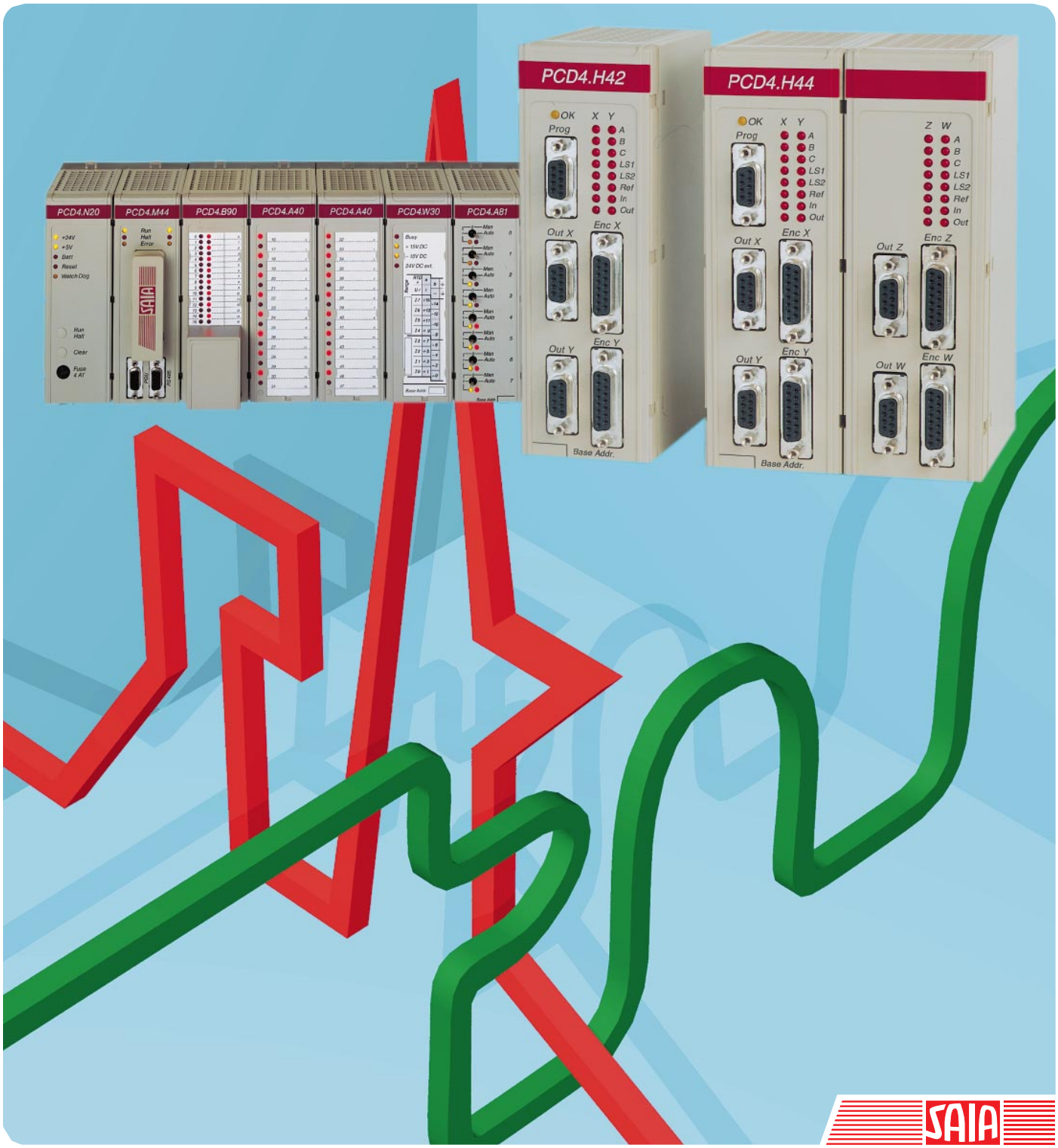
SAIA-Burgess Electronics

SWITCHES • MOTORS • CONTROLLERS

SAIA®PCD

Process Control Devices

Positioniermodul für Servoantriebe mit Linear- und Kreisinterpolation PCD4.H4..



Ausgabe 26/752 D1

SAIA-Burgess Electronics AG
Bahnhofstrasse 18
CH-3280 Murten (Schweiz)
http;//www.saia-burgess.com

GB: Electronic Controllers Telefon 026 / 672 72 72
Telefax 026 / 672 74 99

SAIA-Burgess Gesellschaften

Schweiz
SAIA-Burgess Electronics AG
Freiburgstrasse 33
CH-3280 Murten
☎ 026 672 77 77, Fax 026 670 19 83

Deutschland
SAIA-Burgess Electronics GmbH
Daimlerstrasse 1k
D-63303 Dreieich
☎ 06103 89 060, Fax 06103 89 06 66

Österreich
SAIA-Burgess Electronics Ges.m.b.H.
Schallmooser Hauptstrasse 38
A-5020 Salzburg
☎ 0662 88 49 10, Fax 0662 88 49 10 11

Italien
SAIA-Burgess Electronics S.r.l.
Via Cadamosto 3
I-20094 Corsico MI
☎ 02 48 69 21, Fax 02 48 60 06 92

Frankreich
SAIA-Burgess Electronics Sàrl.
10, Bld. Louise Michel
F-92230 Gennevilliers
☎ 01 46 88 07 70, Fax 01 46 88 07 99

Niederlande
SAIA-Burgess Electronics B.V.
Hanzeweg 12c
NL-2803 MC Gouda
☎ 0182 54 31 54, Fax 0182 54 31 51

Belgien
SAIA-Burgess Electronics Belgium
Avenue Roi Albert 1er, 50
B-1780 Wemmel
☎ 02 456 06 20, Fax 02 460 50 44

Ungarn
SAIA-Burgess Electronics Automation Kft.
Liget utca 1.
H-2040 Budaörs
☎ 23 501 170, Fax 23 501 180

Vertretungen

Gross-britannien
Canham Controls Ltd.
25 Fenlake Business Centre, Fengate
Peterborough PE1 5BQ UK
☎ 01733 89 44 89, Fax 01733 89 44 88

Dänemark
Malthe Winje Automation AS
Håndværkerbyen 57 B
DK-2670 Greve
☎ 70 20 52 01, Fax 70 20 52 02

Norwegen
Malthe Winje Automasjon AS
Haukelivn 48
N-1415 Oppegård
☎ 66 99 61 00, Fax 66 99 61 01

Schweden
Malthe Winje Automation AB
Truckvägen 14A
S-194 52 Upplands Våsby
☎ 08 795 59 10, Fax 08 795 59 20

Suomi/ Finnland
ENERGEL OY
Atomitie 1
FIN-00370 Helsinki
☎ 09 586 2066, Fax 09 586 2046

Portugal
INFOCONTROL Electronica e Automatismo LDA.
Praceta Cesário Verde, No 10 s/cv, Massamá
P-2745 Queluz
☎ 21 430 08 24, Fax 21 430 08 04

Spanien
Tecnosistemas Medioambientales, S.L.
Poligono Industrial El Cabril, 9
E-28864 Ajalvir, Madrid
☎ 91 884 47 93, Fax 91 884 40 72

Tschechische Republik
ICS Industrie Control Service, s.r.o.
Modranská 43
CZ-14700 Praha 4
☎ 2 44 06 22 79, Fax 2 44 46 08 57

Polen
SABUR Ltd.
ul. Druzynowa 3A
PL-02-590 Warszawa
☎ 22 844 63 70, Fax 22 844 75 20

Australien
Siemens Building Technologies Pty. Ltd.
Landis & Staefa Division
411 Ferntree Gully Road
AUS-Mount Waverley, 3149 Victoria
☎ 3 9544 2322, Fax 3 9543 8106

Argentinien
MURTEN S.r.l.
Av. del Libertador 184, 4° “A”
RA-1001 Buenos Aires
☎ 054 11 4312 0172, Fax 054 11 4312 0172

Kundendienst

USA
SAIA-Burgess Electronics Inc.
1335 Barclay Boulevard
Buffalo Grove, IL 60089, USA
☎ 847 215 96 00, Fax 847 215 96 06

Issue : 22.11.2000

SAIA® Process Control Devices

Positioniermodul für Servoantriebe mit Linear- und Kreisinterpolation

PCD4.H4x0

Ausgabe 26/752 D1 - 06.1997

Anpassungen

Handbuch: PCD4.H4x0 - Positioniermodul für Servoantriebe mit Linear-und Kreisinterpolation - Ausgabe D1

Datum	Abschnitt	Seite	Beschreibung

Inhalt

Seite

Notizen

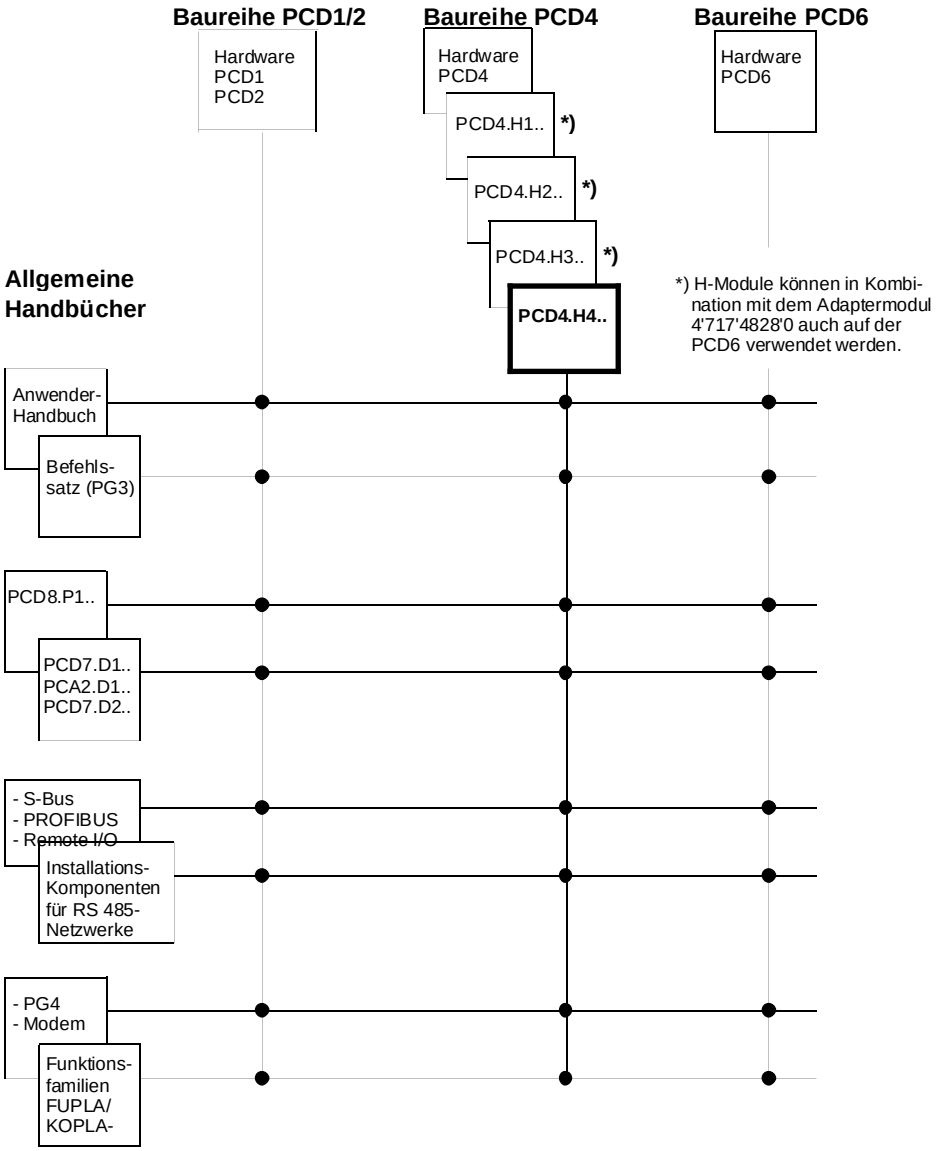
Wichtiger Hinweis:

Um den einwandfreien Betrieb von SAIA® PCD sicherstellen zu können, wurde eine Vielzahl detaillierter Handbücher geschaffen. Diese wenden sich an technisch qualifiziertes Personal, das nach Möglichkeit auch unsere Workshops erfolgreich absolviert hat.

Die vielfältigen Leistungen der SAIA® PCD treten nur dann optimal in Erscheinung, wenn alle in diesen Handbüchern aufgeführten Angaben und Richtlinien bezüglich Montage, Verkabelung, Programmierung und Inbetriebnahme genau befolgt werden.

Damit allerdings werden Sie zum grossen Kreis der begeisterten SAIA® PCD Anwendern gehören.

Übersicht



Zuverlässigkeit und Sicherheit elektronischer Steuerungen

Die Firma SAIA-Burgess Electronics AG konzipiert, entwickelt und stellt ihre Produkte mit aller Sorgfalt her:

- Neuster Stand der Technik
- Einhaltung der Normen
- Zertifiziert nach ISO 9001
- Internationale Approbationen: z.B. Germanischer Lloyd, Det Norske Veritas, CE-Zeichen ...
- Auswahl qualitativ hochwertiger Bauelemente
- Kontrollen in verschiedenen Stufen der Fertigung
- In-Circuit-Tests
- Run-in (Wärmelauf bei 85°C während 48h)

Die daraus resultierende hochstehende Qualität zeigt trotz aller Sorgfalt Grenzen. So ist z.B. mit natürlichen Ausfällen von Bauelementen zu rechnen. Für diese gibt die Firma SAIA-Burgess Electronics AG Garantie gemäss den "Allgemeinen Lieferbedingungen".

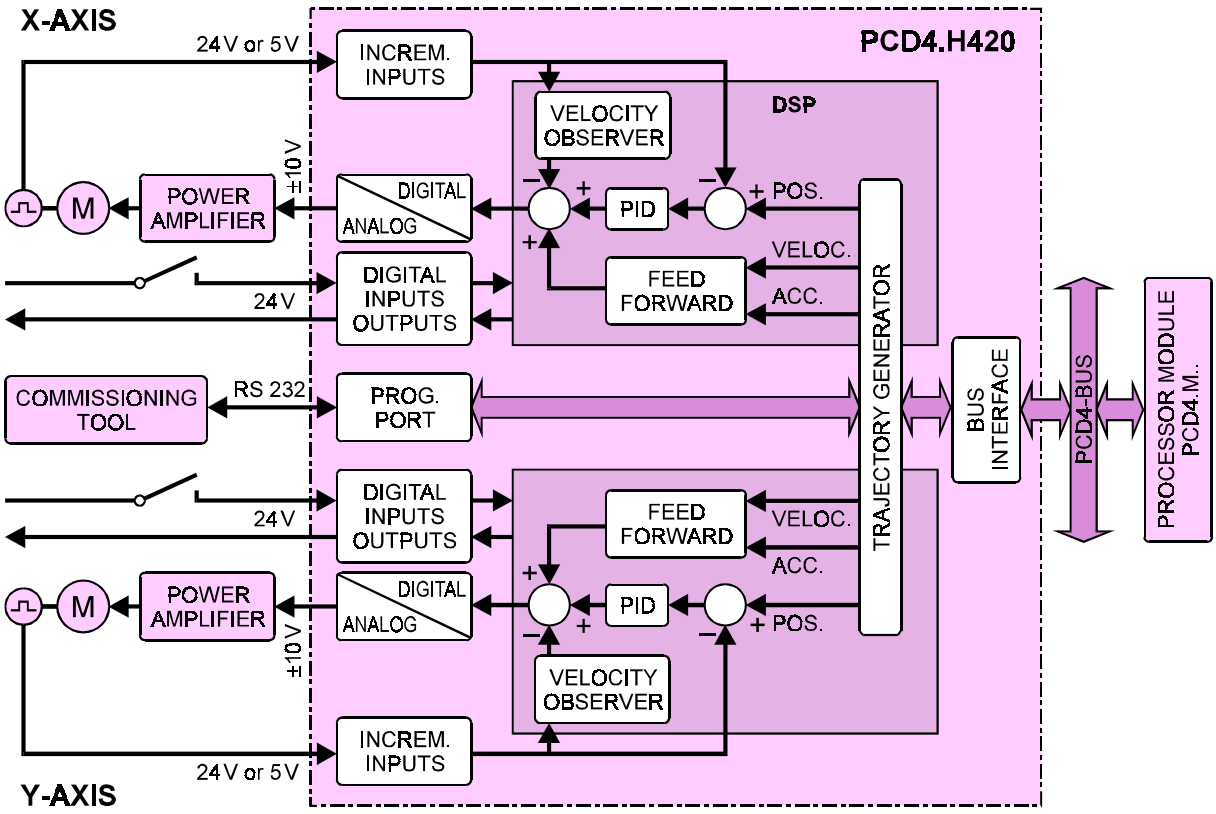
Der Anlagebauer seinerseits muss auch seinen Teil für das zuverlässige Arbeiten einer Anlage beitragen. So ist er dafür verantwortlich, dass die Steuerung datenkonform eingesetzt wird und keine Überbeanspruchungen, z.B. auf Temperaturbereiche, Überspannungen und Störfelder oder mechanischen Beanspruchungen auftreten.

Darüber hinaus ist der Anlagebauer auch dafür verantwortlich, dass ein fehlerhaftes Produkt in keinem Fall zu Verletzungen oder gar zum Tod von Personen bzw. zur Beschädigung oder Zerstörung von Sachen führen kann. Die einschlägigen Sicherheitsvorschriften sind in jedem Fall einzuhalten. Gefährliche Fehler müssen durch zusätzliche Massnahmen erkannt und hinsichtlich ihrer Auswirkung blockiert werden. So sind z.B. für die Sicherheit wichtige Ausgänge auf Eingänge zurückzuführen und softwaremässig zu überwachen. Es sind die Diagnoseelemente der PCD wie Watch-Dog, Ausnahme-Organisations-Blocks (XOB) sowie Test- und Diagnose-Befehle konsequent anzuwenden.

Werden alle diese Punkte berücksichtigt, verfügen Sie mit der SAIA® PCD über eine moderne und sichere programmierbare Steuerung, die Ihre Anlage über viele Jahre zuverlässig steuern, regeln und überwachen wird.

1. Einführung

Blockschema eines Servoantriebes für 2 Achsen

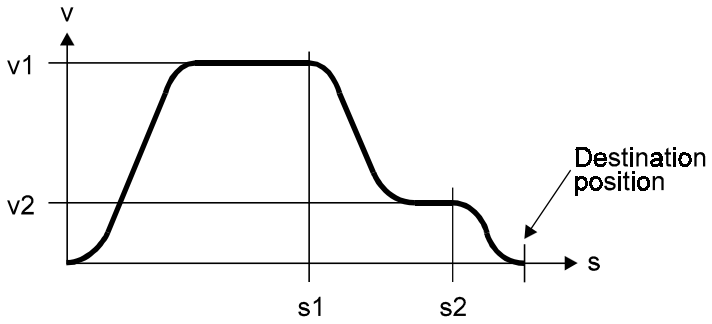


Funktion und Anwendung

Das ..H4..-Modul ist das leistungsfähigste der Achssteuermodule zur SAIA® PCD4. Durch den Einsatz modernster DSP-Technologie (Digital Signal Processor) ist das ..H4..-Modul in der Lage, 2 bzw. 4 Servomotor-Achsen unabhängig, linearinterpoliert oder zirkularinterpoliert zu regeln. Das S-förmige Geschwindigkeitsprofil ergibt dabei schnelle und zugleich weiche Bewegungsabläufe.

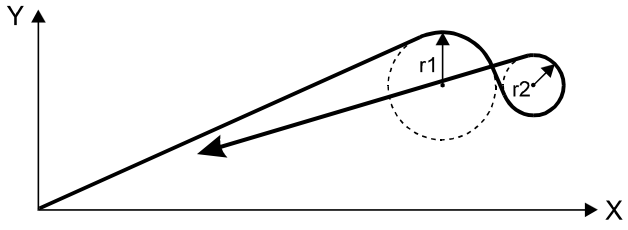
Durch den eigenen Speicher und die hohe Eigenintelligenz kann, je nach Einsatzart des ..H4..-Moduls, die CPU der PCD4 fast vollständig entlastet werden, so dass diese ganz für die eigentliche Prozess-Steuerung frei bleibt. Praxisgerechte Funktionsbausteine und ein leistungsfähiges Softwarepaket machen die Programmierung und Inbetriebnahme äusserst einfach. Sinnvolle Test- und Diagnose-Informationen mit entsprechenden Help-Funktionen unterstützen den Einsteiger und machen die Abläufe transparent.

Geschwindigkeits-/Weg-Profil mit S-förmigem Verlauf und Anfahren der Zielposition mit Langsamvorschub



Die wichtigsten Eigenschaften

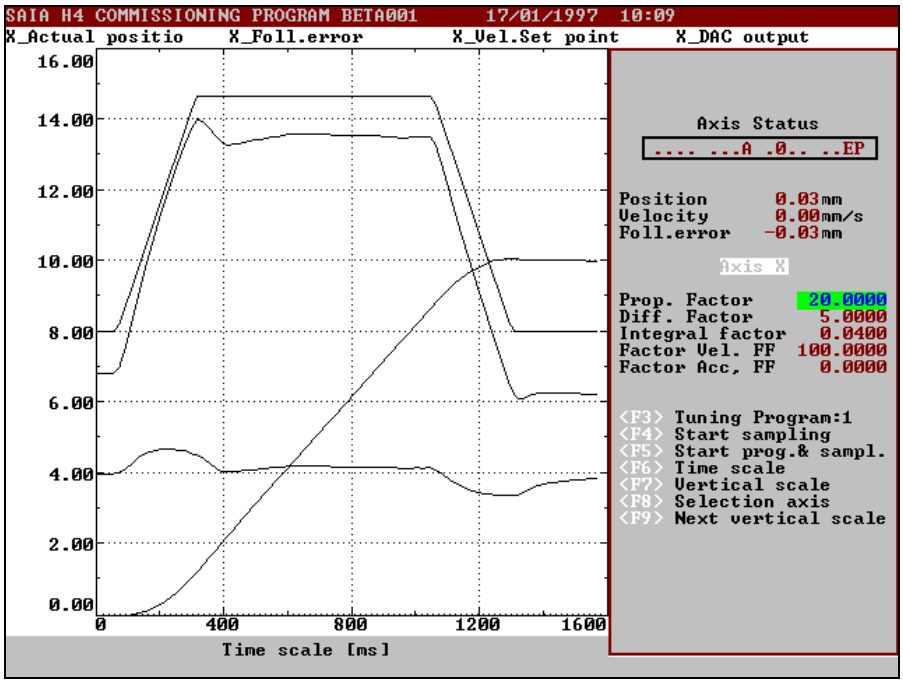
- PID-geregelte Steuerung von 2 bis 4 Achsen, unabhängig voneinander oder mit Linearinterpolation
- Zirkularinterpolation von 2 beliebigen Achsen des gleichen Moduls
- Weiche Bewegungsabläufe dank wählbarem Geschwindigkeitsprofil mit trapez- oder S-förmigem Verlauf
- Hohe Rechengeschwindigkeit (40 MIPS)
- Dank autonomer Achsfunktionen des .. H4..Moduls wird die CPU der Steuerung kaum belastet, steht also ganz für die Prozess-Steuerung zur Verfügung.
- Die Bewegungsparameter können in einem EEPROM nullspannungssicher gespeichert werden
- Als Incremental-Encoder lassen sich Ausführungen für 5V oder 24V einsetzen.
- Hardware- oder Software-Endschalter werden vom Modul selbständig überwacht und verarbeitet
- Als Ausgang zur Leistungselektronik steht pro Achse ein analoger Ausgang ±10V mit 16 Bit-Auflösung zur Verfügung
- Einfache Programmierung dank leistungsfähiger Befehle und praxisgerechter Software-Bibliothek mit Funktionsbausteinen
- Komfortables Programmier- und Inbetriebnahme-Werkzeug, mit welchem alle Bewegungsabläufe überwacht und geändert sowie individuelle Programme direkt in das ..H4..-Modul geladen und anschliessend gefahren werden können



Linear- und zirkular-interpolierter Verfahrweg von 2 Achsen

Typische Einsatzgebiete

- Palettierautomaten
- Bestückungs- und Montageautomaten
- Verpackungsmaschinen
- NC-gesteuerte Schneidemaschinen
- Maschinen für Dichtungs- und Klebstoffauftrag
- Rohrbiegemaschinen
- Werkzeugwechsler
- Lagerhandling
- Handlingroboter
- Poliermaschinen und anderes mehr



Ausdruck aus dem aktuellen CP (Commissioning and Programming Tool)

Programmier- und Inbetriebnahme-Werkzeug (CP)

Mit diesem Softwarepaket hat der Anwender Zugriff auf alle Funktionen des leistungsfähigen ..H4..-Moduls, d.h. Erstellen und Testen von Verfahrprogrammen sowie Optimieren der Regelparameter. Als menügeführte Programme stehen zur Verfügung:

- **Configure:** Eingabe der Kommunikations- und Achsparameter
- **Motion:** Syntax-geführter Editor zur Erstellung und Inbetriebnahme der Verfahrprogramme
- **Grafics:** grafische Wiedergabe eines Bewegungsablaufes (siehe obenstehende Abbildung), welche die Überprüfung und Optimierung der Regelungsparameter erlaubt
- **Utility:** Laden und Sichern von Programmen und Parametern

Notizen

2. Technische Daten

2.1 PCD4.H4xx

Wegerfassung (inkremental, 2 um 90° versetzte Impulse A und B sowie Referenzmarke R)

5V Eingänge	5V differenzielle RS422-Eingänge
Potentialtrennung	nein
Signalfrequenz	max. 150 kHz (intern 600 kHz mit x4-Mode)

24V Eingänge	
Signalbereiche	L (Low) = 0... 4V
	H (High) = 19... 32V
Eingangsstrom	10 mA
Potentialtrennung	nein
Signalfrequenz	max. 100 kHz (intern 400 kHz mit x4-Mode)
Arbeitsmodus	Quellbetrieb

Digitale Eingänge

Gemeinsam für alle Achsen	- Stop - Start
Pro Achse	- Endschalter LS1 ¦ Können auch durch Software- - Endschalter LS2 ¦ Endschalter ersetzt werden. - Referenzschalter - Positionserfassung - Fehler im Leistungsverstärker
Signalpegel	Low = 0 ... 4V, High = 19 ... 32V
Eingangsstrom	10 mA
Eingangsfilter	30 µs
Potentialtrennung	nein
Arbeitsmodus	Quellbetrieb

Digitale Ausgänge

Gemeinsam für alle Achsen	- H4 ready
Pro Achse	- Positions-Trigger-Ausgang - Leistungsverstärker enable/disable
Potentialtrennung	nein
Kurzschlussfest	nein
Ausgangsstrom	1 ... 100 mA (min. Last = 240Ω an 24V)
Arbeitsmodus	Quellbetrieb

Regler-Ausgang (zur Ansteuerung des Leistungsverstärkers)

Pro Achse	±10V, kurzschlussfest, Auflösung 15 Bit plus Vorzeichen, Lastwiderstand ≥ 3 kΩ Offset max. ± 100 mV

Programmier- und Inbetriebnahme-Werkzeug
(PC mit MS-DOS)

Anschluss	RS 232 (mit Standard-Kabel PCD8.K110/111)

Bewegungsparameter (können wahlweise in mm, Inch, Winkelgrad oder Encoder-Impulsen eingegeben werden)

Position	-2 147 483 648 bis +2 147 483 647 Einheiten Bereich: -2^{31}... $+(2^{31}-1)$ Impulse
Geschwindigkeit	-16 384 bis +16 383 Einheiten/Servo-Zyklus Bereich: -2^{14}... $+(2^{14}-1)$ Impulse (limitriert durch das Eingangsfilter auf 100 kHz rsp. 150 kHz).
Beschleunigung	-16 384 bis +16 383 Einheiten/Servo-Zyklus Bereich: -2^{14}... $+(2^{14}-1)$ Impulse
Dauer der S-Form	0.01 bis 99.99s
PID-Regler	P-, I- und D-Faktoren programmierbar Abtastzeit 200 µs bei 2 Achsen, 400 µs bei 4 Achsen
Elektronisches Getriebe	für Übersetzungsverhältnisse 0.0001 bis 10'000

Programmierung	mit Funktions-Blocks (FB) welche als PCD-Quellcode abgeben werden oder mit dem "Programming/Commissioning"-Tool.

Speicher (auf dem..H4..-Modul)

- nullspannungssicheres EEPROM für alle Bewegungsparameter von 4 Achsen
- mit Super-Cap gepuffertes RAM
  ca. 3000 ... 4000 Programmzeilen unterteilbar in 9 Einzelprogramme mit max. 1000 Zeilen pro Programm.

Stromversorgung

Extern durch Anwender	24 VDC (19 ... 32VDC) geglättet, Restwelligkeit max. 10%, max. 0.2A plus Encoder-Speisung
Für 5V Encoder	I_{max} = 300 mA/Achse
Für 24V Encoder	I_{max} = 200 mA/Achse
Intern vom PCD4-Bus +5V	typ. 550 mA + 100 mA pro Achse

Betriebsbedingungen

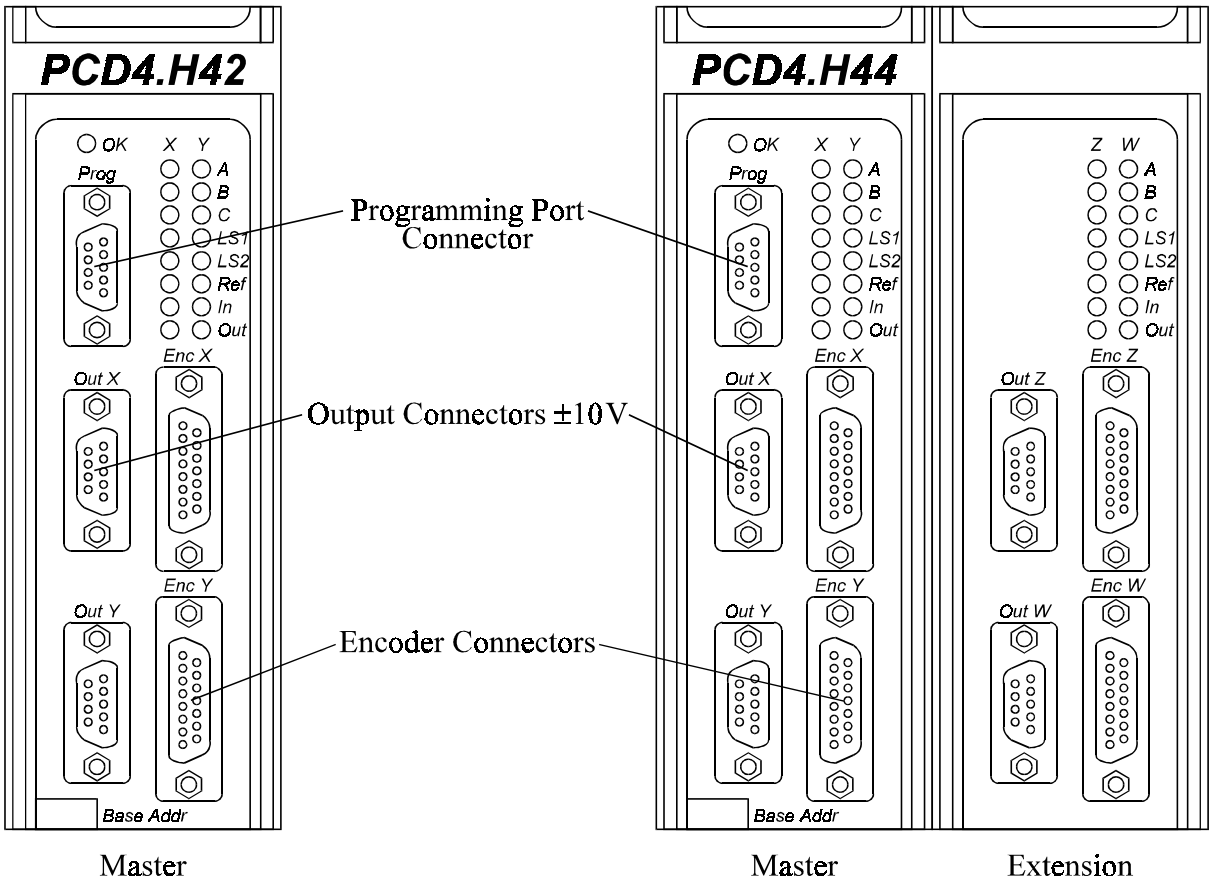
Umgebungs-Temperatur	Betrieb: 0... +55 °C ohne Zwangsbelüftung Lagerung: -20°C... +85 °C Feuchte: 5... 95%
EMV	gemäss CE-Empfehlungen: Immunität gemäss EN 50 082-2, 1995 Emissionen gemäss EN 50 081-2, 1993
Mechanische Festigkeit	gemäss IEC 1131-2
Störfestigkeit	1 kV in kapazitiver Kopplung gemäss IEC 801-4

2.2 PCD4 Konfigurierung

CPU	Jede PCD4 CPU ist einsetzbar
Speisung	PCD4.N210 muss wegen den ±15 VDC, welche vom H4-Modul benötigt werden, eingesetzt werden. Die 5V Stromaufnahme der H4-Module begrenzt die Anzahl Module auf 4 Module H120 oder 3 Module H440.
Speicher	PCD7.R1.. genügt, wenn keine Daten in den DB der CPU abgelegt werden. PCD7.R3.. in allen andern Fällen (siehe Abschnitt 7.7)

3. Präsentation

3.1 Frontplatte und LEDs

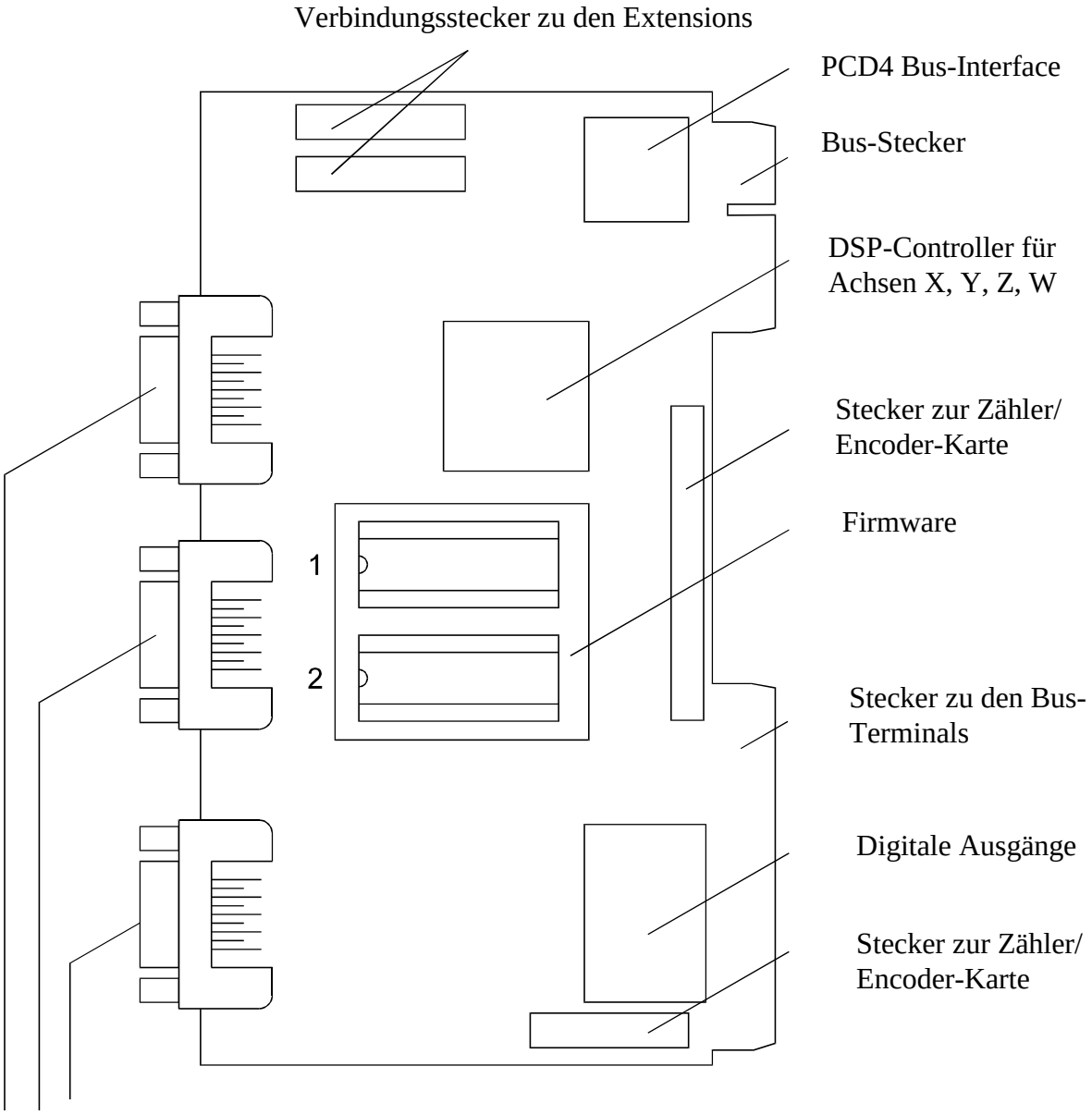


Gruppe	LED		Bedeutung
Systemsteuerung	**OK**	ein	- DSP in Run - Checksumme OK - Kein Fehler im Bewegungsprogramm
		blinkt	- Abnormaler Zustand (z.B. Endschalter erreicht). System läuft korrekt.
		aus	- System blockiert, schwerwiegender Fehler
Encoder	**A**		Encoder Signal A
	B		Encoder Signal B
	C		Encoder Signal C (Referenz) Diese LED zeigt den Status des angewählten Encoder-Eingangs

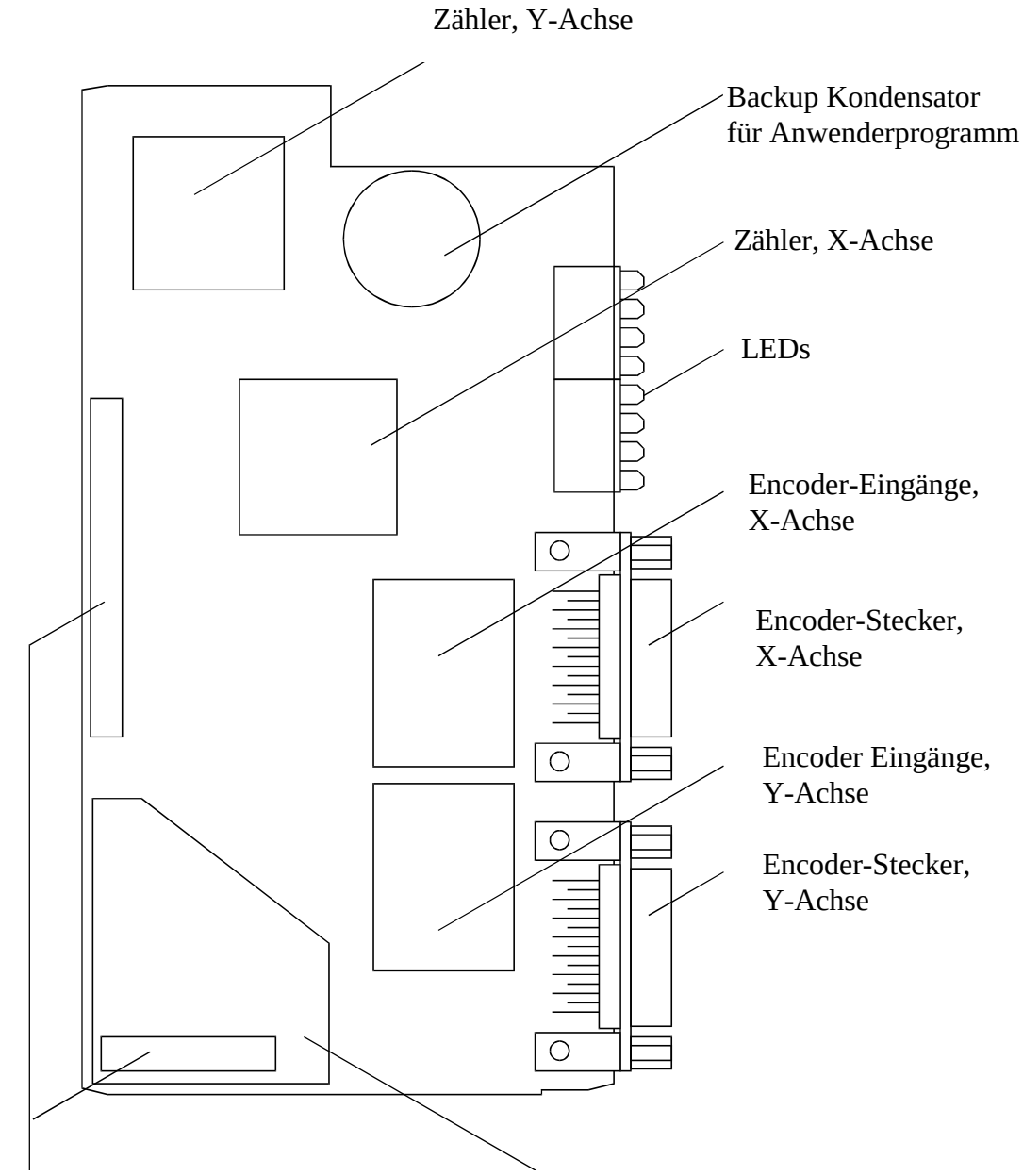
Group	LED		Bedeutung
Digitale Eingänge	**LS1**	ein	Negativer Endschalter erreicht
		aus	Negativer Endschalter nicht erreicht
	LS2	ein	Positiver Endschalter erreicht
		aus	Positiver Endschalter nicht erreicht
	Ref	ein	Referenz-Schalter erreicht
		aus	Referenz-Schalter nicht erreicht
			Diese Eingänge sind im "L"-Status aktiv. **Es werden hier aus Sicherheitsgründen Ruhekontakte (normally closed) eingesetzt.**
Amplifier (Verstärker)	**Out**	ein	Der digitale Ausgang "Amplifier enable" wurde durch das H4.. = H gesetzt. (Befehl "Enable")
	In	ein	Der digitale Eingang "Amplifier OK" wurde durch den Verstärker = H geseetzt. (als Anwort auf "Enable")

3.2 Gedruckte Leiterplatte

Hauptprint, Master

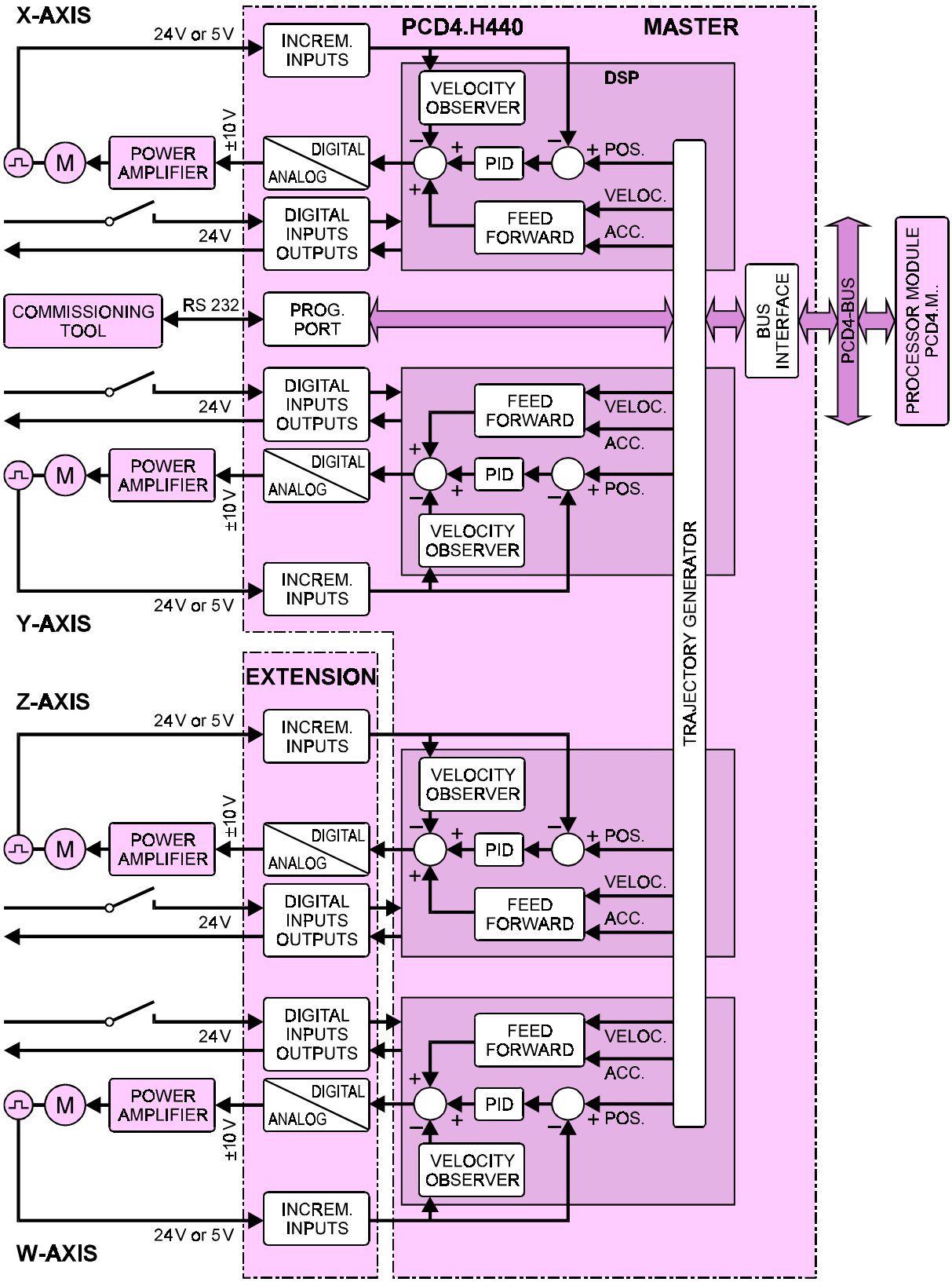


Zähler/Encoder Karte



4. Blockschaltbild

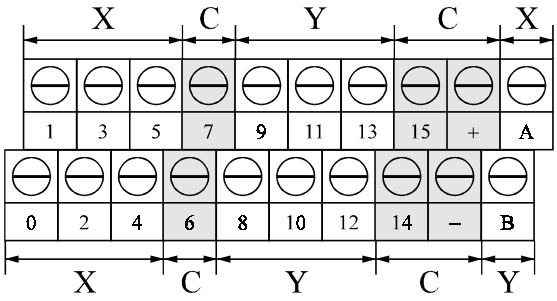
Blockschaltbild eines Servoantriebs für 4 Achsen



Notizen

5. Anschlüsse

5.1 Klemmenanschlüsse des Bus-Moduls (Übersicht)



MASTER

X-Achse (Master) (LS → Limit Switch = Endschalter)

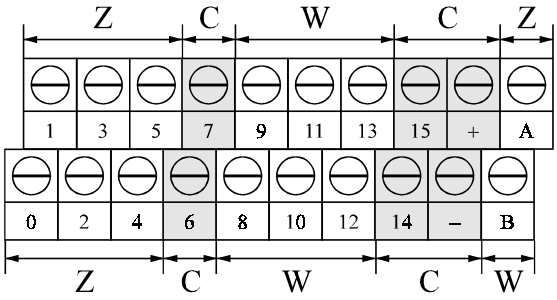
Anschluss	Beschreibung	Typ
0	Verstärker aktiv/inaktiv	Ausgang
1	Trigger-Ausgang	Ausgang
2	Verstärker ok / Fehlersignal	Eingang
3	LS1: Endschalter 1 (neg.)	Eingang
4	LS2: Endschalter 2 (pos.)	Eingang
5	Ref: Referenzpunkt-Schalter	Eingang
A	Positionserfassung	Eingang

Y-Achse (Master)

Anschluss	Beschreibung	Typ
8	Verstärker aktiv/inaktiv	Ausgang
9	Trigger-Ausgang	Ausgang
10	Verstärker ok / Fehlersignal	Eingang
11	LS1: Endschalter 1 (neg.)	Eingang
12	LS2: Endschalter 2 (pos.)	Eingang
13	Ref: Referenzpunkt-Schalter	Eingang
B	Positionserfassung	Eingang

Gemeinsame Anschlüsse (Master)

Anschluss	Beschreibung	Typ
6	..H4.. Bereit	Ausgang
7	Stop Programm	Eingang
14		
15	Start Programm	Eingang
–	GND	
+	+24 V	



Erweiterung

Z-Achse (Erweiterung) **(**LS → Limit Switch = Endschalter)

Anschluss	Beschreibung	Typ
0	Verstärker aktiv/inaktiv	Ausgang
1	Trigger-Ausgang	Ausgang
2	Verstärker ok Fehlersignal	Eingang
3	LS1: Endschalter 1 (neg.)	Eingang
4	LS2: Endschalter 2 (pos.)	Eingang
5	Ref: Referenzpunkt-Schalter	Eingang
A	Positionserfassung	Eingang

W-Achse (Erweiterung)

Anschluss	Beschreibung	Typ
8	Verstärker aktiv/inaktiv	Ausgang
9	Trigger-Ausgang	Ausgang
10	Verstärker ok / Fehlersignal	Eingang
11	LS1: Endschalter 1 (neg.)	Eingang
12	LS2: Endschalter 2 (pos.)	Eingang
13	Ref: Referenzpunkt-Schalter	Eingang
B	Positionserfassung	Eingang

Gemeinsame Anschlüsse (Erweiterung)

Anschluss	Beschreibung	Typ
6	nicht verwendet	
7	nicht verwendet	
14	nicht verwendet	
15	nicht verwendet	
–	GND	
+	+24 V	

5.2. Digitale Ein-/Ausgänge an den Busmodul-Klemmen

Master

Anschluss	Beschreibung	Typ
0	Verstärker aktiv/inaktiv, X-Achse	Ausgang
8	Verstärker aktiv/inaktiv, Y-Achse	Ausgang

Erweiterung

Anschluss	Beschreibung	Typ
0	Verstärker aktiv/inaktiv, Z-Achse	Ausgang
8	Verstärker aktiv/inaktiv, W-Achse	Ausgang

Die meisten Verstärker verfügen über einen Aktiv-Inaktiv-Eingang (Enable), welcher das Abschalten des Verstärkers, unabhängig von der Grösse des Steuersignals ermöglicht.

