an axis to 0 (is not required and must not be used after reference scan, as the position will be lost.	WMSRES4	
	?MXWMSSTRK<n>	Query maximum travel of linear measuring system.	?MXWMSSTRK2	
	WMSFAKZ<n>=<uv>	Set numerator factor for conversion of the resolution of the WMS to the resolution of the actuator for positioning with follow up control.	WMSFAKZ1=1	
	?WMSFAKZ<n>	Query numerator factor for positioning with follow up control.	?WMSFAKZ1	1
	WMSFAKN<n>=<uv>	Set denominator factor for conversion of the resolution of the WMS to the resolution of the actuator for positioning with follow up control.	WMSFAKN1=5	
	?WMSFAKN<n>	Query denominator factor for positioning with follow up control.	?WMSFAKN1	5
	PWMSSET<n>=<sv>	set target position and relative travel, respectively for an axis (preselection is, by analogy to the normal positioning without follow up control, via the commands ABSOL and RELAT, respectively); if absolute position indication is selected, the parameter is interpreted as absolute position with sign. If relative position indication is chosen, the parameter is interpreted as travel with sign. The new absolute target position is now calculated by the sum of the last target position and travel.	PWMSSET2= 100000	
	?PWMSSET<n>	Read out target position and relative travel, respectively, for an axis.	?PWMSSET2	100000
	PWMSGO<n>	Start positioning with WMS at an axis. The axis moves to the new target position either in trapezidol- or S-shaped profile (see PMOD).	PWMSGO2	
	PWMSWIN<n>=<uv>	Set half target window width for the positioning with WMS (whole width of target position = ±PWMSWIN).	PWMSWIN1=10	
	?PWMSWIN<n>	Query half target window width for positioning with WMS.	?PWMSWIN1	10
	WMSVEL<n>=<uv>	Set follow up velocity for positioning with WMS (without sign).	WMSVEL1=100	
	?WMSVEL<n>	Query follow up velocity for positioning with WMS.	?WMSVEL1	
	PWMSSTP<n>	Stop positioning with WMS at an axis. If the axis at position with MS in phase 3, the mode has to be stopped by this command before moving the axis with a new command.	PWMSSTP1	
	PWMSMODE<n>=<uv>	Set positioning mode for positioning with WMS. Mode 0: coarse positioning (phase 1). Mode 1: coarse positioning (phase 1), iteration (phase 2). Mode 2: coarse positioning (phase 1), iteration (phase 2), position correction (phase 3), phase 3 is active. Mode 3: coarse positioning (phase 1), position correction (phase 3), phase 3 is active. Mode 4: coarse positioning (phase 1), iteration (phase 2), position correction (phase 3), phase 3 is finished in target window. Mode 5: coarse positioning (phase 1), position correction (phase 3), phase 3 is finished in target window. Mode 6: coarse positioning (phase 1), iteration (phase 2), hybrid positioning (phase 3), phase 3 is finished in target window. Mode 7: coarse positioning (phase 1), iteration (phase 2), hybrid positioning (phase 3), phase 3 is active. Mode 8: coarse positioning (phase 1), position correction (phase 2), hybrid positioning (phase 3), phase 3 is finished in target window. Mode 9: coarse positioning (phase 1), position correction (phase 2), hybrid positioning (phase 3), phase 3 is active.	PWMSMODE1=1	
	?PWMSMODE<n>	Query positioning mode for positioning with WMS.	?PWMSMODE1	1
	?PWMSSTATE<n>	Read out state of positioning with WMS of a given axis. Bit 0: axis positions with WMS Bit 1: axis positions with WMS and is in phase 1 Bit 2: axis positions with WMS and is in phase 2 Bit 3: axis positions with WMS and is in phase 3 Bit 4: axis reached the default target window	?PWMSSTATE1	16
	?PWMSERR<n>	Read out current position error of an axis when positioning with WMS.	?PWMSERR1	
	WMSINV<n>=<uv>	Reverse count direction of WMS (1=yes / 0=no).	WMSINV1=0	
	?WMSINV<n>	Read out whether count direction of WMS is reversed (yes/no).	?WMSINV1	0
	WMSOFFS<n>=<sv>	Set signed offset for hybrid coarse positioning with WMS.	WMSOFFS1=-80	
	?WMSOFFS<n>	Read out signed offset for hybrid coarse positioning with WMS.	?WMSOFFS1	80