Diese Steuerfunktion ist sicherheitstechnisch sehr wichtig, damit der Vertärker im Bedarfsfall ganz herungergefahren werden kann (dies in einem Fehlerfall oder auch zum Steueren der Einschaltsequenz des Systems). Ein Stopsignal sollte niemals von einem analogen Ausgang, welcher Null Volt ist, abgenommen werden, da sich hier ein Offset aufbauen kann, was dann nicht zu einem Stop führt.

Der Ausgang ist aktiv, wenn dieser = H ist.

Master

Anschluss	Beschreibung	Typ
2	Verstärker ok / Fehlersignal, X-Achse	Eingang
10	Verstärker ok / Fehlersignal, Y-Achse	Eingang

Erweiterung

Anschluss	Beschreibung	Typ
2	Verstärker ok / Fehlersignal, Z-Achse	Eingang
10	Verstärker ok / Fehlersignal, W-Achse	Eingang

An diesen Eingang wird das Verstärker-OK-Signal geführt, so dass im Störungsfall alle Bewegungen und Programme gestoppt und einer oder alle Verstärker ausgeschaltet werden.

Der Eingang ist aktiv, wenn dieser = H ist.

Master (LS → Limit Switch = Endschalter)

Anschluss	Beschreibung	Typ
3	LS1: Endschalter neg. X-Achse	Eingang
11	LS1: Endschalter neg. Y-Achse	Eingang

Erweiterung

Anschluss	Beschreibung	Typ
3	LS1: Endschalter neg. Z-Achse	Eingang
11	LS1: Endschalter neg. W-Achse	Eingang

An diesen Eingang wird der Endschalter 1 für die negative Wegbegrenzung angeschlossen.

Diese Eingänge sind im "L"-Status aktiv, d.h. im Normalfall (Achse nicht auf Endschalter) müssen +24V anliegen.

Diese Schaltung verlangt einen Ruhekontakt (normally closed) als Endschalter.

Master (LS → Limit Switch = Endschalter)

Anschluss	Beschreibung	Typ
4	LS2: Endschalter pos. X-Achse	INPUT
12	LS2: Endschalter pos. Y-Achse	INPUT

Erweiterung

Anschluss	Beschreibung	Typ
4	LS2: Endschalter pos. Z-Achse	INPUT
12	LS2: Endschalter pos. W-Achse	INPUT

An diesen Eingang wird der Endschalter2 für die positive Wegbegrenzung bei der Endposition angeschlossen.

Diese Eingänge sind im "L"-Status aktiv, d.h. im Normalfall (Achse nicht auf Endschalter) müssen +24V anliegen.

Diese Schaltung verlangt einen Ruhekontakt (normally closed) als Endschalter.

Master

Anschluss	Beschreibung	Typ
5	Ref: Referenzpunkt-Schalter, X-Achse	Eingang
13	Ref: Referenzpunkt-Schalter, Y-Achse	Eingang

Erweiterung

Anschluss	Beschreibung	Typ
5	Ref: Referenzpunkt-Schalter, Z-Achse	Eingang
13	Ref: Referenzpunkt-Schalter, W-Achse	Eingang

Dieser Eingang wird in der Routine zum Anfahren des Referenzpunktes verwendet (homing routine).

Diese Schaltung verlangt einen Ruhekontakt (normally closed) als Referenzschalter.

Master

Anschluss	Beschreibung	Typ
6	..H4..-Ready-Ausgang (Bereit-Ausgang)	Ausgang

Dieser Ausgang ist = H, wenn das System zum Arbeiten bereit ist und kein abnormaler Zustand (z.B. Endschalter aktiv) vorliegt.

Der Ausgang wird = L, wenn die OK-LED blinkt oder ganz ausgeschaltet ist.

Master

Anschluss	Beschreibung	Typ
15	Start-Eingang	Eingang

Mit diesem Eingang wird der Start des mit P95 gewählten Programms ausgelöst (wie Befehl 'Start').

Die Polarität dieses Eingangs ist programmierbar (P90).

Master

Anschluss	Beschreibung	Typ
7	Stop-Eingang	Eingang

Mit diesem Eingang wird der Programmablauf des mit P95 gewählten Programms bei der nächsten Warte-Instruktion angehalten (wie Befehl 'Break').

Die Polarität dieses Eingangs ist programmierbar (P91).

Master

Anschluss	Beschreibung	Typ
1	Trigger-Ausgang, X-Achse	Ausgang
9	Trigger-Ausgang, Y-Achse	Ausgang

Erweiterung

Anschluss	Beschreibung	Typ
1	Trigger-Ausgang, Z-Achse	Ausgang
9	Trigger-Ausgang, W-Achse	Ausgang

Der Triggerausgang liefert, wenn softwaremässig aktiviert, eine Signalflanke, wenn eine Achse eine vorgegebene Position erreicht.

Mit dieser Funktion kann eine externe Aktion, abhängig von einer Achsposition, praktisch verzögerungsfrei ausgelöst werden.

Die Polarität dieses Ausgangs ist programmierbar (P'x'62).

Master

Anschluss	Beschreibung	Typ
A	Positionserfassung, X-Achse	Eingang
B	Positionserfassung, Y-Achse	Eingang

Erweiterung

Anschluss	Beschreibung	Typ
A	Positionserfassung, Z-Achse	Eingang
B	Positionserfassung, W-Achse	Eingang

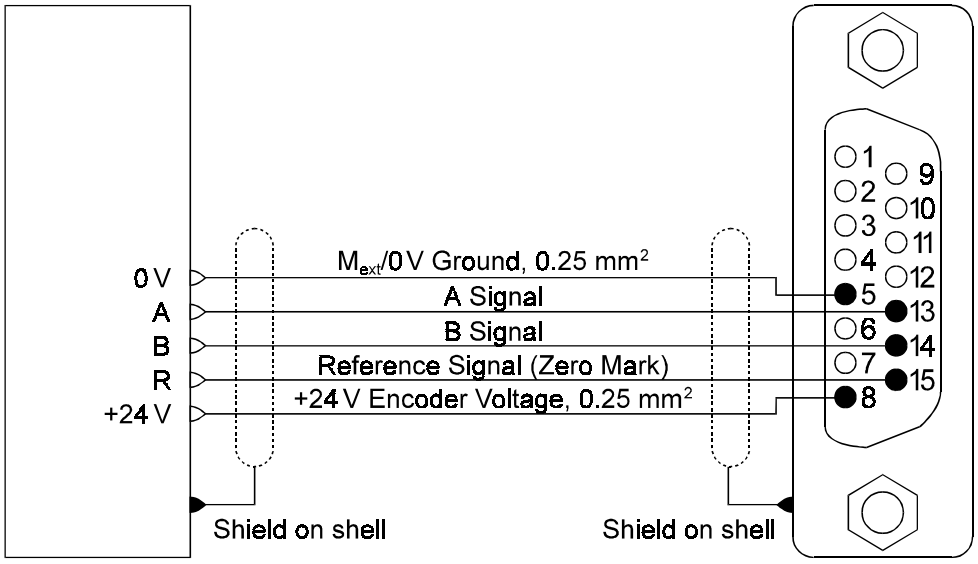
Mit diesem Signal kann, wenn softwaremässig aktiviert, die aktuelle Position der Achsen in Echtzeit-Werten für spätere Auswertungen gespeichert werden.

Der aktive Status dieses Eingangs ist = H.

5.3. Frontstecker und Kabel

Verbindungs-Diagramm für das 24V Encoder-Anschlusskabel

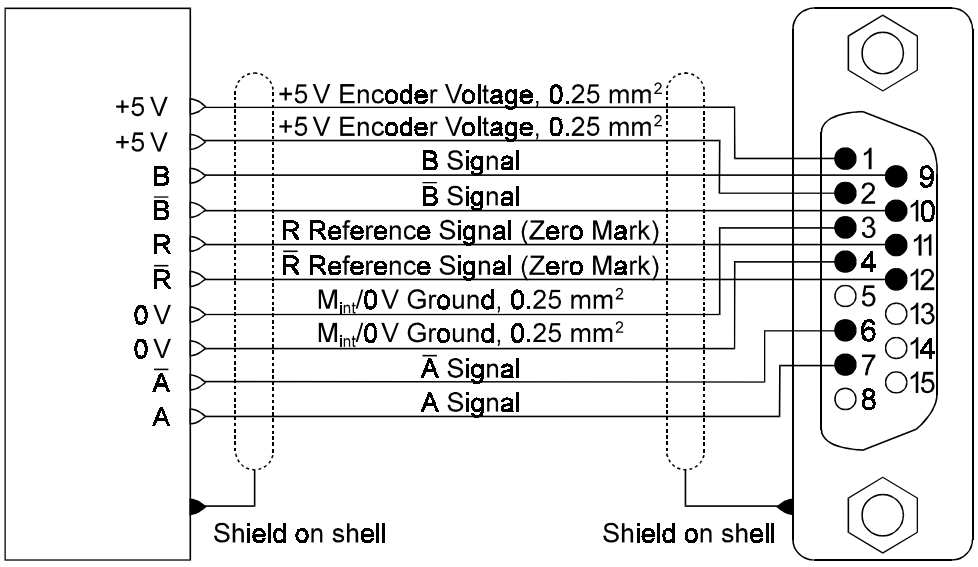
Encoder 24V | abgeschirmtes Kabel (max. 20m) | Kabel-Stecker (Encoder) (Ansicht von der Frontseite)



15-pin D-Sub Stecker, männlich, metallisches Gehäuse mit schraubbarer Verbindung.

Verbindungs-Diagramm für das 5V/RS 422 Encoder-Anschlusskabel

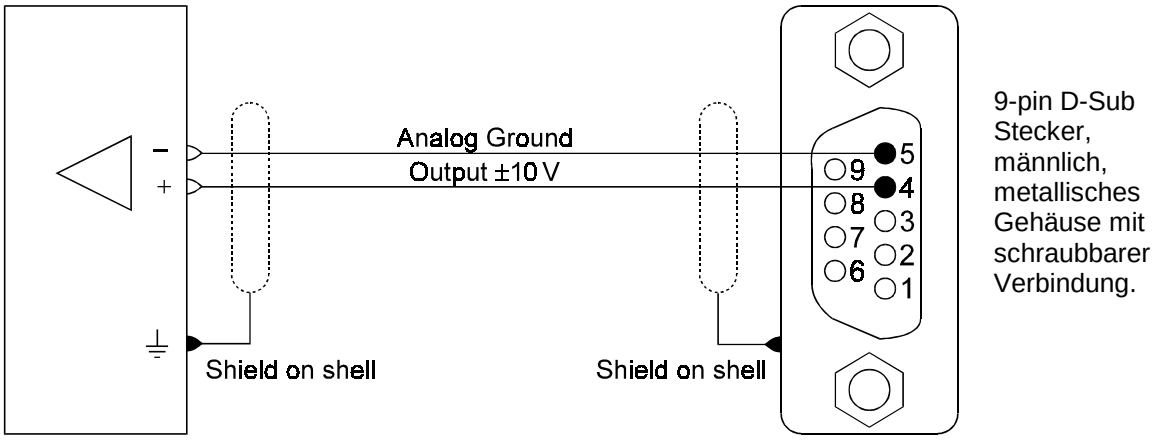
Encoder 5V | abgeschirmtes Kabel (max. 20m) | Kabel-Stecker (Encoder) (Ansicht von der Frontseite)



15-pin D-Sub Stecker, männlich, metallisches Gehäuse mit schraubbarer Verbindung.

Verbindungs-Diagramm für das Analog-Ausgangs Kabel (±10 V)

Leistungs-Verstärker	abgeschirmtes Kabel (max. 20m)	Kabel-Stecker (Encoder) (Ansicht von der Frontseite)



9-pin D-Sub Stecker, männlich, metallisches Gehäuse mit schraubbarer Verbindung.

Programmier-Schnittstelle (PROG)

mit Standard-Kabel PCD8.K110/111

(Anschlussbelegungen siehe PCD4-Handbuch)

Notizen

6. Funktionsbeschreibung

6.1 Einführung

Das Modul

Das PCD4.H4.. "Motion Control"-Modul steuert 1 bis 4 Achsen und kann diese linear und zirkular interpolieren.

Das Modul wird auf den PCD4-Bus aufgesteckt. Das Modul belegt 16 Adressen des PCD4-Bus für den Datenaustausch mit der PCD4-Anwender-Software. Die 5V Stromaufnahme limitiert die Anzahl H4-Module. Es könnten theoretisch bis zu 8 Module (32 Achsen) gesteckt werden und jedes Modul kann seine Achsen interpolieren (siehe Abschnitt 2.2, Speisung). Eine modulübergreifende Interpolation ist nicht möglich, jedoch können mit einem H4-Modul 4 Achsen für eigenständige Bewegungen unabhängig voneinander oder 4 Achsen für eine Bewegung (interpoliert) gesteuert und überwacht werden. Kombinationen dazwischen sind selbstverständlich auch möglich.

Selbständigkeit

Das Modul arbeitet selbständig. Das Modul regelt die Achsen, verfährt diese bahngenau, kommuniziert mit dem CP-Tool (Commissioning and Programming Tool) und/oder mit der PCD-CPU via den SAIA® Standard-Funktionsblöcken und verfügt über einen eigenen Programmspeicher.

Integriert in PCD

Je nach Komplexität der Anwendung (z.B. variable prozessabhängige Daten wie Geschwindigkeit, Position usw.) muss eine PCD-CPU zur Steuerung der H4-Module herangezogen werden.

Benutzerfreundlich

Die Programmierung und Parametrierung ist mit dem CP-Tool am PC Bildschirm möglich aber auch via den FBs von der PCD4-CPU aus. Dies gibt dem Anwender die volle Freiheit, alle Möglichkeiten für sein Projekt auszuschöpfen.

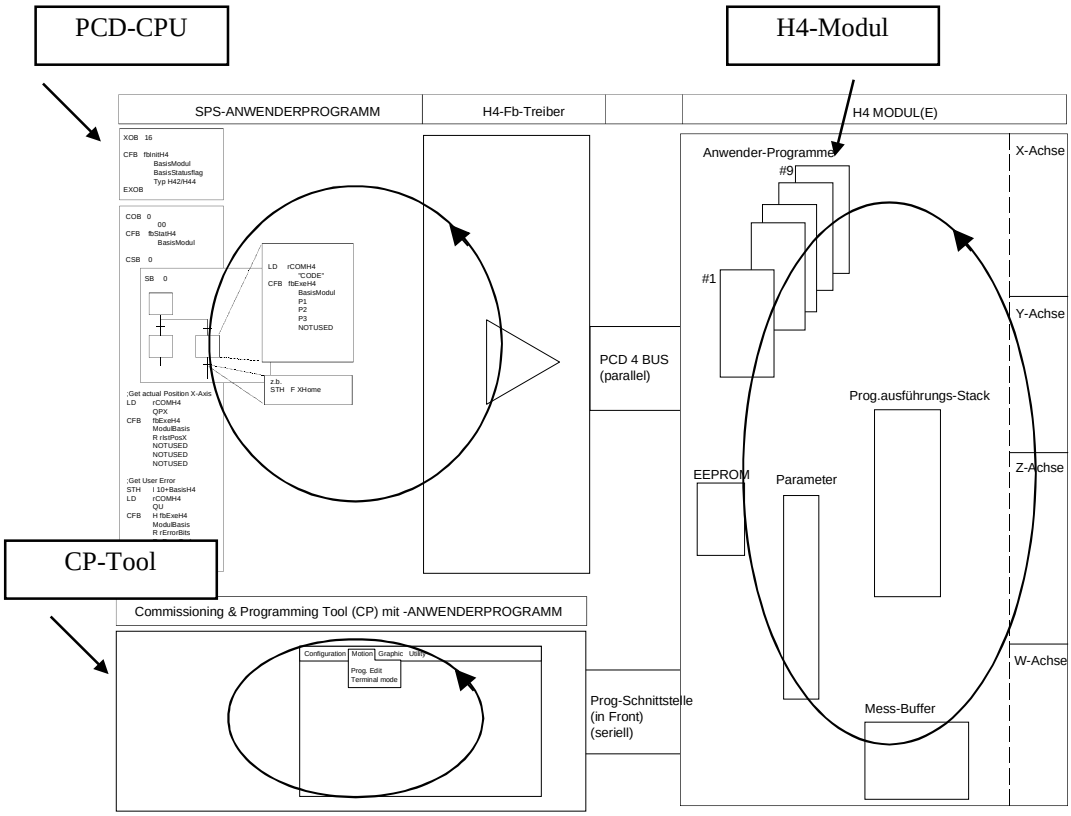


Bild 6.01

Dieses Bild zeigt, dass jedes Modul (PCD, H4-Modul, CP-Tool) selbständig arbeitet und die andern nicht wesentlich belastet.

6.2 Blockdiagramm, Funktionsweise

6.2.1 Übersicht

Das Hardware-Blockschaltbild kann in Kapitel 4 eingesehen werden. Hier wird das H4-Modul in seinen Ressourcen und Funktionen dargestellt. So ist zum Beispiel ersichtlich, dass es einen Parameterbereich (Block) gibt. Alle Funktionen oder Befehle (mit Pfeilen gekennzeichnet) wirken immer auf den entsprechenden Block.

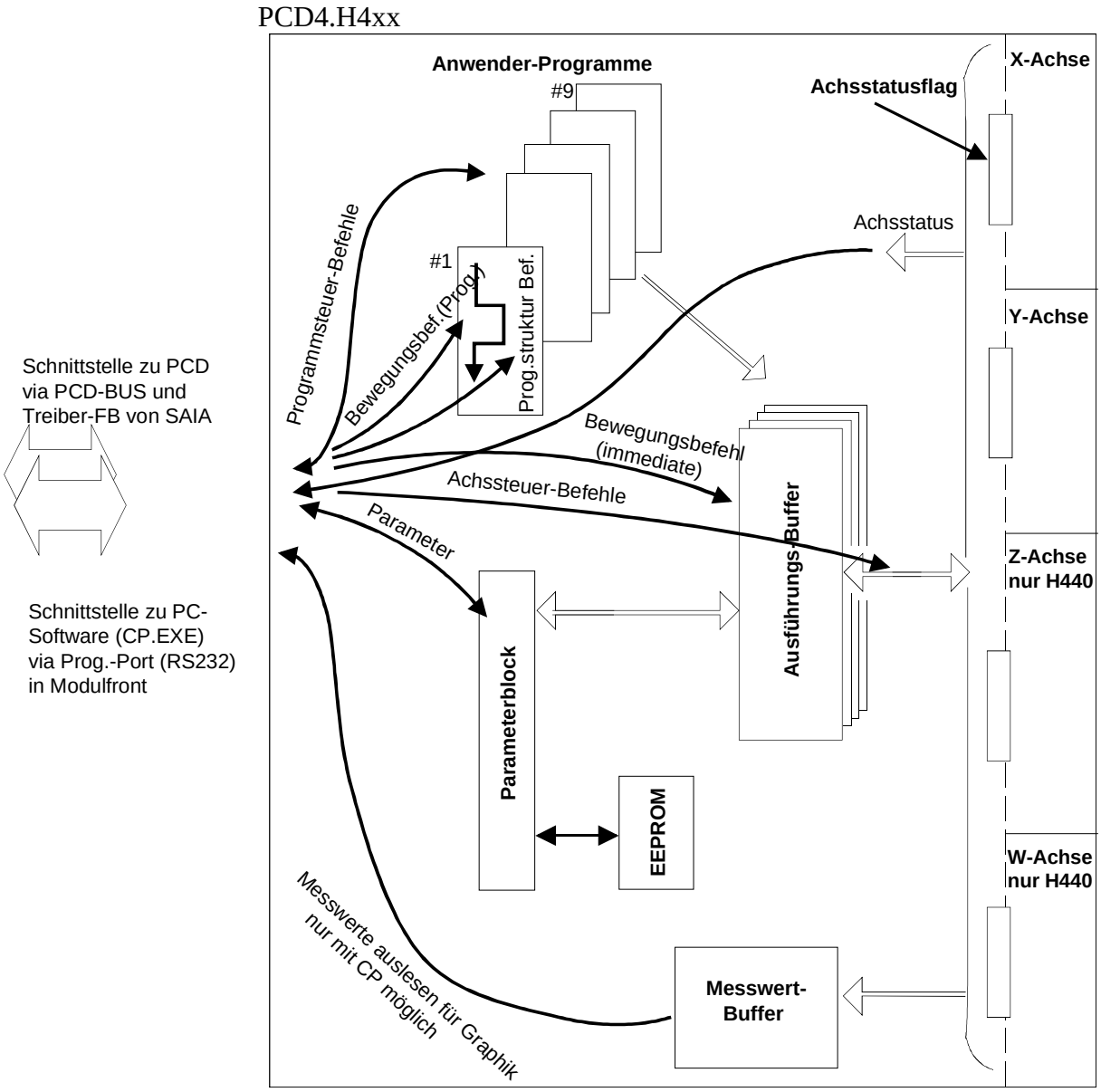


Bild 6.02

Anhand obiger Darstellung ist ersichtlich, dass die Befehle ans H4-Modul in verschiedene Gruppen aufgeteilt werden können. Es spielt dabei keine Rolle, ob diese Befehle von der PCD-CPU (via FBs) oder vom PC (CP.EXE) stammen.

6.2.2 H4-Programmspeicher

Das H4-Modul besitzt einen eigenen Speicher für Anwender-Programme. In diesem Speicher können Befehle aus dem Befehlsvorrat des H4-Moduls abgelegt werden. Ein Programm kann im CP-Tool geschrieben und anschliessend ins H4-Modul transferiert werden. Dies ist ebenso von der PCD4-CPU aus mit den FBs möglich.

Programme
Die Programme werden beim Laden ins H4-Modul einer Nummer zugeordnet. Es stehen 9 Programme zur Verfügung. Es können maximal 4 Programme gleichzeitig laufen.

Programmzeilen
Es können max. 1000 Programmzeilen pro Programm im H4-Modul gespeichert werden. Insgesamt können, je nach verwendeten Befehlen, ca. 3000 - 4000 Programmzeilen im H4-Modul gespeichert werden.

Speicherung
Die Programme im H4-Modul sind gegen Verlust bei Spannungsausfall mit einem Supercap gesichert und werden mindestens 2 Wochen gehalten. Die Parameter sind im EEPROM gesichert und gehen nicht verloren.

6.2.3 Parameter

Etwa 80 Parameter sind im H4-Modul hinterlegt. Diese werden beim Einschalten vom EEPROM in den 'Parameterblock' kopiert mit welchem das H4-Betriebssystem arbeitet. Die Parameter sind in Funktionsgruppen gegliedert und in der Parameterliste, Abschnitt 7.6, zusammengefasst.

Änderungen /EEPROM
Änderungen der Parameter sind bei Spannungsausfall flüchtig. Um dennoch die Einstellungen für eine bestimmte Applikation im H4-Modul zu speichern, können die Parameter ins EEPROM kopiert werden. Dies geschieht mit dem CP-Tool automatisch beim Schreiben der Parameter ins H4-Modul aus dem Parametermenü.. Beim Arbeiten mit den FBs muss das Speichern der Parameter bewusst mit einem speziellen Befehl ausgeführt werden. Die maximale Anzahl Schreibzyklen ist auf 100'000 begrenzt. Aus diesem Grund darf das Abspeichern nicht in einem zyklischen Programmablauf erfolgen.

6.2.4 Ausführungsmodus (Immediate / Program)

FBs:
In Bild 6.02 ist ersichtlich, dass es zwei Typen von Bewegungs-Befehlen gibt. Die Bewegungsbefehle 'Immediate' werden direkt den Ausführungsbuffern übermittelt und fortlaufend ausgeführt. Die Bewegungsbefehle 'Program' werden nicht direkt ausgeführt, sondern in ein H4-Programm (Nr. 1 - 9) geschrieben. In der Befehlsliste (Abschnitt 7.5) ist in der Spalte 'ip-Parameter' angegeben, welcher Befehl in welchem Ausführungsmodus arbeitet. (I nur Immediate, P nur Program, IP Immediate + Program)

CP-Tool: (CP = Commissioning / Programming)
Grundsätzlich bestehen die beiden Ausführungsmodi (Immediate / Program) bei Verwendung des CP-Tools auch. Der Anwender wird jedoch nur indirekt damit konfrontiert. Verwendet und arbeitet man im CP-Tool im Menü 'Motion/Program Edit', in welchem Programme erstellt und anschliessend ins H4-Modul geladen werden, so werden nur die Befehle 'Motion Prog. Cmd' akzeptiert. Im Fenster 'Motion/ Terminal' werden hingegen die Befehle 'Motion immediate' benutzt.

6.2.5 Ausführungs-Buffer

Das H4-Modul ist eigenständig und kann somit Programme bis zum Schluss ohne weitere Unterstützung ausführen. Vom CP-Tool aus oder mit den FBs der PCD-CPU muss anschliessend nur der Programmfluss gesteuert werden (z.B. Run 5).

Für diesen eigenständigen Ausführungsprozess der Programme oder auch für das Ausführen der 'Motion Immediate'-Befehle verfügt das H4 über vier interne Ausführungs-Buffer. Dieser Ausführungs-Buffer oder Ausführungs-Prozess kann nicht umgangen werden. Mit 'Immediate'-Befehlen wird nur ein Ausführungs-Buffer verwaltet.

Der Ausführungs-Buffer kann 50 'Immediate'-Befehle speichern. Wird diese Anzahl überschritten, wird das 'User Error Bit' und das Störungs-Bit 9 gesetzt. Diese Fehlermeldung wird zurückgesetzt, wenn die Anzahl Befehle im Buffer auf 45 sinkt.

Buffer Überfüllung: Werden trotz der Fehlermeldung 'Buffer full' (Bit 9) weitere Immediate-Befehle ans H4 geschickt, gehen diese verloren.

Der Ausführungsbuffer wird sequentiell abgearbeitet, d.h. ein neuer Befehl wird erst ausgeführt, wenn der vorangehende beendet ist.

Wenn mehrere Achsen gleichzeitig (nicht interpoliert) bewegt werden sollen, muss mit verschiedenen Programmen (1 Programm/Achse) gearbeitet werden, welche parallel gestartet werden können.

Jedes Programm benutzt einen Ausführungsbuffer, d.h. max. 4 Programme oder 3 Programme und Immediate-Befehle können gleichzeitig ausgeführt werden.

6.2.6 Achsenstatusflag

In Bild 6.02 ist ersichtlich, dass jede Achse sogenannte Achsenstatusflag beinhaltet. Anhand dieser kann z. B. festgestellt werden, ob eine Achse einen Endlagen-Schalter erreicht hat, die Lageregelung aktiv ist oder z.B. auch ob der Home-Prozess beendet ist. Diese Achsenstatusflag werden mit dem Befehl 'Query status x' abgefragt. Die einzelnen Flags und deren Bedeutung entnehmen Sie bitte der Befehlsliste (Abschnitt 7.5).

Die Statusflags sind in Gruppen aufgeteilt. Die Flags 0-7 werden durch die Standard-FBs belegt und dürfen vom Anwender nicht benutzt werden. Die Flags 8-23 sind für die X-Achse reserviert, 24-39 für Y, 40-55 für Z und 56-71 für die Achse W. Für die Programmierung steht es dem Anwender frei, ob er mit Nummern arbeiten oder ob er dem jeweiligen Flag ein Symbol zuteilen will. Zu allen Flags wird die Basisadresse der Flags (BAF) addiert, welche im Initialisierungs-FB definiert wurde. (siehe Abschnitt 7.4.6).

Flag 0-6: Die Flag 0-6 werden durch die Standard-FBs belegt und dürfen vom Anwender nicht benutzt werden.
7: Fatal Error (siehe Kapitel 8).

	X	Y	Z	W:	Achse
Flag	8	24	40	56:	Achse in Position
	9	25	41	57:	Achse läuft in 'immediate'-Mode
	10	26	42	58:	Achse in Hardware LS
	11	27	43	59:	Achse in Software LS
	12	28	44	60:	Störung Schleppfehler
	13	29	45	61:	Warnung Schleppfehler
	14	30	46	62:	Soll-Geschwindigkeit der Achse = 0
	15	31	47	63:	Capture-Position erfasst
	16	32	48	64:	Drive OK (Status Input AOK)
	17	33	49	65:	Negativ LS Input angesprochen (LSS)
	18	34	50	66:	Positiv LS Input angesprochen (LSE)
	19	35	51	67:	Referenz Switch angesprochen (RPS)
	20	36	52	68:	'Position Capture Input' angesprochen (PCI)
	21	37	53	69:	Trigger-Position erreicht
	22	38	54	70:	Positionsüberlauf
	23	39	55	71:	Homefunktion erfolgreich beendet

6.2.7 **Messwertbuffer** (siehe Bild auf Seite 1-3 und Bild 6-02)

Der Messwertbuffer kann nur vom CP-Tool im Graphic-Menü ausgelesen werden. Der Messwertbuffer dient der Speicherung der vorgängig vom Anwender gewählten Bewegungs-Daten. Diese können grafisch visualisiert werden (Oszilloskop-Funktion) und dienen der Einstellung der Achsregelparameter an Maschinen. Das Bedienen und Einsetzen dieser Funktion ist im Abschnitt 7.3, CP-Tool, beschrieben.

6.3 Funktionsübersicht

Funktionen:	
Positionieren einer linearen Achse	Ja
Positionieren einer Dreh-Achse	Ja
Linearinterpolation bis zu 4 Achsen	Ja
Kreisinterpolation	Ja
Spline-Interpolation	Nein
Lageregelung (Positioniermode)	Ja
Drehzahlregelung	Nein
Elektronik-Getriebe (von zwei und mehr Achsen)	Ja
Blended moves	Ja
S-Kurven Beschleunigungsprofil	Ja
Feedforward für Geschwindigkeit und Beschleunigung	Ja
Justieren der Achsenregelparameter	Ja mit Software (CP-Tool)
Speichern des Bewegungsprog. des H4-Moduls ausserhalb des Moduls	Ja auf PC oder SPS
Spindelsteigung Fehlerkompensation	Nein
Kompensation des Umkehrspiels	Ja
Teach-In	Nein
ISO-Code (CNC)	Nein
Verändern der Parameter im Betrieb	Ja (siehe ‘on the fly’)
M-Befehle wie in CNC	Nein
Jog: Manual-Betrieb	Ja

Die mit 'Nein' bezeichneten Funktionen können grossteils mit der CPU gelöst werden.

Einsatzbeispiele: (siehe auch Kapitel 1)	
Electronic cam programmer (el. Nockenschaltwerk)	Nein
Flying cut possibilities (Fliegende-Schere)	Nein
Kartesischer Roboter	Ja
Handling-Gerät	Ja
Spezial-Maschinen	Ja

6.4 Unterschiede der Module H3 und H4

Unterschiede	**PCD4.H3xx**	**PCD4.H4xx**
Betriebsmodi	Positions- und Geschwindigkeitsmodus	Nur Positionsmodus
Programm Editor für IL-Instruktionen	Jeder ASCII Editor, **ohne** SEDIT (Symbol Offset nicht möglich).	Jeder ASCII Editor, **auch** SEDIT möglich.
Bewegungs- Programm	Kann im Modul nicht gespeichert werden. Alle Daten und Informationen der Achse sind in der PCD-CPU abgelegt.	Bis zu 9 verschiedene Programme können im RAM des Moduls gespeichert werden. Die PCD-CPU wird somit entlastet.
Parameter	Sind im Modul nicht nullspannungssicher gespeichert. Das heisst, diese gehen beim Ausschalten verloren.	Sind im Modul in einem EEPROM gespeichert. Daher gehen diese bei Power down nicht verloren.
Achs Init und Achs Handling	Jede Achse wird durch zwei FBs gesteuert und überwacht. "AxInit" und "AxHndlg". Eine Funktion wird durch Setzen eines Flags ausgelöst.	Jede Funktion wird durch Aufruf des FBs "fbH4.exe" mit einem Befehlswort direkt ausgeführt. Für die Initialisierung und den Modul-Zustand stehen zwei weitere FBs zur Verfügung..
Referenz Prozedur	Muss durch den Anwender gelöst werden.	Auf Verlangen vom H4 automatisch durchgeführt
FB Verschachtelungsebene	3 Ebenen	1 Ebene (bei Prog. Up/Down laden 2 Ebenen)
Synchronisation zwischen Achsen	Durch das Anwender-Programm in der PCD.	Durch das Modul (Mehrachsen Linear- oder Zirkular-Interpolation)
Motion Control Factor : - Einheiten - Übersetzung von Impulsen zu mm / inch oder umgekehrt	 - Encoder Impulse oder mm - durch PCD-CPU	 - Encoder Impulse, mm, Grad oder Inch - Durch H4
Commissioning & Progr. Tool : - Verbindung - Programmierung - Commissioning	 - PGU Stecker auf der PCD-CPU - nicht möglich - nur eine Achse kann programmiert und ausgeführt werden. Ein minimales PCD-Programm muss in der PCD-CPU vorhanden sein.	 - Programmier-Stecker auf dem H4 - ganze Bewegungsabläufe können geschrieben, ausgeführt und auf Diskette gespeichert werden. Dazu ist kein Anwenderprogramm in der PCD-CPU notwendig. - Möglichkeit für direkte Bewegungs-Ausführung - Online Optimierung der Parameter mit grafischer Unterstützung.
Endschalter und Referenz Schalter	Muss durch den Benutzer überwacht werden.	Überwacht durch das H4 Modul.
Adressen	Die erste Adresse und die Anzahl der Module müssen definiert werden. Bei mehreren Modulen müssen alle beieinander liegen (lückenlos).	Flexibler. Die Adresse jedes Moduls muss definiert werden und als Parameter den FBs übergeben werden. Dies ermöglicht die freie Plazierung der Module auf dem PCD4-Bus.
I/O für Motion Control Modul	Alle I/Os müssen durch die PCD-CPU gesteuert werden.	Integriert und kontrolliert im H4.

6.5 Generator für das Geschwindigkeitsprofil

Profilgenerator

Mit dem H4-Modul können entweder Trapezförmige oder S- kurvenförmige Geschwindigkeitsprofile generiert werden. Diese können für jede Achse mit P 'x' 42 gewählt und festgelegt werden. Der Generator produziert die spezifizierte Sollkurve für jede Achse. Der Servo-Lageregler regelt nun die effektive Position so gut wie möglich an die Soll-Position.

6.5.1 Trapez Geschwindigkeitsprofil

Dies ist das einfachste Geschwindigkeitsprofil. Die Achse fährt mit einer bestimmten Geschwindigkeit auf ein Ziel, beschleunigt und bremst mit einer konstanten Rampe. Diese Geschwindigkeiten sind in den folgenden Parametern definiert:

maximum acceleration/deceleration rate:	P 'x' 33
acceleration rate:	P 'x' 43
deceleration rate:	P 'x' 44
Geschwindigkeit mit dem Befehl	SS 'x'
Beschleunigungsmodus	P 'x' 42 = 0 (trapezförmig)

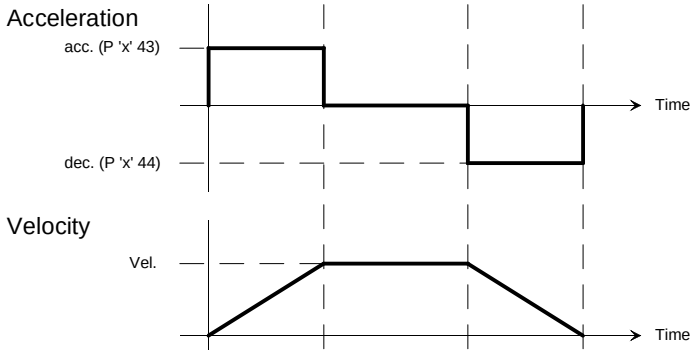


Bild 6-03

Bei Verwendung einer hohen Geschwindigkeit oder beim Verfahren einer sehr kurzen Strecke ist es möglich, dass die gewünschte Geschwindigkeit nicht erreicht wird. In diesem Fall wird das Geschwindigkeitsprofil dreieckförmig.

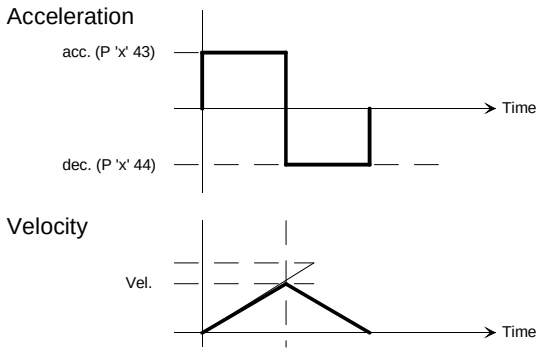


Bild 6-04

6.5.2 S-Kurven Geschwindigkeitsprofil

Ein Trapez-Geschwindigkeitsprofil mit einer konstanten Beschleunigung hat einen harten Wechsel beim Einsetzen der Beschleunigung zur Folge, was die Achse zum Schwingen anregen kann. Für einen weicheren Übergang und zum Überwinden der Haftreibung bei V = 0, wird das S- Kurven-Profil verwendet. Dieses Profil wird durch Ändern der Beschleunigung während dem Beschleunigen erreicht. Die S-Kurvendauer t_s ist einstellbar und in einem Parameter hinterlegt.

S-curve duration time (ts): P 'x' 45
Beschleunigungsmodus: P 'x' 42 = 1 (S-Kurve)

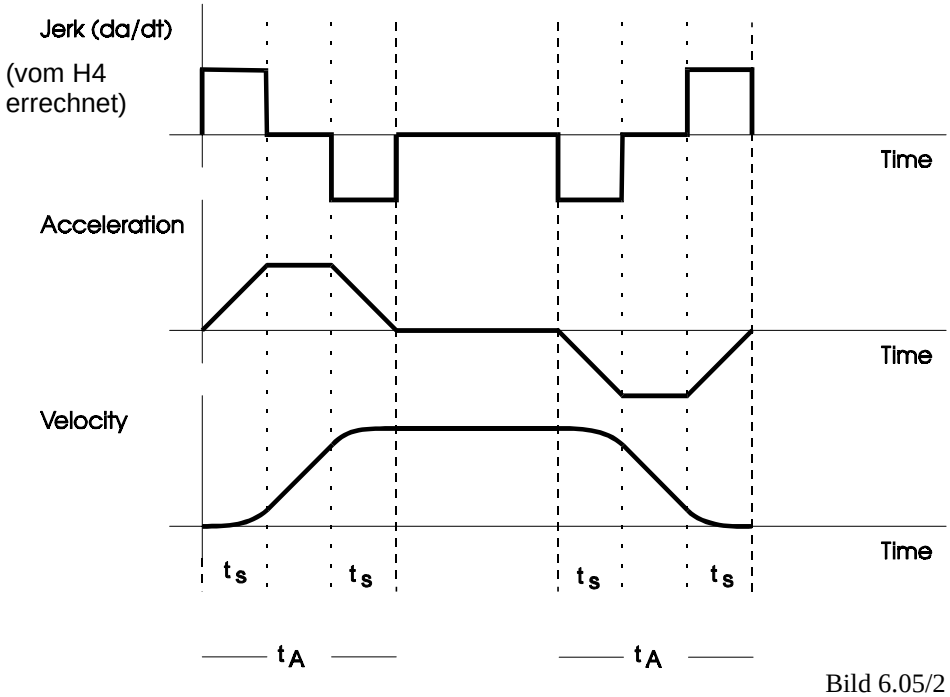


Bild 6.05/2

Wird die S-Kurve verwendet, so gelten die Parameter 43 und 44 als mittlere Acceleration und Deceleration.

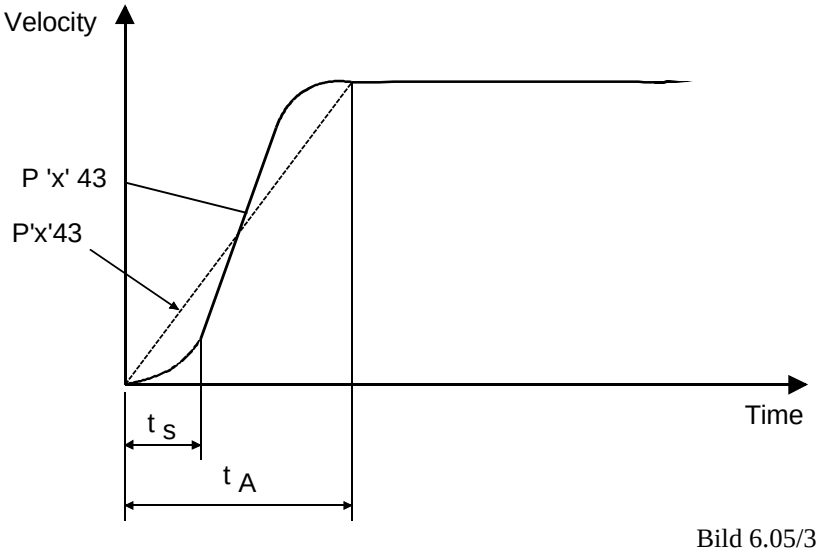


Bild 6.05/3

Wird die S-Kurvendauer $t_s = 0$ spezifiziert, ergibt sich ein rein trapezförmiges Geschwindigkeitsprofil. Wird t_s grösser als die Hälfte der (errechneten) Beschleunigungszeit (t_A) spezifiziert, wird ts auf $t_{A/2}$ begrenzt. Innerhalb der Beschleunigung ergibt sich somit kein linearer Anteil. Dies ergibt eine rein S-förmige Beschleunigung mit einem Maximum, welches 2x der mittleren Beschleunigung (P 'x' 43) entspricht. Die maximale Beschleunigung, eingestellt in P 'x' 33, kann somit bei Verwendung einer reinen S-Kurve auch 2x überschritten werden, was sich durch einen grösseren Schleppfehler während dem Beschleunigen auswirken würde.

In der Praxis erweist sich eine S-Kurvendauer von 5 ... 30% der Beschleunigungszeit t_a als sinnvoll.

Kombination von zwei Achsen mit unterschiedlichem Rampenprofil

Wird zum Beispiel die X-Achse mit dem trapezförmigen Rampenprofil und die Y-Achse mit der S-Kurve definiert und eine Bahn-Bewegung ausgelöst, so werden beide Achsen mit der S-Kurve beschleunigt.

Ist die Zeit t_s 'Duration time', welche für jede Achse definiert wird unterschiedlich, so wird die grösste Zeit verwendet.

Für interpolierte Bewegungen gelten folgende Befehle:

SV anstelle SS 'x'	(Geschwindigkeit)
SA anstelle P 'x' 43	(Beschleunigung)
SD anstelle P 'x' 44	(Verzögerung)

max. Velocity	P 'x' 30	¦ wird auch
max. Acc./Dec.	P 'x' 33	¦ berücksichtigt

Die Bahngeschwindigkeit SV wird entsprechend der Verfahrwege auf die einzelnen Achsen aufgeteilt.

$SV = \sqrt{V_x^2 + V_y^2}$	für 2-Achs-Interpolation
$SV = \sqrt{V_x^2 + V_y^2 + V_z^2}$	für 3-Achs-Interpolation
$SV = \sqrt{V_x^2 + V_y^2 + V_z^2 + V_w^2}$	für 4-Achs-Interpolation

Pro Modul existiert nur 1 Bahngeschwindigkeit SV. Beim Ausführen einer interpolierten Bewegung wird immer der aktuelle Wert von SV zur Berechnung verwendet. Danach kann SV verändert werden, ohne dass die laufende Bewegung beeinflusst wird. (nicht 'on the fly'). Damit können bei einem H440 zwei Achsenpaare gleichzeitig mit unterschiedlichen Bahngeschwindigkeiten bewegt werden.

6.6 Blended move (Weicher Übergang)

Dem H4-Modul können ganze Abläufe in Auftrag gegeben werden. Das heisst, mehrere einzelne Bewegungen zusammen ergeben einen Ablauf. Ist eine einzelne Bewegung eines Ablaufs beendet, würden alle beteiligten Achsen die Geschwindigkeit auf Null reduzieren, um anschliessend für die nächste Bewegung erneut zu beschleunigen. Mit der Funktion 'Blended move' wird die neue Geschwindigkeit übernommen und es findet nur eine Änderung von der ersten zur zweiten Geschwindigkeit statt. Diese Änderung beginnt bei der Position wo die Bremsrampe (ohne 'blended move') einsetzen würde. Die Überblendung von einer Geschwindigkeit auf eine neue, wird immer trapezförmig ausgeführt.

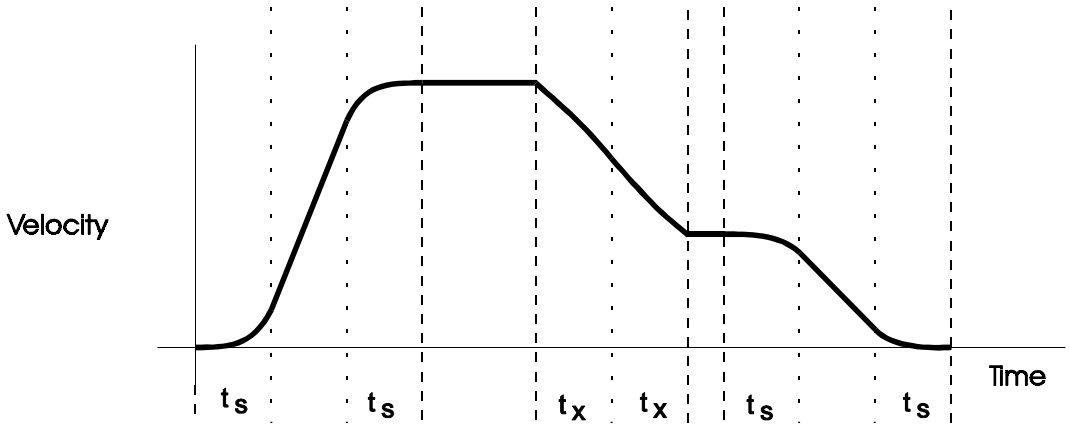


Bild 6.06

Beispiele:

ohne Blended move:	mit Blended move:
XR500,YR0	XR500,YR0
WAIT0	
XR200YR100	XR200YR100
SV100	SV100
XR50,YR0	XR50,YR0
WAIT0	
SV120	SV120
XR200,YR100	XR200,YR100
XR500	XR500,YR0
YR200	XR0,YR200

Parameter P97 "Blended move Winkel"

Mit dem Parameter 97 kann ein Winkel α gesetzt werden, ab welchem das H4-Modul 'Blended move' anwendet.

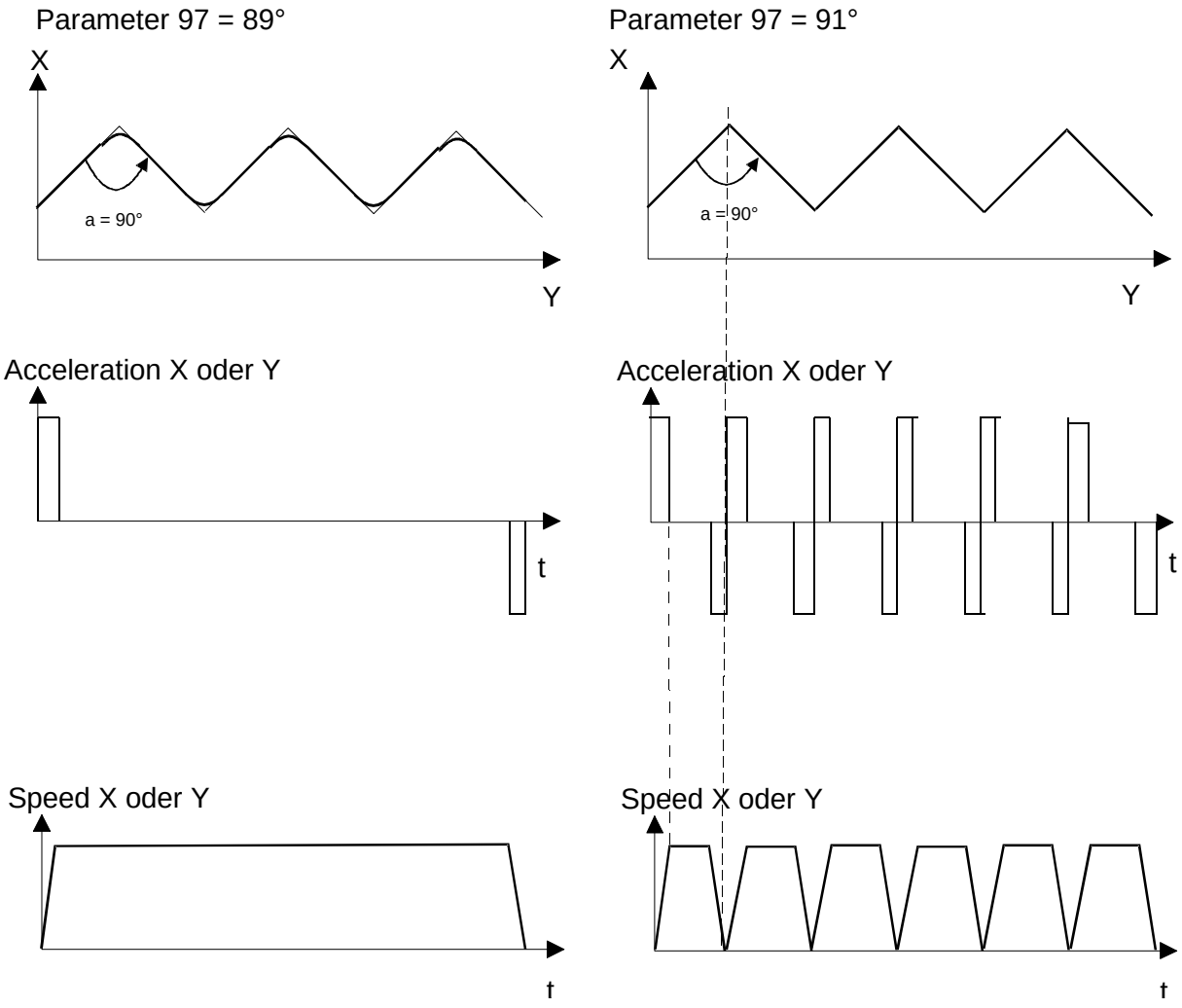


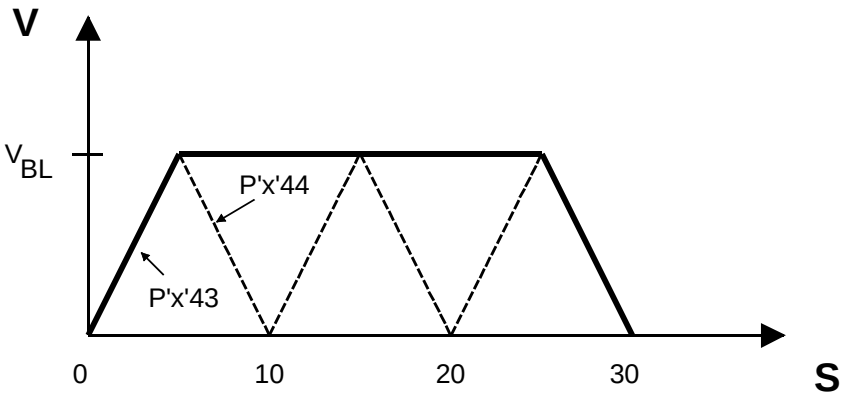
Bild 6.06.1

Ist P97 kleiner als der beim Verfahren der Achsen entstehende Winkel, so wird 'Blended move' angewendet. Dies ist sinnvoll, wenn zum Beispiel mit einem Handlinggerät eine Strecke zurückgelegt werden muss, ohne dazwischen anzuhalten. Sollen jedoch bestimmte Punkte exakt angefahren werden, so kann der Winkel so gesetzt werden, dass kein 'Blended move' angewendet wird. (Mit P97 = 181° wird die Funktion ganz ausgeschaltet). Es ist jedoch zu beachten, dass mit 'Blended move' die Zwischenpositionen nicht ganz erreicht werden, um die Bahngeschwindigkeit konstant zu halten.

Geschwindigkeit bei 'Blended move'

Sind die einzelnen Bewegungsschritte im Vergleich zur gewählten Geschwindigkeit klein ist es möglich, dass diese nicht erreicht wird.

Die Geschwindigkeit bei 'blended move' entspricht derjenigen, welche im Maximum bei einer einzelnen Bewegung erreicht würde und ist abhängig von der Beschleunigung P'x'43 bzw. P'x'44 und dem Weg 's':



Mit Acc. P'x'43 = Desc. P'x'44 = a ergibt sich die folgende max. Geschwindigkeit:

$$V_{BL} = \sqrt{2 \times a \times s}$$

Wenn die gewählte Geschwindigkeit erreicht werden soll, kann:

- die Geschwindigkeit reduziert werden (SSX bzw. SV) oder
- die Beschleunigungen P'x'43 und P'x'44 erhöht werden oder
- die einzelnen Bewegungsschritte vergrössert werden

6.7 Home Funktion / Referenzfahren

Das H4-Modul kann die Referenzfahrt eigenständig ausführen. Diese Funktion ist bereits ein kleiner Programmablauf und muss daher auch definiert werden. Die Definitionen stehen in den Parametern 20-24 und sind im Abschnitt 7.6.3 aufgelistet.

Die zu referenzierende Achse muss aktiv sein (ENABLE).
(Die nachfolgende Beschreibung ist auf Bild 6.07 bezogen)

1. Die Suche des Referenzschalters wird mit der in Parameter 22 definierten Geschwindigkeit durchgeführt. Im Parameter 20 wird die Suchrichtung definiert. Wird der Referenzschalter nicht gefunden und trifft die Achse auf einen Endschalter (HW oder SW), wird die Suchrichtung geändert.

2. Wurde der Referenzschalter gefunden, beginnt das Freifahren. Die Richtung des Freifahrens ist in Parameter 21 definiert, die Geschwindigkeit in P 'x' 24.

3. Nachdem der Refernzschalter abgefallen ist, wird die Achse weiter bewegt, bis das Index-Signal des Encoders (Kanal C) detektiert wird. Die Wiederholgenauigkeit des Referenzschalters ist nicht relevant, da auf das C-Signal des Encoders referenziert wird. Die Wiederholgenauigkeit ist somit incrementgenau.

4. Beim Erkennen des C-Signals wird die Achse mit dem in Parameter 23 festgelegten Wert geladen (oft '0') und anschliessend abgebremst. Die Achse steht daher nach dem Stoppen nicht genau auf dem Referenzpunkt.

   Die Home Funktion ist erfolgreich beendet und das Statusflag wird 'H' gesetzt (siehe Abschnitt 7.5.5).

Hinweise:

- Wird kein Referenzschalter oder kein C-Signal des Encoders gefunden, blinkt die OK-LED und im User-Error Code wird das Störungs-Bit 7 hinterlegt. (siehe Abschnitt 7.5.5, Zelle 2.13). Liefert der Encoder kein C-Signal, muss dieser Eingang auf logisch H verdrahtet werden (siehe Abschnitt 5.3).
- Die Referenzposition P 'x' 23 kann zum Beispiel auch als Abgleichung der Achsen von mehreren gleichen Maschinen dienen.
- Die Home-Funktion wird nicht PID geregelt sondern nur gesteuert ausgeführt. Die Steuerspannung ist:

$$Vout = 10V \frac{\text{Home speed}}{\text{max. speed}}$$

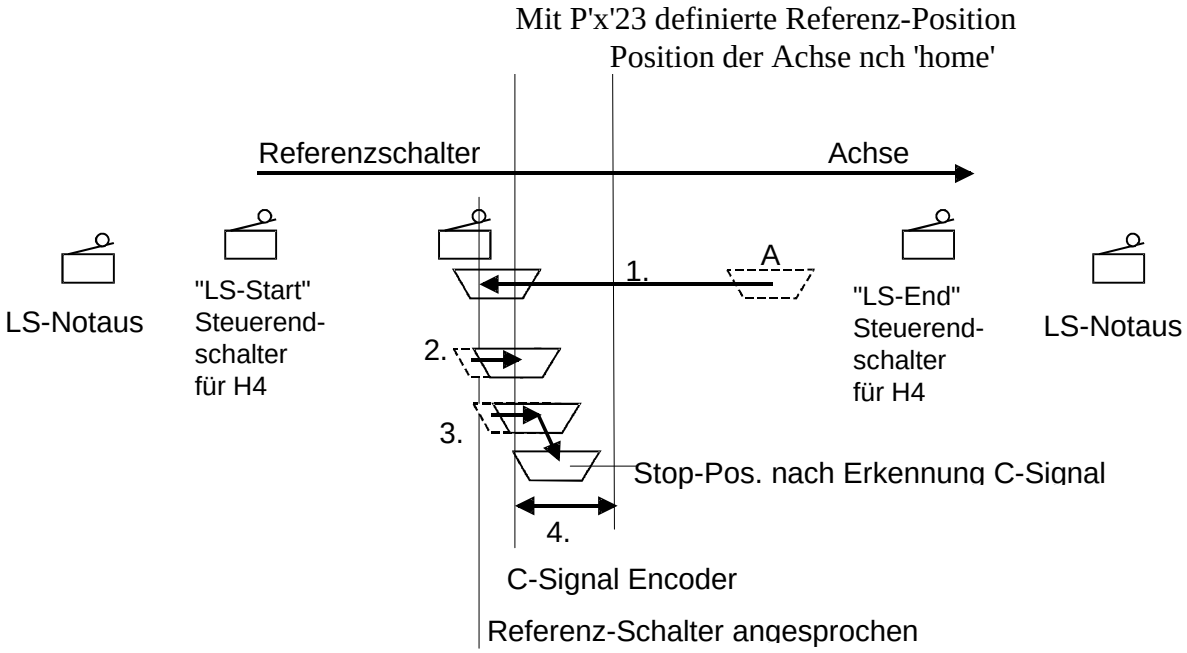


Bild 6.07

6.8 PID-Regler

Der digitale Lageregler mit 'Geschwindigkeit feedforward' und 'Beschleunigung feedforward' kann in allen seinen Regelgrössen während der Bewegung verändert werden. (Change "on the fly" in Abschnitt 7.6).

Regler-Parameter:

Proportional- Faktor	Parameter 50
Differential- Faktor	Parameter 51
Abtastfaktor für D	Parameter 56
Integral Faktor	Parameter 52
Integrallimit	Parameter 53
Integralmode	Parameter 16
Geschwindigkeit feedforward	Parameter 54
Beschleunigung feedforward	Parameter 55
Dead band	Parameter 10

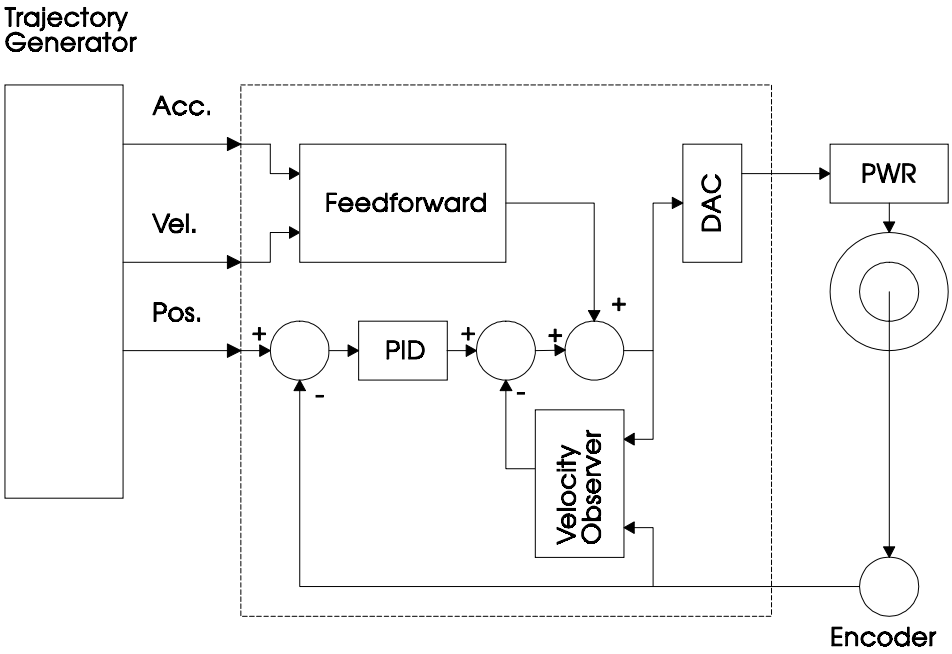


Bild 6.08

Der Profil-Generator (Trajectory Generator) erzeugt alle digitalen Informationen, um das System zu steuern (Zielposition, Geschwindigkeit, Beschleunigung). Ebenso wird die Funktion 'jerk' erzeugt (Beschleunigungs-Änderung).

6.9 Encoder

Die genaue Position der Achse wird mit einem Incremental-Drehgeber oder Incremental-Massstab erfasst. Die Position wird im 'x4'-Modus ausgewertet. Damit wird die vierfache Auflösung erreicht.

6.9.1. Encodertyp

Die Incremental-Geber können verschiedene Signale aufweisen. Das H4 Modul kann 24 VDC oder 5 VDC-Encoder verarbeiten (Hardwaredetails siehe Kapitel 2 "Technische Daten").

Der Encodertyp kann im Parameter 92 ausgewählt und eingestellt werden. Die Einstellung umfasst jeweils ein Achsenpaar X und Y oder Z und W. Da die LEDs A, B und C nach der Encoderwahl angesteuert werden, kann mit diesen die korrekte Einstellung überprüft werden.

6.9.2. Drehrichtung

Die Signalreihenfolge der Signale A und B bestimmen die Drehrichtung, und dementsprechend wird die aktuelle Position incrementiert oder decrementiert. Abhängig davon, an welchem Wellenende und in welcher Lage der Incremental-Geber montiert ist, wird die Zählrichtung positiv oder negativ. Durch Vertauschen der Signale A und B ist eine Zählrichtungsänderung möglich. Das H4-Modul ermöglicht dies jedoch auch ohne Verdrahtungsänderung, indem der Parameter 08 geändert wird. Dies erlaubt auch im Schema eine Vereinheitlichung der Verdrahtung, unabhängig von der Lage des Gebers.

6.9.3. Auflösung / Einheiten

Das H4-Modul arbeitet direkt in physikalischen Einheiten. So wird mit dem Parameter 01 die Einheit gewählt, in der gearbeitet werden soll (mm / Zoll / Grad / Impulse). Auf diese Weise ist es möglich, z.B. den Weg 233,56 mm direkt dem H4 Modul zu übergeben. Um eine optimale Zusammenarbeit mit der PCD-CPU und dem H4 zu ermöglichen und dennoch eine hohe Auflösung zu erreichen, wurde für floating point Werte das 'Virtual Integer' Format gewählt, d.h. alle Werte werden ganzzahlig, mit einer virtuellen Kommastelle, welche in Parameter 96 definiert ist, übergeben.

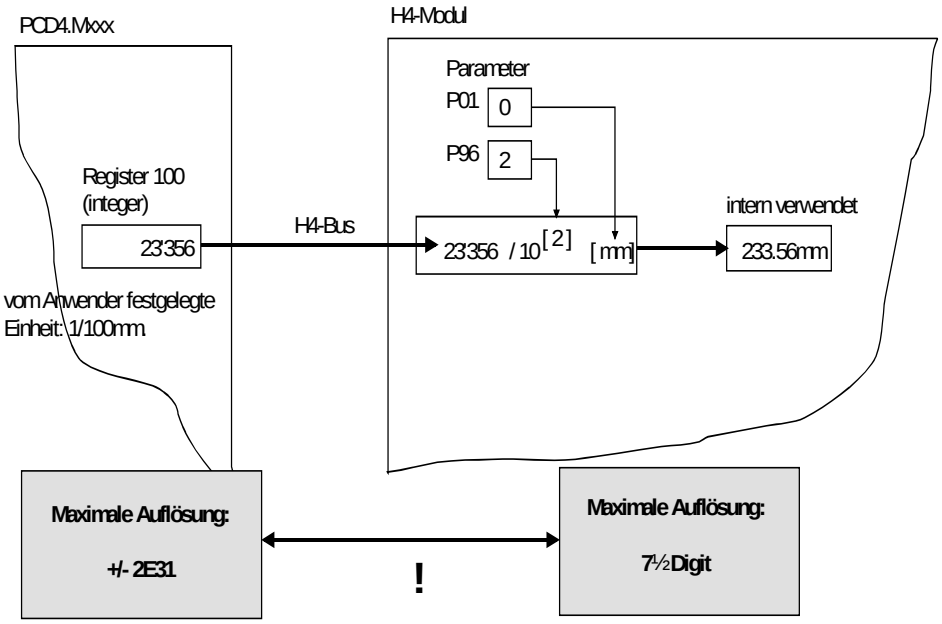


Bild 6.09

Ganzzahlige Werte (integer) sind von dieser Umwandlung nicht betroffen. In der Befehlsliste sind die Befehle, welche für Parameter das "Virtual Integer" Format verwenden mit "VI" bezeichnet.

Die Grundgenauigkeit für die Fliesskommawerte beträgt 7½ Digit, d.h. alle Werte welche in diesem Format berechnet werden, wie z.B. die Position, sind bis zur 7. Dezimalstelle exakt. (Bsp.: 0,001 mm auf 10m Weg).

Um die Achsposition (ohne Umrechnung) inkrementgenau anzuzeigen stehen spezielle Befehle zur Verfügung (QPI'x', ...)

6.10 Umkehrspiel / Backlash

Eine Achse mit einem Spindelantrieb hat meist ein Umkehrspiel. Wird die Wegrichtung geändert, so fährt die Achse den gewünschten Weg minus das Umkehrspiel, also nicht exakt den gewünschten Weg.

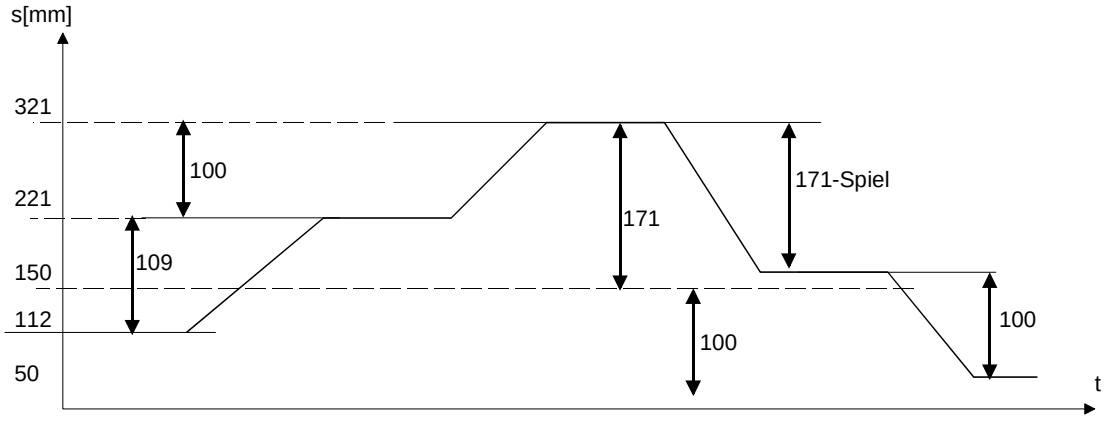


Bild 6.10

Die Genauigkeit einer Maschine wird durch das Umkehrspiel erheblich verschlechtert. Entweder wird eine spielfreie Mechanik eingesetzt, die oft sehr teuer ist oder aber dieses Spiel wird durch die Umkehrkompensation auf 'Null' gesetzt.

Das H4-Modul besitzt diese Möglichkeit. Das Umkehrspiel kann im Parameter 14 eingestellt werden (siehe Parameterliste 7.6.4) und wird beim Ändern der Richtung zum Verfahrweg addiert. Wird die Richtung nicht geändert, ist das Spiel bereits aufgehoben und es wird keine Korrektur vorgenommen.

Umkehr Korrekturgeschwindigkeit

Mit dem Parameter 63 kann die Geschwindigkeit eingestellt werden, mit welcher das Umkehrspiel aufgehoben wird. Diese kann unter Umständen höher sein als die normale Geschwindigkeit, da nur der Motor und die Spindel bewegt werden müssen.

Bereich = 10... 100% der max. Geschwindigkeit (Parameter 30).

6.11 Elektronisches Getriebe

Bei der Kopplung (Link) der Y-Achse an die X-Achse werden alle Wegbefehle auf die X-Achse auch auf der Y-Achse ausgeführt. Das Übersetzungsverhältnis kann in Parameter 07 eingestellt werden, in diesem Fall Y/X. Diese Kopplung (Link) ist nur in eine Richtung wirksam (Y übernimmt die X-Befehle und nicht umgekehrt). Ist die Kopplung in beide Richtungen gewünscht, so müssen die Kopplungen in beiden Richtungen gesetzt werden. Der Link Y an X und der Link X an Y. Es ist dabei darauf zu achten, dass die zweite Kopplung das inverse Übersetzungsverhältnis zur ersten hat (1/2 und 2/1). Das elektronische Getriebe wird als lineare Interpolation gerechnet. Somit sind die Vektorgeschwindigkeit SV und die Vektorbeschleunigung SA resp. SD wirksam.

Beispiel:

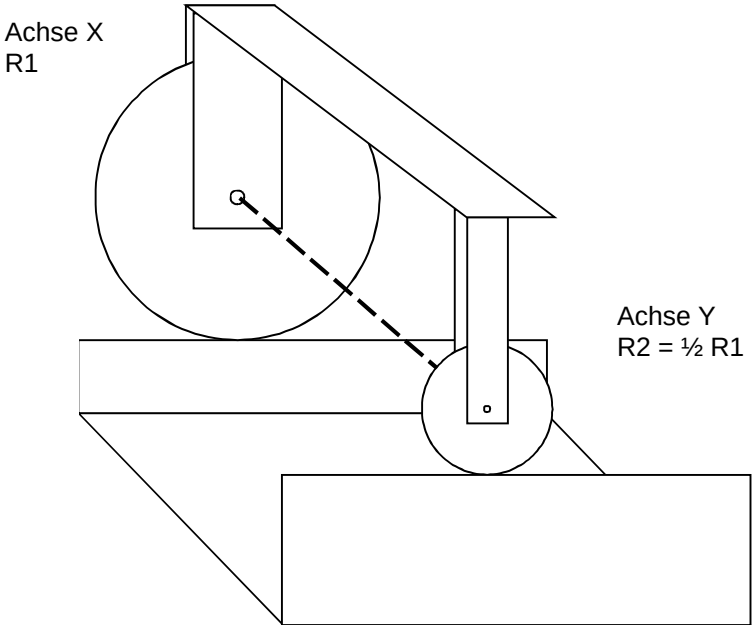


Bild 6.11

Eine Maschine hat zwei weit auseinander liegende Achsantriebe, die synchron miteinander fahren sollen. Die mechanische Kopplung ist schwierig zu lösen oder sehr teuer. Es bietet sich also eine Lösung mit zwei einzelnen Antrieben an, welche synchron miteinander verfahren und dadurch, wie durch ein Getriebe verbunden sind. Man spricht daher von einem 'Elektronischen Getriebe'.

Im Bild 6.11 ist eine virtuelle Achsverbindung der beiden Räder eingezeichnet. Wäre diese vorhanden, könnte der Wagen nicht geradeaus fahren, da der Umfang der Räder unterschiedlich ist. Durch das 'Elektronische Getriebe' ist es möglich, diese beiden Räder mit unterschiedlichen Durchmessern mit Einzelantrieben auszurüsten und im richtigen Verhältnis miteinander synchron laufen zu lassen, so dass der Wagen geradeaus fährt. Wenn X als Master gewählt wird, muss P 'x' 06 = 2 (Achse Y) und P 'x' 07 = Y/X gesetzt werden.

6.12 Funktion "Trigger-Out"-Signal

Pro Achse steht ein "Trigger-Out"-Signal zur Verfügung. Mit diesem Ausgang kann eine sehr genaue und schnelle Reaktion, abhängig von der Achsposition, realisiert werden. Überschreitet die Achse den durch den Befehl SOx (Abschnitt 7.5.5, Zelle 2.14) gesetzten Wert P, wird das "Trigger-Out"-Signal aktiviert. (Dieses Signal steht dem Anwender an der Busklemme 1 oder 9 des H4-Moduls zur Verfügung).

Der Positionswert, welcher das "Trigger-Out"-Signal aktiviert, wird mit dem SOx-Befehl in das H4-Modul geschrieben. Der Wert wird in der im H4 gewählten Einheit verwendet. Ist diese Einheit ungenügend, so kann der Positionswert mit dem SOIx-Befehl impulsgenau geladen, respektive gesetzt werden. Beim Ausführen des SO..-Befehls wird das Statusbit (21 bei X-Achse) gelöscht und das "Trigger-Out"-Signal desaktiviert. SOx (Abschnitt 7.5.5, Zelle 2.12)

Überfährt die Achse nun den gesetzten Positionswert in der einen oder anderen Richtung, so wird das "Trigger-Out"-Signal aktiviert und das Statusbit gesetzt.

Die Polarität des Trigger-Ausgangs kann mit dem Parameter 62 gewählt werden.

Bei der Überschreitung der Triggerposition und P62 = 0 wird der Ausgang = H (positive Logik). Der inaktive Zusatnd ist = L.

Bei der Überschreitung der Triggerposition und P62 = 1 wird der Ausgang = L (negative Logik). Der inaktive Zusatnd ist = H.

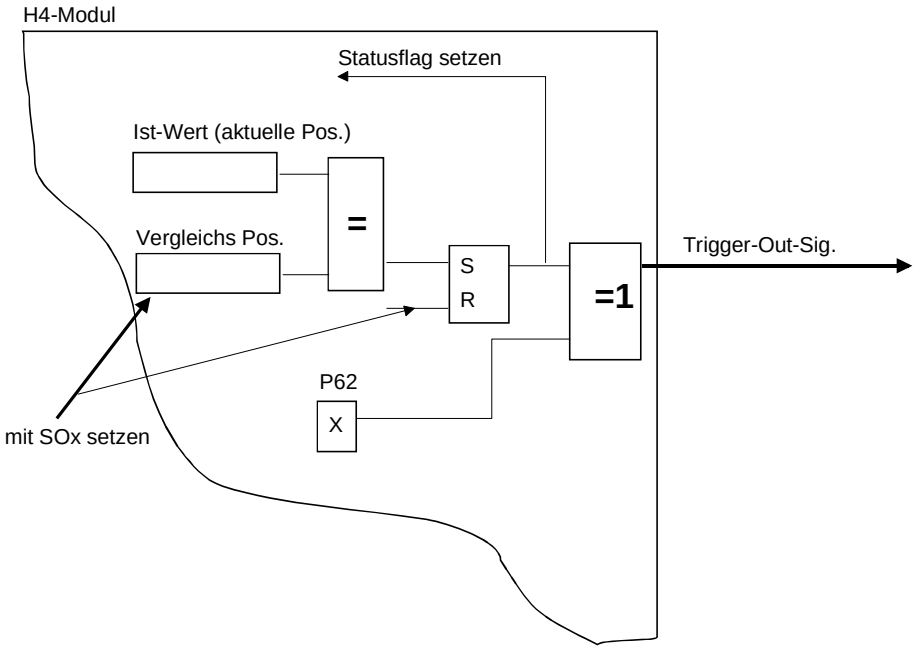


Bild 6.12

6.13 Funktion "Position Capture Input"-Signal

Dieser Hardware-Eingang auf dem H4-Modul ist für jede Achse einmal vorhanden. Wird diese Funktion aktiviert (SC 'x'), so wird beim Ansprechen des Eingangs 'PCI' (Busklemme A oder B) der aktuelle Positionswert der Achse im "Position Capture"-Register gespeichert. Dieser Wert kann anschliessend aus dem H4 ausgelesen werden (QC 'x'). Wurde eine Capture-Position erfasst, wird das entsprechende Statusflag 'C' der Achse gesetzt (Abschnitt 7.5.5, Zelle 2.12).

Folgende Befehle, Parameter und Flag gehören zur "Position Capture"-Funktion:

SC 'x'- / QC "x"- Befehl
Flag ‘PCI eingetragen’ (für X-Achse = Flag 15)

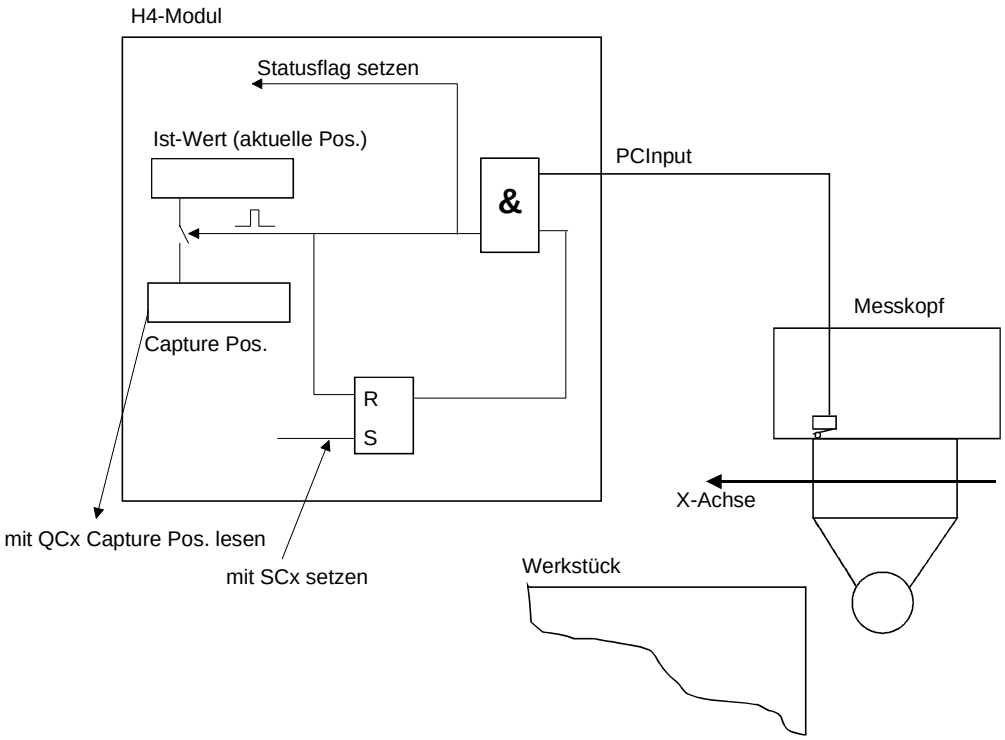


Bild 6.13

Beispiel:

Mit einem Messkopf an einem kartesischen Roboter wird ein Werkstückrand ausgemessen. Spricht der Messkopf an, wird sofort die aktuelle Position gespeichert. Diese kann dann aus dem H4-Modul ausgelesen werden.

6.14 "Change on the fly" - Funktion

Verschiedene Parameter werden beim Ausführen von Befehlen verrechnet oder berücksichtigt. Wurde der Befehl im H4-Modul interpretiert und an den Ausführungsprozess weitergegeben, können die Parameter verändert werden, haben jedoch erst auf den nächsten Befehl eine Auswirkung.

Die in der Parameterliste (Abschnitt 7.6) in der Rubrik "change on the fly" mit 'JA' gekennzeichneten Befehle haben jederzeit Einfluss auf die Achse. Werden diese während einer Bewegung geändert, wirken diese sofort auf die Achse.

So können z.B. die Regelparameter während der Bewgung verändert werden und haben auch sofort Einfluss auf die Regelung. Im Jog-Betrieb kann auch während des Verfahrens die Jog-Geschwindigkeit verändert werden und die Geschwindigkeit wird "fliegend" geändert.

6.15 Beschreibung von zirkularen periodischen Achsen

Mit dem Parameter P'x'05 wird angegeben, ob eine Achse linear **(n=0)** oder rotierend **(n>0)** behandelt wird. Eine lineare Achse ist hardwaremässig in der Bewegung durch Endschalter begrenzt. Eine rotierende Achse hat keine Begrenzungen.

Dieser Parameter definiert die Überlaufposition (Periode) der rotierenden Achse. Der Anzeigebereich des 'OPX'-Befehls liegt immer innerhalb der definierten Periode, da die Endposition und die Anfangsposition am physikalisch selben Ort liegen. Am Bereichsende wechselt die Position vom Maximalwert auf Null oder von Null auf den Maximalwert, abhängig von der Drehrichtung. Es kann also maximal ein Weg von P'x'05/2 (halbe Periode) gefahren werden. die Bewegung wird immer in der Richtung des kürzeren Weges ausgeführt.

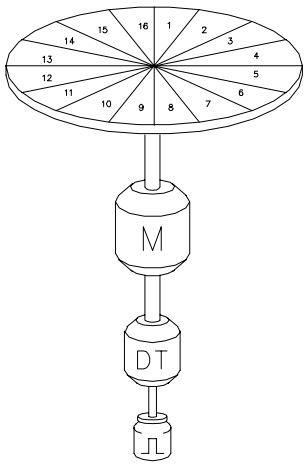
Wird in mm oder inches gearbeitet (P'x'01 und P'x'03), entspricht die Überlaufposition P'x'05 dem Umfang der Rotationsachse. Wird in Winkelgraden gearbeitet, wird P'x'05 normalerweise auf 360° gesetzt. Für die Einheit 'Impulse' siehe Beispiel 1.

Beispiel 1: Eine rotierende Scheibe mit 16 Positionen ist von einem Motor direkt angetrieben.

Ein Encoder mit 2000 Impulsen/Umdrehung ist auf der Motorenachse befestigt. Die Bewegung soll in 16 Schritten unterteilt werden.

Die Parameter für die Achse 'x' werden:

P'x'01 = 2 (Einheit = Winkelgrade)
P'x'02 = 2000 (Impulse /Umdrehung)
P'x'03 = 16 (16 Schritte/Umdrehung)
P'x'05 = 16 (16 Schritte/Umdrehung)

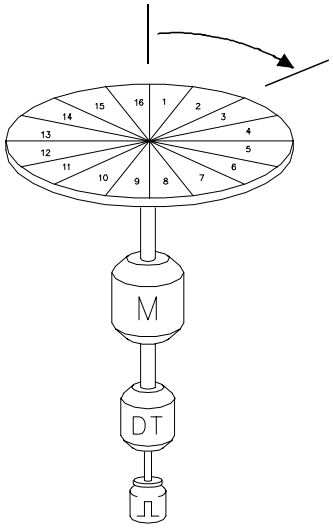


Die Bewegungsbefehle können somit direkt in Schritten eingegeben werden

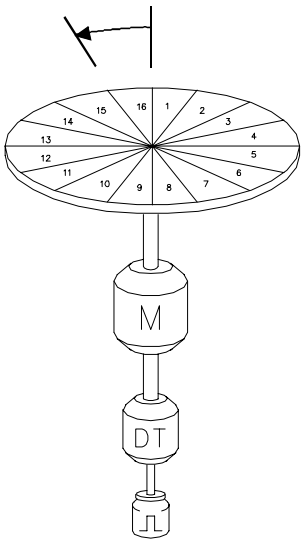
'x'A = 2 Achse 'x' fährt zum 2. Segment der Drehscheibe. (Die Bewegung erfolgt im Gegenuhrzeigersinn

mit:
'x'A = 18 führt die Achse die genau gleiche Bewegung wie 'x'A2 aus (18 - 16 = 2)
Die zusätzliche Umdrehung wird nicht ausgeführt.

Die Endposition ist in beiden Fällen OP'x' = 2.



Aber mit:
'x'A = 15 bewegt sich die Achse rückwärts zum 15. Segment (im Uhrzeigersinn), da 15 - 16 = -1.



Beispiel 2: Laufband-System mit Bewegungseinheit in Schritten.

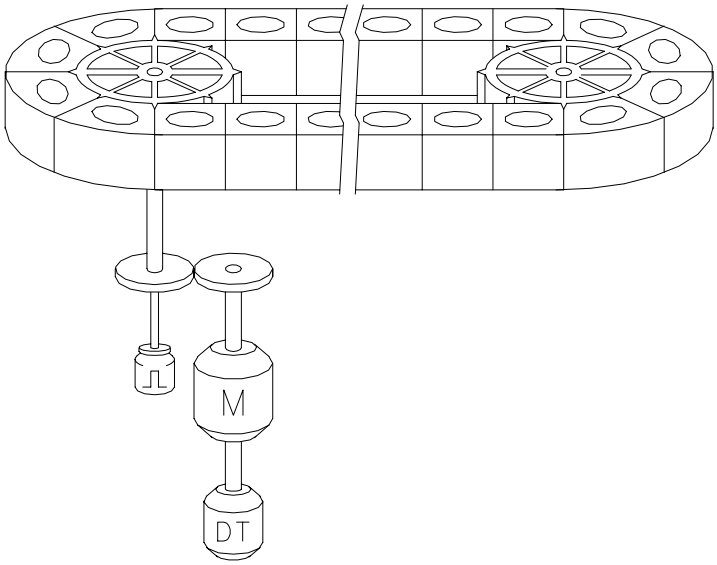
Die Zahnräder haben 8 Zähne.
Ein Encoder mit 200 Impulsen/Umdrehung ist auf der Zahnradachse befestigt.
Das Laufband ist in 48 Schritte unterteilt.

Die Parameter werden.

P'x'01 = 2 (Einheit = Winkelgrade)
P'x'02 = 2000 (Impulse/Umdrehung)
P'x'03 = 8 (Schritte/Umdrehung Motor)
P'x'05 = 48 (Schritte/Umdrehung Laufband)

Die Bewegung wird in Schritten eingegeben:

'x'A = 36
'x'A = 56.7
'x'R = 49
'x'R = -3.67



Notizen

7. Programmerstellung

7.1. Einführung

Die "PCD4.H4..Motion Control"- Einheit hat eine Vielzahl von Befehlen und Parametern. Die Parameter für Modul, System oder Achseinstellungen können in verschiedenen Betriebslagen geändert werden. Nebst den Befehlen für "Motion Control" stehen auch spezielle Befehle für "System Control" und "Program Control" zur Verfügung. Alle Befehle und Parameter können durch zwei verschiedene Arten des Zugriffs geändert oder gelesen werden.

a.) von der CPU der PCD aus mit Standard-FBs via PCD-Bus.
b.) vom PC aus via Frontstecker am H4.