command group	command	function description	example	response
follow up control	PWMSPWIN<n>=<uv>	Set half target window width for hybrid positioning with WMS (phase 3).	PWMSPWIN1=0	
	?PWMSPWIN<n>	Read out half target window width for hybrid positioning with WMS (phase 3).	?PWMSPWIN1	0
	?PVOLTG<n>	Read out current hybrid output value.	?PVOLTG1	487
	?DACINPUTS	Read hybrid error state (as bit pattern). Bit 0: error, hybrid axis 1 Bit 1: error, hybrid axis 2 Bit 2: error, hybrid axis 3 Bit 3: error, hybrid operating voltage 1 Bit 4: error, hybrid axis 4 Bit 5: error, hybrid axis 5 Bit 6: error, hybrid axis 6 Bit 7: error, hybrid operating voltage 2	?DACINPUTS	11110001
	PWMSPTIM<n>=<uv>	Set cycle time of the hybrid positioning of an axis with follow-up control.	PWMSPTIM1=1	
	?PWMSPTIM<n>	Read out cycle time of the hybrid positioning of an axis with follow-up control.	?PWMSPTIM1	1
	PWMSPMXO<n>=<uv>	Set maximum hybrid output value of an axis.	PWMSPMXO1=4095	
	?PWMSPMXO<n>	Read out maximum hybrid output value of an axis.	?PWMSPMXO1	4095
holding brake control	HBCH<n>=<uv>	Assign PWM output for holding brake to an axis: <AxisNumber> = <PWM port> PWM port = 0 for holding break function off.	HBCH8=3	
	?HBCH<n>	Query holding brake assignment PWM port to an axis.	?HBCH8	3
	HBFV<n>=<uv>	Set first PWM value to activate the holding brake: <AxisNumber> = <Percent Value>.	HBFV8=50	
	?HBFV<n>	Query first PWM value for activation of the holding brake.	?HBFV8	50
	HBSV<n>=<uv>	Set second PWM value for clamping the holding brake: <AxisNumber> = <Percent Value>.	HBSV8=20	
	?HBSV<n>	Query second PWM value for clamping the holding brake.	?HBSV8	20
	HBTI<n>=<uv>	Set settling time for the holding brake. The first PWM value will be set for this amount of time after activation of the holding brake: <AxisNumber> = <Time for first PWM value in ms>.	HBTI8=300	
	?HBTI<n>	Query settling time for the holding brake. The first PWM value will be set for this amount of time after activation of the holding brake.	?HBTI8	300
reset	RESETAC	Activate motor driver board Reset.	RESETAC	
	RESETMB	Activate main board Reset.	RESETMB	
stand-alone programming	SAMEM	Set flag value for stand-alone programming.	SAMEM38=50	
	?SAMEM	Query flag value for stand-alone programming.	?SAMEM38	50
	SAEXEC	Start (1) / Stop (0) stand-alone program execution.	SAEXEC0	
	SASTEP	Run a stand-alone program line, the line number is sent and the line number of the next line is returned.	SASTEP1	2
	SALOAD	Download a stand-alone program line, the corresponding line number and the contents of the program line as Hex-Dump (16 byte) in ASCII format are sent.	SALOAD11=04000015F900...	
	SACHKS	Update checksum of the stand-alone program, after a new stand-alone program has been loaded.		
Anybus® module	ABNETADR=<uv>	Set fixed Anybus IPv4 address as decimal number. Active after global save and restart.	(ABNETADR=168428041 (for 10.10.2.9) ABNETADR=3232235628 (for 192.168.0.108)	

command group	command	function description	example	response
Anybus® module	ABNETADR1=<uv>	Set fixed Anybus IPv4 address in dot notation. Active after global save and restart.	ABNETADR1= 10.10.2.9 ABNETADR1= 192.168.0.108	
	?ABNETADR	Request fixed Anybus IPv4 address as decimal number. Returns last saved address.	?ABNETADR	168428041
	?ABNETADR1	Request fixed Anybus IPv4 address in dot notation. Returns last saved address.	?ABNETADR1	10.10.2.9
	ABNETSUB=<uv>	Set fixed Anybus IPv4 netmask as decimal number. Active after global save and restart.	ABNETSUB= 168428032 (for 10.10.2.0)	
	ABNETSUB1=<uv>	Set fixed Anybus IPv4 netmask in dot notation. Active after global save and restart.	ABNET- SUB1=10.10.2.0	
	?ABNETSUB	Request fixed Anybus IPv4 netmask as decimal number. Returns last saved netmask.	?ABNETSUB	168428032
	?ABNETSUB1	Request fixed Anybus IPv4 netmask in dot notation. Returns last saved netmask.	?ABNETSUB1	10.10.2.0
	ABNETGW=<uv>	Set fixed Anybus IPv4 gateway address as decimal number. Active after global save and restart.	ABNETGW= 168248287 (for 10.10.2.255)	
	ABNETGW1=<uv>	Set fixed Anybus IPv4 gateway address in dot notation. Active after global save and restart.	ABNETGW1= 10.10.2.255	
	?ABNETGW	Request fixed Anybus IPv4 gateway address as decimal number. Returns last saved address.	?ABNETGW	168248287
	?ABNETGW1	Request fixed Anybus IPv4 gateway address in dot notation. Returns last saved address.	?ABNETGW1	10.10.2.255
	ABNETDHCP=<uv>	Activate (=1) or deactivate (=0) Anybus DHCP Active after global save and restart.	DHCP=0 DHCP=1	
	?ABNETDHCP	Request Anybus DHCP status. Returns last saved staus.	?DHCP	0
	ABNETCOM=<uv>	Set Anybus network communication parameters, e.g. bit rate.	ABNETCOM=10	
	?ABNETCOM	Request Anybus network communication parameters, e.g. bit rate..	?ABNETCOM	10