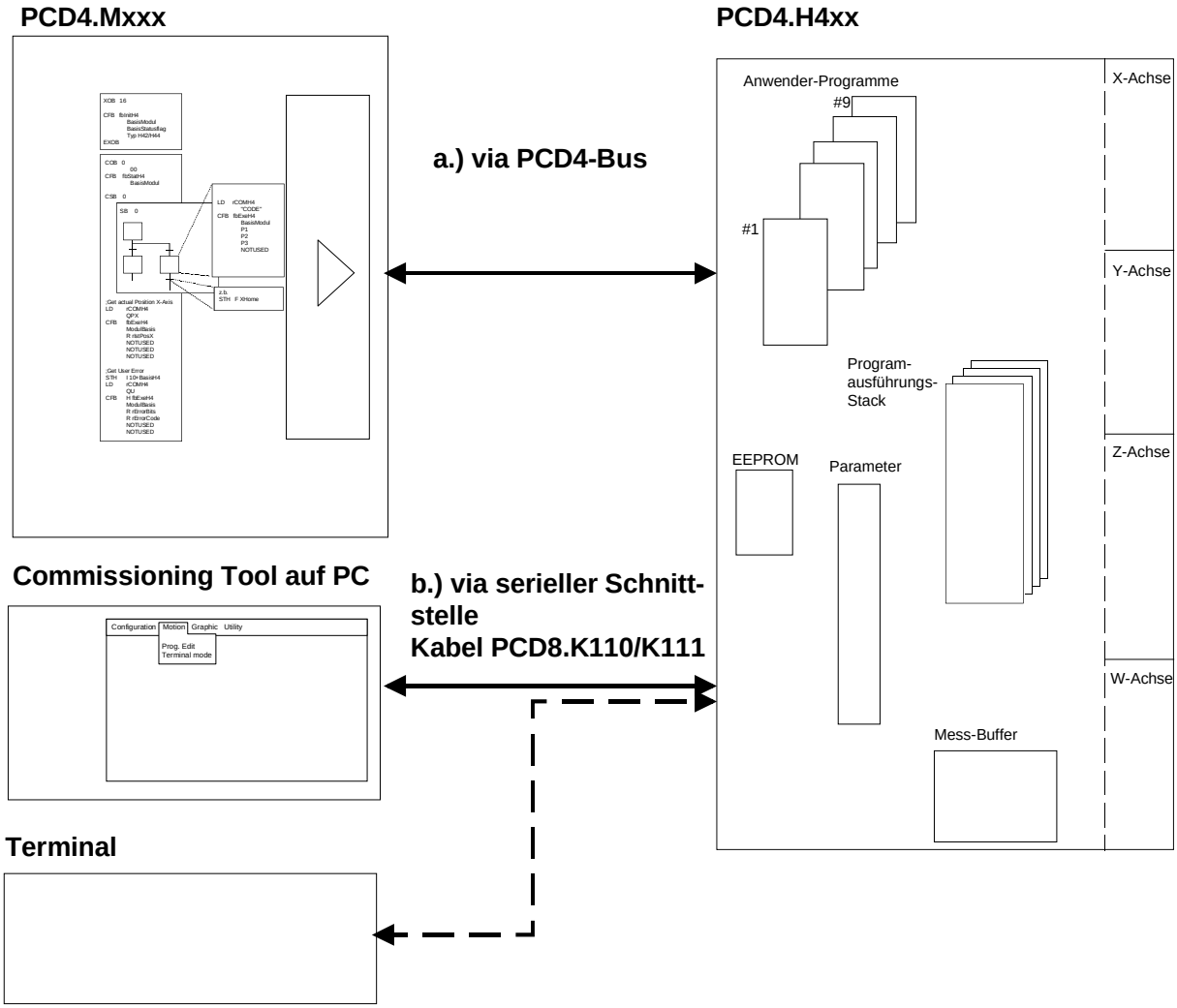


Bild 7.01

7.2 Programmierkonzept

Die Programmierung des H4-Moduls kann grundsätzlich in zwei Arten aufgeteilt werden:

a) Programmierung des H4-Moduls mit dem CP-Tool

b) Programmierung des H4-Moduls durch die PCD-Anwendersoftware

Für beide Möglichkeiten gilt jedoch:

- Der Benutzer kann mit verschiedenen Befehlen oder Parametern das Modul einstellen und justieren.

- Ganze Bewegungs-Programme können ins H4-Modul geladen oder dem Modul zur Ausführung übergeben werden. (Programm Mode). Selbstverständlich können auch einzelne Befehle ausgeführt oder einzelne Parameter geändert werden (Immediate Mode). Verschiedene Anwender-Bewegungs-Programme können im H4-Modul hinterlegt und bei Bedarf zur Ausführung gebracht werden.

a) Programmierung des H4-Moduls mit dem CP-Tool

Das Commissioning/ Programming-Tool, "CP" genannt, läuft auf einem Personalcomputer (PC) und ist via serieller Schnittstelle mit dem PCD4.H4 verbunden. Dieses Tool ist speziell für das H4-Modul entwickelt worden und kann alle Funktionen des H4 ausführen. Das CP-Tool kann nicht nur Bewegungs-Programme editieren und in das H4-Modul laden, es kann diese Programme auch auf der Festplatte speichern. Nebst den Programmen können auch alle Parameter geändert und ebenfalls auf der Festplatte gespeichert werden.

b) Programmierung des H4-Moduls durch die PCD-Anwendersoftware.

Zur Ansteuerung des H4-Moduls von der PCD-Anwendersoftware her, stellt die Firma SAIA-Burgess Electronics Standard-FBs zur Verfügung. Dadurch kann der Anwender durch einfaches Aufrufen von FBs alle gewünschten Funktionen im H4-Modul ausführen lassen. So können komplexe Bewegungs-Programme in der PCD- Aplikationssoftware erstellt werden. Ebenso ist das Ändern von Parametern durch Aufrufen von FBs in der Applikationssoftware sehr einfach gelöst. Durch spezielle Funktionen können auch ganze Verfahr-Programme in das H4-Modul geladen werden. Zur Sicherung von Verfahrprogrammen, welche mit dem CP-Tool editiert wurden, können diese aus dem H4-Modul ausgelesen, in der PCD gespeichert und auch wieder ins H4-Modul geladen werden.

Synchronisierung von Programmen

Erstellte Programme können mit RUN'p' (p = Programm Nr. 1…9) unmittelbar und unabhängig voneinander gestartet werden.

Mit BREAK'p' können diese nach der Ausführung des laufenden Befehls gestoppt werden. Werden mehrere Befehle 'blended' ausgeführt, werden diese wie ein Befehl behandelt und können mit BREAK nicht einzeln unterbrochen werden.

Die Funktion RUN kann auch hardwaremässig mit dem Starteingang (Klemme 15) und die Funktion BREAK mit dem Stopeingang (Klemme 7) realisiert werden.

Mit dem Parameter 95 wird angewählt auf welches Programm die Start- und Stop-Eingänge wirken.

Sollen Programme verschachtelt werden (max. 4 Ebenen), kann der RUN Befehl innerhalb eines Programms aufgerufen werden:

```
Bsp.    1 - XA20
        2 - RUN3
        3 - XA40
        4 - END
```

Wenn zwischen zwei 'blended' ausgeführten Bewegungen ein anderes Programm gestartet werden soll, ist zu beachten, dass der RUN-Befehl in dem Moment ausgeführt wird, wo die Überblendung einsetzt, also etwas vor der Position 20 (abhängig von P44 und der Geschwindigkeit). (Siehe auch Abschnitt 6.6 'Blended move').

Für positionsabhängige Aktionen kann der Trigger-Ausgang verwendet werden.

Sollen Programme nur bis zu einer bestimmten Zeile abgearbeitet werden, kann mit dem Befehl STOP ein 'Break point' gesetzt werden:

```
Bsp.    1 - XA20
        2 - STOP
        3 - XA40
        4 - END
```

Der Programmstop kann mit RUN'p' unmittelbar oder aus einem anderen Programm verschachtelt oder auch mit dem HW-Eingang Start wieder aufgehoben werden.

Um den Programmablauf zu verfolgen, kann mit dem Befehl QL'p' (p = Programm Nr.) die Programmzeile abgefragt werden, welche momentan ausgeführt wird. Ist das Programm beendet (oder noch nicht gestartet), wird 0 zurückgegeben.

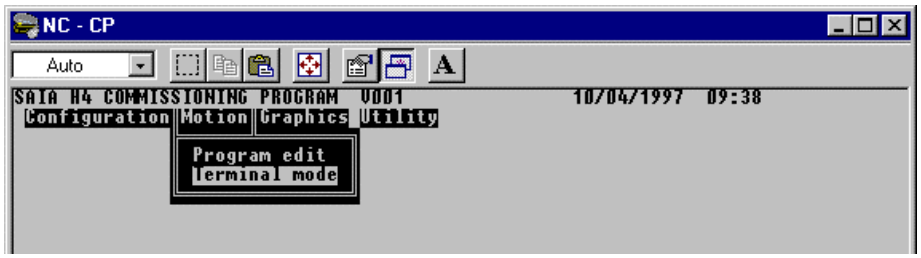
7.3 Programmierung mit CP-Tool (Commissioning / Programming tool)

Das 'Commissioning / Programming Tool' (CP) ist eine Software für einen IBM kompatiblen PC. Das Tool läuft unter dem Betriebssystem DOS. Die Installation und das Programmieren sind sehr einfach.

7.3.1 Installation

Durch Kopieren aller Dateien von der Diskette auf die Festplatte z.B. ins Verzeichnis \SAIA-H4 ist die Installation erfolgt. Es werden ca. 2 MByte freier Speicher auf der Harddisk sowie ca. 600 kBytes freier Arbeitsspeicher benötigt. Durch Aufrufen aus dem DOS-Prompt 'CP' <CR> wird das Programm gestartet. Nach dem Starten des CP erscheint das SAIA-Signet.

Durch Betätigen einer beliebigen Taste erscheint das "Pull down"-Menü.



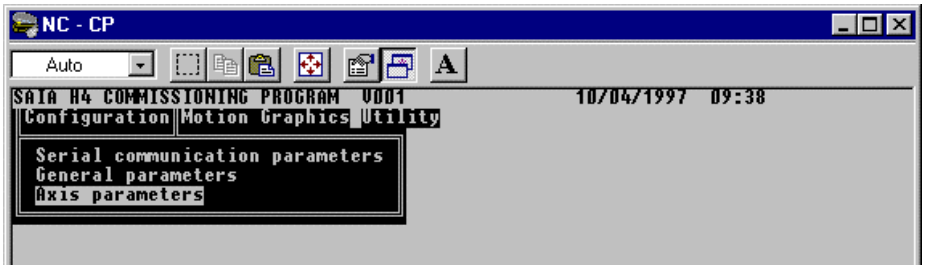
Mit den Tasten <↓>, <↑>, <←> und <→> kann der Cursor zur Menüwahl verschoben werden.

Das Öffnen eines Menüs erfolgt mit <CR>. Erklärungen der Menüpunkte siehe Abschnitt 7.3.3 Menü 'Erklärung'.

<ESC> = Menü
Durch Betätigen der <ESC>-Taste wird das Pulldown-Menü aktiviert.

Die Taste **F1** wird nicht dargestellt. Diese dient zum Aufrufen der Hilfe.

7.3.2 Menü-Übersicht




```
- Configuration
      - Serial communication parameter
      - General parameter
      - Axis parameter
- Motion
      - Program edit
      - Terminal mode
- Graphics
      - Graphics
- utility
      - Language change
      - Color change
      - Graphics color change
      - Select color for Text in graphic
      - DOS Shell
      - Quit
```

Wird ein Menü gewählt, so werden noch weitere Funktionstasten dargestellt.

Für 'Program Edit' sind z.B. die folgenden Tasten verfügbar:

Folgende Funktionstasten stehen zur Auswahl:'

<F4> = From/To H4 mod
Die Funktionstaste 4 setzt das Ziel und die Quelle der Daten für die beiden Funktionen F2 und F3 fest (PC oder H4).

<F2> = Read/Upload Prg.
wenn <F4> = PC:

- Liest die Datei .cnf (config) in den CP-Arbeitsspeicher

wenn <F4> = H4:

- Liest die Konfiguration aus dem H4 in den CP-Arbeitsspeicher

<F3> = Save/Downl. Prg.
wenn <F4> = PC:

- Speichert die Konfig. aus dem CP-Arbeitsspeicher auf die Festplatte.

wenn <F4> = H4:

- Lädt die Konfiguration aus dem CP-Arbeitsspeicher ins H4.

7.3.3 Menü Erläuterung

Die nachfolgend beschriebenen Funktionen sind entsprechend dem Hauptmenü gegliedert.

Configuration

'Serial communication parameters'
Es kann das COM-Port des PCs gewählt werden. (COM1: oder COM2:). Die Kommunikationsparameter sind fix (9600,8,E,1).

'General parameters'
Die relevanten Parameter, welche für das H4-Modul generell gelten, können eingestellt werden. Es sind dies die Parameter 90 ... 98.

Funktionstasten-Beschreibung:

Sind alle Parameter richtig eingestellt, so können diese mit <F3> gespeichert werden. Das Ziel der Speicherung muss vorgängig mit der <F4>-Taste festgelegt werden (H4 oder PC). Mit <F2> können die Parameter wieder gelesen werden (mit <F4> anwählen ob vom H4 oder vom PC).

'Axis parameters'
Alle Parameter der Achsen werden angezeigt. Einige Parameter sind von den Einstellungen der anderen abhängig. Es empfiehlt sich daher, die Parameter von oben nach unten auszufüllen. Die Limiten werden im Edit-Fenster angezeigt.

Durch das Betätigen der Funktionstaste <F8> kann die nächste Achse gewählt werden. Speicherung siehe 'General Parameter'.

Motion

'Program edit'
Der Programmeditor arbeitet offline, d. h. es können Programme geschrieben, geändert und gespeichert werden ohne mit dem H4-Modul verbunden zu sein. Wird eine Verbindung zum H4 hergestellt und mit der <F4>-Taste angewählt, so können Programme vom H4 gelesen oder ins H4 geladen werden. Es können max. 9 Programme ins H4 geladen werden (siehe auch Abschnitt 6.2.2). Die Befehle werden im H4-Modul abgelegt, jedoch noch nicht ausgeführt.

Um die Eingabe eines Befehls auszuführen, wird die <Enter>-Taste (<↵>) betätigt. Ein Editierfenster am unteren Rand des Bildschirms wird geöffnet und erlaubt die Eingabe des gewünschten Kommandos. Die möglichen Kommandos oder Parameter und ihre Funktion können dem Abschnitt 7.5 'Befehlsliste' und Abschnitt 7.6 'Parameterliste' entnommen werden.

Folgende Funktionstasten stehen zur Auswahl:'

<F4> = From/To H4 mod
Die Funktionstaste 4 setzt das Ziel und die Quelle der Daten für die beiden Funktionen F2 und F3 fest (from/to PC dis oder from/to H4 Mod).

<F2> = Read/Upload Prg.
wenn <F4> = from/to PC dis

- Liest die Datei .prg (program) in den CP-Arbeitsspeicher

wenn <F4> = from/to H4 Mod

- Liest ein Programm aus dem H4 in den CP-Arbeitsspeicher

<F3> = Save/Downl. Prg.
wenn <F4> = from/to PC dis

- Speichert ein Programm. aus dem CP-Arbeitsspeicher auf die Festplatte.

wenn <F4> = from/to H4 Mod

- Lädt ein Programm aus dem CP-Arbeitsspeicher ins H4.

<F8> = Analyse

- prüft die eingegebenen Programmzeilen auf Syntaxfehler und zeigt in einem Fenster die fehlerhaften Zeilen an.

<F5> = Delete line

- löscht die aktuelle Programmzeile und schliesst die nachfolgenden Linien auf. Die gelöschte Zeile wird zwischengespeichert (Funktion 'Cut')

<F6> = Insert line

- schiebt eine leere Programmzeile auf der aktuellen Zeile ein und verschiebt die nachfolgenden Zeilen.

<F7> = Paste line

- schiebt die zuletzt gelöschte Programmzeile auf der aktuellen Linie ein und verschiebt die nachfolgenden Zeilen.

Vor dem Laden eines Programms ins H4-Modul, wird die Syntax überprüft und bei Fehlern auf die fehlerhafte Zeile verwiesen. Wurden Fehler erkannt, wird das Programm nicht geladen. Ein korrekt geladenes Programm kann im Terminal-Modus gestartet werden.

Die Einstellung der Funktionstaste <F4> ist immer zu beachten, um sicherzustellen, dass das Programm am gewünschten Ort gespeichert wird.

Befehle, welche gemäss Befehlsliste nur im Immediate-Modus ausgeführt werden können, werden im Programm nicht akzeptiert. Wenn
P94 = 2 (axis present = x, y) werden Befehle und Parameter für Achse Z und W nicht akzeptiert.

'Terminal mode'
Im Gegensatz zum Programmeditor werden hier die eingegebenen Befehle oder Parameter sofort ausgeführt (Immediate: siehe auch Abschnitt 7.5.1 Syntaxerklärung, Kasten 'Ausführungsmode'). In diesem Fenster erscheinen weitere Funktionstasten.

<**ESC** Menu> <**F9** Previous commands REV> <**F10** Previous commands FOR.>

'ESC Menu': Verlassen des Terminal-Modus, zurück zum Menü.

F9 Previous commands reverse

F10 Previous commands forward

Die Eingabezeile besitzt einen 10-zeiligen Ringbuffer. Dieser kann rückwärts <F9> oder vorwärts <F10> gerollt werden, um vorgängig eingegebene Befehlszeilen auf einfache Weise wieder zu verwenden. mit <CR> (<↵>) wird auch die letzte Eingabe in die Befehlszeile geholt. Es ist möglich, mehrere Befehle, durch Leerschläge getrennt, in die Eingabezeile zu schreiben, wobei diese Befehle dann praktisch gleichzeitig ausgeführt werden.

Beispiel: QPX QVX QEX <CR>

Programm ausführen:
Ein ins H4 geladene Programm kann durch den Befehl "RUN" + Programmnummer gestartet werden. Es ist jedoch zu beachten, dass die Parameter gesetzt und geladen sind und dass die relevanten Achsen korrekt initialisiert worden sind (es ist 'Enable' und 'Home' zu beachten).

Zum oben (rechts) erscheinenden Achsenstatus-Fenster siehe Abschnitt 7.5.5, Zelle 2.12, wobei die aktiven Bit mit einem Buchstaben oder Zeichen angezeigt werden (z.B. 'Limit switch positiv erreicht', '+' und 'h')

Graphics

'Graphics'
Dieser Mode erlaubt dem Benutzer verschiedene Funktionen der Achse zu betrachten. Der Benutzer kann diese Grafik beim Einrichten der Achse nutzen. Verändert der Benutzer die PID-Parameter, so kann er anhand der Grafik die Auswirkungen direkt erkennen. Die veränderten Regelparameter werden erst beim Auslösen der Datenerfassung mit <F4>/<F5> aus dem H4 übermittelt. Anschliessend an eine Testbewegung benötigt das CP eine kurze Zeit, um die erfassten Messwerte vom Modul an den PC zu übertragen und darzustellen. Es können max. 4 Kurven der mit <F8> angewählten Achse aufgezeichnet werden. Zur Auswahl stehen 'Ist-Position', 'Soll-Position', 'Schleppfehler', 'Soll-Geschwindigkeit', 'Soll-Beschleunigung', 'Analog-Ausgangsspannung'.

Die dadurch gefundenen Parameter müssen anschliessend ins EEPROM nullspannungssicher abgelegt werden.

Die meisten Funktionstasten und anderen Angaben auf diesem Bildschirm sind selbsterklärend. Hier in Kürze einige ergänzende Hinweise:

Mit <F5> werden das mit <F3> gewählte Programm gestartet, die Datenerfassung ausgelöst und die Regelparameter an das H4 übertragen.

Mit <F4> werden die Datenerfassung gestartet und die Regelparameter an das H4 übertragen. Diese Funktion ist für eine sich bereits in Bewegung befindliche Achse vorgesehen.

Utilitiy

'Language change'
Die Sprache des CP kann hier gewählt werden. (ENG, ITA, [GER, FR])

'Color change'
Die Farben der Fensterdarstellungen mit Vordergrund und Hintergrund können gewählt werden. Monochrom ist auch wählbar. Durch das Betätigen der Pfeiltasten <←> oder <→> können die verschiedenen Fensterbereiche gewählt werden. Mit <CR> und dann der <Space> Taste erfolgt die Farbwahl; <F4> monochrom, <F5> Grundeinstellung.

'Graphics color change'
Die Farben im Grafikfenster mit Vordergrund und Hintergrund können gewählt werden.

'Select color for Text in graphic'
Die Farben der Texte in der Grafik können gewählt werden, um so die optimale Darstellung der Grafik zu erreichen. Farbwahl siehe oben.

'DOS Shell'
Temporäres Verlassen des CP ohne Datenverlust im Arbeitsspeicher. (Zurück mit 'Exit')

'Quit'
Verlassen des CP. Beim Beenden gehen die Daten im Arbeitsspeicher verloren. 'Quit' kann auch mit <Ctrl> + <z> veranlasst werden.

Notizen

7.4 Programmierung mit FBs

In diesem Teil des Handbuchs wird erklärt, wie die Standard-FBs von SAIA-Burgess Electronics für das H4-Modul zu verwenden sind. Die Parameter und das Aufrufen der FBs werden beschrieben. Die Erklärungen zu den einzelnen H4-Befehlen (Commands) oder Parametern wird in den Abschnitten 7.5 'Befehlsliste' und 7.6 'Parameterliste' erklärt.

7.4.1 Einführung

Der Anwender erstellt eigene Programme (siehe Bild 7.04 "Anwenderprogramm") indem er die drei Treiber-FBs für das H4-Modul aufruft. (siehe Bild 7.4 "H4-FBs"). Zuerst muss mit dem 'fbInitH4' das Modul initialisiert werden (meist im XOB 16). Im COB muss der 'fbStatH4' zyklisch aufgerufen werden, damit der Status des H4 aufgefrischt wird. (Die aktualisierten Statusflags können im Anwenderprogramm weiterverarbeitet werden.). Um Befehle zu senden oder Parameter zu verändern, muss zuerst das entsprechende Kommando in das Laderegister (rComH4) geladen werden. und anschliessend der 'fbExecH4' aufgerufen werden. Um hier einen leichten Überblick zu erhalten, ist das Bild 7.04 zu betrachten. Die beiden FBs 'fbUpLdH4' und 'fbDnLdH4' sind für den Transfer von ganzen Programmen vorgesehen (Abschnitt 7.7).

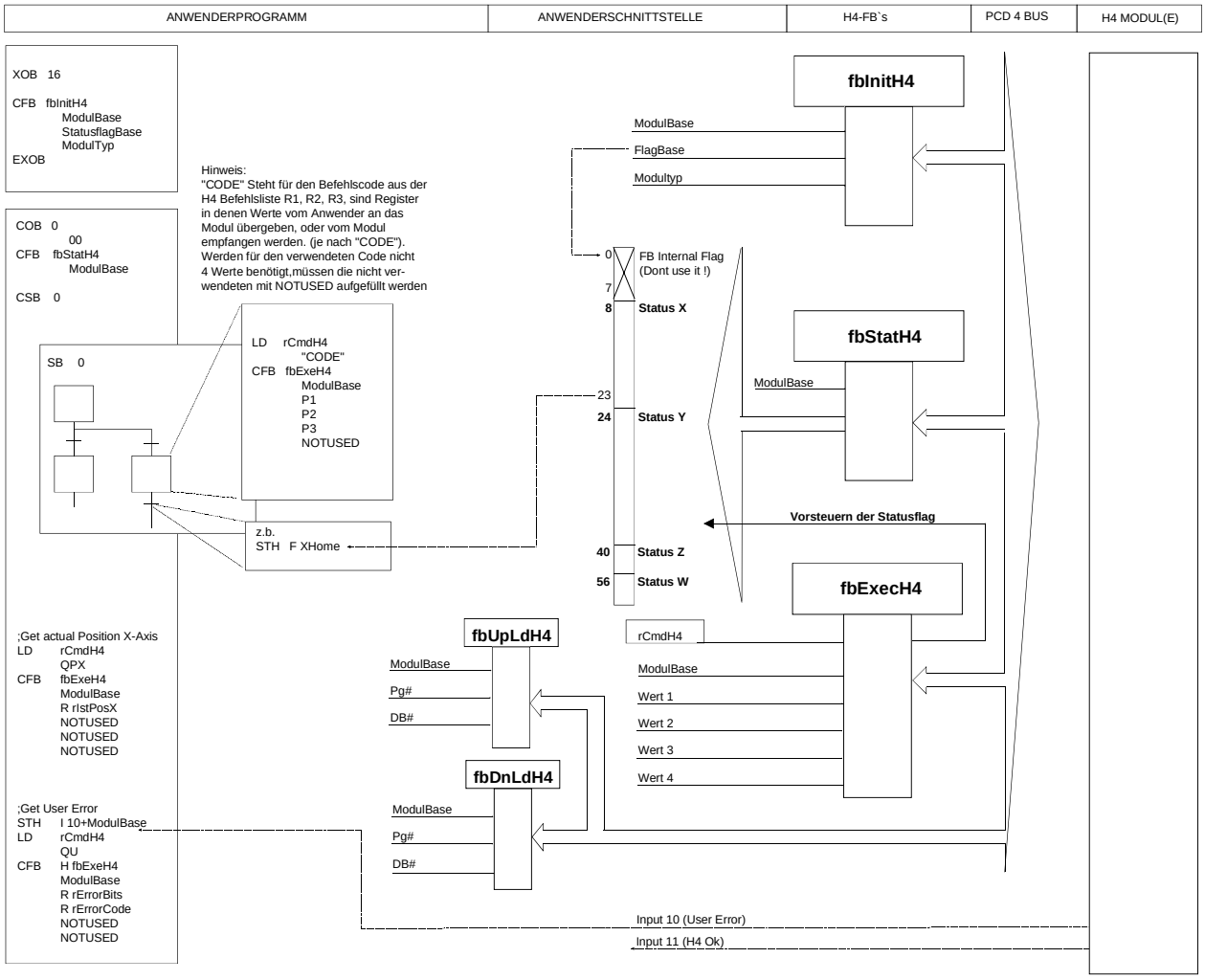


Bild 7.04

Ein Anwender-Programm muss assembliert, gelinkt und anschliessend in die CPU geladen werden. Das Einbinden der Treiber-FBs wird im Abschnitt 7.4.4 erklärt.

7.4.2 Adressierung des H4-Moduls

Das Modul benötigt, wie alle andern PCD4-Module, 16 Adressen. Die tiefste Adresse des Moduls ist die Basis-Adresse. alle weiteren Adressen werden entsprechend verwendet. Der Anwender muss also die Basis-Adresse des Moduls und die Basis-Adresse der Flags definieren. (siehe 'fbInitH4'-FB).

Die Bedeutung der der einzelnen Adressen: (+ Basis-Adressen)

	Data read	**Data write**
0	Data bit 0 (LSB)	Data bit 0 (LSB)
1	Data bit 1	Data bit 1
2	Data bit 2	Data bit 2
3	Data bit 3	Data bit 3
4	Data bit 4	Data bit 4
5	Data bit 5	Data bit 5
6	Data bit 6	Data bit 6
7	Data bit 7 (MSB)	Data bit 7 (MSB)
8	Data available	Write (WR)
9	Channel busy	Read (RD)
10	User Error Set	Clear channel
11	DSP Ready	Reset DSP
12	Axis X "in-position"	
13	Axis Y "in-position"	
14	Axis Z "in-position"	
15	Axis W "in-position"	

Die Eingänge 12 .. 15 (Axis in position) dienen nur zur Information im Debugger. Für die Programmierung sollten, um Timingproblemen vorzubeugen, die entsprechenden Statusflags abgefragt werden.

7.4.3 Vorsteuern der Statusflags

(Statusflag siehe Abschnitt 7.5.4, Zelle 2.12)

Durch das Ausführen bestimmter Befehle mit dem Funktionsblock (FB) 'fbExecH4' werden einige Statusflags sofort nachgeführt (Vorsteuern), bevor durch den FB 'fbStatH4' die Statusflag aufgefrischt werden. Dies erleichtert das Verwenden der FBs. Das Aufrufen des FB 'fbStatH4' zwischen zwei 'fbExecH4' ist daher nicht notwendig. Ein zyklischer, asynchroner Aufruf des FB 'fbStatH4' ist ausreichend.

Die Statusflags werden nur durch den 'fbStatH4' aufgefrischt, was bei der Programmierung berücksichtigt werden muss.

Folgende Statusflag werden vorgesteuert:

- Achse in Pos. (8)
- Immediate Befehl in Ausführung (9)
- Capture-Pos. eingetroffen (15)
- Trigger-Position erreicht' (21)
- Home ist erfolgreich beendet (23)

7.4.4 Software Bibliothek mit Funktionsbausteinen (PCD9.H4..)

Das Standard-FB-Paket zum H4-Modul beinhaltet drei Dateien:

H4DEF.SRC In dieser Datei befinden sich nur die Definitionen der Basisadressen der Resourcen (Flags, Register, FBs). Der Anwender kann hier seine Basisadressen definieren. Diese Datei wird im H4FB included.

H4EXTN.DEF In dieser Datei befinden sich alle notwendigen Deklarationen, welche der Anwender in seinem Programm verwenden kann. Diese Datei muss im Anwenderprogramm mit '$include' eingebunden werden.

H4FB.SRC Diese Datei enthält den FB-Sourcecode sowie alle Deklarationen, welche innerhalb der FBs verwendet werden. Diese Datei darf nicht verändert und nicht mit SEDIT editiert werden.

Benötigte FB-Verschachtelungen:	1
Grösse der H4FB-Software (Zeilen):	< 4200

7.4.5 Dateien assemblieren und linken

Das folgende Bild 7.4.4 zeigt, wie die FBs des H4-Moduls in die Anwender-Software integriert werden können.

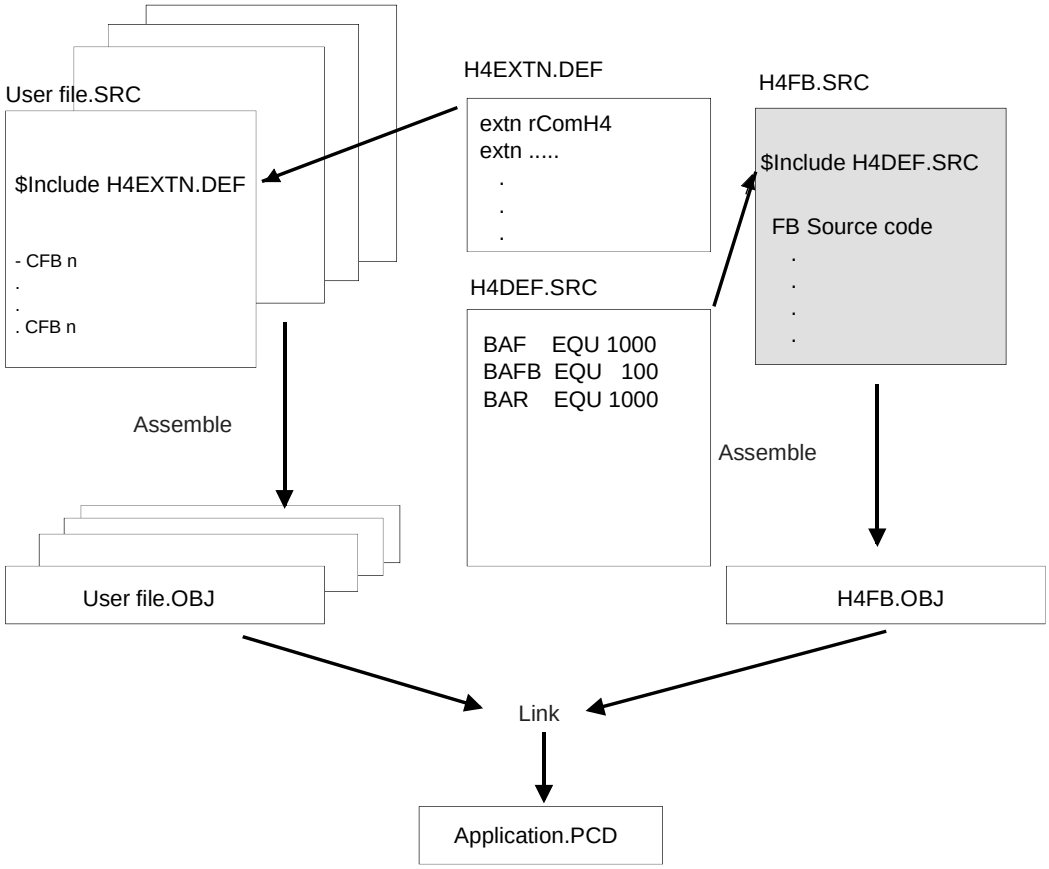


Bild 7.4.4

Die in den H4-FBs verwendeten Basisadressen werden in der Datei H4DEF.SRC vom Anwender für seine Anwendung eingestellt/definiert. Er muss also die Datei H4DEF.SRC verändern. Die ebenfalls von SAIA gelieferten Dateien H4FB.SRC und H4EXTN.DEF sollten **nie** durch den Anwender verändert werden.

Um die verschiedenen H4-Funktionen aus dem Anwender-Programm aufzurufen, muss dieses auch den Zugriff auf die Definitionsdatei H4EXTN.DEF haben. Daher wird die H4-Definitionsdatei bei jeder Anwenderdatei eingebunden (mit $include H4EXTN.DEF).

Der Anwender kann seine projektbezogenen Definitionen in einer eigenen Deklarationsdatei definieren, welche ebenfalls ins Anwenderprogramm eingebunden werden muss. Falls der Anwender die H4-Definitionen BAF, BAR und BAFB in seinem Programm auch verwenden will, muss er die Datei H4DEF.SRC entweder im Anwenderprogramm oder in seiner eigenen Deklarationsdatei einbinden.

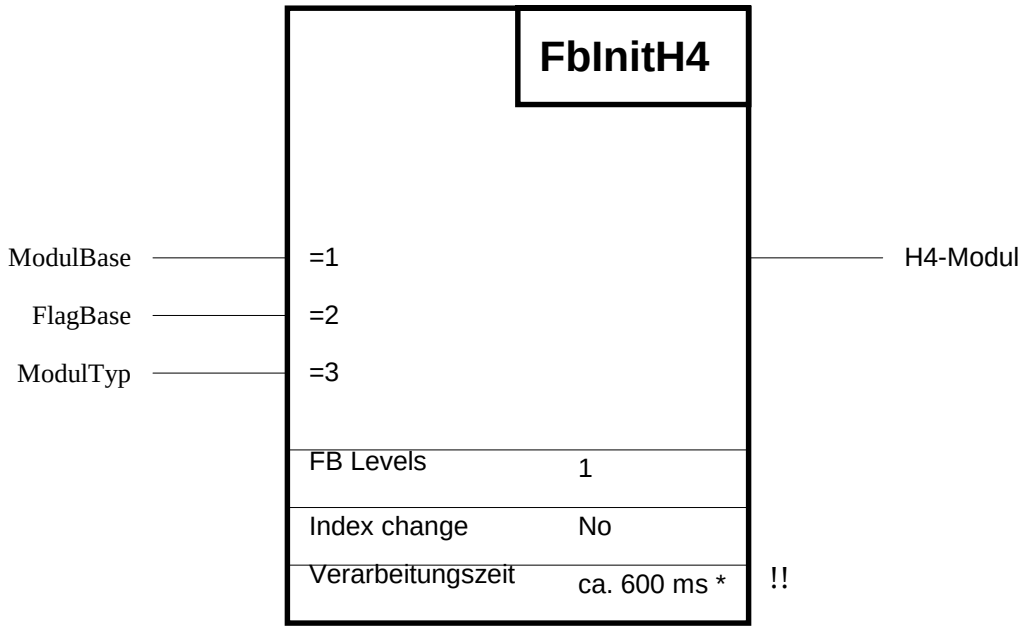
7.4.6 Beschreibung der FB

Zur Vereinfachung sind alle FB auf die gleiche Art beschrieben. Das nachfolgende Bild erklärt die graphische Darstellung der nächsten Seiten.

Für das Format und die Einheiten der eingegebenen Werte sind die Parameter P01 und P96 massgebend (siehe Abschnitt 6.9.3)

FbInitH4 Funktion: - H4-Modul Initialisieren FbInitH4

Achtung:
Der FB prüft die CPU Leistung, es ist daher zwingend, den FB nur in der Kaltstart-Routine (XOB16) aufzurufen. (Hier muss dieser FB nach den PCD-Befehlen und **vor dem DEFTB** eingefügt werden).



* abhängig von der CPU Leistung

Funktionsbeschreibung:

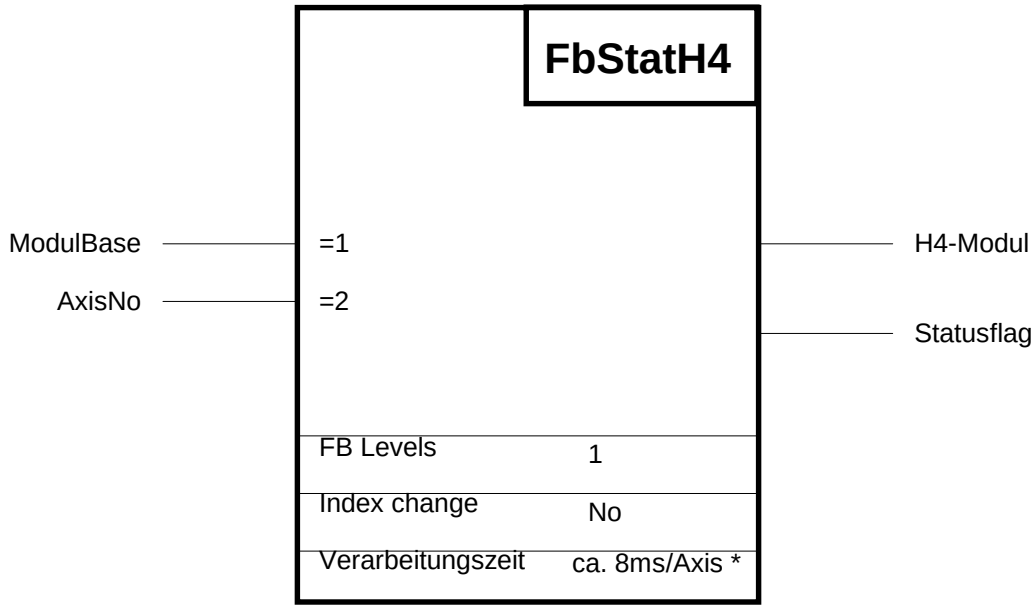
Diese Funktion wird zur Initialisierung des H4-Moduls und den dazu benötigten PCD Recourcen verwendet. Für jedes H4-Modul muss beim 'PowerUp' (XOB16) ein 'fbInitH4'-FB aufgerufen werden.

Werden mehrere H4-Module in einem PCD4-System verwendet, muss mit der Flag-Basisadresse sichergestellt werden, dass sich die Flag-Bereiche der verschiedenen Module nicht überschneiden.

Beschreibung der Ein- und Ausgänge:

Symbol	Beschreibung	Para-meter	Media		
			Type	Format	Adr-Bereich
ModulBase	Modul Basisadresse	Ja	K	Integer	0 -512
FlagBase	Flag Basisadresse	Ja		Integer	0 -8192
ModulTyp	Modulwahl H420 / H440	Ja	K	Integer	2 oder 4

FbStatH4 Funktion: - Lese Status des H4-Moduls FbStatH4



* abhängig von der CPU Leistung

Funktionsbeschreibung:

Diese Funktion liest das Status-Register für jede Achse aus dem H4-Modul. Das Status-Register wird anschliessend auf die Statusflags ausgegeben und kann somit vom Anwender abgefragt werden. Die Basis für die Statusflags wird von der Funktion 'fbInitH4' übernommen. Die Offset und die Bedeutung der einzelnen Statusflags sind in der Befehlsliste Abschnitt 7.5.3 (2.11) oder 6.2.6 beschrieben. Dieser FB muss zyklisch aufgerufen werden, damit die Achsenstatusflags aufgefrischt werden.

Beschreibung der Ein- und Ausgänge::

Symbol	Beschreibung	Parameter	Media		
			Type	Format	Adr-Bereich
ModulBase	Modul Basisadresse	Ja	K	Integer	0 -512
AxisNo	Achsnummer	Ja	-	Integer	0 - 15 (0-F)

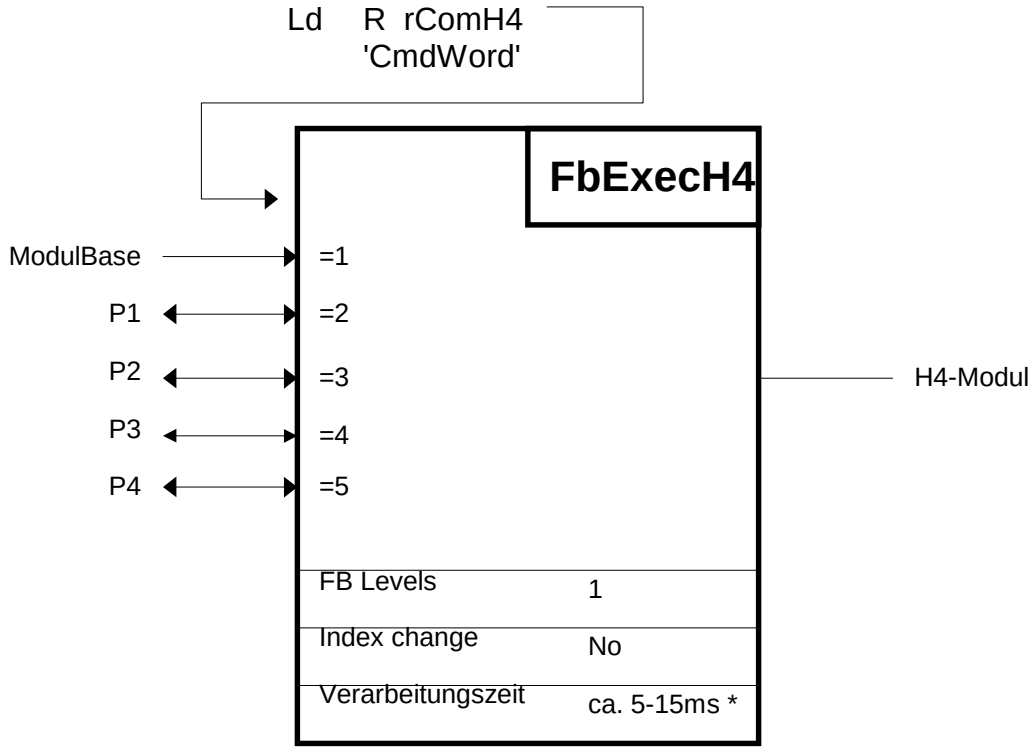
Achsnummer:

00h : eine Achse/ Zyklus
01h : Achse X
02h : Achse Y
04h : Achse Z
08h : Achse W
0Fh : alle vorhandenen Achsen in einem Zyklus

Kombinationen sind auch möglich:
z.B: Ch oder 12d = : Es werden die Statusflag der Achsen Z und W aufgefrischt. (h = Hex; d= Dezimal)

Wird eine Achse adressiert, welche nicht existent ist, wird keine Funktion ausgeführt.

FbExecH4 Funktion: - Execute Command to H4 FbExecH4



* abhängig von der CPU Leistung und Anzahl der Parameter

Funktionsbeschreibung:

Die Funktion 'fbExecH4' ist die befehlsausführende Funktion für alle Befehle und H4-Parameter. Diese Funktion kann die Daten des H4-Moduls schreiben oder lesen. Vor dem Aufruf dieses FBs muss das Register 'rComH4' mit dem entsprechenden Befehlswort geladen werden. Der FB selbst liest die ganze Information, welche zu erledigen ist, aus dem Befehlswort. Alle möglichen Befehlswörter sind im Abschnitt 7.5 aufgelistet.

Beschreibung Inputs und Outputs:

Symbol	Beschreibung	Parameter	Media		
			Type	Format	Adr-Bereich
ModulBase	Modul Basisadresse	Ja	K	Integer♥	0 -512
P1*	Parameter 1	Ja	R	Integer♥	0 - 4095
P2*	Parameter 2	Ja	R	Integer♥	0 - 4095
P3*	Parameter 3	Ja	R	Integer♥	0 - 4095
P4*	Parameter 4	Ja	R	Integer♥	0 - 4095

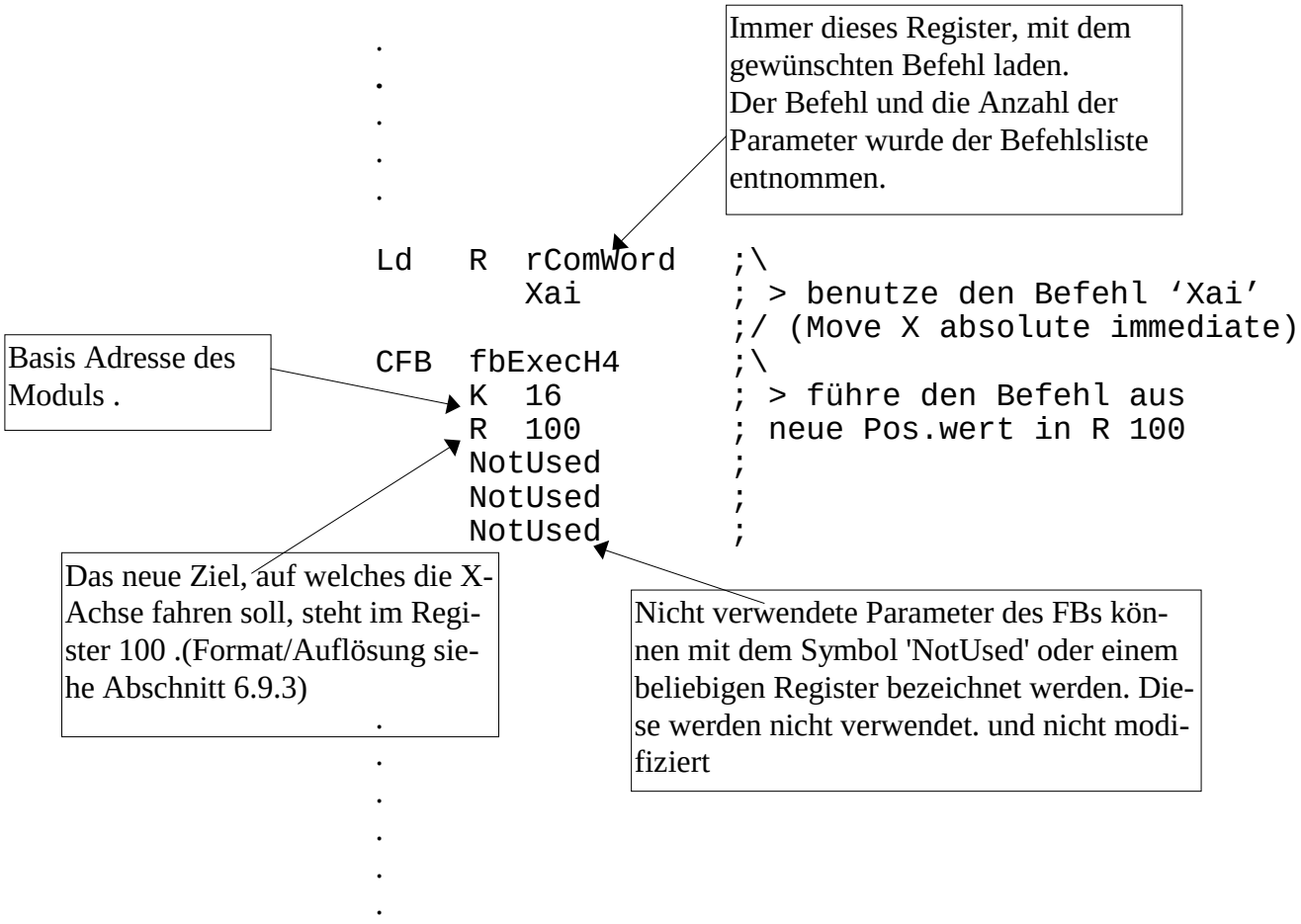
* Die Anzahl der Parameter ist aus der Befehlsliste zu entnehmen.

♥ Integer bzw. "Virtual Integer", je nach Befehl (siehe Abschnitt 6.9.3 und Befehlsliste Abschnitt 7.5)

Beispiel: siehe nächste Seite

Beispiel zu FbExecH4:

Dieses Beispiel zeigt eine absolute Bewegung (Immediate) der Achse X (siehe Abschnitt 7.5.4, Zelle 1.10)




```
.
.
.
.
.

Ld   R  rComWord   ;\
        Xai        ; > benutze den Befehl 'Xai'
                   ;/ (Move X absolute immediate)
CFB  fbExecH4      ;\
     K  16         ; > führe den Befehl aus
     R  100        ; neue Pos.wert in R 100
     NotUsed       ;
     NotUsed       ;
     NotUsed       ;

.
.
.
.
.
.
```

Bemerkung:

Die FBs 'FbUpLdH4': H4-Programm lesen und
'FbDnLdH4': H4-Programm schreiben

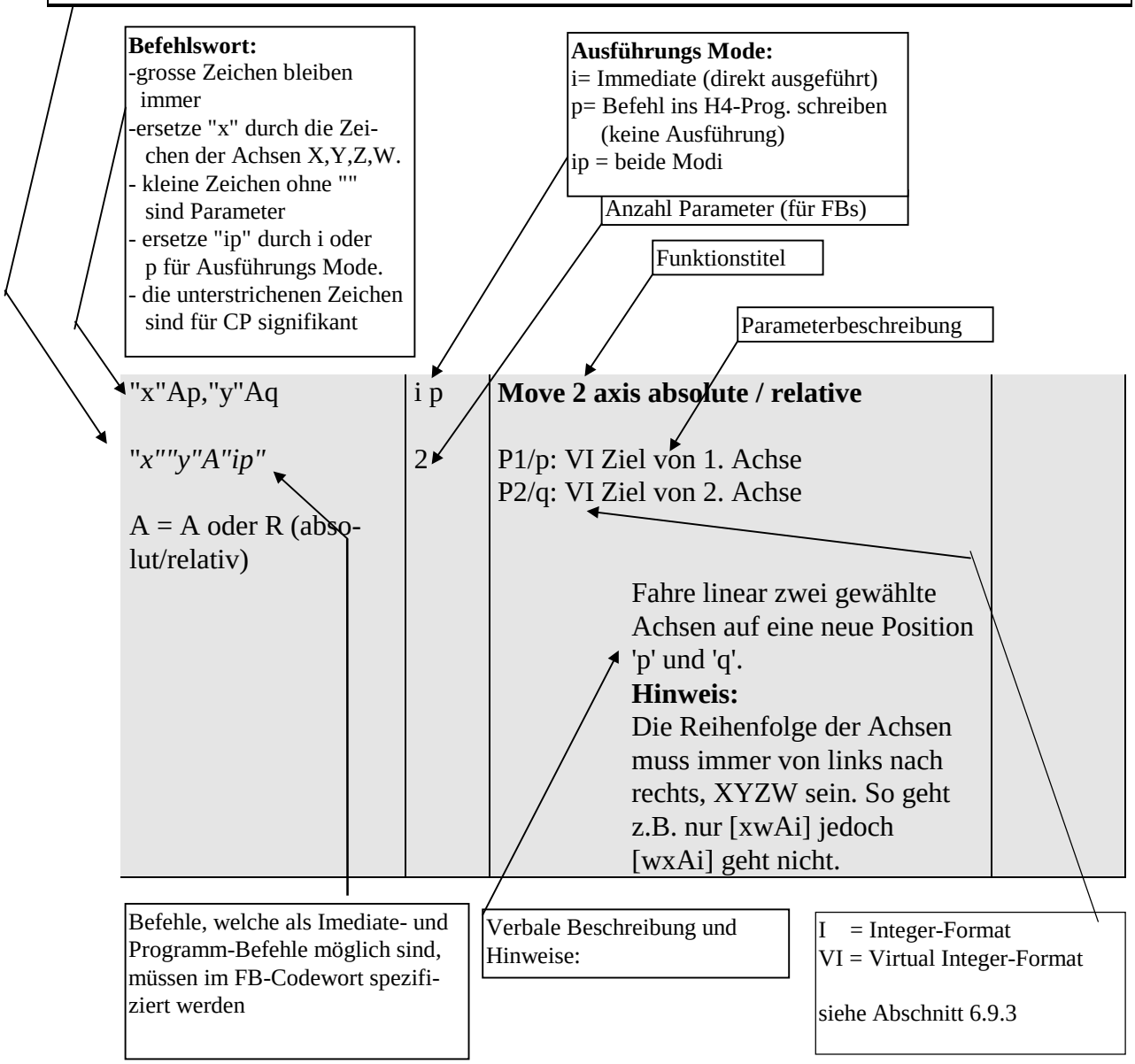
sind im Abschnitt 7.7 "H4-Programme mit FBs schreiben und lesen" behandelt.

Notizen

7.5. Befehlsliste

7.5.1 Syntaxerklärung der Befehlsliste

Die Befehlsliste ist für das Programmieren mit dem CP-Tool oder mit den FB ausgelegt. Die Befehlswörter in gerader Schrift gelten für das CP-Tool, jene in kursiver Schrift für die FB.



7.5.2 Übersicht Befehlsgruppen:

Gruppe:		Beispiele:
7.5.4	Bewegungsbefehle	(XA100 / cir... / Home / Zero / ...)
7.5.5	Achsensteuer-Befehle:	(Enable / Kill / Query Pos. Acc. / Decel.)
7.5.6	Spezial-Befehle:	(Out / Lock / Unlock)
7.5.7	Parameter-Befehle	(read & write Param.)
7.5.8	Programmsteuer-Befehle:	(run / step / ...
7.5.9	Programmstruktur-Befehle	(for / next / ...)
7.5.10	Programm List-Funktionen für nur Terminal (nur für CP)	(list)
7.5.11	Programm-Erstellungsbefehle	(open / close / erase)

7.5.3 Alphabetische Befehls- und Parameterliste mit Angabe der Zellennummer der nachfolgenden Tabellen

Query capture position	2.10-11
Query current execution line	5.7
Query memory lines free	8.6
Query Posotion	2.4
Query Status	2.12
Query user error	2.13
QV	2.6
RAPID	2.16
RESUME	5.6
RETURN	6.5
RUN	5.1
SA	1.8
SC	2.8-9
SD	1.9
Set acceleration	1.8
Set capture position	2.8-9
Set deceleration	1.9
Set output compare	2.14-15
Set parameter	4.1
Set position	2.11
Set Speed	1.6
Set vector speed	1.7
SO; SOI	2.14-15
SPLOCK	3.2
SPUNLOCK	3.3
Status	2.12
SS	1.6
STEP	5.3
STOP	6.6
SV	1.7
Verfahre eine Achse	1.10
VOUT	3.1
WA /WR	1.10
WAIT	6.7
XA /XR	1.10
XAp,YAq,ZAr,WAs / XRp,YRq,ZRr,WRs	1.13
XAYA / XRYR	1.11
XAYAZA /XRYRZR	1.12
YA /YR	1.10
ZA /ZR	1.10
ZERO	1.1

7.5.4 Bewegungsbefehle / Motion commands

	Befehlswort **für CP** ***für FBs***	**i p** Mode	**Kurzbezeichnung** **Kurzerklärung**	
1.1	**ZERO "x"** ***ZERO"x""ip"***	i p 0	**Set axis to zero** Setze die Position der Achse auf Null. Egal wo dieAchse gerade steht, der Positionswert geht verloren und wird auf '0' gesetzt.	
1.2	**HOME "x"** ***HOME"x"***	i 0	**Synchronize axis** Synchronisiert die Achse mit der Mechanik. Die Suchart der Home Pos. ist in den Param. 20-24 bestimmt. Dieses kleine Ablaufprog. wird ohne Belastung der PCD-CPU im H4 durchgeführt. Durch Abfrage des Flags 'Home' aus den H4-Status-Flag kann festgestellt werden, wann der Befehl 'Home' beendet ist. Anschliessend wird die Ref. Position (Par. 23) als Istposition geladen. Home- und Jogbewegungen sind nicht PID gelregelt (siehe Abschnitt 7.6.3)	Vorsteuern: Das Status-Flag (23 für x-Achse) wird gelöscht.
1.3	**J+ "x"** ***JUP"x"***	i 0	**Jog positive** (manuelles Fahren) Jog - Betrieb: In dieser Betriebsart kann die Achse im gesteuerten Mode verfahren werden (nicht PID geregelt). Die Achse beschleunigt mit der Beschl. P'x'43 auf die gesetzte Geschw. und fährt bis ein 'Jog stop'-Befehl gesendet wird oder bis der Pos.-LS (HW/SW) erreicht ist. Für die Verfahrgeschwindigkeit im Jog-Betrieb siehe Befehl 'RAPID' und 'NORMAL'.	

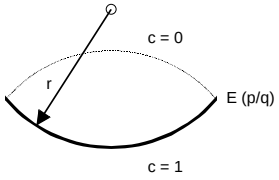
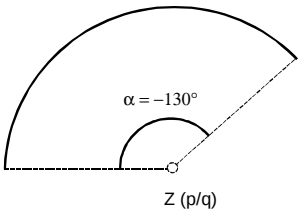
7.5.4

1.4	**J- "x"** ***JDN"x"***	i 0	**Jog negative** Fährt bis LS negativ bzw. 'Jog stop'-Befehl siehe 'J+ - Befehl	
1.5	**JS "x"** ***JS"x"***	i 0	**Jog stop** Der Jog - Betrieb der entsprechenden Achse wird mit der Descelleration P'x'44 gestoppt.	
1.6	**SS "x",s** ***SS"x""ip"***	i p 1	**Set motion speed** P1/s: VI [unit/sec] Nie 0 max = V_{max} (P 'x' 30) Default = 1/10 V_{max} Setzt die Bewegungsgeschwindigkeit der Achse. Gilt nur beim Verfahren einer Achse.	
1.7	**SV s** ***SV"ip"***	i p 1	**Set vector speed** P1/s: VI [unit/sec] Nie 0 Setzt die Vektor-Geschwindigkeit der Bewegung. Die Achsgeschwindigkeiten werden vektoriell berechnet. Gilt nur beim interpolierten Verfahren mehrerer Achsen. 2 Achsen: $V = \sqrt{V_x^2 + V_y^2}$ 3 Achsen: $V = \sqrt{V_x^2 + V_y^2 + V_z^2}$ 4 Achsen: $V = \sqrt{V_x^2 + V_y^2 + V_z^2 + V_w^2}$	
1.8	**SA a** ***SA"ip"***	i p 1	**Set motion acceleration** P1/a: VI [unit/sec^2] Nie 0 max = A_{max} (P 'x' 33) pro Achse Die Beschleunigung für interpolierte Bewegungen wird geladen.	
1.9	**SD d** ***SD"ip"***	i p 1	**Set motion deceleration** P1/d: VI [unit/sec^2] Nie 0 max = A_{max} (P 'x33) Die Verzögerung für interpolierte Bewegungen wird geladen.	

7.5.4

1.10	**"x"Ap** ***"x"A"ip"*** A = A oder R (absolut/relativ)	i p 1	**Move axis absolute / relative** P1/p: VI [unit] Der max. Positionierbereich ist -2^{31} ... $+2^{31}$ Impulse ($-2{,}147*10^9$ ... $+ 2{,}147*10^9$) Die Auflösung ist jedoch durch das Fliesskomma-Format auf 7½ Stellen begrenzt. Details siehe Abschnitt: 6.9.3 Fahre mit der gewählten Achse auf die Position 'p'. Wird ein LS während der Bewegung aktiv, so wird die Achse "gekillt" und das Störungsbit 2 sowie das entsprechende Achsenstatusflag gesetzt. Ein Freifahren aus dem LS ist nur mit den Jogbefehlen möglich.	Vorsteuern: Die 'In Pos'-Flags werden gelöscht
1.11	**"x"Ap,"y"Aq** ***"x""y"A"ip"*** A = A oder R (absolut/relativ) für alle Achsen gleich z.B. XAp, ZAq XRp, WRq	i p 2	**Move 2 axis absolute / relative** P1/p: VI Position/Weg der X-Achse [unit] P2/q: VI Position/Weg der Y-Achse [unit] Fahre linearinterpoliert zwei gewählte Achsen auf eine neue Position 'p' und 'q'. (Gilt nur für absolute Bewegung, für relativ gilt: Weg). **Hinweis:** Die Achsen müssen immer in der Reihenfolge x,y,z,w eingesetzt werden. So geht z.B. nur [xwAi] jedoch [wxAi] nicht.	Vorsteuern: Die 'In Pos'-Flags werden gelöscht
1.12	**"x"Ap,"y"Aq,"z"Ar** ***"x""y""z"A"ip"*** A = A oder R (absolut/relativ) für alle Achsen gleich	i p 3	**Move 3 axis absolute / relative** P1/p: VI Position/Weg der X-Achse [unit] P2/q: VI Position/Weg der Y-Achse [unit] P3/r : VI Position/Weg der Z-Achse [unit] Fahre linearinterpoliert drei gewählte Achsen auf eine neue Pos. 'p', 'q' und 'r'.	Vorsteuern: Die 'In Pos'-Flags werden gelöscht.
1.13	**XAp,YAq,ZAr,WAs** **XYZWA"ip"** A = A oder R (absolut/relativ) für alle Achsen gleich	i p 4	**Move 4 axis absolute / relative** P1/p: VI Position/Weg der X-Achse [unit] P2/q: VI Position/Weg der Y-Achse [unit] P3/r : VI Position/Weg der Z-Achse [unit] P4/s : VI Position/Weg der W-Ach. [unit] Fahre linearinterpoliert die vier Achsen auf eine neue absolute Pos. 'p', 'q', 'r' und 's'.	Vorsteuern: Die 'In Pos'-Flags werden gelöscht.

7.5.4

1.14	**CIRr,c,"x"Ap,"y"Aq** **CIR"x""y"A"ip"** A = A oder R (absolut/relativ)  	i p 4	**Move circle with radius (abs. / rel.)** P1/r: VI Radius [unit] P2/c: I Richtung: 0 = Uhrzeigersinn 1 = Gegenuhrzeigersinn (wenn pos. Koordinatensystem definiert ist) P3/p : VI End-Position/Weg der 1. Achse P4/q : VI End-Position/Weg der 2. Achse [unit] Fahre einen Kreisbogen von der akt. Pos. zur neuen Pos. 'p', 'q', mit einem Radius von 'r' und Drehrichtung 'c'. Hinweis: Ist der Kreisbogen >100° dann wird alle 5° ein Punkt errechnet. Ist der Bogen <100° werden immer 20 Punkte gerechnet. Achtung: Der Radius muss mindestens die Hälfte der Distanz (bis zur Endposition) betragen, ansonsten wird der Befehl nicht ausgeführt.	Vorsteuern: Die 'In Pos'-Flags werden gelöscht. End-Position abs./relativ
1.15	**CIRa,2,"x"Ap,"y"Aq** **CIR"x""y"A"ip"** A = A oder R (absolut/relativ)  	i p 4	**Move circle with angle (abs. / rel.)** (Zentrum-Modus) P1/a: I Winkel [unit] P2/c: 2 Zentrum-Modus P3/p : VI Zentr. des Kreises auf 1. Achse P4/q : VI Zentr. des Kreises auf 2. Achse [unit] Fahre einen Kreisbogen von der akt. Pos. mit dem Winkel 'a' dessen Zentrum auf der Pos. 'p', 'q', liegt. Ist der Winkel negativ, dreht die Drehrichtung des Bogens Hinweis: (siehe CIR-Befehl mit Radius)	Vorsteuern: Das Status-Flag (8 für x-Achse) wird gelöscht.

7.5.5 Achsensteuer-Befehle / Axis control commands

7.5.5	**Befehlswort** **für CP** ***für FBs***	**i p** Mode	**Kurzbezeichnung** **Kurzerklärung**	
2.1	**FO=x** ***FO***	i 1	**Feed Override** P1/x: I in % von 1 bis 120 führt die Bewegungsgeschwindigkeiten des ganzen Moduls in Prozenten aus. Nach PowerUp ist dieser Wert auf 100% gesetzt. Diese Funktion hat keinen Einfluss auf den 'Jog-Betrieb'. Der Wert muss vor einer Bewegung gewählt werden. (nicht 'on the fly')	
2.2	**ENABLE "x"** ***ENABLE"x"***	i p 0	**Enable axis** Durch 'Enable..' der gewählten Achse wird der Lageregler eingeschaltet und die Position aktiv gehalten. Ebenso wird der Ausgang 'Ampl. Ena' (Kl. 0/8) gesetzt. Mit 'Kill..' wird der Regler und der Ausgang wieder ausgeschaltet.	
2.3	**KILL "x"** ***KILL"x"***	i 0	**Disable axis** Mit 'Kill..' wird der Regler ausgeschaltet, der Ausgang (Kl. 0/8) ausgeschaltet, der Analogausgang auf 0V gesetzt und der Programmpointer auf 0 gesetzt. Siehe auch Befehl 'Enable..' In Interpolation werden alle beteiligten Achsen gekillt.	
2.4	**QP "x"** ***QP"x"***	i 1	**Query actual position** P1: VI Istposition [unit] Lese die aktuelle Position der gewählten Achse. Istwert der Achse. (z.B. für Anzeige auf MMI)	

7.5.5				
2.5	**QPI "x"** ***QP"x"***	i 1	**Query actual position in Impulsen** P1: I +/- 2E31 (unabhängig von P96) [Imp x 4] Lese die aktuelle Position des Encoders der gewählten Achse. Der Istwert ist in Encoder Impulse aufgelöst. (Mode x4) + Erklärung bei 2.10.	
2.6	**QV "x"** ***QV"x"***	i 1	**Query actual velocity** P1: VI Geschwindigkeit der Achse [unit/s] Lese aktuelle Geschwindigkeit der Achse.	
2.7	**QE "x"** ***QE"x"***	i 1	**Query actual position error** P1: VI Positionierfehler der Achse [unit] Lese aktuellen Schleppfehler/rsp. Positionierfehler im Stand.	
2.8	**SC "x"** **SC"x""ip"**	i p 0	**Set capture position** Aktivieren der Funktion: 'Position speichern', wenn der Input 'PCI' (Kl.A/B) des Moduls angesprochen hat. Der Wert kann mit dem Befehl QC abgefragt werden. Das Setzen der Capture-Funktion löscht das Statusflag ‘PCI erfasst’.	Vorsteuern: Das Status-Flag (8 für x-Achse) wird gelöscht
2.9	**QC "x"** **QC"x"**	i 1	**Query capture position** P1: VI Gespeicherte Position [unit] Lese den durch die Funktion 'Capture' erfassten Wert. Durch Abfrage des Statusflag (Flag 15 bei X-Achse) kann festgestellt werden, ob der Wert schon erfasst wurde. (siehe auch SC-Befehl)	

7.5.5				
2.10	**QCI "x"** **QCI"x"**	i 1	**Query capture position in Impulsen** P1: VI Gespeicherte Position in Impulsen $\pm 2^{31}$ (unabhängig von P96) Lese den durch die Funktion 'Capture' erfasste Wert direkt vom Encoder. Hinweis: Ein Encoder mit 1000 Pulsen erzeugt bei einer Umdrehung 4000 Impulse im H4 Modul (im 'x 4'-Modus) (siehe auch SC- und QC-Befehl)	
2.11	**SP "x", p** **SP"x" "ip"**	ip 1	**Set position** P1/p: VI [unit] Position, welche als Istwert geladen wird. Max. Positionsbereich -2^{31} .. $+2^{31}$ Impulse. Die aktuelle Achsposition geht verloren und wird mit der Position p überschrieben.	

7.5.5

2.12	**QS "x"** **special** ***CFB FbStatH4***	i 0	**Query Axis status** Der Achsenstatus wird im Terminal-Mode online angezeigt, weshalb der QS'x'-Befehl nicht speziell gebraucht wird. (Das Format ist Hex Code Dezimal) 16 Status Flag /Achse Die Flag-Basisadresse wird im FBInitH4 definiert. Lese die Status-Flag der entsprechenden Achse. Flag : 0-6 nur für FB intern +BAF **7:** Fatal Error H4 (kein Link zu H4-Modul). Nur Lesezugriff. Für x-Achse gilt: pro Achse 8: P Achse in Pos. 9: E Immediate-Befehl in Ausführung 10: h Achse in Hardware LS 11: s Achse in Software LS 12: F Schleppfehler Störung 13: W Schleppfehler Warnung 14: 0 Soll-Geschw. = 0 15: C Capture-Pos. eingetreten 16: A Drive OK (Input AOK) 17: - Neg. LS (Input LSS) 18: + Pos. LS (Input LSE) 19: R Ref. switch (Input RPS) 20: I Input capture (PCI) 21: c Trigger-Position erreicht 22: V Positionsüberlauf 23: H Home ist erfolgreich beendet *für Y-Achse 'Flag+16'* *für Z-Achse 'Flag+32'* *für W-Achse 'Flag+48'* FB: Die Abfrage für die Status-Flag werden durch einen speziellen FB ausgeführt. Dieser muss zyklisch im Programm aufgerufen werden.	

7.5.5

2.13	**QU** ***QU***	i 2	**Query user error code** Der User-Error wird im Terminal-Menü online angezeigt, mit QU wird diese Fehlermeldung sowie auch der User-Error-Eingang E 10 gelöscht. (auf PCD4-Bus) P1: I 16 Errorbits P2: I Errorcode Lese das Wort 'User error code' :	L = gespeichert
			Störungsbit:	Fehler löschen durch:
			0: EEPROM nicht bereit	L
			1: EEPROM Checksum Error	L
			2: LS angesprochen	L
			3: (Reserviert Intern)	
			4: Max. Schleppfehler mit Stop	
			5: Errechneter Positions-Überlauf	L ENA
			6. Max. Schleppfehler	
			7: Fehler in Home-Routine	L ENA
			8: LS angesprochen	online
			9: Ausführungsbuf. overflow	
			10: Falscher Befehl in Prog. Prog. Nr. siehe Bit 12-15	online
			11: Checksum Error in Prog.	L Run 'p'
			12: 20 ¦ LSB	
			13: 21 ¦ Programm hat eine	L
			14: 22 ¦ Störung	
			15: 23 ¦ MSB für Progr. Nr.	
			Störungs- Code:	
			0: keine Störung	
			1 Start Prog. mit Err. Bit 11	
			2 Start Prog. mit Err Bit 10	
			3 mehr als 4 Prog. in RUN	
			4 Start Prog. das nicht existiert	
			5 Start Prog. auf einer nicht existierenden Zeile	
			6: Ein Prog. in Run kann nicht verändert werden	
			7: Bewege eine Achse die disabled ist.	
			8: 'Home' verlangt von einer disabled Achse.	
			50: Data overflow in PCD interface	
			100: Befehl von PCD unmöglich	
			200: kein H440 erkannt	
			255: GENERAL ERROR	

7.5.5

2.14	**SO "x",p** **SO"x""ip"**	i p 1	**Set output compare** P1/p: VI Position [unit] Das H4 Modul besitzt einen Output 'Trigger Out', welcher beim Überschreiten, der durch diesen Befehl gesetzten Position, aktiviert wird. Beim Ausführen dieses Befehls wird der Output gelöscht.	Vorsteuern: Das Status-Flag (21 für x-Achse) wird gelöscht.
2.15	**SOI "x",p** **SOI"x""ip"**	i p 1	**Set output compare in impulses** P1/p: I Position $\pm 2^{31}$ (unabhängig von P96) Beschreibung siehe Befehl SO. Die Position wird jedoch in Impulsen angegeben. Hinweis: Beachte, die Encoder Auflösung wird mit 4 multipliziert. (siehe Kap. 6.9 Encoder)	Vorsteuern: Das Status-Flag (21 für x-Achse) wird gelöscht.
2.16	**RAPID** ***Rapid***	i 0	**Use rapid speed for jog** Setzt die Geschwindigkeit für den Jog-Betrieb auf P'x'32. Falls die Achse bereits mit 'Normal speed' fährt, wird die Geschwindigkeit 'on the fly' mit der Beschleunigung P'x'43 auf Rapid geändert. 'Rapid' und 'Normal' wirken auf alle Achsen, jedoch können diese pro Achse unterschiedliche Geschwindigkeiten (P'x'31 und P'x'32) haben.	
2.17	**NORMAL** ***NORMAL***	i 0	**Use normal speed for jog** . 'Jog'-Geschwindigkeit auf normal setzen (Parameter 31). Zusatzinformationen siehe Befehl 'Rapid'. Falls die Achse bereits mit 'Rapid speed' fährt, wird diese 'on the fly' mit P'x'44 auf 'Normal' abgebremst.	

7.5.5

2.18	**DRIFT "x"** ***DRIFT"x"***	i 0	**Drift compensation** (Offset-Abgleich) Führt eine Drift Kompensation für die angewählte Achse aus. Addiert die Offset-Spannung, welche abhängig vom Positionsfehler im Stand und dem Proportional-Faktor P'x'50 ist, zum Analog-Output. Deshalb muss der Integralanteil ausgeschaltet werden, damit ein Positionsfehler auftritt und eine Kompensation vorgenommen werden kann.	

7.5.6 Spezial-Befehle / Special commands

	Befehlswort **für CP** ***für FBs***	**i p** Mode	**Kurzbezeichnung** **Kurzerklärung**	
3.1	**OUT "x",v** ***VOUT"x"***	i 1	**Output to DAC** P1: VI v: ±0.00 ... 10V Der Analogausgang, welcher im Regelkreis zur Ansteuerung des Verstärkers benötigt wird, kann mit dem Befehl 'Out..' direkt angesteuert werden. Dazu muss der Regler der entsprechenden Achse ausgeschaltet sein ('Kill'). Hinweis: Dieser Befehl wird meist nur für Tests und Inbetriebsetzungen verwendet.	
3.2	- ***SPLOCK***	(i) 0	**Disable serial port (nur für FBs)** Sperrt die Schnittstelle zu CP.	
3.3	- ***SPUNLOCK***	(i) 0	**Enable serial port (nur für FBs)** Gibt die Schnittstelle zum CP wieder frei.	

7.5.7 Parameter-Befehle / Parameter commands

	Befehlswort **für CP** ***für FBs***	**i p** Mode	**Kurzbezeichnung** **Kurzerklärung**	
4.1	**Pxn=y**	ip 3	**Set parameter** x: Achse (für FBs: x=1, ...W=4) n: Parameternummer y: I/VI Parameterwert (siehe Parameterliste) Lade Parameter 'n' der gewählten Achse mit dem Wert 'yyy'	

	P"n" (0-89,92)
	Für H4-Prog. verwenden.
		p
		
		3
			Set axis parameter in 'Program' mode
			P1: Achse
			(für FBs: x = 1,....W = 4)
			P2: Parameternummer 'n' = 0 - 89,92
			P3: I/VI Parameterwert, (siehe Parameterliste)
			Lade Parameter ('n' und P2) der gewählten Achse (P1) mit dem Wert aus P3
			Hinweis:
			Für nn muss das Befehlswort (z.B. P01) mit dem Parameter 2 übereinstimmen, das heisst gleich sein.
				Beispiel:
				
				`Ld rComH4`
				`P01`
				`CFB fbExecH4`
				`K 32`
				`R rr (1)`
				`R rr (01)`
				`R rr(0)`

	P"n" (90-99,~~92~~)
	Für H4-Prog. verwenden.
		p
		
		3
			Set general paramters in 'Program' mode
			
			P1: Parameternummer 'n'=90-99
			P2: I Parameterwert (siehe Parameterliste)
			Lade Parameter ('n' und P2) der gewählten Achse (P1) mit dem Wert aus P2
			Hinweis:
			Für nn muss das Befehlswort (z.B. P01) mit dem Parameter 1 übereinstimmen, das heisst gleich sein.
				Beispiel:
				
				`Ld rComH4`
				`P97`
				`CFB fbExecH4`
				`K 32`
				`R rr(97)`
				`R rr(45`

7.5.7				

4.2	***P"x""n"W***	i	**Set parameter im 'Immediate' Mode**
			
			P1: Parameterwert (siehe Parameterliste)
				Beispiel:
				
				`Ld rComH4 Py01W`
				`CFB fbExecH4 K 32 R rr(2)`

	(n= 0-98) für allgemeine Parameter (90-98) ohne Achse ist 'x' wegzulassen.	1	Lade Parameter 'n' der gewählten Achse mit dem Wert aus P1	`Ld rComH4 P96W` `CFB fbExecH4 K32 R rr (3)`
4.2	**Pxnn?** ***P"x""nn"R*** *(siehe Parameter-Liste).*	i 1	**Get parameter** P1/nn: Parameterwert (Zielregister) Lese Parameter 'nn' der gewählten Achse. Siehe Parameterliste für P1 Inhalt.	`Ld rComH4 Px50R` `CFB fbExecH4 K 32 R rr`
4.3	**EREAD** ***EREAD***	i 0	**Read from EEPROM** Lese die Parameter vom EEPROM in das H4-RAM. Dies wird auch beim PowerUp durchgeführt.	
4.4	**EWRITE** ***EWRITE***	i 0	**Store in EEPROM** Speichere die Parameter vom H4-RAM in das EEPROM. Dies wird automatisch bei einem 'Paramter Download' vom CP aus durchgeführt.	

7.5.8 Programmsteuer-Befehle / Program control commands

	Befehlswort **für CP** ***für FBs***	**i p** Mode	**Kurzbezeichnung** **Kurzerklärung**	
5.1	**RUN p** ***RUN"ip"***	i p 1	**Run program** P1: I Programmnummer 1-9 Startet das ausgewählte Programm auf Zeile 1. Wenn der 'Run'-Befehl aus einem Programm gestartet wird, muss berücksichtigt werden, dass max. 4 Programme gleichzeitig laufen können (siehe auch P98).	
5.2	**Gpn** ***G"ip"***	i p 2	**Run program on line N** P1/p: I Programmnummer (1-9) P2/n: I Programmzeile (1-1000) Startet das ausgewählte Programm auf der Zeile n.	
5.3	**STEP p** ***STEP***	i 1	**Step program** P1/p: I Programmnummer (1-9) Führe einen einzelnen Programm-Befehl aus.	
5.4	**BREAK p** ***BREAK"ip"***	i p 1	**Break program** (Abbrechen mit 'Run') P1/p:I Programmnummer (1-9) Stoppe das Programm nach dem laufenden Befehl. (Wenn mehrere Befehle 'blended' ausgeführt werden, gelten diese als ein einziger Befehl). Mit 'Run' läuft das Programm weiter.	
5.5	**HALT** ***HALTALL***	i 0	**Stop all program and motion** Stoppe jegliche Bewegung sofort (alle Achsen) mit der max. Verzögerung. Die Regelung bleibt aktiv und die Position erhalten. Dieser Befehl kann nur mit dem Kommando 'Resume' (oder 'Kill') aufgehoben werden.	

5.6	**RESUME** ***RESUME***	i 0	**Resume comand Halt** Löst den Haltezustand des Moduls, der durch den Befehl 'HALT' eingeleitet wurde. Die durch den Halt-Befehl unterbochene Bewegung wird zu Ende geführt.	
5.7	**QL"p"** ***QL"p"*** ersetze „p" durch die Prog. Nummer (1-9)	i 1	**Query current execution line** P1: I Programmlinie (1-1000) Gibt die laufende Programmzeile aus. Ist das Programm beendet (oder nicht gestartet worden) wird 0 zurückgegeben.	

7.5.9 Programmstruktur-Befehle / Program structure commands

	Befehlswort **für CP** ***für FBs***	**i p** Mode	**Kurzbezeichnung** **Kurzerklärung**	
6.1	**FOR n** ***FOR***	p 1	**Starts a repeat block** P1/n: I Schlaufenanzahl (0 - 32767) Markiert den Beginn eines Wiederholungs-Blocks. Dieser wird n-mal wiederholt. Dieser Befehl kann bis zu 8 Ebenen tief verschachtelt werden.	
6.2	**NEXT** ***NEXT***	p 0	**Ends a repeat block** Markiert das Ende eines Wiederholungs-Blocks.	
6.3	**GOTO n** ***GOTO***	p 1	**Jump to program line** P1/n: I Programmlinie (1 - 1000) Sprung zu einer bestimmten Zeile im Programm.	
6.4	**GOSUB n** ***GOSUB***	p 1	**Jump to subroutine** P1/n: I Programmlinie (1 - 1000) Ruft ein Unterprogramm auf Zeile n auf. Dieser Befehl kann bis zu 8 Ebenen tief verschachtelt werden.	
6.5	**RETURN** ***RETURN***	p 0	**End of the subroutine** Markiert das Ende des Unterprogramms	
6.6	**STOP** ***STOP***	p 0	**Stop program** Hält das eigene Programm auf der Zeile, wo der Stop-Befehl steht an (wait endless) bis ein erneuter Befehl RUN oder STEP erfolgt. Wirkt wie ein 'Break Point'.	

6.7	**WAIT n** ***WAIT***	p 1	**Wait** P1/n: I Wartezeit 0 - 65535 [msec] Führt beim Abarbeiten des Programms eine Pause der Grösse n in msec. aus. Hinweis: Durch Einfügen eines WAIT 0 zwischen zwei Bewegungsbefehlen, wird die 'Blended move'-Funktion unterdrückt.	
6.8	**END** ***END***	p 0	**Identifies the end of program** Jedes Programm im H4 muss mit diesem Befehl enden.	

7.5.10 Programm List-Funktionen für Terminal (nur für CP) Program list commands

	Befehlswort für CP *für FBs*	**i p** Mode	**Kurzbezeichnung** **Kurzerklärung**	
7.1	**Lpn=.....**	i	**Function of list: Set programline** Überschreibe die Zeile n des Programms p mit den nachfolgenden Befehlszeichen. Das Programm wird erst beim 'Close'-Befehl abgespeichert. Für FBs siehe 'OPEN'	
7.2	**Lpn?**	i	**Function of list: Get programline** Anzeige der Zeile n des Programms p in Befehlszeichen.	
7.3	**Lpn,m?**	i	**Function of list: Get area of programline** Anzeige der Zeile n bis m (max. 20 Zeilen) des Programms p in Befehlszeichen.	

7.5.11 Programm-Erstellungsbefehle / Program build commands

	Befehlswort **für CP** ***für FBs***	**i p** Mode	**Kurzbezeichnung** **Kurzerklärung**	

8.1	-
	
	OPEN"p"
	
	ersetze "p" durch die Prog. Nummer (1-9).
		i
		
		1
			Open program for edit
			
			P1: I Programmlinie
			(1 - 1000)
			
			Öffnet das gewählte Programm, um anschliessend Befehle ins Programm zu schreiben.
			Hinweis:
			Dies geschieht im CP automatisch mit dem Lpn = ... Befehl und benötigt daher für das CP keine spez. Instruktion.
				Beispiel:
				
				`Ld  rComH4`
				`    OPEN5`
				`CFB fbExecH4`
				`    K 32`
				`    R rr(1)`

8.2	**CLOSE** ***CLOSE***	i 0	**Close and save program under edit** Schliesse das geöffnete Programm und speichere dieses. Wird auch benötigt, wenn mit Lpm = ... ein Porgramm modifiziert wurde. Ist das editierte Programm in Ausführung, wird der Close-Befehl nicht ausgeführt.	
8.3	**Ep** ***EP***	i 1	**Erase program** P1/p: I Programmnummer (1-9) Lösche das gewählte Programm. Ist dieses Programm in Ausführung, wird das Löschen nicht durchgeführt. Siehe auch Bits in 'User Error'.	

8.4	**Epn** -	i 2	**Erase program line** p: Programmnummer (1-9) n: Programmlinie (1-1000) Lösche eine Zeile im gewählten Programm. Die gelöscht Zeile bleibt leer, die nachfolgenden rücken nicht nach. Dieser Befehl kann mit den FB nicht ausgeführt werden.	
8.5	**Epn,m** -	i 3	**Erase area of program line** p: Programmnummer (1 - 9) n: Programmzeile Beginn (1 - 1000) m: Programmzeile Ende (n - 1000) Lösche einen Bereich von Zeilen n bis m im gewählten Programm. Dieser Befehl kann mit den FB nicht ausgeführt werden.	
8.6	**QM** ***QM***	i 1	**Query memory lines free** P1: I Anzahl freie Programmlinien (0 - 1000) Abfrage, wieviel Programmzeilen im Editierbuffer noch frei sind. P2: I Freier Speicher im H4-Modul in Worten (max. ca. 11K Worte, was ca. 3000 .. 4000 Programmzeilen entspricht) je nach verwendeten Befehlen	

7.6. Parameterliste

7.6.1 Modul Parameter (generell)

In der folgenden Liste verwendete Abkürzungen:

E = Basis-Einheit
s = Sekunde (siehe auch die Syntaxerläuterungen
V = Volt im Abschnitt 7.5.1)
mV = Millivolt

Para. Nr	Bezeichnung	Einheit	changed on the fly	default	For-mat	Werte :	ip-Mode
94	Modul Type	-		2 4	I	2: für H420 4: für H440	i
95	Programmnummer, welche mit den Eingägnen 'Start' & 'Stop' ausgelöst werden.	-		1	I	1-9: für Pro-gramm-Nummer	i
96	Anzahl Nachkommas; nur für VI-Werte via FBs	-		3	I	0-6: Anzahl Nachkommas	i
90	Flanke des 'Start' Signals, auf welche das Programm ge-startet wird	-		Positv	I	0: Positiv 1: Negativ	i
91	Flanke des 'Stop' Signals, auf welche das Programm ge-stoppt wird	-		Positv	I	0: Positiv 1: Negativ	i
92	Encoder Spannung mit P'x'92 werden X und Y eingestellt, mit p'z'92 die beiden Achsen Z und W.	-		5V	I	0: 5V 1: 24V	i
97	Winkel, ab welchem Blended move nicht mehr ausgeführt wird.	-Grad		0°	I	0-181°	i
98	Kontrolle ob mehrere Pro-gramme auf eine Achse zu-greifen	-		1		0: keine Kontrolle 1: Kontrolle	i

Wenn P98 = 1 (Check aktiv), wird verhindert, dass mehrere Programme, welche auf die gleiche Achse wirken, gleichzeitig laufen. Der Run-Befehl für das 2. Programm wird nicht akzeptiert und die Fehlermeldung 'Axis locked' erscheint.
Wenn mehrere Programme parallel laufen sollen und Bewegungsbefehle für eine gemeinsame Achse enthalten, muss der Check ausgeschaltet werden (P98 = 0). Der Anwender ist dann für die Synchronisation der Programme verantwortlich.

ip-Mode: i nur 'Immediate' (nicht in Programm verwendbar)
ip 'Immediate' + 'Programm'-Mode

7.6.2 Maschinen Parameter

Para. Nr pro Achse	Bezeichnung	Einheit	change on the fly	default	For-mat	Werte :	ip-Mo-de
01	Achs-Einheit (E)	E		mm	I	0: mm 1: inch 2: grad 3: Impuls	i
02	Encoder Impulse/Umdrehung	-		1000		0..65535	i
03	Weg/Encoder-Umdrehung (in Einheiten gemäss P01)	E		5	VI	7-stellig 0- 100’000	i
04	Zählrichtung Encoder	-		positiv	I	0: positiv 1: negativ	i
08	Analogausgang Polarität	-		positiv	I	0: positiv 1: negativ	i
30	Maximale Geschwindigkeit (bei V_{out} = 10V)	E/s		200	VI	7-stellig 0-150kHz*P03/P02	ip
33	Maximale Beschleunigung (positiv & negativ)	E/s2		1000	VI	7-stellig 0 - 1'000'000	ip
40	Positive Wegbegrenzung Softwarelimit *)	E	ja	0	VI	7-stellig $\pm 2^{31}$ Schritte 0 = keine Grenze	ip
41	Negative Wegbegrenzung Softwarelimit *)	E	ja	0	VI	7-stellig $\pm 2^{31}$ Schritte 0 = keine Grenze	ip
11	Schleppfehler- Grenze Störungssignal	E	ja	2	VI	7-stellig 0 - 8192*P03/P02	ip
12	Schleppfehler- Grenze Warnungssignal	E	ja	0,5	VI	7-stellig 0 - 8192*P03/P02	ip
13	Verhalten der Achse bei Schleppfehler Störungssignal	-	ja	Stop	I	0: kein Stop 1: Stop	ip

*) Wird kein Softwarelimit benutzt, müssen **beide** Grenzen auf Null gesetzt werden, damit die Funktion ausgeschaltet ist.

Achtung: Die Softwarelimits sind relativ, d.h. wenn der Positionswert verändert wird (Zero'x' oder SetPos'x') verschieben sich auch die Softwarelimits.

Die 'Home'-Funktion (Ref.-Position anfahren) behandelt die Software-LS gleich wie die Hardware-LS. Falls sich der Referenzschalter ausserhalb der Software-LS befindet, müssen diese für die 'Home'-Funktion ausgeschaltet werden.

7.6.3 Jog und Referenzfahren

Para. Nr pro Achse	Bezeichnung	Einheit	changed on the fly	default	For-mat	Werte :	ip-Mode
31	Jog Geschwindigkeit (normal)	E/s	ja	20	VI	7-stellig 0 - P30 *)	ip
32	Jog Geschwindigkeit (rapid)	E/s	ja	40	VI	7-stellig 0 - P30 *)	ip
22	Referenzschalter Such- Geschwindigkeit	E/s		20	VI	7-stellig 0 - P30 *)	ip
24	Encoder C-Signal Such- Geschwindigkeit	E/s		10	VI	7-stellig 0 - P30 *)	ip
20	Referenzschalter Such-Richtung	-		positiv	I	0: positiv 1: negativ	ip
21	Referenzschalter Freifahrrichtung	-		positiv	I	0: positiv 1: negativ	ip
23	Achspositionswert nach Referenzfahren (Preset) (Referenzposition)	E		0	VI	7-stellig $\pm 2^{31}$ Schritte	ip

*) Die Jog- und Referenzfahrtgeschwindigkeiten werden nicht PID geregelt, sondern nur gesteuert.

Die Steuerspannung wird wie folgt berechnet:

$$U_{out} = 10V * \frac{\text{Jog Geschwindigkeit}}{\text{max. Geschwindigkeit}}$$

Deshalb ist es wichtig, vorgängig die max. Geschwindigkeit P 'x' 30 zu definieren.

Die max. Steuerspannung wird erreicht, wenn die Beschleunigungsrampe beendet ist (abhängig von P'x'43).