II Parameter Relevance for the different Motor Types

parameter	DC brush	2-phase step motor open-loop	2-phase step motor closed-loop	BLDC
MOTYPE	+	+	+	+
AXIS	+	+	+	+
FKP	+	–	+	+
FKD	+	–	+	+
FDT	+	–	+	+
FKI	+	–	+	+
FIL	+	–	+	+
FST	+	–	+	+
MAXOUT	+	+[1)]	+	+
MXPOSERR	+	–	+	+
SMK	+	+	+	+
SPL	+	+	+	+
RMK	+	+	+	+
RPL	+	+	+	+
RVELF	+	+	+	+
RVELS	+	+	+	+
ACC	+	+	+	+
DACC	+	+	+	+
JACC	+	+	+	+
PVEL	+	+	+	+
EDACC	+	+	+	+
FVEL	+	+	+	+
ABSOL	+	+	+	+
RELAT	+	+	+	+
PMOD	+	+	+	+
AMPPWMF	+	+	+	+
MCSTP	–	+	–	–
DRICUR	–	+	+	–
HOLCUR	–	+	+	–
AMPSHNT	+	+	+	+

parameter	DC brush	2-phase step motor open-loop	2-phase step motor closed-loop	BLDC
MOTPOLES	–	–	+	+
ENCLINES	–	–	+	+
ELCYCNT	–	–	+	+
BLDCCT	–	–	+	+
PHINTIM	–	–	+	+
PHINAMP	–	–	+	+
ATOT	+	+	+	+
INPOSTIM	+	–	+	+
INPOSWND	+	–	+	+
INPOSMOD	+	–	+	+
HBCH	(+)	(+)	(+)	(+)
HBFV	(+)	(+)	(+)	(+)
HBTI	(+)	(+)	(+)	(+)
HBSV	(+)	(+)	(+)	(+)
JPLAX	(+)	(+)	(+)	(+)
JPLAY	(+)	(+)	(+)	(+)
JPLAZ	(+)	(+)	(+)	(+)
JOYACC	(+)	(+)	(+)	(+)
JVEL	(+)	(+)	(+)	(+)
JZONE	(+)	(+)	(+)	(+)
JZEROX	(+)	(+)	(+)	(+)
JZEROY	(+)	(+)	(+)	(+)
JBUTTON	(+)	(+)	(+)	(+)

\+ neccessary
-- not neccessary
(+) optional

1) The command may be used, however, it is important that the value set here is larger than or equal to the maximum PWM value for DRICUR or HOLCUR. In any case, the output is limited to the value defined by MAXOUT. If a too small value is selected, the micro-step operation will not work properly.

III Connecting Table

TTL In-/Outputs

pin assignment of the 25-pin D-Sub male connector

TTL-I/O	pin
input 1	16
input 2	17
input 3	18
input 4	19
input 5	20
input 6	21
input 7	22
input 8	23
output 1	3
output 2	4
output 3	5
output 4	6
output 5	7
output 6	8
output 7	9
output 8	10
+ 5 V, max. 300 mA total current	1, 2, 14, 15
GND	11, 12, 24, 25
n. c.	13

SPS In-/Outputs

pin assignment of the 25-pin D-Sub female connector

SPS-I/O	pin
input 1	16
input 2	17
input 3	18
input 4	19
input 5	20
input 6	21
input 7	22
input 8	23
output 1	3
output 2	4
output 3	5
output 4	6
output 5	7
output 6	8
output 7	9
output 8	10
+24 V, max. 1000 mA total current	1, 2, 14, 15
GND	11, 12, 24, 25
n. c.	13