7.6.4 Regel Parameter

Para. Nr pro Achse	Bezeichnung	Einheit	ch. on the fly	default	For-mat	Werte :	ip-Mode
50	Kp Regler Proportionalfaktor	V/E	ja	1	VI	0- 40*P02/P03	ip
51	Kd Regler Differenzialfaktor	DA-Step/Zyklus/ Impuse	ja	0	VI		ip
56	Sampling time des Kd	Servo-zyklen	ja	100	I	1-1000	ip
52	Ki Regler Integralfaktor	DAStep/ Impul-se/Zyklus	ja	0	VI		ip
53	Integration Limit des Ki (Anti-windup-protec.)	V	ja	2	VI	0-10	ip
16	Intergral Mode Der Regler benutzt den Integralfaktor nur gemäss Einstellung	-	ja	immer	I	0: immer 1: nur im Stillstand	ip
54	Geschwindigkeit Auf-schaltung (feedforward)	mV/E/s	ja	0	VI	0 - 15000/P30	ip
55	Beschleunigungs Auf-schaltung (feedforward)	mV/E/s2	ja	0	VI	0 - 10000/P33	ip
10	Totzone	E	ja	0	VI	7-stellig 0 - 2^{31} Schritte	ip
14	Umkehrspiel	E	?	0	0	0 - 8192 0 - 8192*P03/P02	ip
63	Umkehrspiel Korrektur-geschwindigkeit	%	?	10	I	10-100	ip
15	In-Position-Zone (für 'in-position'-Flag)	E	ja	0,2	VI		ip

7.6.5 Beschleunigungs Parameter

Para. Nr pro Achse	Bezeichnung	Einheit	changed on the fly	default	For-mat	Werte :	ip-Mode
42	Beschleunigungs Mode (bei Interpolationen wird für alle Achsen die weichste Form verwendet)	-		Trapez	I	0: Trapez 1: S-Kurve	ip
43	Beschleunigung (bei Interpolationen wird die tiefste aller Achsen verwen-det)	E/s2	ja	100	VI	0- P33	ip
44	Verzögerung/Abbremsung (bei Interpolationen wird die tiefste aller Achsen verwen-det)	E/s2	ja	100	VI	0- P33	ip
45	Dauer der S-Kurven- Be-schleunigung	s		0	VI	0-99.99	ip

7.6.6 Achsmode Parameter

Para. Nr pro Achse	Bezeichnung	Einheit	changed on the fly	default	For-mat	Werte :	ip-Mode
05	Kreisachse (Überlaufposition)	E		0	VI	0: Linear > 0 - 9999.999	ip
06	Elektronik-Getriebe (Koppelt die ausgewählte Achse an die Master-Achse) Hinweis: Die Kopplung gilt nur von der Slave-Achse zur Master-Achse, jedoch nicht umgekehrt. Soll beides erreicht werden, muss P06 auch bei der Slave-Achse eingestellt werden.	-		0	I	0: keine Kupplung 1: koppelt X an 2: koppelt Y an 3: koppelt Z an 4: koppelt W an	ip
07	Übersetzung des Elektronikgetriebes (Slave-/Master-Achse)	-		0	VI	0 - 9999.9999	ip

7.6.7 Spezielle Parameter

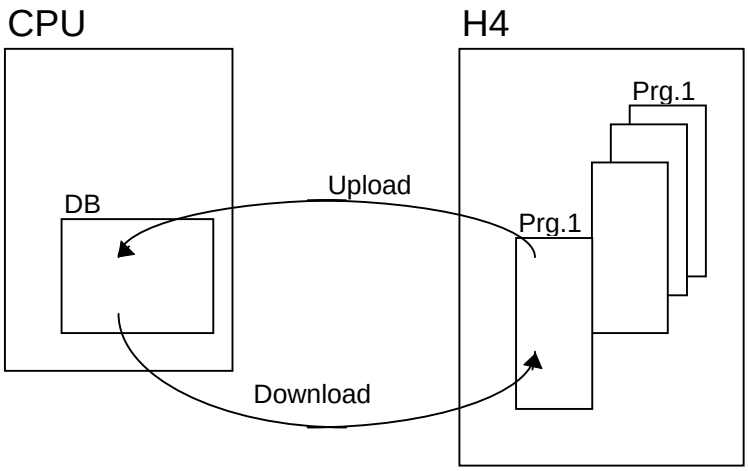
Para. Nr pro Achse	Bezeichnung	Einheit	changed on the fly	default	For-mat	Werte :	ip-Mode
62	Polarität des 'Trigger-Output'-Signals	-	ja	0	I	0: positiv 1: negativ	ip

7.7. H4 Programme mit FBs schreiben und lesen

Beschreibung

Das H4-Modul kann bis zu 9 Programme verwalten. Diese Programme sind im H4 durch einen Superkondensator gepuffert. Liegt das Modul länger als zwei Wochen nicht an Spannung, kann das Anwenderprogramm verändert sein. Durch die Funktionen 'Upload' und 'Download' können die Programme aus dem H4 in der PCD gespeichert und wieder geschrieben werden.

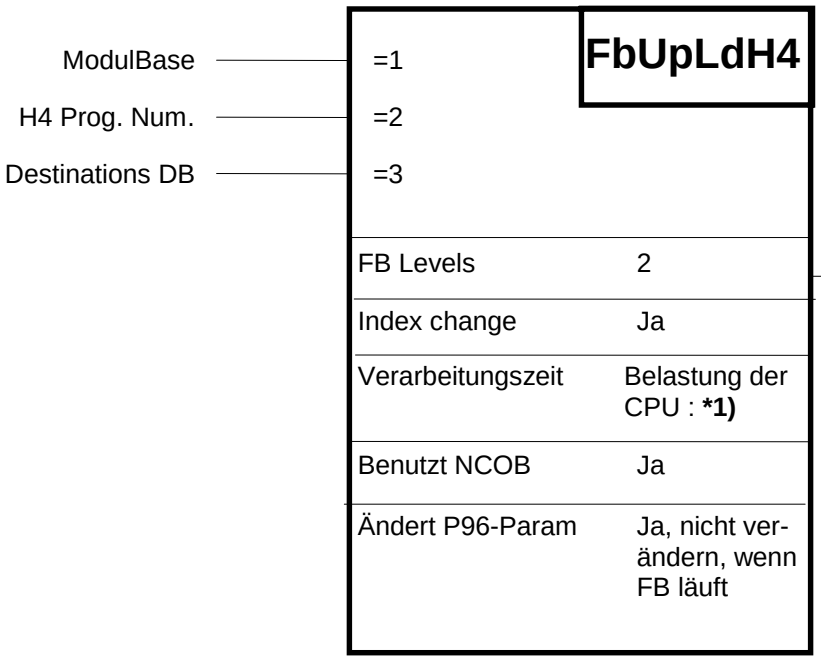
Die graphische Darstellung der FBs ist in Abschnitt 7.4.6 erklärt.



FbUpLdH4 Funktion: - H4-Programm lesen FbUpLdH4

```
Ld   R  rHelp      ;z.B.3 Nachkommastellen
     3             ;
Ld   R  rComH4
     P96W
CFB  fbExecH4
     ModulBase
     rHelp
     NotUsed
     NotUsed
     NotUsed
```

Die Anzahl Nachkommastellen mit welcher das Programm im DB gespeichert werden soll, muss vorgängig gesetzt werden. Der Wert aus P96 wird im DB ebenfalls gespeichert.



Bemerkung *1):
ca. 30ms/H4-Prog.Linie / CPU-Prog.Zyklus im Exended Memory

Hinweis:
abhängig von der CPU Leistung und Linienlänge.

Funktionsbeschreibung:

Diese Funktion liest ein Programm des H4-Moduls und speichert es in der PCD-CPU in einem DB. Detailfunktionen siehe nächste Seite.

Beschreibung Inputs und Outputs:

Symbol	Beschreibung	Parameter	Media		
			Typ	Format	Adr-Bereich
ModulBasic	Modul Basisadresse	Ja	K	Integer	0 - 496
H4-Prog.Num.	Programmnummer im H4	Ja	K	Integer	1 -9
Destination DB	Ziel / Speicher DB	Ja	DB	Integer	(0 -) 4000-7999 ***2)**

Bemerkung *2):
Vorzugsweise nur DB >4000 verwenden, da der Zugriff auf den DB schneller und die DB auch grösser sein können. DB >4000 befinden sich im Extended Memory, benötigen also ein PCD7.R3.. Memory-Modul.

Funktionsbeschreibung:

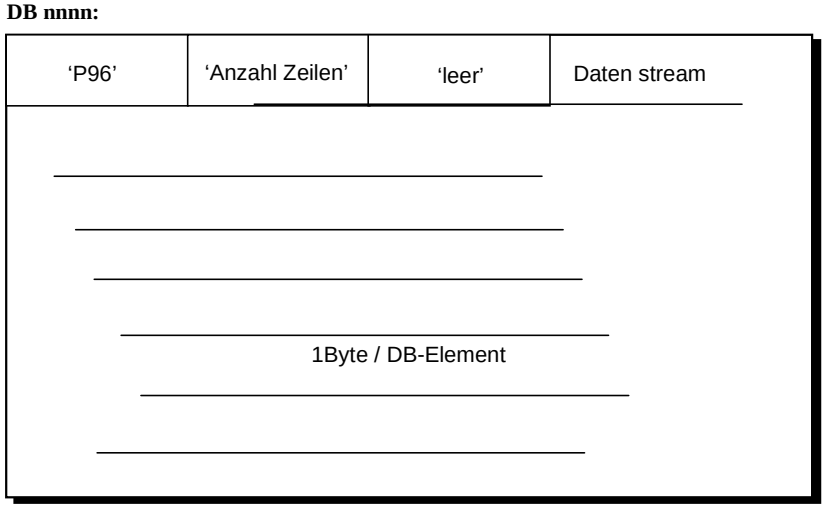
Das gelesene H4-Programm wird in einem einzigen DB abgelegt. Dieser DB muss durch den Anwender definiert werden (Grösse siehe Berechnung). Wird dieser zu klein definiert, so wird der Upload abgebrochen und das Error-Flag gesetzt. Dieser DB kann somit nicht ins H4 geladen werden.

Während ein Upload- oder Download-FB arbeitet, können alle Immediate-Befehle weiterhin aufgerufen werden. So können zum Beispiel die Statusflags oder die aktuelle Position einer Achse in einem andern COB weiterhin gelesen werden.

Alle Befehle, welche auf den Programmspeicher des H4-Moduls zugreifen, dürfen nicht aufgerufen werden.

Der FbUpldH4 wird erst beendet, wenn das ganze H4-Programm in den DB geladen ist. Der FB verwendet den Next-COB-Befehl, womit in einem weiteren COB auch andere Aufgaben behandelt werden.

DB-Aufbau:



Berechnung der DB-Grösse: (Annäherung)

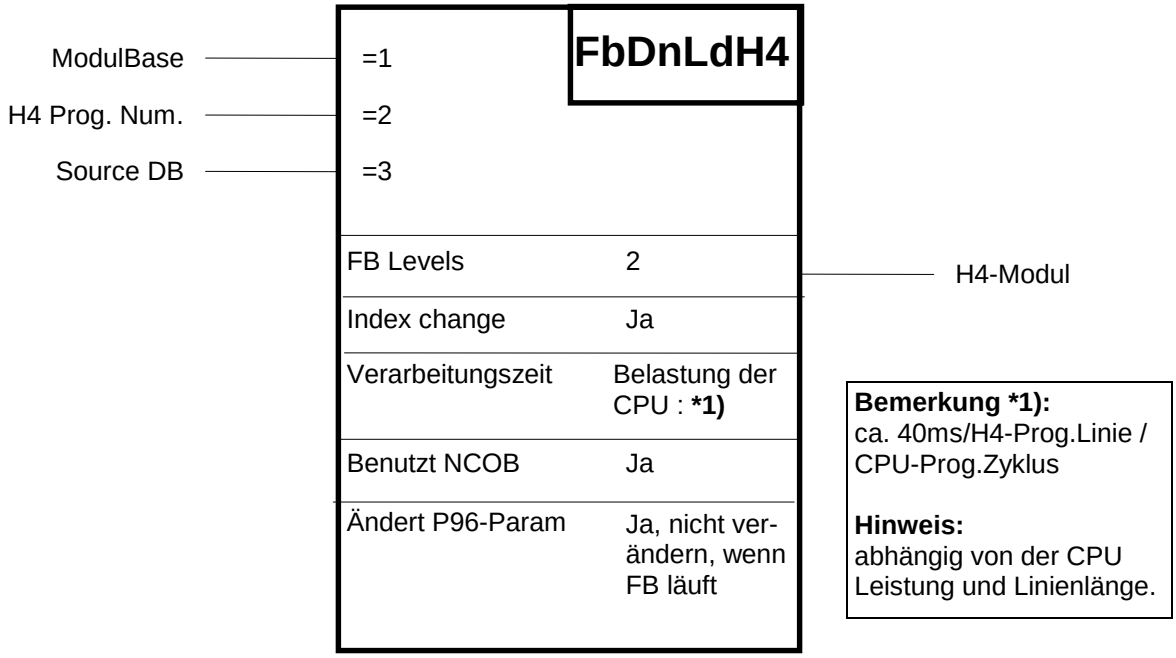
DB-Grösse = Anzahl H4-Programm-Zeilen * 9

Beispiel:

Ein Programm habe 120 Zeilen:

```
  :
DB      3600     [120*9]      ; (max. 16384)
  :
```

FbDnLdH4 Funktion: - H4-Programm schreiben FbDnLdH4



Funktionsbeschreibung:

Diese Funktion schreibt ein Programm in das H4-Modul. Das Programm wird aus einem DB der PCD geholt. Detailfunktionen siehe nächste Seite.

Beschreibung Inputs und Outputs:

Symbol	Beschreibung	Parameter	Media		
			Type	Format	Adr-Bereich
ModulBasic	Modul Basisadresse	Ja	K	Integer	0 -512
H4-Prog.Num.	Programmnummer im H4	Ja	K	Integer	1 -9
Source DB	Quell DB	Ja	DB	Integer	(0 -) 4000-7999 ***2)**

Bemerkung *2):
Vorzugsweise nur DB >4000 verwenden, da der Zugriff auf den DB schneller und die DB auch grösser sein können. DB >4000 befinden sich im Extended Memory, benötigen also ein PCD7.R3.. Memory-Modul.

Funktionsbeschreibung:

Während ein Upload- oder Download-FB arbeitet, können alle Immediate-Befehle weiterhin aufgerufen werden. So können zum Beispiel die Statusflags oder die aktuelle Position einer Achse weiterhin gelesen werden. Alle Befehle, welche auf den Programmspeicher des H4-Moduls zugreifen, dürfen nicht aufgerufen werden.

Hat das Programm aus dem DB keinen Platz im H4, wird der Download abgebrochen und kein Programm ins H4 geschrieben und die CPU in 'Halt' gebracht..

Der FbUpldH4 wird erst beendet, wenn das ganze H4-Programm in den DB geladen ist. Der FB verwendet den Next-COB-Befehl, womit in einem weiteren COB auch andere Aufgaben behandelt werden.

8. Fehler-Behandlung / Vorbeugung

8.1 Installation

Um Positionierungsfehlern in stark gestörter Umgebung vorzubeugen, sind die nachfolgend aufgelisteten Punkt zu beachten:

- Das PCD-System ist sauber zu erden. Es ist eine kurze Verbindung von der Erdungsklemme der PCD (GND) zur Erdschiene zu legen.
- Es sind zwischen dem H4, den Encodern und den Leistungsverstärkern abgeschirmte Leitungen einzusetzen und die Abschirmung ist beidseitig zu erden. Maximale Länge = 20m.
- Es sind ganzmetall D-Typ Stecker zu verwenden. (Abschirmung am Steckergehäuse).
- Sind zwischen der PCD4-Masse und der Masse der Maschine Potentialdiffernzen vorhanden, ist die Abschirmung maschinenseitig über ein paralleles RC-Filter anzuschliessen.
- Die H4-Leitungen (für Encoder und DAC-Ausgang) dürfen nicht parallel zu Leistungskabel für Motoren, Schützen, Schweissmaschinen usw. verlaufen.
- Es sind Leistungsverstärker mit differentiellen Eingängen einzusetzen (Spannung ~ Geschwindigkeit)

8.2 Checkliste zur Fehlersuche

1. Ist die 24V-Speisung des PCD4-Systems sauber geerdet?
2. Ist das H4-Modul mit einer **geglätteten** Gleichspannung 19 bis 32V gespeist?
3. Ist die Verdrahtung der Achsen korrekt:
   - arbeiten die Endschalter / Referenzschalter richtig? Negative Logik (siehe LEDs am H4)
   - arbeiten die Encoder? (vom H4 gespeist)
4. Die allgemeinen Parameter sind korrekt zu setzen
5. Wurde der richtige Encodertyp gewählt? 5V/24V (siehe LEDs A, B, C am H4)
6. Die Maschinen-Parameter sind korrekt zu setzen (max. Geschwindigkeit, max. Beschleunigung, Encoderauflösung, mm/Auflösung, Schleppfehler, Folgeaktion = Stop usw.)
7. Stimmt die Zählrichtung? (Achse von Hand bewegen oder mit den Jog-Befehlen für kleine Geschwindigkeit)
8. Ist die Zählposition korrekt? (QP'x')
9. Setzen der Parameter für Jog-Betrieb (nicht PID-geregelt)
10. Leistungsverstärker einschalten (mit einer Hand am Notaus-Schalter)
11. Freigeben der Achsen des H4 ENA'x' (mit einer Hand am Notaus-Schalter).
    Ist das OK-Signal vom Verstärker gekommen?
    (siehe LED 'IN' am H4)
12. Ist die DAC-Polarität korrekt? (Positiver Jog-Befehl bewegt die Achse in die positive Richtung)
13. Parameter für die Home-Routine setzen. (Einstellungen für Geschwindigkeit, Richtung und der Homeposition)
14. Ausführen des Home-Befehls. Ist die Position korrekt?
15. Ausführen eines Bewegungs-Befehls, z.B. R10 (PID-geregelt)
16. Ermitteln der optimalen Regelparameter mittels einem kleinen Bewegungsprogramm (im Grafikmodus)
17. Abspeichern der optimalen Parameter in das EEPROM mittels EW (EPROM Write)

Fehlersuche

Falls sich die Achse nicht bewegen sollte, ist der Achsenstatus im Terminalmenü zu überprüfen und der Status der Flags mit der Tabelle 2.12 im Abschnitt 7.5.5 zu vergleichen.

Führt das H4 die Befehle nicht aus, ist der Error-Code zu überprüfen und mit der Tabelle 2.13 im Abschnitt 7.5.5 zu vergleichen.

Kommt keine Verbindung zwischen dem CP (Commissioning / Programming tool) und dem H4 zustande (Kommunikationsfehler) ist folgendes zu prüfen:

- ist die richtige COM-Schnittstelle gewählt?
- hat die PCD den Ausgang RESET DSP (BA+11) nicht gesetzt?
- läuft das H4 (OK-LED hell)?
- ist das richtige Kabel eingesetzt?

8.3 Fehlerbehandlung mit FB

In den Standard-FBs befindet sich keine Struktur, welche eine Maschine bei einer Störung des H4-Moduls stoppt oder ausser Betrieb setzt. Dies muss der Anwender spezifisch für seine Maschine oder Anlage überlegen und lösen. Die Standard-FBs unterstützen jedoch den Anwender und lassen ihm trotzdem die grösstmögliche Freiheit zur Lösungsfindung.

Es werden zwei verschiedene Störungen unterschieden: "Fatal Error"-Störung und "User Error"-Störung.

Verhalten der FBs bei User Error:

Ein "User Error" wird durch das H4-Modul erzeugt. Die FBs funktionieren weiterhin. Der Anwender erkennt ein "User Error" anhand des Zustandes des Eingangs I 10 des H4-Moduls (siehe dazu Abschnitt 7.4.2). Durch das Aufrufen des FB 'fbExecH4' mit dem Befehl 'QU' erhält der Anwender detailliertere Informationen über den "User Error". Die Auswertung des "Error code" ist in Abschnitt 7.5.5 Zelle 2.13 beschrieben. Anhand dieser Informationen kann er dann das Verhalten seiner Maschine oder Anlage programmieren.

Fatal Error

Die "Fatal Error"-Störung (F7 + Flag BA, siehe Abschnitt 7.5.5 Zelle 2.12) wird durch den FB 'StatH4' erzeugt, welcher zyklisch aufgerufen werden muss. Erst wenn es unmöglich ist, das H4-Modul anzusprechen oder dieses falsch reagiert, wird diese Störung gesetzt. Dies kann vorkommen, wenn das H4-Modul defekt ist oder wenn ein Fehler auf dem PCD-Bus vorliegt. Bei einer falschen Adressierung des H4-Modul kann ebenfalls ein "Fatal Error" auftreten, da das Modul nicht anspricht und dadurch das Verhalten auf dem Bus falsch ist.

Das 'Fatal Error'-Flag kann durch den Anwender nicht gelöscht werden. Es würde auch wenig Sinn ergeben, wenn ein defektes Modul durch einfache Quittierung weiter betrieben werden könnte.

Verhalten der FBs bei Fatal Error:

FB 'fbExecH4':
Dieser FB wird nicht mehr ausgeführt. Die Vorsteuerung der Flags werden jedoch weiterhin behandelt. Entsteht ein "Fatal Error" während der Arbeit des FBs mit dem Modul, wird der FB über einen Timeout verlassen, dabei wird das 'Fatal Error'-Flag jedoch nicht gesetzt. (Die Timeout-Zeit von ca. 200 ms ist fix und wird ohne Timer erzeugt).

FB 'fbStatH4':
Erkennt dieser FB einen "Fatal Error", wird das 'Fatal Error Flag' gesetzt. Nur dieser FB kann das 'Fatal Error-Flag 7' der Statusflags setzen. Weitere Zugriffe auf das H4-Modul sind gesperrt. Die Zustände der Statusflags sind nicht mehr aktuell und werden nicht mehr verändert.

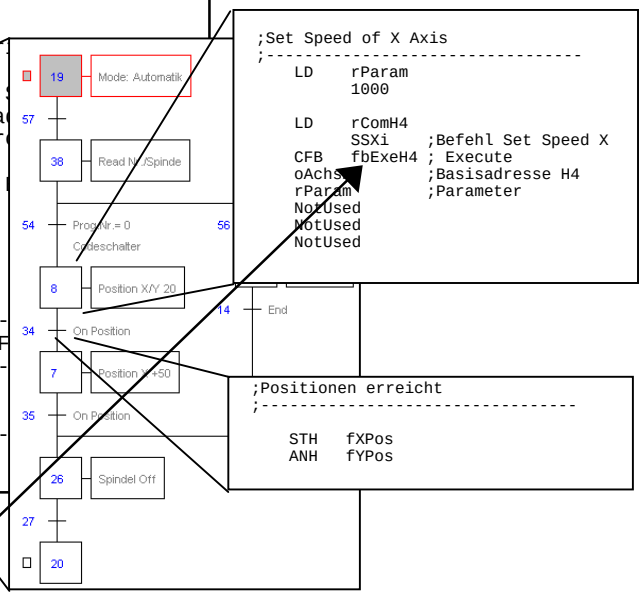
FB 'fbInitH4':
Tritt ein "Fatal Error" bereits beim Aufruf des Initial-FBs auf, wird ein QIO-Fehler erzeugt (XOB 5) und die PCD-CPU blockiert. Dies tritt zum Beispiel bei fehlendem H4-Modul auf. Ein Weiterarbeiten ist nicht möglich.

FB 'fbUpLdH4':
Der Upload Fb wird abgebrochen und der Ziel DB wird ungültig. Ein gültiger DB liegt vor, wenn im DB selbst das zweite Element (Nr. 1) ungleich 0 (Null) ist.

FB 'fbDnLdH4':
Der Download wird abgebrochen. Das angewählte Programm im H4-Modul wird nicht verändert.

Beispiel mit Erklärungen zu verschiedenen Situationen:

```
;=============
;Drehmaschine
;=============

;-------------------------------------
;Einbinden der Definitionen
;-------------------------------------
$INCLUDE        DREHDEF.SRC

;-------------------------------------
;Kaltstart
;-------------------------------------

 XOB    16

 CFB    fbInitH4        ;Init H4
        oAchsH4         ;Base Adress
        fStatus         ;Base Status
        K 2             ;Moduletype

 EXOB

;-------------------------------------
;Zyklus / Monitoring
;-------------------------------------

 COB    0
        0

 COM    O 255           ;auffr

 CFB    fbStatH4        ;Read
        oAchsH4         ;Basea
        0               ;0= pr

 CSB    0               ;Call

 ECOB


;-------------------------------
;Einbinden der Saia-Standard F
;-------------------------------
$INCLUDE        H4FB.SRC

;---------Ende Source-Code----
```




```
;Set Speed of X Axis
;---------------------------------
    LD    rParam
          1000

    LD    rComH4
          SSXi    ;Befehl Set Speed X
    CFB   fbExeH4 ; Execute
    oAchs         ;Basisadresse H4
    rParam        ;Parameter
    NotUsed
    NotUsed
    NotUsed
```

```
;Positionen erreicht
;---------------------------------

    STH   fXPos
    ANH   fYPos
```

Fatal Error:

Das 'Fatal Error Flag' wird gesetzt. Anhand dessen können die Bewegungsabläufe abgebrochen werden (z.B. mit RSB).

Hier könnte eine Abfrage des "User Error" eingebaut und das Register mit QU ausgelesen werden. Abhängig vom Fehler kann dann das Weiterarbeiten der Maschine oder Anlage programmiert werden.

Fatal Error:

Der FB bricht ab und setzt die Vorsteuerflags. Der SB wird verlassen. Die nachfolgende Transition wird nicht ausgeführt. Würde diese ausgeführt, läuft der SB nicht weiter, weil die Statusflags vorgesteuert sind. Der Anwender kann nun den SB stoppen und so ein Fehlverhalten verhindern.

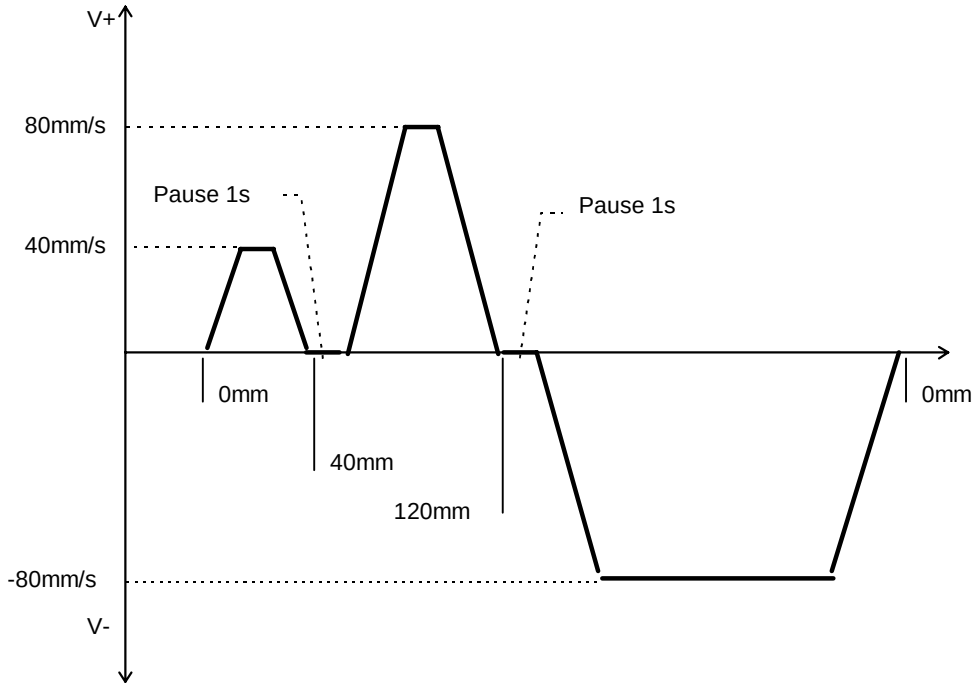
9. Anwendungsbeispiele

9.1 Verfahren eines einfachen Weges

9.1.1 Beispiel

Für dieses Beispiel ist lediglich eine Achse (X) nötig. Das Beispiel zeigt, wie ein einfacher Bewegungsablauf erstellt werden kann. Es werden die beiden Varianten, Steuerung durch H4-Ablaufprogramm oder Steuerung durch PCD-Programm gegenübergestellt.

Folgender Bewegungsablauf soll realisiert werden:



a) Software installieren

Es wird angenommen, dass die SAIA-Tools (PG3 oder PG 4) bereits auf Ihrem PC installiert sind. Ausserdem muss das CP-Tool für H4 installiert werden. Für die Installation und Bedienung siehe Abschnitt 7.3 'Programmierung mit CP-Tool'.

b) Einrichten

Richten Sie die Achse X betriebsbereit ein. Siehe Kapitel 5 für die elektrischen Anschlüsse und Abschnitt 8.2 "Checkliste". Sie benötigen eine PCD4 Steuerung mit einer CPU M1.., ein Speichermodul R..., ein Positioniermodul PCD4.H4.., ein Netzteil PCD4.N210 und Busmodule. Eine zweite Achse wird für dieses Beispiel nicht benötigt.

Die Achse sollte in der Ausgangposition so stehen, dass eine Bewegung von 120 mm ausgeführt werden kann ohne die Limit-Switches zu überfahren.

9.1.2 Variante mit CP-Tool

Die Programmierung über das CP-Tool ist die einfachste Möglichkeit. Sie benötigen dazu kein PCD-Programm und keine PCD4-CPU. Stellen Sie zuerst die Parameter des H4-Moduls entsprechend der verwendeten Achse ein. Siehe Abschnitt 7.6 'Parameterliste'.

Achtung:

> Stellen Sie die Encoderspannung (Parameter 92) richtig auf 5V oder 24V ein.
>
> Der Parameter 30 muss entsprechend der maximal möglichen Geschwindigkeit der verwendeten Achse eingestellt werden.
>
> Stellen Sie die Parameter für Zählrichtung und DAC polaritätsrichtig ein, ansonst die Achse unkontrolliert beschleunigen kann.

a) Programmierung

Anschliessend geben Sie diese Befehle im Programm-Editor des CP-Tools ein. Sie erstellen somit Ihr erstes H4-Programm. (Zum Eingeben der Befehle muss immer zuerst mit <CR> das Eingabefenster geöffnet werden).

```
ZEROX
SSX,40
XR40
WAIT1000
SSX,80
XR80
WAIT1000
XA0
END
```

Übertragen Sie das Programm als Programm Nr.1 ins H4-Modul (F4: From/To H4mod). Falls ein Kommunikationsfehler angezeigt wird, überprüfen Sie das Kabel, das im CP-Tool gewählte COM-Port und die Versorgungsspannung der PCD. (Das Programm ist bereits im CP-Tool unter Example 1 gespeichert)

b) Betrieb

Schalten Sie die X Achse ein und starten Sie das Programm. Geben Sie dazu folgende Befehle im Terminalmode des CP ein:

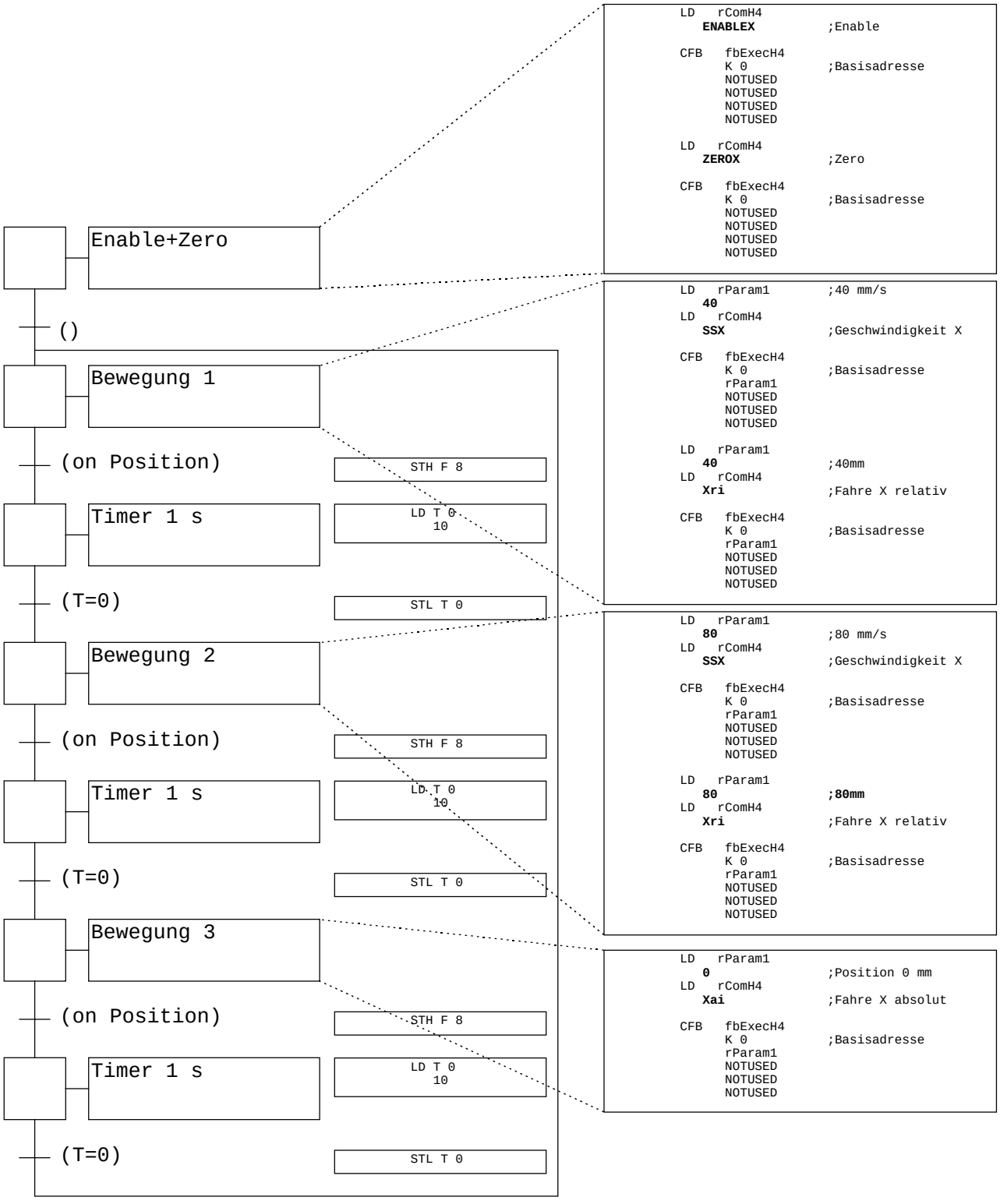
```
ENABLEX
RUN1
```

9.1.3 Variante mit PCD Programm.

Das Programm wird in GRAFTEC programmiert.

a) Programmierung

SB 0



XOB 16

```
XOB     16          ;Kaltstart

CFB     fbInitH4    ;H4-Modul initialisieren
      K 0           ;Basisadresse
        0           ;Basis Flags
      K 2           ;Modultyp 2 Achsen

LD      RHelp
        0
LD      rComH4
        P96w
CFB     fbExecH4
      K 0
        RHelp
        NotUsed
        NotUsed
        NotUsed

EXOB
```

COB 0

```
COB     0           ;Zyklischer Block
        0

CFB     fbStatH4    ;Lesen Status H4
      K 0           ;Basisadresse
        0           ;Lesen Zyklisch

CSB     0           ;Call SB

ECOB
```

b) Betrieb

Nach dem Eingeben und Laden des Programms, kann dieses gestartet werden. Es wird der gleiche Bewegungsablauf wie die Variante mit CP-Tool ausgeführt. Der Ablauf wird allerdings zyklisch wiederholt.

Beachten Sie die fettgedruckten Befehle und deren Analogie zum vorangehenden Beispiel.

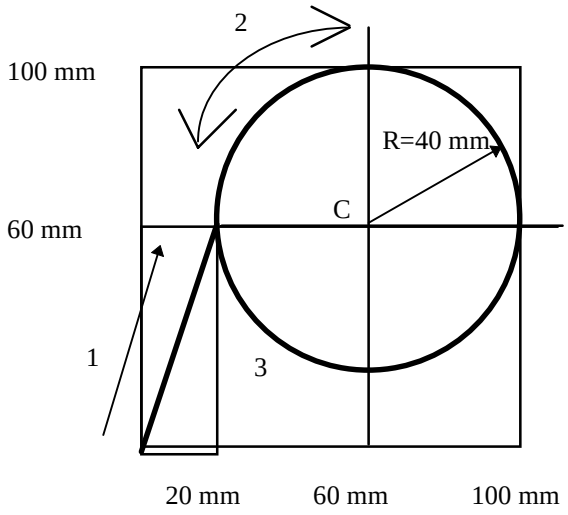
9.2 Anwendungsbeispiel mit Kreisinterpolation

Dieses Beispiel beschreibt das Arbeiten mit der Linear - und Zirkularinterpolation.

Für das Einrichten der Achsen und das Installieren der Software gelten dieselben Angaben wie im Beispiel 9.1. Es sind jedoch zwei Achsen vorgesehen. Die Eingaben erfolgen mit dem CP-Tool.

Aufgabenstellung:

Das folgende Bewegungsprogramm soll ausgeführt werden:



Die momentane Achsposition wird als Nullpunkt definiert. Die Bewegung soll von diesem Punkt aus eine Linearinterpolation mit einer Bahngeschwindigkeit von 20mm/s starten. Danach wird, mit 80mm/s eine Kreisbahn 10 mal abgefahren. Die Bewegung erfolgt im Uhrzeigersinn.

Variante 1:
Dieselbe Aufgabe, jedoch mit Kreisbahn im Gegenuhrzeigersinn und endloser Wiederholung.

Variante 2:
Ursprüngliche Aufgabenstellung aber Kreis mit Centermode programmiert. Hinzu kommt, dass ohne "Blended move" zwischen den beiden Interpolationen verfahren werden soll.

Programm: Example 2

ZEROX	Achse X Nullen
ZEROY	Achse Y Nullen
SV20	Bahngeschwindigkeit = 20mm/s
XR20,YR60	Bewegung relativ auf 20,60mm
WAIT0	Verhindert "Blended move"
SV80	Bahngeschwindigkeit = 80mm/s
FOR10	Start 10x Schlaufe
CIR40,**0**,XR80,YR0	Halbkreis Im Uhrzeigersinn mit Radius 40 und Ziel +80
CIR40,**0**,XR-80,YR0	Halbkreis Im Uhrzeigersinn mit Radius 40
NEXT	Ende Schlaufe
END	Ende Programm

mit Variante 1: Example 3

Die Unterschiede zum ursprünglichen Programm sind fett gedruckt und unterstrichen**.**

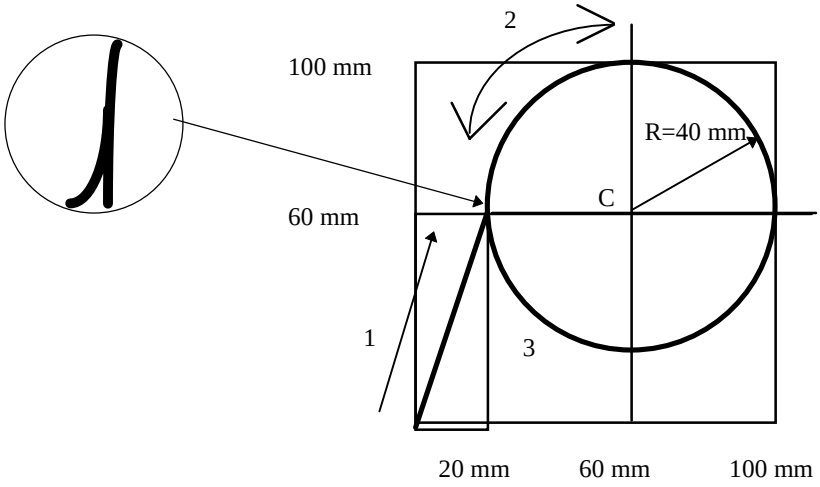
ZEROX	Achse X Nullen
ZEROY	Achse Y Nullen
SV20	Bahngeschwindigkeit = 20mm/s
XR20,YR60	Bewegung relativ auf 20,60mm
WAIT0	Verhindert "Blended move"
SV80	Bahngeschwindigkeit = 80mm/s
FOR**0**	Start Endlos-Schlaufe
CIR40,**1**,XR80,YR0	Halbkreis im Gegenuhrzeigersinn mit Radius 40
CIR40,**1**,XR-80,YR0	Halbkreis im Gegenuhrzeigersinn mit Radius 40
NEXT	Ende Schlaufe
END	Ende Programm

mit Variante 2: Example 4

Die Unterschiede zum ursprünglichen Programm sind fett gedruckt und unterstrichen**.**

	ZEROX	Achse X Nullen
	ZEROY	Achse Y Nullen
	SV20	Bahngeschwindigkeit = 20mm/s
	XR20,YR60	Bewegung relativ auf 20,60mm
*)	WAIT0	Verhindert "Blended move"
	SV80	Bahngeschwindigkeit = 80mm/s
	FOR**0**	Start Endlos-Schlaufe
	CIR**360**,**2**,XR40,YR0	360° Kreis mit Angabe des Zentrums
	NEXT	Ende Schlaufe
	END	Ende Programm

*) Ohne diese Anweisung wird "Blended move" ausgeführt. Siehe auch Skizze mit Detail und Abschnitt 6.6.



9.3 Anwendungsbeispiel Drehautomat

Das Beispiel beschreibt eine einfache Drehmaschine. Die Maschine referenziert beim Einschalten die Achsen automatisch. An einem Bedienpult können die beiden Betriebsarten 'Manuell' und 'Automatik' gewählt werden. Im manuellen Mode kann die Spindel ein- und ausgeschaltet, die Achsen vor- und zurückgefahren werden. Der Automatik-Mode startet Programme aus dem H4.

Folgende Hardware ist dazu notwendig:

- PCD4.M1.. CPU
- PCD7.R... RAM-Speicher
- PCD4.E1.. Eingangsmodul (BA 0)
- PCD4.A4.. Ausgangsmodul (BA 16)
- PCD4.N210 Netzteil
- PCD4.C1.. CPU-Busmodul
- PCD4.C2.. I/O-Busmodul
- PCD4.H4.. Achsmodul (BA 32)
- Schalterbox mit minimal 12 Schaltern
- Achsmodell: Zwei Achsen mit Inkrementalgebern, Referenzschalter und zwei Endschaltern

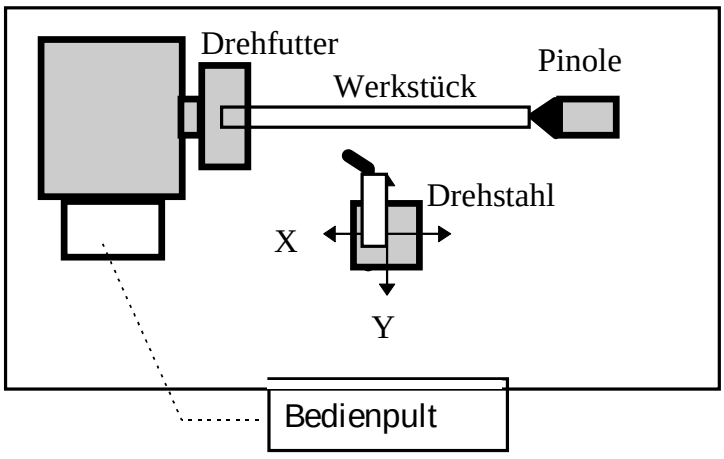
Folgende Software ist dazu notwendig:

- SAIA 'PG3' Assembler komplett mit Editor
- Commissioning Tool CP.EXE für H4-Modul
- Standard-Funktionsblöcke von SAIA mit Definitionsdatei H4FB.SRC, H4EXTN.DEF und H4DEF.SRC

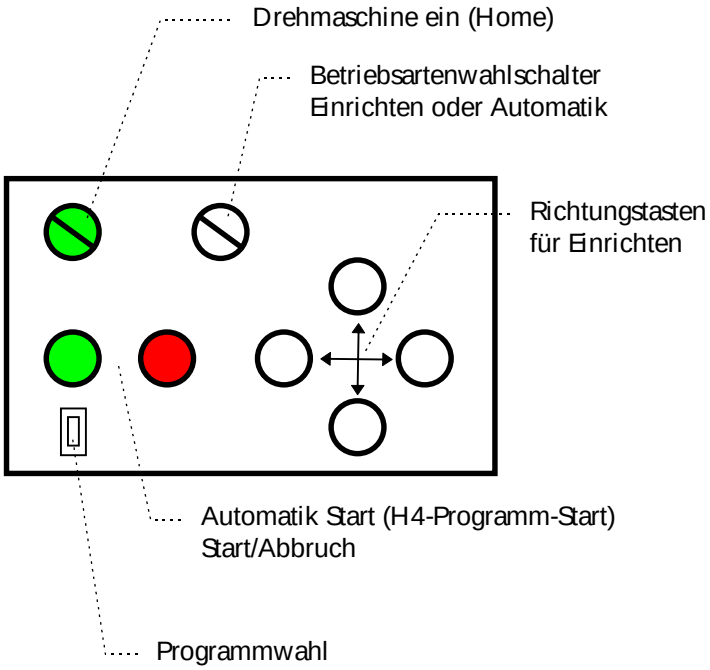
Source-Code für Anwendungsbeispiel:

- DREHDEF.INC
- DREH_SB.SRC
- DREH_MP.SRC

Skizze der Drehmaschine



Bedienpult:



Beschreibung der Funktion, welche in der Datei 'Dreh_xx.SRC' implementiert ist.

Nach dem Einschalten der Drehmaschine wird der Home Befehl ausgeführt. Sobald die Achsen referenziert sind, ist die Maschine betriebsbereit. Es kann zwischen den beiden Betriebsarten 'Einrichten' und 'Automatik' gewählt werden. Beim 'Einrichten' können die Achsen mit den Richtungstasten verfahren werden. Mit der Starttaste kann, in dieser Betriebsart, der Spindelantrieb eingeschaltet und mit der Austaste wieder ausgeschaltet werden. Wird der Automatikbetrieb gewählt, kann über dieselbe Starttaste ein H4-Programm gestartet werden. Die Starttaste leuchtet solange, wie dieses Programm läuft.

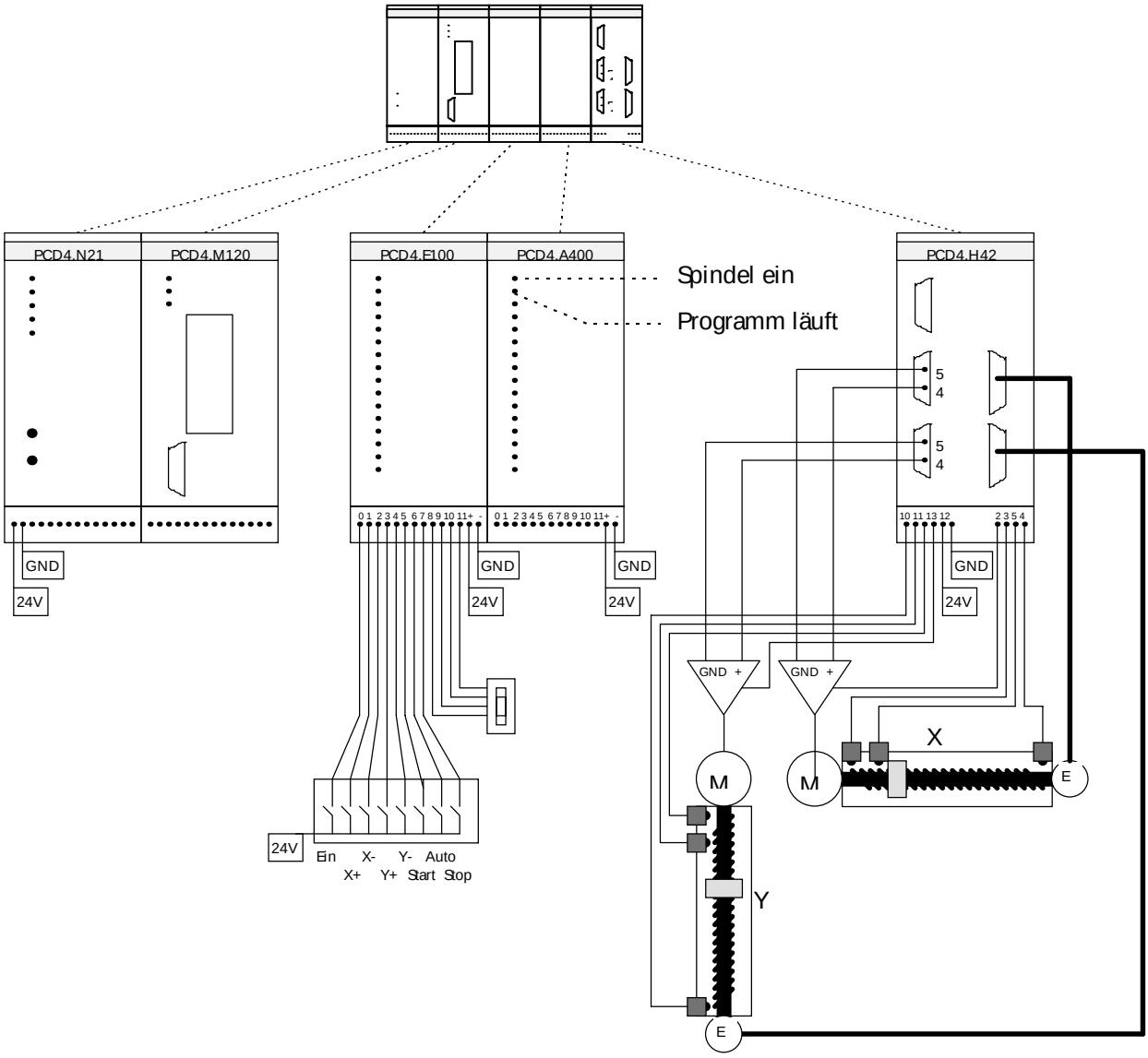
Mittels eines Codierschalters kann eine Programmnummer (1-9) gewählt werden, wobei diese vor dem Betätigen der Starttaste eingestellt werden muss. Diese Programme sind im H4 hinterlegt und können mit dem "Commissioning Tool" verändert werden. Die PCD-CPU ist nur für den Start der H4-Programme zuständig.

Wird am Codierschalter die "0" gewählt, so läuft nicht ein H4-Programm, sondern die PCD-CPU generiert einen Ablauf, in dem mit den FBs jede einzelne Bewegung als 'Immediate'-Befehl ans H4-Modul übertragen wird. Das H4-Modul führt diese Befehle unmittelbar aus, d.h: der Bewegungsablauf ist nicht im H4 sonder in der PCD-CPU hinterlegt.

Installation des Anwendungsbeispiels:

Die Eingänge der Bedienelemente werden mit einer Schalterbox simuliert. Die verwendeten E/A-Adressen sind in der Datei DREHDEF.INC definiert.

Der Zustand der Ausgänge kann auf dem A400-Modul beobachtet werden.

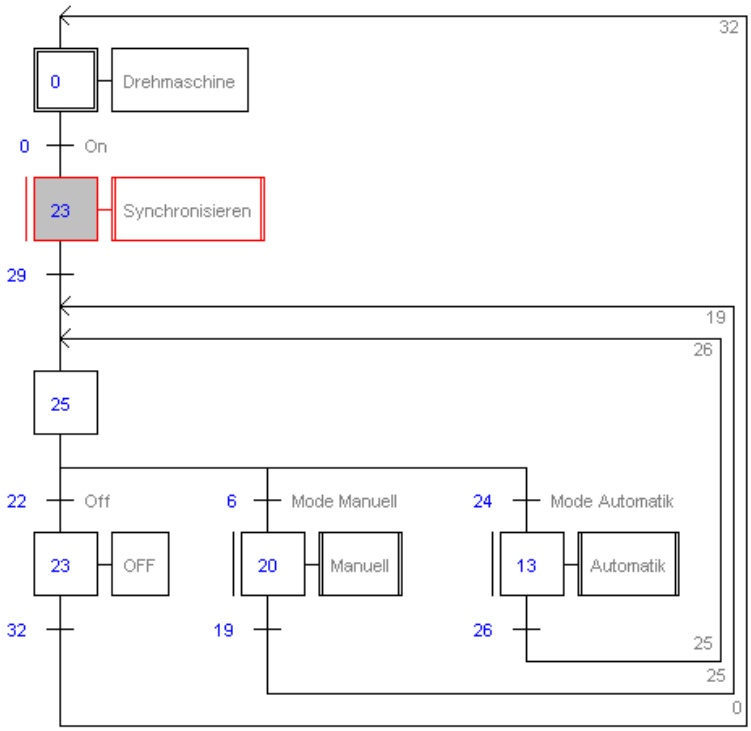


PCD Grundstruktur für Drehmaschine:

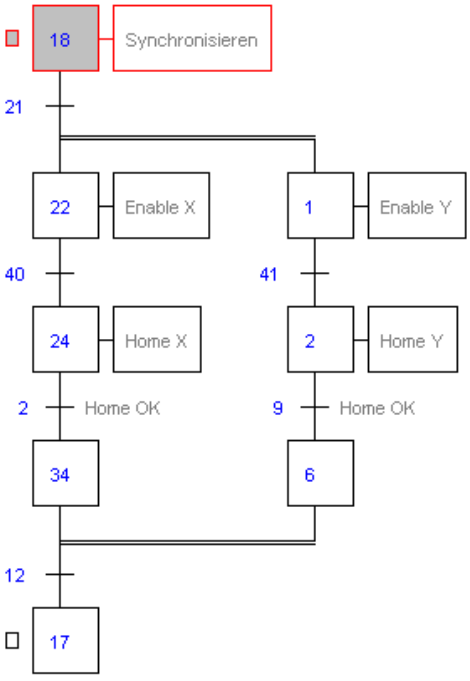
```
;=============
;Drehmaschine
;=============


;-------------------------------------
;Einbinden der Definitionen
;-------------------------------------
$INCLUDE           DREHDEF.SRC
$INCLUDE           H4EXTN.DEF

;-------------------------------------
;Kaltstart
;-------------------------------------

XOB     16

CFB     fbInitH4   ; Init H4
        oAchsH4    ; Base Adress Module
        fStatus    ; Base Statusflags
        K_2        ; Moduletype

EXOB


-------------------------------------
Zyklus / Monitoring
-------------------------------------

COB     0
        0

COM   O 255        ; auffrischen des Watchdog

CFB     fbStatH4   ; Read Status H4
        oAchsH4    ; Base address of H4
        0          ; 0 = pro Zyklus eine Achse

CSB     0          ;Call Basis-Ablaufstruktur der Drehm.


ECOB


;-------------------------------------
;Einbinden der SAIA-Standard FBs für H4 Modul
;-------------------------------------
$INCLUDE           H4FB.SRC

;-------- Ende Source-Code ------------------------
```

Ablauf der Drehmaschine in SB 0

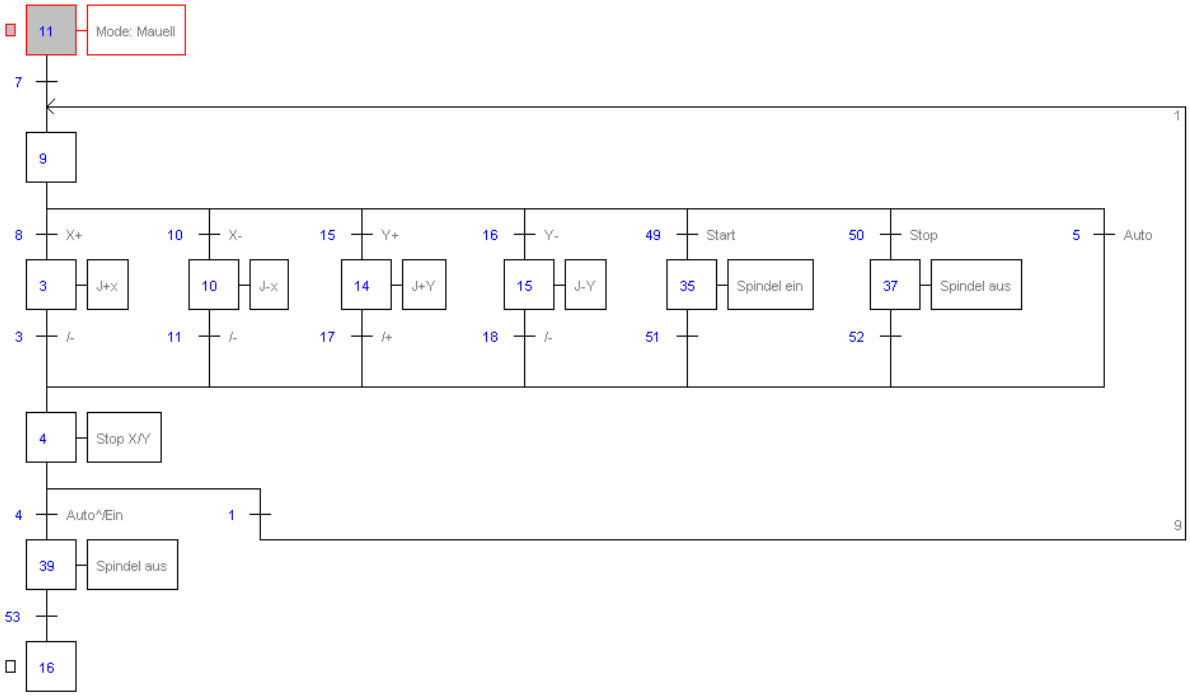
Basis-Ablaufstruktur



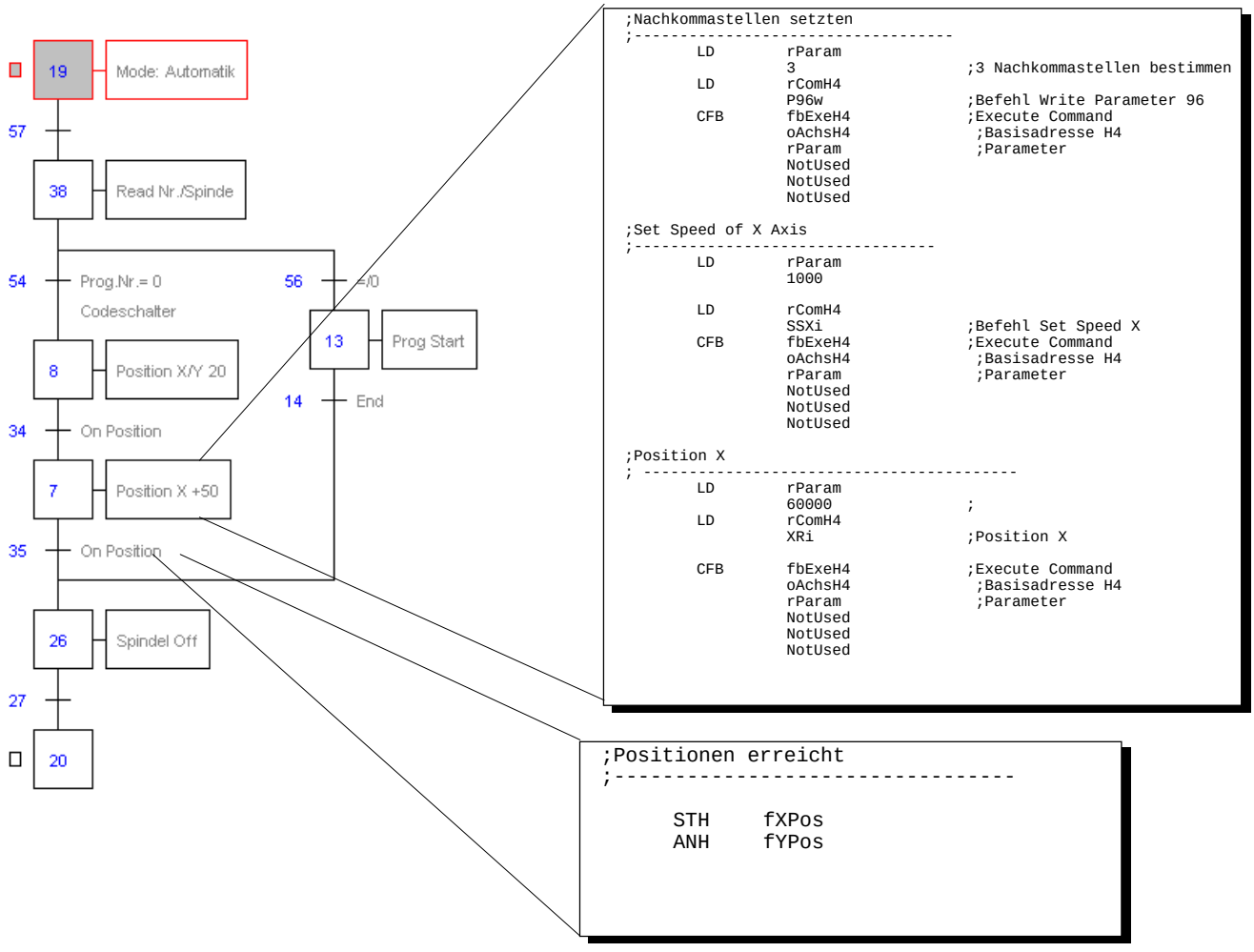
Synchronisierung:



Betriebsmode Manuell



Betriebsmode Automatik



9.4 Anwendungsbeispiel mit unabhängigen Achsen (mit OPEN/CLOSE Funktion)

Dieses Beispiel beschreibt das Arbeiten mit unabhängigen Achsen, welche gleichzeitig rsp. überlappend aber nicht interpoliert verfahren werden sollen.

Zu diesem Zweck muss für jede Achse ein Programm geschrieben werden. Die Programme können anschliessend unabhängig voneinander (immediate) oder abhängig voneinander (verschachtelter Programmstart innerhalb eines Programmes) gestartet werden.

Mit den FBs werden die Befehle OPEN und CLOSE zum Erstellen eines Programmes verwendet:

```
LD      R 0
        K 1          ; Zeilen Nr. ab welcher das Programm editiert wird
LD      rComH4
        OPEN5        ; Programm 5 wird zum editieren geöffnet
                     ;   (auf Zeile 1)
CFB     fbExeH4      ; Execute Command
        K 0          ; Basisadresse des H4
        R 0          ; Zeilen Nr.
        R 1          ; nicht verwendet für diesen Befehl
        R 2          ;   "
        R 3          ;   "

LD      R 0
        K 20000      ; entspricht 20 bei P96 = 3
LD      rComH4       ; XA20 wird in Programm 5 auf Zeile 1 geschrieben
        XAp          ; das p bezeichnet den Programmbefehl
                     ;  Xai würde unmittelbar (immediate) ausgeführt
                     ;  und nicht in den Programmspeicher geschrieben
CFB     fbExeH4      ; Execute Command
        K 0          ; Basisadresse
        R 0          ; nicht verwendet für diesen Befehl
        R 1          ;   "
        R 2          ;   "
        R 3          ;   "

LD      rComH4       ; END wird auf Zeile 2 geschrieben
        END          ; (END existiert nur als Programmbefehl und muss
                     ;  nicht speziell mit einem p gekennzeichnet werden)
```

```
CFB     fbExeH4     ; Execute Command
        K 0         ; Basisadresse
        R 0         ; nicht verwendet für diesen Befehl
        R 1         ;  "
        R 2         ;  "
        R 3         ;  "

LD      rComH4      ; Mit CLOSE wird das editierte Programm in dem
        CLOSE       ; Speicherplatz 5 abgespeichert
CFB     fbExeH4     ; Execute Command
        K 0         ; Basisadresse
        R 0         ; nicht verwendet für diesen Befehl
        R 1         ;  "
        R 2         ;  "
        R 3         ;  "
```

Befindet sich das editierte Programm in Ausführung, wird der CLOSE Befehl nicht akzeptiert (Störungs-Code 6) und das Programm nicht überschrieben.

Ansonsten wird im Programmspeicher Nr. 5 folgendes Programm stehen:

```
1 - XA20
2 - END
```

Es werden nur die editierten Zeilen überschrieben. Soll ein längeres Programm überschrieben werden, muss dieses vor dem CLOSE mit dem Befehl EP (erase program) gelöscht werden.

Notizen

Anhang A: Kommandocode-Definitionen für die Programmierung mit FBs

```
;
;-----------------------------------------------------------------
; Kommandocode-Definitionen für FBs Version V001
;-----------------------------------------------------------------
;
; Alle Codes sind im Hex-Format und werden vor der
; Verwendung ins Register 'rComH4' (BAR+0) geladen.
; Dieses Register ist im Debugger im Hex-Format
; anzuzeigen
;

;motion commands
;---------------
ZeroXi          EQU    0A0010000h       ;Zero X immediate
ZeroYi          EQU    0A0020000h       ;Zero Y immediate
ZeroZi          EQU    0A0030000h       ;Zero Z immediate
ZeroWi          EQU    0A0040000h       ;Zero W immediate

ZeroXp          EQU    0C0010000h       ;Zero X program
ZeroYp          EQU    0C0020000h       ;Zero Y program
ZeroZp          EQU    0C0030000h       ;Zero Z program
ZeroWp          EQU    0C0040000h       ;Zero W program

HomeX           EQU    0A0055100h       ;Home X
HomeY           EQU    0A0065200h       ;Home Y
HomeZ           EQU    0A0075400h       ;Home Z
HomeW           EQU    0A0085800h       ;Home W

Rapid           EQU    0A0090000h       ;Select Jog Rapid speed
Normal          EQU    0A08F0000h       ;Select Jog Normal speed

JUpX            EQU    0A00A0000h       ;Jog + on X
JdnX            EQU    0A00B0000h       ;Jog - on X
JSX             EQU    0A00C0000h       ;Jog Stop on X

JUpY            EQU    0A10A0000h       ;Jog + on Y
JDnY            EQU    0A10B0000h       ;Jog - on Y
JSY             EQU    0A10C0000h       ;Jog Stop on Y

JUpZ            EQU    0A20A0000h       ;Jog + on Z
JDnZ            EQU    0A20B0000h       ;Jog - on Z
JSZ             EQU    0A20C0000h       ;Jog Stop on Z

JUpW            EQU    0A30A0000h       ;Jog + on W
JDnW            EQU    0A30B0000h       ;Jog - on W
JSW             EQU    0A30C0000h       ;Jog Stop on W

QPX             EQU    0200E0003h       ;Read (Query) Position of X
QPY             EQU    0210E0003h       ;Read (Query) Position of Y
QPZ             EQU    0220E0003h       ;Read (Query) Position of Z
QPW             EQU    0230E0003h       ;Read (Query) Position of W

QSX             EQU    0200F0002h       ;Read Status of X Axis
QSY             EQU    0210F0002h       ;Read Status of Y Axis
QSZ             EQU    0220F0002h       ;Read Status of Z Axis
QSW             EQU    0230F0002h       ;Read Status of W Axis
```



```
QVX             EQU     020100003h      ;Read actual Velocity X
QVY             EQU     021100003h      ;Read actual Velocity Y
QVZ             EQU     022100003h      ;Read actual Velocity Z
QVW             EQU     023100003h      ;Read actual Velocity W

QEX             EQU     020110003h      ;Read actual Pos.error X
QEY             EQU     021110003h      ;Read actual Pos.error Y
QEZ             EQU     022110003h      ;Read actual Pos.error Z
QEW             EQU     023110003h      ;Read actual Pos.error W

SSXi            EQU     0A0120003h      ;Set motion Speed of X
SSYi            EQU     0A0130003h      ;Set motion Speed of Y
SSZi            EQU     0A0140003h      ;Set motion Speed of Z
SSWi            EQU     0A0150003h      ;Set motion Speed of W

SSXp            EQU     0C0120003h      ;Set motion Speed of X
SSYp            EQU     0C0130003h      ;Set motion Speed of Y
SSZp            EQU     0C0140003h      ;Set motion Speed of Z
SSWp            EQU     0C0150003h      ;Set motion Speed of W

SPXi            EQU     0A0480003h      ;Set actual Position X
SPYi            EQU     0A0490003h      ;Set actual Position Y
SPZi            EQU     0A04A0003h      ;Set actual Position Z
SPWi            EQU     0A04B0003h      ;Set actual Position W

SPXp            EQU     0C0480003h      ;Set actual Position X
SPYp            EQU     0C0490003h      ;Set actual Position Y
SPZp            EQU     0C04A0003h      ;Set actual Position Z
SPWp            EQU     0C04B0003h      ;Set actual Position W

SVi             EQU     0A0160003h      ;Set Vector motion Speed
SAi             EQU     0A0170003h      ;Set motion acceleration
SDi             EQU     0A0180003h      ;Set motion deceleration

SVp             EQU     0C0160003h      ;Set Vector motion Speed
SAp             EQU     0C0170003h      ;Set motion acceleration
SDp             EQU     0C0180003h      ;Set motion deceleration

XAi             EQU     0A01E1103h      ;Move X Absolute
YAi             EQU     0A01F1203h      ;Move Y Absolute
ZAi             EQU     0A0201403h      ;Move Z Absolute
WAi             EQU     0A0211803h      ;Move W Absolute

XAp             EQU     0C01E0003h      ;Move X Absolute
YAp             EQU     0C01F0003h      ;Move Y Absolute
ZAp             EQU     0C0200003h      ;Move Z Absolute
WAp             EQU     0C0210003h      ;Move W Absolute

XYAi            EQU     0A028130Fh      ;Move X,Y Absolute
XZAi            EQU     0A029150Fh      ;Move X,Z Absolute
XWAi            EQU     0A02A190Fh      ;Move X,W Absolute
YZAi            EQU     0A02B160Fh      ;Move Y,Z Absolute
YWAi            EQU     0A02C3A0Fh      ;Move Y,W Absolute
ZWAi            EQU     0A02D3C0Fh      ;Move Z,W Absolute

XYAp            EQU     0C028000Fh      ;Move X,Y Absolute
XZAp            EQU     0C029000Fh      ;Move X,Z Absolute
XWAp            EQU     0C02A000Fh      ;Move X,W Absolute
YZAp            EQU     0C02B000Fh      ;Move Y,Z Absolute
YWAp            EQU     0C02C000Fh      ;Move Y,W Absolute
ZWAp            EQU     0C02D000Fh      ;Move Z,W Absolute
```

```
XYZAi           EQU     0A034173Fh      ;Move X,Y,Z Absolute
YZWAi           EQU     0A0361E3Fh      ;Move Y,Z,W Absolute
XZWAi           EQU     0A0901D3Fh      ;Move X,Z,W Absolute
XYWAi           EQU     0A0351B3Fh      ;Move X,Y,W Absolute

XYZAp           EQU     0C034003Fh      ;Move X,Y,Z Absolute
YZWAp           EQU     0C036003Fh      ;Move Y,Z,W Absolute
XZWAp           EQU     0C090003Fh      ;Move X,Z,W Absolute
XYWAp           EQU     0C035003Fh      ;Move X,Y,W Absolute

XYZWAi          EQU     0A03A1FFFh      ;Move X,Y,Z,W Absolute

XYZWAp          EQU     0C03A00FFh      ;Move X,Y,Z,W Absolute

XRi             EQU     0A0221103h      ;Move X relative
YRi             EQU     0A0231203h      ;Move Y relative
ZRi             EQU     0A0241403h      ;Move Z relative
WRi             EQU     0A0251803h      ;Move W relative

XRp             EQU     0C0220003h      ;Move X relative
YRp             EQU     0C0230003h      ;Move Y relative
ZRp             EQU     0C0240003h      ;Move Z relative
WRp             EQU     0C0250003h      ;Move W relative

XYRi            EQU     0A02E130Fh      ;Move X,Y relative
XZRi            EQU     0A02F150Fh      ;Move X,Z relative
XWRi            EQU     0A030190Fh      ;Move X,W relative
YZRi            EQU     0A031160Fh      ;Move Y,Z relative
YWRi            EQU     0A0321A0Fh      ;Move Y,W relative
ZWRi            EQU     0A0331C0Fh      ;Move Z,W relative

XYRp            EQU     0C02E000Fh      ;Move X,Y relative
XZRp            EQU     0C02F000Fh      ;Move X,Z relative
XWRp            EQU     0C030000Fh      ;Move X,W relative
YZRp            EQU     0C031000Fh      ;Move Y,Z relative
YWRp            EQU     0C032000Fh      ;Move Y,W relative
ZWRp            EQU     0C033000Fh      ;Move Z,W relative

XYZRi           EQU     0A037173Fh      ;Move X,Y,Z relative
YZWRi           EQU     0A0391E3Fh      ;Move Y,Z,W relative
XZWRi           EQU     0A0911D3Fh      ;Move X,Z,W relative
XYWRi           EQU     0A0381B3Fh      ;Move X,Y,W relative

XYZRp           EQU     0C037003Fh      ;Move X,Y,Z relative
YZWRp           EQU     0C039003Fh      ;Move Y,Z,W relative
XZWRp           EQU     0C091003Fh      ;Move X,Z,W relative
XYWRp           EQU     0C038003Fh      ;Move X,Y,W relative

XYZWRi          EQU     0A03B1FFFh      ;Move X,Y,Z,W relative

XYZWRp          EQU     0C03B00FFh      ;Move X,Y,Z,W relative

CirXYRi         EQU     0A04213F7h      ;Circle X,Y relative
CirXZRi         EQU     0A04315F7h      ;Circle X,Z relative
CirXWRi         EQU     0A04419F7h      ;Circle X,W relative
CirYZRi         EQU     0A04516F7h      ;Circle X,W relative
CirYWRi         EQU     0A0461AF7h      ;Circle Y,W relative
CirZWRi         EQU     0A0471CF7h      ;Circle Z,W relative
```

```
CirXYRp         EQU    0C04200F7h       ;Circle X,Y relative
CirXZRp         EQU    0C04300F7h       ;Circle X,Z relative
CirXWRp         EQU    0C04400F7h       ;Circle X,W relative
CirYZRp         EQU    0C04500F7h       ;Circle X,W relative
CirYWRp         EQU    0C04600F7h       ;Circle Y,W relative
CirZWRp         EQU    0C04700F7h       ;Circle Z,W relative

CirXYAi         EQU    0A03C13F7h       ;Circle X,Y absolute
CirXZAi         EQU    0A03D15F7h       ;Circle X,Z absolute
CirXWAi         EQU    0A03E19F7h       ;Circle X,W absolute
CirYZAi         EQU    0A03F16F7h       ;Circle X,W absolute
CirYWAi         EQU    0A0401AF7h       ;Circle Y,W absolute
CirZWAi         EQU    0A0411CF7h       ;Circle Z,W absolute

CirXYAp         EQU    0C03C00F7h       ;Circle X,Y absolute
CirXZAp         EQU    0C03D00F7h       ;Circle X,Z absolute
CirXWAp         EQU    0C03E00F7h       ;Circle X,W absolute
CirYZAp         EQU    0C03F00F7h       ;Circle X,W absolute
CirYWAp         EQU    0C04000F7h       ;Circle Y,W absolute
CirZWAp         EQU    0C04100F7h       ;Circle Z,W absolute


;Program Control Commands
;------------------------
END             EQU    0C05D0000h       ;End of Programm
FOR             EQU    0C05E0002h       ;Beginn Loop
NEXT            EQU    0C05F0000h       ;End Loop
GOTO            EQU    0C0600002h       ;Jump
GOSUB           EQU    0C0610002h       ;Jump to Subroutine
RETURN          EQU    0C0620000h       ;End of Subroutine
STOP            EQU    0C0630000h       ;Stop Programm
WAIT            EQU    0C0640002h       ;Wait

RUNp            EQU    0C0810001h       ;Run Program
BREAKp          EQU    0C0820001h       ;Break Program
Gp              EQU    0C0880009h       ;Set program execution pointer to line


;System Control Commands
;-----------------------
FO              EQU    0A0500002h       ;Set Feed Override (0-120%)

DriftX          EQU    0A0510000h       ;Execute drift compensation X
DriftY          EQU    0A1510000h       ;Execute drift compensation Y
DriftZ          EQU    0A2510000h       ;Execute drift compensation Z
DriftW          EQU    0A3510000h       ;Execute drift compensation W

QPIX            EQU    020930003h       ;Query Position X in Pulses
QPIY            EQU    021930003h       ;Query Position Y in Pulses
QPIZ            EQU    022930003h       ;Query Position Z in Pulses
QPIW            EQU    023930003h       ;Query Position W in Pulses

QU              EQU    02092000Ah       ;Read User-Error information

QL1             EQU    0218B0002h       ;Read Execution Line Program 1
QL2             EQU    0228B0002h       ;Read Execution Line Program 2
QL3             EQU    0238B0002h       ;Read Execution Line Program 3
QL4             EQU    0248B0002h       ;Read Execution Line Program 4
QL5             EQU    0258B0002h       ;Read Execution Line Program 5
QL6             EQU    0268B0002h       ;Read Execution Line Program 6
QL7             EQU    0278B0002h       ;Read Execution Line Program 7
QL8             EQU    0288B0002h       ;Read Execution Line Program 8
QL9             EQU    0298B0002h       ;Read Execution Line Program 9
```

```
KILLX           EQU     0A0520000h      ;Kill X
KILLY           EQU     0A0530000h      ;Kill Y
KILLZ           EQU     0A0540000h      ;Kill Z
KILLW           EQU     0A0550000h      ;Kill W

ENAXi           EQU     0A0560000h      ;Enable X
ENAYi           EQU     0A0570000h      ;Enable Y
ENAZi           EQU     0A0580000h      ;Enable Z
ENAWi           EQU     0A0590000h      ;Enable W

ENAXp           EQU     0C0560000h      ;Enable X
ENAYp           EQU     0C0570000h      ;Enable Y
ENAZp           EQU     0C0580000h      ;Enable Z
ENAWp           EQU     0C0590000h      ;Enable W

VOUTX           EQU     0A05A0003h      ;Output Voltage X
VOUTY           EQU     0A15A0003h      ;Output Voltage Y
VOUTZ           EQU     0A25A0003h      ;Output Voltage Z
VOUTW           EQU     0A35A0003h      ;Output Voltage W

SPLOCK          EQU     0A05B0000h      ;Lock serial port
SPUNLOCK        EQU     0A05C0000h      ;Unlock serial Port
EREAD           EQU     0A0650000h      ;Read EEPROM
EWRITE          EQU     0A0660000h      ;Write EEPROM

SCXi            EQU     0A0678100h      ;Set Capture function X
SCYi            EQU     0A0688200h      ;Set Capture function Y
SCZi            EQU     0A0698400h      ;Set Capture function Z
SCWi            EQU     0A06A8800h      ;Set Capture function W

SCXp            EQU     0C0670000h      ;Set Capture function X
SCYp            EQU     0C0680000h      ;Set Capture function Y
SCZp            EQU     0C0690000h      ;Set Capture function Z
SCWp            EQU     0C06A0000h      ;Set Capture function W

QCX             EQU     0208D0003h      ;Query Capture Position X
QCY             EQU     0218D0003h      ;Query Capture Position Y
QCZ             EQU     0228D0003h      ;Query Capture Position Z
QCW             EQU     0238D0003h      ;Query Capture Position W

QCIX            EQU     020940003h      ;Query Capture Position X in Pulses
QCIY            EQU     021940003h      ;Query Capture Position Y in Pulses
QCIZ            EQU     022940003h      ;Query Capture Position Z in Pulses
QCIW            EQU     023940003h      ;Query Capture Position W in Pulses

SOXi            EQU     0A06B2103h      ;Set Output Compare X
SOYi            EQU     0A06C2203h      ;Set Output Compare Y
SOZi            EQU     0A06D2403h      ;Set Output Compare Z
SOWi            EQU     0A06E2803h      ;Set Output Compare W

SOXp            EQU     0C06B0003h      ;Set Output Compare X
SOYp            EQU     0C06C0003h      ;Set Output Compare Y
SOZp            EQU     0C06D0003h      ;Set Output Compare Z
SOWp            EQU     0C06E0003h      ;Set Output Compare W

SOIXi           EQU     0A0702103h      ;Set Output Compare X in Pulses
SOIYi           EQU     0A0712203h      ;Set Output Compare Y in Pulses
SOIZi           EQU     0A0722403h      ;Set Output Compare Z in Pulses
SOIWi           EQU     0A0732803h      ;Set Output Compare W in Pulses
```

```
SOIXp           EQU    0C0700003h       ;Set Output Compare X in Pulses
SOIYp           EQU    0C0710003h       ;Set Output Compare Y in Pulses
SOIZp           EQU    0C0720003h       ;Set Output Compare Z in Pulses
SOIWp           EQU    0C0730003h       ;Set Output Compare W in Pulses

STEP            EQU    0A0800001h       ;Execution a single instruction

RUNi            EQU    0A0810001h       ;Execute program
BREAKi          EQU    0A0820001h       ;Break Program
HALTALL         EQU    0A0830000h       ;Stop program and hold position
RESUME          EQU    0A0840000h       ;Resume all program simulatneously

EP              EQU    0A0850001h       ;Erase program (all line)

Gi              EQU    0A0880009h       ;Set program execution pointer to line

QM              EQU    02086000Ah       ;Query free memory

OPEN1           EQU    0E0010002h       ;Open Program 1 for Edit
OPEN2           EQU    0E0020002h       ;Open Program 2 for Edit
OPEN3           EQU    0E0030002h       ;Open Program 3 for Edit
OPEN4           EQU    0E0040002h       ;Open Program 4 for Edit
OPEN5           EQU    0E0050002h       ;Open Program 5 for Edit
OPEN6           EQU    0E0060002h       ;Open Program 6 for Edit
OPEN7           EQU    0E0070002h       ;Open Program 7 for Edit
OPEN8           EQU    0E0080002h       ;Open Program 8 for Edit
OPEN9           EQU    0E0090002h       ;Open Program 9 for Edit

CLOSE           EQU    0A0870000h       ;Close and save program under edit
;

;General parameter to read from the modul
;----------------------------------------
P90r            EQU     005A0001h       ;Parameter 90 Read
P91r            EQU     005B0001h       ;Parameter 91 Read
Px92r           EQU     005C0001h       ;Parameter 92 Read on X
Pz92r           EQU     025C0001h       ;Parameter 92 Read on Z
P94r            EQU     005E0001h       ;Parameter 94 Read
P95r            EQU     005F0001h       ;Parameter 95 Read
P96r            EQU     00600001h       ;Parameter 96 Read
P97r            EQU     00610001h       ;Parameter 97 Read
P98r            EQU     00620001h       ;Parameter 98 Read

;General parameter to write to the modul
;---------------------------------------
P90w            EQU     805A0001h       ;Parameter 90 Write
P91w            EQU     805B0001h       ;Parameter 91 Write
Px92w           EQU     805C0001h       ;Parameter 92 Write on X
Pz92w           EQU     825C0001h       ;Parameter 92 Write on Z
P94w            EQU     805E0001h       ;Parameter 94 Write
P95w            EQU     805F0001h       ;Parameter 95 Write
P96w            EQU     80600001h       ;Parameter 96 Write
P97w            EQU     80610001h       ;Parameter 97 Write
P98w            EQU     80620001h       ;Parameter 98 Write
```

```
;Parameter to write on Axis 'X'
;-----------------------------
PX01w           EQU     80010001h       ;Parameter 1 Write on X
PX02w           EQU     80020002h       ;Parameter 2 Write on X
PX03w           EQU     80030003h       ;Parameter 3 Write on X
PX04w           EQU     80040001h       ;Parameter 4 Write on X
PX05w           EQU     80050003h       ;Parameter 5 Write on X
PX06w           EQU     80060001h       ;Parameter 6 Write on X
PX07w           EQU     80070003h       ;Parameter 7 Write on X
PX08w           EQU     80080001h       ;Parameter 8 Write on X

PX10w           EQU     800A0003h       ;Parameter 10 Write on X
PX11w           EQU     800B0003h       ;Parameter 11 Write on X
PX12w           EQU     800C0003h       ;Parameter 12 Write on X
PX13w           EQU     800D0001h       ;Parameter 13 Write on X
PX14w           EQU     800E0003h       ;Parameter 14 Write on X
PX15w           EQU     800F0003h       ;Parameter 15 Write on X
PX16w           EQU     80100001h       ;Parameter 16 Write on X

PX20w           EQU     80140001h       ;Parameter 20 Write on X
PX21w           EQU     80150001h       ;Parameter 21 Write on X
PX22w           EQU     80160003h       ;Parameter 22 Write on X
PX23w           EQU     80170003h       ;Parameter 23 Write on X
PX24w           EQU     80180003h       ;Parameter 24 Write on X

PX30w           EQU     801E0003h       ;Parameter 30 Write on X
PX31w           EQU     801F0003h       ;Parameter 31 Write on X
PX32w           EQU     80200003h       ;Parameter 32 Write on X
PX33w           EQU     80210003h       ;Parameter 33 Write on X

PX40w           EQU     80280003h       ;Parameter 40 Write on X
PX41w           EQU     80290003h       ;Parameter 41 Write on X
PX42w           EQU     802A0001h       ;Parameter 42 Write on X
PX43w           EQU     802B0003h       ;Parameter 43 Write on X
PX44w           EQU     802C0003h       ;Parameter 44 Write on X
PX45w           EQU     802D0003h       ;Parameter 45 Write on X

PX50w           EQU     80320003h       ;Parameter 50 Write on X
PX51w           EQU     80330003h       ;Parameter 51 Write on X
PX52w           EQU     80340003h       ;Parameter 52 Write on X
PX53w           EQU     80350003h       ;Parameter 53 Write on X
PX54w           EQU     80360003h       ;Parameter 54 Write on X
PX55w           EQU     80370003h       ;Parameter 55 Write on X
PX56w           EQU     80380002h       ;Parameter 56 Read on X

PX62w           EQU     803E0001h       ;Parameter 62 Write on X
PX63w           EQU     803F0001h       ;Parameter 63 Write on X


;Parameter to read on Axis 'X'
;----------------------------
PX01r           EQU     00010001h       ;Parameter 1 Read on X
PX02r           EQU     00020002h       ;Parameter 2 Read on X
PX03r           EQU     00030003h       ;Parameter 3 Read on X
PX04r           EQU     00040001h       ;Parameter 4 Read on X
PX05r           EQU     00050003h       ;Parameter 5 Read on X
PX06r           EQU     00060001h       ;Parameter 6 Read on X
PX07r           EQU     00070003h       ;Parameter 7 Read on X
Px08r           EQU     00080001h       ;Parameter 8 Read on X
```

```
PX10r           EQU     000A0003h       ;Parameter 10 Read on X
PX11r           EQU     000B0003h       ;Parameter 11 Read on X
PX12r           EQU     000C0003h       ;Parameter 12 Read on X
PX13r           EQU     000D0001h       ;Parameter 13 Read on X
PX14r           EQU     000E0003h       ;Parameter 14 Read on X
PX15r           EQU     000F0003h       ;Parameter 15 Read on X
PX16r           EQU     00100001h       ;Parameter 16 Read on X

PX20r           EQU     00140001h       ;Parameter 20 Read on X
PX21r           EQU     00150001h       ;Parameter 21 Read on X
PX22r           EQU     00160003h       ;Parameter 22 Read on X
PX23r           EQU     00170003h       ;Parameter 23 Read on X
PX24r           EQU     00180003h       ;Parameter 24 Read on X

PX30r           EQU     001E0003h       ;Parameter 30 Read on X
PX31r           EQU     001F0003h       ;Parameter 31 Read on X
PX32r           EQU     00200003h       ;Parameter 32 Read on X
PX33r           EQU     00210003h       ;Parameter 33 Read on X

PX40r           EQU     00280003h       ;Parameter 40 Read on X
PX41r           EQU     00290003h       ;Parameter 41 Read on X
PX42r           EQU     002A0001h       ;Parameter 42 Read on X
PX43r           EQU     002B0003h       ;Parameter 43 Read on X
PX44r           EQU     002C0003h       ;Parameter 44 Read on X
PX45r           EQU     002D0003h       ;Parameter 45 Read on X

PX50r           EQU     00320003h       ;Parameter 50 Read on X
PX51r           EQU     00330003h       ;Parameter 51 Read on X
PX52r           EQU     00340003h       ;Parameter 52 Read on X
PX53r           EQU     00350003h       ;Parameter 53 Read on X
PX54r           EQU     00360003h       ;Parameter 54 Read on X
PX55r           EQU     00370003h       ;Parameter 55 Read on X
PX56r           EQU     00380002h       ;Parameter 56 Read on X

PX62r           EQU     003E0001h       ;Parameter 62 Read on X
PX63r           EQU     003F0001h       ;Parameter 63 Read on X


;Parameter Write to Program for X
;--------------------------------
PX10             EQU    0C0A0000Dh       ;Parameter 10 Write-Prog on X
PX11             EQU    0C0A0000Dh       ;Parameter 11 Write-Prog on X
PX12             EQU    0C0A0000Dh       ;Parameter 12 Write-Prog on X
PX13             EQU    0C0A00005h       ;Parameter 13 Write-Prog on X
PX14             EQU    0C0A0000Dh       ;Parameter 14 Write-Prog on X
PX15             EQU    0C0A0000Dh       ;Parameter 15 Write-Prog on X
PX16             EQU    0C0A00005h       ;Parameter 16 Write-Prog on X

PX20             EQU    0C0A00005h       ;Parameter 20 Write-Prog on X
PX21             EQU    0C0A00005h       ;Parameter 21 Write-Prog on X
PX22             EQU    0C0A0000Dh       ;Parameter 22 Write-Prog on X
PX23             EQU    0C0A0000Dh       ;Parameter 23 Write-Prog on X
PX24             EQU    0C0A0000Dh       ;Parameter 24 Write-Prog on X

PX30             EQU    0C0A0000Dh       ;Parameter 30 Write-Prog on X
PX31             EQU    0C0A0000Dh       ;Parameter 31 Write-Prog on X
PX32             EQU    0C0A0000Dh       ;Parameter 32 Write-Prog on X
PX33             EQU    0C0A0000Dh       ;Parameter 33 Write-Prog on X
```

```
PX40            EQU    0C0A0000Dh       ;Parameter 40 Write-Prog on X
PX41            EQU    0C0A0000Dh       ;Parameter 41 Write-Prog on X
PX42            EQU    0C0A00005h       ;Parameter 42 Write-Prog on X
PX43            EQU    0C0A0000Dh       ;Parameter 43 Write-Prog on X
PX44            EQU    0C0A0000Dh       ;Parameter 44 Write-Prog on X
PX45            EQU    0C0A0000Dh       ;Parameter 45 Write-Prog on X

PX50            EQU    0C0A0000Dh       ;Parameter 50 Write-Prog on X
PX51            EQU    0C0A0000Dh       ;Parameter 51 Write-Prog on X
PX52            EQU    0C0A0000Dh       ;Parameter 52 Write-Prog on X
PX53            EQU    0C0A0000Dh       ;Parameter 53 Write-Prog on X
PX54            EQU    0C0A0000Dh       ;Parameter 54 Write-Prog on X
PX55            EQU    0C0A0000Dh       ;Parameter 55 Write-Prog on X
PX56            EQU    0C0A00009h       ;Parameter 56 Write-Prog on X

PX62            EQU    0C0A00005h       ;Parameter 62 Write-Prog on X
PX63            EQU    0C0A00005h       ;Parameter 63 Write-Prog on X

; Note: The codes for parameters on axis Y, Z and W are the same as
;       for axis X, only the two first numbers are changed
;
;
;Parameter to write on Axis 'Y'
;-----------------------------
PY01w           EQU     81010001h       ;Parameter 1 Write on Y
PY02w           EQU     81020002h       ;Parameter 2 Write on Y
PY03w           EQU     81030003h       ;Parameter 3 Write on Y
PY04w           EQU     81040001h       ;Parameter 4 Write on Y
...
...
PY63w           EQU     813F0001h       ;Parameter 63 Write on Y

;Parameter to read on Axis 'Y'
;-----------------------------
PY01r           EQU     01010001h       ;Parameter 1 Read on Y
PY02r           EQU     01020002h       ;Parameter 2 Read on Y
PY03r           EQU     01030003h       ;Parameter 3 Read on Y
PY04r           EQU     01040001h       ;Parameter 4 Read on Y
...
...
PY63r           EQU     013F0001h       ;Parameter 63 Read on Y

;Parameter Write to Program for Y
;-------------------------------
PY10            EQU    0C0A1000Dh       ;Parameter 10 Write-Prog on Y
PY11            EQU    0C0A1000Dh       ;Parameter 11 Write-Prog on Y
PY12            EQU    0C0A1000Dh       ;Parameter 12 Write-Prog on Y
PY13            EQU    0C0A10005h       ;Parameter 13 Write-Prog on Y
...
...
PY63            EQU    0C0A10005h       ;Parameter 63 Write-Prog on Y


;Parameter to write on Axis 'Z'
;-----------------------------
PZ01w           EQU     82010001h       ;Parameter 1 Write on Z
PZ02w           EQU     82020002h       ;Parameter 2 Write on Z
PZ03w           EQU     82030003h       ;Parameter 3 Write on Z
PZ04w           EQU     82040001h       ;Parameter 4 Write on Z
...
...
PZ63w           EQU     823F0001h       ;Parameter 63 Write on Z
```

```
;Parameter to read on Axis 'Z'
;-----------------------------
PZ01r           EQU     02010001h       ;Parameter 1 Read on Z
PZ02r           EQU     02020002h       ;Parameter 2 Read on Z
PZ03r           EQU     02030003h       ;Parameter 3 Read on Z
PZ04r           EQU     02040001h       ;Parameter 4 Read on Z
...
...
PZ63r           EQU     023F0001h       ;Parameter 63 Read on Z

;Parameter Write to Program for Z
;--------------------------------
PZ10             EQU    0C0A2000Dh       ;Parameter 10 Write-Prog on Z
PZ11             EQU    0C0A2000Dh       ;Parameter 11 Write-Prog on Z
PZ12             EQU    0C0A2000Dh       ;Parameter 12 Write-Prog on Z
PZ13             EQU    0C0A20005h       ;Parameter 13 Write-Prog on Z
...
...
PZ63             EQU    0C0A20005h       ;Parameter 63 Write-Prog on Z


;Parameter to Write on Axis 'W'
;------------------------------
PW01w           EQU     83010001h       ;Parameter 1 Write on W
PW02w           EQU     83020002h       ;Parameter 2 Write on W
PW03w           EQU     83030003h       ;Parameter 3 Write on W
PW04w           EQU     83040001h       ;Parameter 4 Write on W
...
...
PW63w           EQU     833F0001h       ;Parameter 63 Write on W

;Parameter to read on Axis 'W'
;-----------------------------
PW01r           EQU     03010001h       ;Parameter 1 Read on W
PW02r           EQU     03020002h       ;Parameter 2 Read on W
PW03r           EQU     03030003h       ;Parameter 3 Read on W
PW04r           EQU     03040001h       ;Parameter 4 Read on W
...
...
PW63r           EQU     033F0001h       ;Parameter 63 Read on W

;Parameter Write to Program for W
;--------------------------------
PW10             EQU    0C0A3000Dh       ;Parameter 10 Write-Prog on W
PW11             EQU    0C0A3000Dh       ;Parameter 11 Write-Prog on W
PW12             EQU    0C0A3000Dh       ;Parameter 12 Write-Prog on W
PW13             EQU    0C0A30005h       ;Parameter 13 Write-Prog on W
...
...
PW63             EQU    0C0A30005h       ;Parameter 63 Write-Prog on W
```