Analog In-/Outputs

pin assignment of the 25-pin D-Sub male connector

analog-I/O	pin
input 1	6
input 2	5
input 3	4
input 4	3
input 5	10
input 6	9
input 7	8
input 8	7
output 1	23
output 2	22
output 3	21
output 4	20
output 5	19
output 6	18
output 7	17
output 8	16
+ 5 V, max. 300 mA total current	1, 2, 14, 15
GND	11, 12, 24, 25
U_{ref} output 4.096 V	13

RS-232

pin assignment oft he 9-pin D-Sub (female)

RS-232	pin
CD	1
RX	2
TX	3
DTR	4
GND	5
DSR	6
RTS	7
CTS	8
RI	9

Universal Motor Connector

The positioning units are connected using the suitable OWIS® connecting cable. The universal motor connector enables the current supply of the motor, control of the motor holding brake, where applicable, and the transfer of the encoder, limit-switch or Hall-effect sensor signals (if any).

Pin assignment of the 37-pin D-Sub female connector:

	pin	DC motor	step motor OL	BLDC
performance	19	motor +	phase 1 +	U
	18	motor -	phase 1 -	V
	17	motor +	phase 2 +	W
	16	motor -	phase 2 -	–

signals	15	motor type encoding
	14	motor type encoding
	13	GND
	12	+ 5 V
	11	encoder A
	10	encoder $\overline{A}$
	9	encoder B
	8	encoder $\overline{B}$
	7	encoder Index
	6	encoder $\overline{Index}$

switches + signals	5	MINSTOP		
	4	MINDEC		
	3	MAXDEC		
	2	MAXSTOP		
	1	GND		
	37	motor holding brake +24 V		
	36	motor holding brake -		
	35	(reserved)		
	34	(reserved)		
	33	(reserved)		
	32	(reserved)		
	31	GND		
	30	+ 5 V		
	29	(reserved)		
	28			Hall sensor A+
	27			Hall sensor A–
	26			Hall sensor B+
	25			Hall sensor B–
	24			Hall sensor C+
	23			Hall sensor C–
	22	+ 5 V		
	21	GND		
	20	+24 V		

Connecting Cable

1. signal cable "Twisted Pair" 8 x 2 x 0.15 mm² with overall shielding and star quad core, with additional shield, 4 x 0.25 mm²

pair no.	color wire 1	color wire 2	cross sectional area
1	red	blue	0.15 mm²
2	white	brown	0.15 mm²
3	green	yellow	0.15 mm²
4	grey	pink	0.15 mm²
5	black	violet	0.15 mm²
6	grey/pink	red/blue	0.15 mm²
7	orange	orange/black	0.15 mm²
8	transparent	transparent/red	0.15 mm²
9 a	green/white	green/brown	0.25 mm²
9 b	yellow/white	yellow/brown	0.25 mm²

2. motor cable with overall shielding

core no.	color	cross sectional area
1	red	0.6 mm²
2	blue	0.6 mm²
3	white	0.6 mm²
4	black	0.6 mm²
5	brown	0.6 mm²
6	pink	0.5 mm²
7	grey	0.5 mm²

Recommendation for a RS-232 Interface Cable

To connect to a PC a standard cable with 1:1 connctions is used.

EU/UE Konformitätserklärung/Declaration of conformity

Wir
We

OWIS GmbH
Im Gaisgraben 7
79219 Staufen / Germany
+49(0)7633/9504-0
+49(0)7633/9504-44
www.owis.eu
info@owis.eu

erklären in alleiniger Verantwortung, dass das Produkt
declare under our sole responsibility that the product

PS 90+

auf das sich diese Erklärung bezieht, mit den folgenden Normen oder normativen Dokumenten übereinstimmt.
to which this declaration relates is in conformity with the following standards or other normative documents.

EN 55011:2016 + A1:2017; EN 61000-6-2:2005; EN 61000-3-2:2014; EN 61000-3-3:2013
EN 61010-1:2010; EN 50581:2012

Gemäss den Bestimmungen der Richtlinie:
Following the provisions of directive:

2014/30/EU; 2014/35/EU; 2011/65/EU

Ort und Datum der Ausstellung
Place and date of issue

Staufen im Breisgau, 31.07.2020

Name und Unterschrift
Name and signature

D. J. Schuhen

P. Hilgers

Aktuelle Ausgabe: 10.10.03 DB / DSCH, 2.01.112 FO Konformitätserklärung