Anhang B: Programmbeispiele mit FBs

Beispiel 1

```
;; SAIA PCD SOURCE MODULE - SEDIT V2.0
;; MODULE: E1EX1.SRC
;; DATE:   16.04.97 15:37
;;
            DOC   I 0
            DOC   I 60
            DOC   F 8
            DOC   F 9
            DOC   F 23
            DOC   F 400
            DOC   R 0
            DOC   R 1
            DOC   R 2
            DOC   R 3
            DOC   R 100
            DOC   R 101
            DOC   R 102
            DOC   R 103
            DOC   R 104
            DOC   R 105
            DOC   COB 0
            DOC   XOB 16
            DOC   PB 0
;
;-------------------------------------------------------------------------
;  SAIA-Burgess Electronics AG, CH-3280 Murten,
;  Programm Beispiel für das Modul PCD4.H4xx                    BLOCTEC
;  Bewegung ohne 'blended move'.
;           -------------------
;  Datei:        E1EX1.SRC
;
;  Beschreibung: Dieses Programm besteht aus den folgenden Bewegungen:
;                1.- bewege X vom Referenzpunkt X zu 40mm mit 20mm/s
;                2.- bewege X von der aktuellen Pos. X zu 80mm mit 80mm/s
;                3.- bewege X zurück zum Referenzpunkt
;
;                Dieses Programm ist in BLOCTEC editiert. Zum Start der
;                Bewegung ist der Eingang I 0 zu betätigen. Die voll-
;                ständige Bewegung (Schritte 1 bis 3) wird bei jeder
;                ansteigenden Flanke am Eingang I 0 einmal ausgeführt.
;
;                Die aktuelle Position und die aktuelle Geschwindigkeit
;                kann im Debugger durch die aufgefrischte Anzeige der
;                Register R 100 (aktuelle Position) und R 101 (aktuelle
;                Geschwindigkeit) angezeigt werden.

;                Es wird vorausgesetzt, dass alle Maschinen- und Modul-
;                parameter vorgängig ins PCD4.H4xx-Modul geladen wurden.
;
;
```

```
Anmerkung:    Der FB 'FbStatH4' muss pro Zyklus mindestens einmal auf-
;                gerufen werden, ansonst die Statusflags 'axix-in position'
;                oder 'home procedure executed' nicht aufgefrischt werden.
;
;                - Wird das Signal 'axis in position' (Eingänge 12 .. 14
;                  des der H4-Adressen) am Ende der Bewegung nicht gesetzt,
;                  sind die PID-Parameter (z.B. erhöhen des P-Faktors) oder
;                  der Param. P15 (tolerance axis in position) zu prüfen.
;                - Werden die Flags 'Axis in position' (des FB 'FbStatH4')
;                  nicht gesetzt, sind die Parameter 'Axis No.' dieses FB
;                  dahin zu prüfen, ob die richtige Achse behandelt wird.

;  Revision history:
;   16.04.97    N. JUNG         creation
;-------------------------------------------------------------------------
;
$INCLUDE H4EXTN.DEF
;=======Set general parameters
      XOB   16
;=======Axis init
      CFB   fbInitH4    ;Init H4
            K 48        ;Base Adress Module
            0           ;Base Statusflags
            K 2         ;Moduletype
;=======Set 'ENABLE AXIS X'
      LD    rComH4
            ENAXi       ;Enable axis X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 0
            R 1
            R 2
            R 3
;=======Move axes X to reference point (Limit Switch Reference)
;-------Move axis X to Limit Switch Reference (HOME procedure)
      LD    rComH4
            HomeX       ;Home X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 0
            R 1
            R 2
            R 3
;=======Query axes status and wait for the end of HOME procedure
status: CFB fbStatH4
            K 48
            1           ;axis X
;
      STH   F 23        ;HOME procedure axis X finished?
      JR    L status
      EXOB
;
;
;
```

```
;=======Main program
      COB   0
            0
;
;=======Start motion program
      STH   I 0         ; Start input OK?
      DYN   F 400
      CPB   H 0
      ECOB
;
      PB    0
;------------------------------
;=======Set motion speed at 20mm/s
      LD    R 0
            20000       ;(per default, number of decimals af
      LD    rComH4
            SSXi        ;Instruction Set Speed X
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
;=======Motion 1 : Move axis X to 40mm with 20 mm/sec
;-------Motion axis X
      LD    R 0
            40000
      LD    rComH4
            XAi         ;Instruction Move axis X to absolute
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
;=======Verify Axis X inposition/Start input OK?
Wait1:      CFB   fbStatH4
            K 48
            1
;=======Query actual position axis
      LD    rComH4
            QPX         ;Axis X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 100       ;register for actual position
            R 1
            R 2
            R 3
;=======Query actual velocity axis
      LD    rComH4
            QVX         ;Axis X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 101       ;register for velocity
            R 1
            R 2
            R 3
      STH   I 0
      ANH   F 8         ;On Position flag = 1 when position
      ANL   F 9         ;F 9 = 0 when instruction is execute
      JR    L wait1
```

```
;=======Set motion speed at 80mm/s
      LD    R 0
            80000       ;(per default, number of decimals af
      LD    rComH4
            SSXi        ;Instruction Set Speed X
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
;=======Motion 1 : Move axis X to 80mm with 80 mm/sec
;-------Motion axis X
      LD    R 0
            80000
      LD    rComH4
            XAi         ;Instruction Move axis X to absolute
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
;=======Verify Axis X inposition/Start input OK?
wait2:      CFB   fbStatH4
            K 48
            1
;=======Query actual position axis
      LD    rComH4
            QPX         ;Axis X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 100       ;register for actual position
            R 1
            R 2
            R 3
;=======Query actual velocity axis
      LD    rComH4
            QVX         ;Axis X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 101       ;register for velocity
            R 1
            R 2
            R 3
      STH   I 0
      ANH   F 8         ;On Position flag = 1 when position
      ANL   F 9         ;F 9 = 0 when instruction is execute
      JR    L wait2
;=======Motion 3 : back to the start point with 80mm/s
      LD    R 0
            0
      LD    rComH4
            XAi         ;Instruction Move axis X to absolute
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
```

```
;-------Motion axis X
;=======Verify Axis X inposition/Start input OK?
wait3:      CFB   fbStatH4
            K 48
            1
;=======Query actual position axis
      LD    rComH4
            QPX         ;Axis X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 100       ;register for actual position
            R 1
            R 2
            R 3
;=======Query actual velocity axis
      LD    rComH4
            QVX         ;Axis X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 101       ;register for velocity
            R 1
            R 2
            R 3
      STH   I 0
      ANH   F 8         ;On Position flag = 1 when position
      ANL   F 9         ;F 9 = 0 when instruction is execute
      JR    L wait3
      EPB
```

Notizen

Beispiel 2

```
;; SAIA PCD SOURCE MODULE - SEDIT V2.0
;; MODULE: E1EX1BL.SRC
;; DATE:   16.04.97 15:38
;;
            DOC   I 0
            DOC   I 60
            DOC   F 8
            DOC   F 23
            DOC   F 400
            DOC   R 0
            DOC   R 1
            DOC   R 2
            DOC   R 3
            DOC   R 100
            DOC   R 101
            DOC   R 102
            DOC   R 103
            DOC   R 104
            DOC   R 105
            DOC   COB 0
            DOC   XOB 16
            DOC   PB 0
;
;
;-------------------------------------------------------------------------
;  SAIA-Burgess Electronics AG, CH-3280 Murten,
;  Programm Beispiel für das Modul PCD4.H4xx                    BLOCTEC
;  Bewegung mit 'blended move'.
;           ------------------
;  Datei:        E1EX1BL.SRC
;
;  Beschreibung: Dieses Programm besteht aus den folgenden Bewegungen:
;                1.- bewege X vom Referenzpunkt X zu 40mm mit 20mm/s
;                2.- bewege X von der aktuellen Pos. X zu 80mm mit 80mm/s
;                3.- bewege X zurück zum Referenzpunkt
;
;                Dieses Programm ist in BLOCTEC editiert. Zum Start der
;                Bewegung ist der Eingang I 0 zu betätigen. Die voll-
;                ständige Bewegung (Schritte 1 bis 3) wird bei jeder
;                ansteigenden Flanke am Eingang I 0 einmal ausgeführt.
;
;                Die aktuelle Position und die aktuelle Geschwindigkeit
;                kann im Debugger durch die aufgefrischte Anzeige der
;                Register R 100 (aktuelle Position) und R 101 (aktuelle
;                Geschwindigkeit) angezeigt werden.

;                Es wird vorausgesetzt, dass alle Maschinen- und Modul-
;                parameter vorgängig ins PCD4.H4xx-Modul geladen wurden.
;
;  Anmerkung:    Der FB 'FbStatH4' muss pro Zyklus mindestens einmal auf-
;                gerufen werden, ansonst die Statusflags 'axix-in position'
;                oder 'home procedure executed' nicht aufgefrischt werden.
;
;                - Wird das Signal 'axis in position' (Eingänge 12 .. 14
;                  des der H4-Adressen) am Ende der Bewegung nicht gesetzt,
;                  sind die PID-Parameter (z.B. erhöhen des P-Faktors) oder
;                  der Param. P15 (tolerance axis in position) zu prüfen.
;
;                - Werden die Flags 'Axis in position' (des FB 'FbStatH4')
;                  nicht gesetzt, sind die Parameter 'Axis No.' dieses FB
;                  dahin zu prüfen, ob die richtige Achse behandelt wird.
;
```

```
;  Revision history:
;   16.04.97    N. JUNG         creation
;-------------------------------------------------------------------------
;
$INCLUDE H4EXTN.DEF
;=======Set general parameters
      XOB   16
;=======Axis init
      CFB   fbInitH4    ;Init H4
            K 48        ;Base Adress Module
            0           ;Base Statusflags
            K 2         ;Moduletype
;=======Set 'ENABLE AXIS X'
      LD    rComH4
            ENAXi       ;Enable axis X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 0
            R 1
            R 2
            R 3
;=======Move axes X to reference point (Limit Switch Reference)
;-------Move axis X to Limit Switch Reference (HOME procedure)
      LD    rComH4
            HomeX       ;Home X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 0
            R 1
            R 2
            R 3
;=======Query axes status and wait for the end of HOME procedure
status: CFB fbStatH4
            K 48
            1           ;axis X
;
      STH   F 23        ;HOME procedure axis X finished?
      JR    L status
      EXOB
;
;
;
;=======Main program
      COB   0
            0
;=======Query actual position axis
      LD    rComH4
            QPX         ;Axis X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 100       ;register for actual position
            R 1
            R 2
            R 3
```

```
;=======Query actual velocity axis
      LD    rComH4
            QVX         ;Axis X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 101       ;register for velocity
            R 1
            R 2
            R 3
;=======Start motion program
      STH   I 0         ; Start input OK?
      DYN   F 400
      CPB   H 0
      ECOB
;
      PB    0
;------------------------------
;=======Set motion speed at 20mm/s
      LD    R 0
            20000       ;(per default, number of decimals af
      LD    rComH4
            SSXi        ;Instruction Set Speed X
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
;=======Motion 1 : Move axis X to 40mm with 20 mm/sec
;-------Motion axis X
      LD    R 0
            40000
      LD    rComH4
            XAi         ;Instruction Move axis X to absolute
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
;=======Set motion speed at 80mm/s
      LD    R 0
            80000       ;(per default, number of decimals af
      LD    rComH4
            SSXi        ;Instruction Set Speed X
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
```

```
;=======Motion 1 : Move axis X to 80mm with 80 mm/sec
;-------Motion axis X
      LD    R 0
            80000
      LD    rComH4
            XAi         ;Instruction Move axis X to absolute
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
;=======Motion 3 : back to the start point with 80mm/s
      LD    R 0
            0
      LD    rComH4
            XAi         ;Instruction Move axis X to absolute
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
      EPB
```

Beispiel 3

```
;; SAIA PCD SOURCE MODULE - SEDIT V2.0
;; MODULE: E1EX2G.SRC
;; DATE:   16.04.97 15:38
;;
            DOC   I 0
            DOC   F 8
            DOC   F 9
            DOC   F 23
            DOC   F 400
            DOC   R 0
            DOC   R 1
            DOC   R 2
            DOC   R 3
            DOC   R 100
            DOC   R 101
            DOC   R 102
            DOC   R 103
            DOC   R 104
            DOC   R 105
            DOC   T 0
            DOC   COB 0
            DOC   XOB 16
;
;-------------------------------------------------------------------------
;  SAIA-Burgess Electronics AG, CH-3280 Murten,
;  Programm Beispiel für das Modul PCD4.H4xx                    GRAFTEC
;  Bewegung ohne 'blended move'.
;           -------------------
;  File:          E1EX2G.SRC
;
;  Beschreibung: Dieses Programm besteht aus den folgenden Bewegungen:
;                1.- bewege X vom Referenzpunkt X zu 40mm mit 20mm/s
;                2.- bewege X von der aktuellen Pos. X zu 80mm mit 80mm/s
;                3.- bewege X zurück zum Referenzpunkt
;
;                Dieses Programm ist in GRAFTEC editiert. Zum Start der
;                Bewegung ist der Eingang I 0 zu betätigen. Die voll-
;                ständige Bewegung (Schritte 1 bis 3) wird bei jeder
;                ansteigenden Flanke am Eingang I 0 einmal ausgeführt.
;
;                Die aktuelle Position und die aktuelle Geschwindigkeit
;                kann im Debugger durch die aufgefrischte Anzeige der
;                Register R 100 (aktuelle Position) und R 101 (aktuelle
;                Geschwindigkeit) angezeigt werden.

;                Es wird vorausgesetzt, dass alle Maschinen- und Modul-
;                parameter vorgängig ins PCD4.H4xx-Modul geladen wurden.
;
;  Anmerkung:    Der FB 'FbStatH4' muss pro Zyklus mindestens einmal auf-
;                gerufen werden, ansonst die Statusflags 'axix-in position'
;                oder 'home procedure executed' nicht aufgefrischt werden.
;
;                - Wird das Signal 'axis in position' (Eingänge 12 .. 14
;                  des der H4-Adressen) am Ende der Bewegung nicht gesetzt,
;                  sind die PID-Parameter (z.B. erhöhen des P-Faktors) oder
;                  der Param. P15 (tolerance axis in position) zu prüfen.
;
;                - Werden die Flags 'Axis in position' (des FB 'FbStatH4')
;                  nicht gesetzt, sind die Parameter 'Axis No.' dieses FB
;                  dahin zu prüfen, ob die richtige Achse behandelt wird.
```

```
;  Revision history:
;   16.04.97    N. JUNG         creation
;-------------------------------------------------------------------------
;
$INCLUDE H4EXTN.DEF
;=======Set general parameters
      XOB   16
;=======Axis init
      CFB   fbInitH4    ;Init H4
            K 48        ;Base Adress Module
            0           ;Base Statusflags
            K 2         ;Moduletype
;=======Set 'ENABLE AXIS X'
      LD    rComH4
            ENAXi       ;Enable axis X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 0
            R 1
            R 2
            R 3
;=======Move axes X to reference point (Limit Switch Reference)
;-------Move axis X to Limit Switch Reference (HOME procedure)
      LD    rComH4
            HomeX       ;Home X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 0
            R 1
            R 2
            R 3
;=======Query axes status and wait for the end of HOME procedure
status: CFB fbStatH4
            K 48
            1           ;axis X
;
      STH   F 23        ;HOME procedure finished?
      JR    L status
      EXOB
;
;
;
;=======Main program
      COB   0
            0
;
;=======Refresh axis status
      CFB   fbStatH4
            K 48
            1           ;axis X
;=======Query actual position axis
      LD    rComH4
            QPX         ;Axis X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 100       ;register for actual position
            R 1
            R 2
            R 3
```

```
;=======Query actual velocity axis
      LD    rComH4
            QVX         ;Axis X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 101       ;register for velocity
            R 1
            R 2
            R 3
;=======Start motion program
      CSB   0
      ECOB
;
      SB    0
;------------------------------
      IST   0           ;Rectilinear motion
            O 0         ;I0 = 1?
      EST               ;0
;------------------------------
      ST    1           ;Motion 1
            I 0         ;I0 = 1?
            I 6         ;T=0 & I0 = 1?
            O 1         ;on Position ?
;=======Set motion speed at 20mm/s
      LD    R 0
            20000       ;(per default, number of decimals af
      LD    rComH4
            SSXi        ;Instruction Set Speed X
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
;=======Motion 1 : Move axis X to 40mm with 20 mm/sec
;-------Motion axis X
      LD    R 0
            40000
      LD    rComH4
            XAi         ;Instruction Move axis X to absolute
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
      EST               ;1
;------------------------------
      ST    2           ;Pause 1 sec
            I 1         ;on Position ?
            O 2         ;T=0 & I0 = 1?
      LD    T 0
            10
      EST               ;2
;------------------------------
      ST    3           ;Motion 2
            I 2         ;T=0 & I0 = 1?
            O 3         ;on Position ?
```

```
;=======Set motion speed at 80mm/s
      LD    R 0
            80000       ;(per default, number of decimals af
      LD    rComH4
            SSXi        ;Instruction Set Speed X
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
;=======Motion 2 : Move axis X to 80mm with 80 mm/sec
;-------Motion axis X
      LD    R 0
            80000
      LD    rComH4
            XAi         ;Instruction Move axis X to absolute
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
      EST               ;3
;------------------------------
      ST    4           ;Pause 1 sec
            I 3         ;on Position ?
            O 4         ;T=0 & I0 = 1?
      LD    T 0
            10
      EST               ;4
;------------------------------
      ST    5           ;Motion 3
            I 4         ;T=0 & I0 = 1?
            O 5         ;on Position ?
;=======Motion 3 : back to the start point with 80mm/s
      LD    R 0
            0
      LD    rComH4
            XAi         ;Instruction Move axis X to absolute
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
      EST               ;5
;------------------------------
      ST    6           ;Pause 1 sec
            I 5         ;on Position ?
            O 6         ;T=0 & I0 = 1?
      LD    T 0
            10
      EST               ;6
;------------------------------
      TR    0           ;I0 = 1?
            I 0         ;Rectilinear motion
            O 1         ;Motion 1
      STH   I 0
      ETR               ;0
```

```
;------------------------------
      TR    1           ;on Position ?
            I 1         ;Motion 1
            O 2         ;Pause 1 sec
      STH   F 8         ;On Position flag = 1 when position
      ANL   F 9         ;F 9 = 0 when instruction is execute
      ETR               ;1
;------------------------------
      TR    2           ;T=0 & I0 = 1?
            I 2         ;Pause 1 sec
            O 3         ;Motion 2
      STL   T 0
      ANH   I 0
      ETR               ;2
;------------------------------
      TR    3           ;on Position ?
            I 3         ;Motion 2
            O 4         ;Pause 1 sec
      STH   F 8         ;On Position flag = 1 when position
      ANL   F 9         ;F 9 = 0 when instruction is execute
      ETR               ;3
;------------------------------
      TR    4           ;T=0 & I0 = 1?
            I 4         ;Pause 1 sec
            O 5         ;Motion 3
      STL   T 0
      ANH   I 0
      ETR               ;4
;------------------------------
      TR    5           ;on Position ?
            I 5         ;Motion 3
            O 6         ;Pause 1 sec
      STH   F 8         ;On Position flag = 1 when position
      ANL   F 9         ;F 9 = 0 when instruction is execute
      ETR               ;5
;------------------------------
      TR    6           ;T=0 & I0 = 1?
            I 6         ;Pause 1 sec
            O 1         ;Motion 1
      STL   T 0
      ANH   I 0
      ETR               ;6
;
      ESB               ;0
```

Notizen

Beispiel 4

```
;; SAIA PCD SOURCE MODULE - SEDIT V2.0
;; MODULE: E1EX2GB.SRC
;; DATE:   16.04.97 15:38
;;
            DOC   I 0
            DOC   F 8
            DOC   F 9
            DOC   F 23
            DOC   F 400
            DOC   R 0
            DOC   R 1
            DOC   R 2
            DOC   R 3
            DOC   R 100
            DOC   R 101
            DOC   R 102
            DOC   R 103
            DOC   R 104
            DOC   R 105
            DOC   T 0
            DOC   COB 0
            DOC   XOB 16
;
;-------------------------------------------------------------------------
;  SAIA-Burgess Electronics AG, CH-3280 Murten,
;  Programm Beispiel für das Modul PCD4.H4xx                    GRAFTEC
;  Bewegung mit 'blended move'.
;            ------------------
;  File:          E1EX2GB.SRC
;
;  Beschreibung: Dieses Programm besteht aus den folgenden Bewegungen:
;                1.- bewege X vom Referenzpunkt X zu 40mm mit 20mm/s
;                2.- bewege X von der aktuellen Pos. X zu 80mm mit 80mm/s
;                3.- bewege X zurück zum Referenzpunkt
;
;                Dieses Programm ist in GRAFTEC editiert. Zum Start der
;                Bewegung ist der Eingang I 0 zu betätigen. Die voll-
;                ständige Bewegung (Schritte 1 bis 3) wird bei jeder
;                ansteigenden Flanke am Eingang I 0 einmal ausgeführt.
;
;                Die aktuelle Position und die aktuelle Geschwindigkeit
;                kann im Debugger durch die aufgefrischte Anzeige der
;                Register R 100 (aktuelle Position) und R 101 (aktuelle
;                Geschwindigkeit) angezeigt werden.

;                Es wird vorausgesetzt, dass alle Maschinen- und Modul-
;                parameter vorgängig ins PCD4.H4xx-Modul geladen wurden.
;
;  Anmerkung:    Der FB 'FbStatH4' muss pro Zyklus mindestens einmal auf-
;                gerufen werden, ansonst die Statusflags 'axix-in position'
;                oder 'home procedure executed' nicht aufgefrischt werden.
;
;                - Wird das Signal 'axis in position' (Eingänge 12 .. 14
;                  des der H4-Adressen) am Ende der Bewegung nicht gesetzt,
;                  sind die PID-Parameter (z.B. erhöhen des P-Faktors) oder
;                  der Param. P15 (tolerance axis in position) zu prüfen.
;
;                - Werden die Flags 'Axis in position' (des FB 'FbStatH4')
;                  nicht gesetzt, sind die Parameter 'Axis No.' dieses FB
;                  dahin zu prüfen, ob die richtige Achse behandelt wird.
;
```

```
;  Revision history:
;   16.04.97    N. JUNG         creation
;-------------------------------------------------------------------------
;
$INCLUDE H4EXTN.DEF
;=======Set general parameters
      XOB   16
;=======Axis init
      CFB   fbInitH4    ;Init H4
            K 48        ;Base Adress Module
            0           ;Base Statusflags
            K 2         ;Moduletype
;=======Set 'ENABLE AXIS X'
      LD    rComH4
            ENAXi       ;Enable axis X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 0
            R 1
            R 2
            R 3
;=======Move axes X to reference point (Limit Switch Reference)
;-------Move axis X to Limit Switch Reference (HOME procedure)
      LD    rComH4
            HomeX       ;Home X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 0
            R 1
            R 2
            R 3
;=======Query axes status and wait for the end of HOME procedure
status: CFB fbStatH4
            K 48
            1           ;axis X
;
      STH   F 23        ;HOME procedure finished?
      JR    L status
      EXOB
;
;
;
;=======Main program
      COB   0
            0
;=======Query actual position axis
      LD    rComH4
            QPX         ;Axis X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 100       ;register for actual position
            R 1
            R 2
            R 3
```

```
;=======Query actual velocity axis
      LD    rComH4
            QVX         ;Axis X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 101       ;register for velocity
            R 1
            R 2
            R 3
;=======Refresh axis status
      CFB   fbStatH4
            K 48
            1
;=======Start motion program
      CSB   0
      ECOB
;
      SB    0
;------------------------------
      IST   0           ;Rectilinear motion
            I 1         ;on Position ?
            O 0         ;I0 = 1?
      EST               ;0
;------------------------------
      ST    1           ;Motion 1
            I 0         ;I0 = 1?
            O 1         ;on Position ?
;=======Set motion speed at 20mm/s
      LD    R 0
            20000       ;(per default, number of decimals af
      LD    rComH4
            SSXi        ;Instruction Set Speed X
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
;=======Motion 1 : Move axis X to 40mm with 20 mm/sec
;-------Motion axis X
      LD    R 0
            40000
      LD    rComH4
            XAi         ;Instruction Move axis X to absolute
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
;=======Set motion speed at 80mm/s
      LD    R 0
            80000       ;(per default, number of decimals af
      LD    rComH4
            SSXi        ;Instruction Set Speed X
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
```

```
;=======Motion 2 : Move axis X to 80mm with 80 mm/sec
;-------Motion axis X
      LD    R 0
            80000
      LD    rComH4
            XAi         ;Instruction Move axis X to absolute
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
;=======Motion 3 : back to the start point with 80mm/s
      LD    R 0
            0
      LD    rComH4
            XAi         ;Instruction Move axis X to absolute
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
      EST               ;1
;------------------------------
      TR    0           ;I0 = 1?
            I 0         ;Rectilinear motion
            O 1         ;Motion 1
      STH   I 0
      ETR               ;0
;------------------------------
      TR    1           ;on Position ?
            I 1         ;Motion 1
            O 0         ;Rectilinear motion
      STH   F 8         ;On Position flag = 1 when position
      ANL   F 9         ;F 9 = 0 when instruction is execute
      ETR               ;1
;
      ESB               ;0
```

Beispiel 5

```
;; SAIA PCD SOURCE MODULE - SEDIT V2.0
;; MODULE: E3EX3GB.SRC
;; DATE:   16.04.97 15:38
;;
            DOC   I 0
            DOC   F 8
            DOC   F 9
            DOC   F 23
            DOC   F 24
            DOC   F 25
            DOC   F 39
            DOC   F 400
            DOC   R 0
            DOC   R 1
            DOC   R 2
            DOC   R 3
            DOC   R 100
            DOC   R 101
            DOC   R 102
            DOC   R 103
            DOC   R 104
            DOC   R 105
            DOC   T 0
            DOC   COB 0
            DOC   XOB 16
            DOC   PB 0
;
;-------------------------------------------------------------------------
;  SAIA-Burgess Electronics AG, CH-3280 Murten,
;  Programm Beispiel für das Modul PCD4.H4xx                    GRAFTEC
;  Bewegung mit 'blended move'.
;           ------------------
;  File:         E3EX3GB.SRC
;
;  Beschreibung: Dieses Programm besteht aus den folgenden Bewegungen:
;                1.- bewege X, Y vom Ref.punkt X=40mm, Y=40mm mit 20mm/s
;                2.- bewege X, Y zu X=80mm, Y=80mm mit 80mm/s
;                3.- bewege X zurück zum Referenzpunkt
;
;                Dieses Programm ist in GRAFTEC editiert. Zum Start der
;                Bewegung ist der Eingang I 0 zu betätigen. Die voll-
;                ständige Bewegung (Schritte 1 bis 3) wird bei jeder
;                ansteigenden Flanke am Eingang I 0 einmal ausgeführt.
;
;                Die aktuelle Position und die aktuelle Geschwindigkeit
;                können im Debugger durch die aufgefrischte Anzeige der
;                folgenden Register eingesehen werden:
;                - X-Achse: R 100 (aktuelle Pos.), R 101 (aktuelle Geschw.)
;                - Y-Achse: R 102 (aktuelle Pos.), R 103 (aktuelle Geschw.)
;
;                Es wird vorausgesetzt, dass alle Maschinen- und Modul-
;                parameter vorgängig ins PCD4.H4xx-Modul geladen wurden.
;
;  Anmerkung:    Der FB 'FbStatH4' muss pro Zyklus mindestens einmal auf-
;                gerufen werden, ansonst die Statusflags 'axix-in position'
;                oder 'home procedure executed' nicht aufgefrischt werden.
;
```

```
;                - Wird das Signal 'axis in position' (Eingänge 12 .. 14
;                  des der H4-Adressen) am Ende der Bewegung nicht gesetzt,
;                  sind die PID-Parameter (z.B. erhöhen des P-Faktors) oder
;                  der Param. P15 (tolerance axis in position) zu prüfen.
;
;                - Werden die Flags 'Axis in position' (des FB 'FbStatH4')
;                  nicht gesetzt, sind die Parameter 'Axis No.' dieses FB
;                  dahin zu prüfen, ob die richtige Achse behandelt wird.
;
;  Revision history:
;   16.04.97    N. JUNG         creation
;-------------------------------------------------------------------------
$INCLUDE H4EXTN.DEF
;
;=======Set general parameters
      XOB   16
;=======Axis init
      CFB   fbInitH4    ;Init H4
            K 48        ;Base Adress Module
            0           ;Base Statusflags
            K 2         ;Moduletype
;=======Set 'ENABLE AXIS X'
      LD    rComH4
            ENAXi       ;Enable axis X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 0
            R 1
            R 2
            R 3
;=======Set 'ENABLE AXIS Y'
      LD    rComH4
            ENAYi       ;Enable axis Y
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 0
            R 1
            R 2
            R 3
;=======Move axes X to reference point (Limit Switch Reference)
;-------Move axis X to Limit Switch Reference (HOME procedure)
      LD    rComH4
            HomeX       ;Home X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 0
            R 1
            R 2
            R 3
;=======Move axes Y to reference point (Limit Switch Reference)
;-------Move axis Y to Limit Switch Reference (HOME procedure)
      LD    rComH4
            HomeY       ;Home Y
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 0
            R 1
            R 2
            R 3
```

```
;=======Query axes status and wait for the end of HOME procedure
status: CFB fbStatH4
            K 48
            00000000FH  ;all available axes in one cycle
;
      STH   F 23        ;HOME procedure axis X finished?
      JR    L status
      STH   F 39        ;HOME procedure axis Y finished?
      JR    L status
      EXOB
;
;
;
;=======Main program
      COB   0
            0
;=======Refresh axis status
      CFB   fbStatH4
            K 48
            00000000FH  ;all available axes in one cycle
;=======Query actual position axis X
      LD    rComH4
            QPX         ;Axis X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 100       ;register for actual position
            R 1
            R 2
            R 3
;=======Query actual velocity axis X
      LD    rComH4
            QVX         ;Axis X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 101       ;register for velocity
            R 1
            R 2
            R 3
;=======Query actual position axis Y
      LD    rComH4
            QPY         ;Axis Y
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 102       ;register for actual position
            R 1
            R 2
            R 3
;=======Query actual velocity axis Y
      LD    rComH4
            QVY         ;Axis Y
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 103       ;register for velocity
            R 1
            R 2
            R 3
;=======Start motion program
      CSB   0
      ECOB
;
```

```
      SB    0
;------------------------------
      IST   0           ;Rectilinear motion
            I 1         ;on Position ?
            O 0         ;I0 = 1?
      EST               ;0
;------------------------------
      ST    1           ;Motion 1
            I 0         ;I0 = 1?
            O 1         ;on Position ?
;=======Set Vector motion speed at 20mm/s
      LD    R 0
            20000       ;(per default, number of decimals af
      LD    rComH4
            SVi         ;Instruction Set Vector Speed
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
;=======Motion 1 : Move axes X,Y to X=40mm, Y=40mm with 20 mm/sec
      LD    R 0
            40000
      LD    R 1
            40000
      LD    rComH4
            XYAi        ;Instr. Move axes X,Y (linear interp
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            R 1         ;Parameter
            NotUsed
            NotUsed
;=======Set Vector motion speed at 80mm/s
      LD    R 0
            80000       ;(per default, number of decimals af
      LD    rComH4
            SVi         ;Instruction Set Vector Speed
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
;=======Motion 2 : Move axes X,Y to X=80mm, Y=80mm with 80 mm/sec
      LD    R 0
            80000
      LD    R 1
            80000
      LD    rComH4
            XYAi        ;Instr. Move axes X,Y (linear interp
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            R 1         ;Parameter
            NotUsed
            NotUsed
```

```
;=======Motion 3 : back to the start point with 80mm/s
      LD    R 0
            0
      LD    R 1
            0
      LD    rComH4
            XYAi        ;Instr. Move axes X,Y (linear interp
      CFB   fbExeH4     ;Execute Command
            K 48        ;Base address H4
            R 0         ;Parameter
            R 1         ;Parameter
            NotUsed
            NotUsed
      EST               ;1
;------------------------------
      TR    0           ;I0 = 1?
            I 0         ;Rectilinear motion
            O 1         ;Motion 1
      STH   I 0
      ETR               ;0
;------------------------------
      TR    1           ;on Position ?
            I 1         ;Motion 1
            O 0         ;Rectilinear motion
      STH   F 8         ;F 8 = 1 when position axis X is rea
      ANH   F 24        ;F24 = 1 when position axis Y is rea
      ANL   F 9         ;F 9 = 0 when instruction XAp is exe
      ANL   F 25        ;F25 = 0 when instruction YAp is exe
      ETR               ;1
;
      ESB               ;0
```

Notizen

Beispiel 6

```
;; SAIA PCD SOURCE MODULE - SEDIT V2.0
;; MODULE: OPENPR1.SRC
;; DATE:   16.04.97 15:38
;;
            DOC   I 0
            DOC   F 8
            DOC   F 23
            DOC   F 39
            DOC   F 400
            DOC   R 0
            DOC   R 1
            DOC   R 2
            DOC   R 3
            DOC   R 100
            DOC   R 101
            DOC   R 102
            DOC   R 103
            DOC   R 104
            DOC   R 105
            DOC   T 0
            DOC   COB 0
            DOC   XOB 16
            DOC   PB 0
;
;-------------------------------------------------------------------------
;  SAIA-Burgess Electronics AG, CH-3280 Murten,
;  Programm Beispiel für das Modul PCD4.H4xx           OPEN/CLOSE Progr.
;  Bewegung mit 'blended move'.
;            ------------------
;  File:         OPENPR1.SRC
;
;  Beschreibung: Dieses Programm ist mit den OPEN/CLOSE-Befehlen editiert.
;                Dies bedeutet, dass das ganze Bewegungs-Programm zusammen-
;                gesetzt ist und nur einmal ins PCD4.H4xx-Modul übertragen
;                wird. Das geladene Programm wird mittels dem Befehl RUNp
;                (run program number) von der CPU, z.B. in einem COB
;                ausgeführt.
;
;                Dieses Programm besteht aus den folgenden Bewegungen:
;                1.- bewege X, Y vom Ref.punkt X=40mm, Y=40mm mit 20mm/s
;                2.- bewege X, Y zu X=80mm, Y=80mm mit 80mm/s
;                3.- bewege X zurück zum Referenzpunkt
;
;                Dieses Programm ist in BLOCTEC editiert. Zum Start der
;                Bewegung ist der Eingang I 0 zu betätigen. Die voll-
;                ständige Bewegung (Schritte 1 bis 3) wird bei jeder
;                ansteigenden Flanke am Eingang I 0 einmal ausgeführt.
;
;                Die aktuelle Position und die aktuelle Geschwindigkeit
;                können im Debugger durch die aufgefrischte Anzeige der
;                folgenden Register eingesehen werden:
;                - X-Achse: R 100 (aktuelle Pos.), R 101 (aktuelle Geschw.)
;                - Y-Achse: R 102 (aktuelle Pos.), R 103 (aktuelle Geschw.)
;
;                Es wird vorausgesetzt, dass alle Maschinen- und Modul-
;                parameter vorgängig ins PCD4.H4xx-Modul geladen wurden.
;
;  Anmerkung:    Der FB 'FbStatH4' muss pro Zyklus mindestens einmal auf-
;                gerufen werden, ansonst die Statusflags 'axix-in position'
;                oder 'home procedure executed' nicht aufgefrischt werden.
;
```

```
;                - Wird das Signal 'axis in position' (Eingänge 12 .. 14
;                  des der H4-Adressen) am Ende der Bewegung nicht gesetzt,
;                  sind die PID-Parameter (z.B. erhöhen des P-Faktors) oder
;                  der Param. P15 (tolerance axis in position) zu prüfen.
;
;                - Werden die Flags 'Axis in position' (des FB 'FbStatH4')
;                  nicht gesetzt, sind die Parameter 'Axis No.' dieses FB
;                  dahin zu prüfen, ob die richtige Achse behandelt wird.

;  Revision history:
;   16.04.97    N. JUNG         creation
;-------------------------------------------------------------------------
;
$INCLUDE H4EXTN.DEF
;=======Set general parameters
      XOB   16
;=======Axis init
      CFB   fbInitH4    ;Init H4
            K 48        ;Base Adress Module
            0           ;Base Statusflags
            K 2         ;Moduletype
;=======Set 'ENABLE AXIS X'
      LD    rComH4
            ENAXi       ;Enable axis X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 0
            R 1
            R 2
            R 3
;=======Set 'ENABLE AXIS Y'
      LD    rComH4
            ENAYi       ;Enable axis Y
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 0
            R 1
            R 2
            R 3
;=======Move axes X to reference point (Limit Switch Reference)
;-------Move axis X to Limit Switch Reference (HOME procedure)
      LD    rComH4
            HomeX       ;Home X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 0
            R 1
            R 2
            R 3
;=======Move axes Y to reference point (Limit Switch Reference)
;-------Move axis Y to Limit Switch Reference (HOME procedure)
      LD    rComH4
            HomeY       ;Home Y
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 0
            R 1
            R 2
            R 3
```

```
;=======Query axes status and wait for the end of HOME procedure
status: CFB fbStatH4
            K 48
            00000000FH  ;all available axes in one cycle
;
      STH   F 23        ;HOME procedure axis X finished?
      JR    L status
      STH   F 39        ;HOME procedure axis Y finished?
      JR    L status
;=======Motion program
;-------Open program
      LD    R 1
            1
      LD    rComH4
            OPEN1       ;OPEN Program 1
      CFB   fbExeH4     ;Basisadress H4
            K 48        ;Parameter
            R 1         ;Parameter
            NotUsed
            NotUsed
            NotUsed
;------------------------------
;=======Set Vector motion speed at 20mm/s
      LD    R 0
            20000
      LD    rComH4
            SVp         ;Set Vector Speed
      CFB   fbExeH4     ;Execute Command
            K 48        ;Basisadress H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
;=======Motion 1 : Move axes X,Y to X=40mm, Y=40mm with 20 mm/sec
;-------Motion axis X,Y (XYAp)
      LD    R 0
            40000
      LD    R 1
            40000
      LD    rComH4
            XYAp        ;Move absolute axes X,Y
      CFB   fbExeH4     ;Execute Command
            K 48        ;Basisadress H4
            R 0         ;Parameter
            R 1         ;Parameter
            NotUsed
            NotUsed
;------Wait loop-----------------
      LD    R 0
            1000
      LD    rComH4
            WAIT        ;Wait for 1000 ms
      CFB   fbExeH4     ;Execute Command
            K 48        ;Basisadress H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
```

```
;=======Set Vector motion speed at 80mm/s
      LD    R 0
            80000
      LD    rComH4
            SVp         ;Set Vector Speed
      CFB   fbExeH4     ;Execute Command
            K 48        ;Basisadress H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
;=======Motion 2 : Move axes X,Y to X=80mm, Y=80mm with 80 mm/sec
;-------Motion axis X,Y (XYAp)
      LD    R 0
            80000
      LD    R 1
            80000
      LD    rComH4
            XYAp        ;Move absolute axes X,Y
      CFB   fbExeH4     ;Execute Command
            K 48        ;Basisadress H4
            R 0         ;Parameter
            R 1         ;Parameter
            NotUsed
            NotUsed
;------Wait loop-----------------
      LD    R 0
            1000
      LD    rComH4
            WAIT        ;Wait for 1000 ms
      CFB   fbExeH4     ;Execute Command
            K 48        ;Basisadress H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
;=======Motion 3 : back to the start point with 80mm/s
;-------Motion axis X,Y(XYAp)
      LD    R 0
            0
      LD    R 1
            0
      LD    rComH4
            XYAp        ;Move absolute axes X,Y
      CFB   fbExeH4     ;Execute Command
            K 48        ;Basisadress H4
            R 0         ;Parameter
            R 1         ;Parameter
            NotUsed
            NotUsed
;------Wait loop-----------------
      LD    R 0
            1000
      LD    rComH4
            WAIT        ;Wait for 1000 ms
      CFB   fbExeH4     ;Execute Command
            K 48        ;Basisadress H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
```

```
;-------Program END and CLOSE-------------
      LD    rComH4
            END         ;Move absolute axis X
      CFB   fbExeH4     ;Execute Command
            K 48        ;Basisadress H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
      LD    rComH4
            CLOSE       ;Move absolute axis X
      CFB   fbExeH4     ;Execute Command
            K 48        ;Basisadress H4
            R 0         ;Parameter
            NotUsed
            NotUsed
            NotUsed
      EXOB
;=======Main program
      COB   0
            0
;
;=======Refresh axis status
      CFB   fbStatH4
            K 48
            00000000FH  ;all available axes in one cycle
;=======Query actual position axis X
      LD    rComH4
            QPX         ;Axis X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 100       ;register for actual position
            R 1
            R 2
            R 3
;=======Query actual velocity axis X
      LD    rComH4
            QVX         ;Axis X
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 101       ;register for velocity
            R 1
            R 2
            R 3
;=======Query actual position axis Y
      LD    rComH4
            QPY         ;Axis Y
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 102       ;register for actual position
            R 1
            R 2
            R 3
```

```
;=======Query actual velocity axis Y
      LD    rComH4
            QVY         ;Axis Y
;
      CFB   fbExeH4     ;Execute Command
            K 48
            R 103       ;register for velocity
            R 1
            R 2
            R 3
;=======Start motion program
      LD    R 0
            1           ;Set program number = 1
      LD    rComH4
            RUNi        ;RUN Program 1
;
      STH   I 0         ;If Input I0=1
      DYN   F 400
      CFB   H fbExeH4   ;Execute Command
            K 48        ;Basisadress H4
            R 0         ;Parameter
            R 1         ;Parameter
            R 2         ;Parameter
            R 3         ;Parameter
      ECOB
```

Absender:

Firma
Abteilung
Name
Adresse

Tel.

Datum

An:

SAIA-Burgess Electronics AG
Bahnhofstrasse 18
CH-3280 Murten (Schweiz)
http://www.saia-burgess.com

GB: Electronic Controllers

Handbuch PCD4.H4x0
Positioniermodul für Servoantriebe
mit Linear-und Kreisinterpolation

Falls Sie Vorschläge zu SAIA® PCD zu machen oder Fehler in diesem Handbuch gefunden haben, sind wir Ihnen für einen kurzen Bericht dankbar.

Ihre Vorschläge: