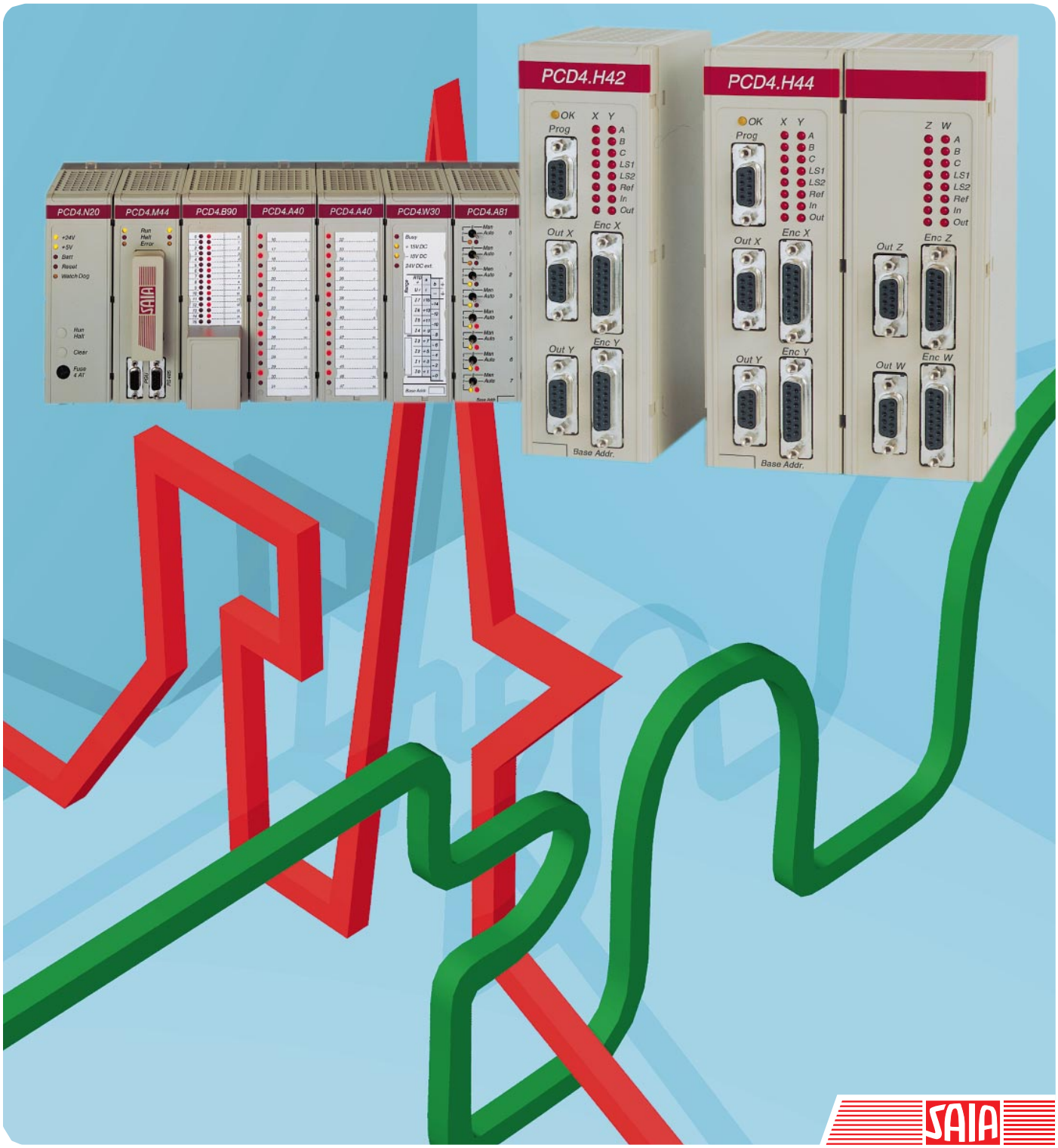
SAIA-Burgess Electronics

SWITCHES • MOTORS • CONTROLLERS

SAIA®PCD

Process Control Devices

Motion control modules for servo drives with linear and circular interpolation PCD4.H4..



Edition 26/752 E1

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

BA: Electronic Controllers Telephone 026 / 672 72 72
Telefax 026 / 672 74 99

SAIA-Burgess Companies

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

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

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

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

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

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

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

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

Representatives

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

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

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

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

Suomi/ Finland 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

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

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

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

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

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

After sales service

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

Motion control modules for servo drives with linear and circular interpolation

PCD4.H4x0

Edition 26/752 E1 - 04.1997

Subject to technical changes

Updates

Manual : PCD4.H4x0 - Motion control modules for servo drives with linear and circular interpolation - Edition E1

Date	Chapter	Page	Description

Contents

Notes

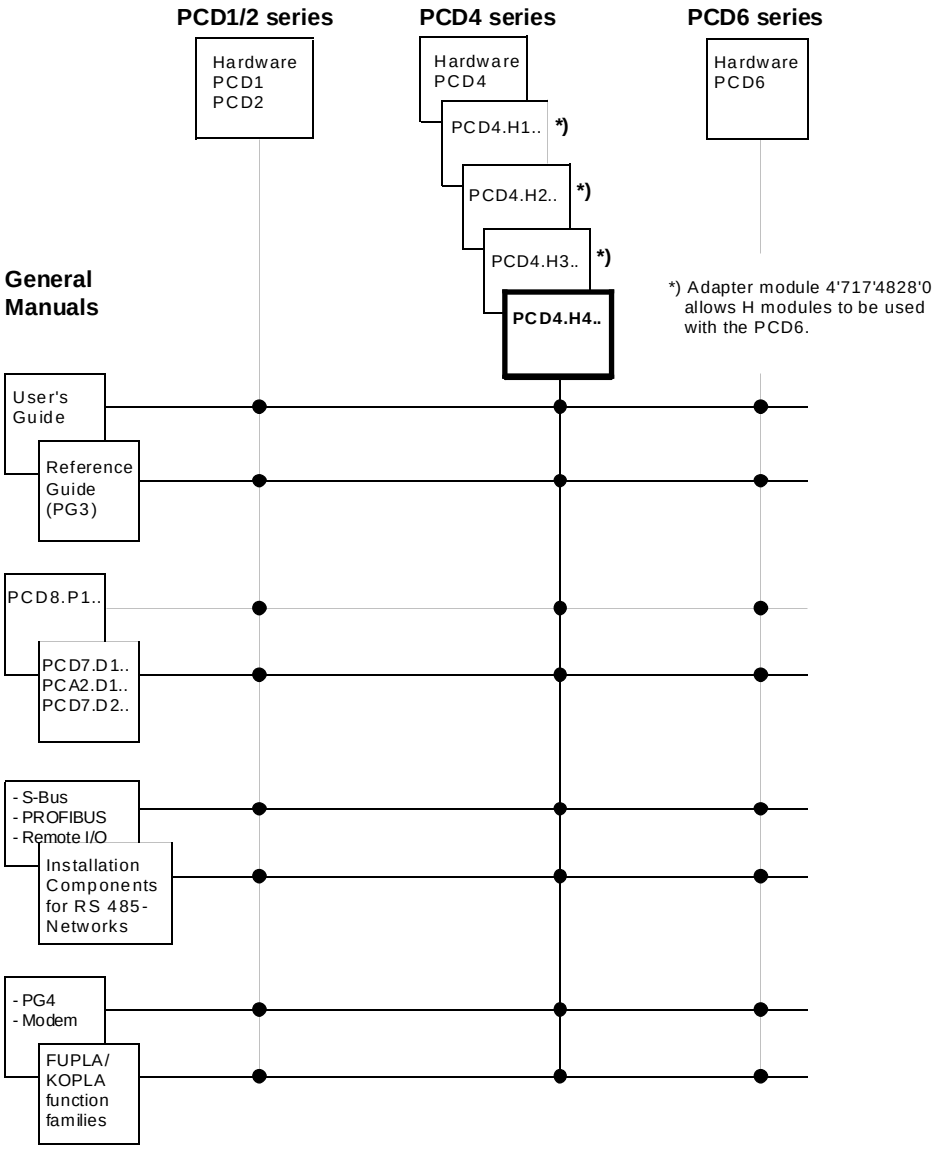
Please note:

A number of detailed manuals are available to aid installation and operation of the SAIA PCD. These are for use by technically qualified staff, who may also have successfully completed one of our "workshops".

To obtain the best performance from your SAIA PCD, closely follow the guidelines for assembly, wiring, programming and commissioning given in these manuals. In this way, you will also become one of the many enthusiastic SAIA PCD users.

If you have any technical suggestions or recommendations for improvements to the manuals, please let us know. A form is provided on the last page of this manual for your comments.

Summary



Reliability and safety of electronic controllers

SAIA-Burgess Electronics Ltd. is a company which devotes the greatest care to the design, development and manufacture of its products:

- state-of-the-art technology
- compliance with standards
- ISO 9001 certification
- international approvals: e.g. Germanischer Lloyd, Det Norske Veritas, CE mark ...
- choice of high-quality componentry
- quality control checks at various stages of production
- in-circuit tests
- run-in (burn-in at 85°C for 48h)

Despite every care, the excellent quality which results from this does have its limits. It is therefore necessary, for example, to reckon with the natural failure of components. For this reason SAIA-Burgess Electronics Ltd. provides a guarantee according to the "General terms and conditions of supply".

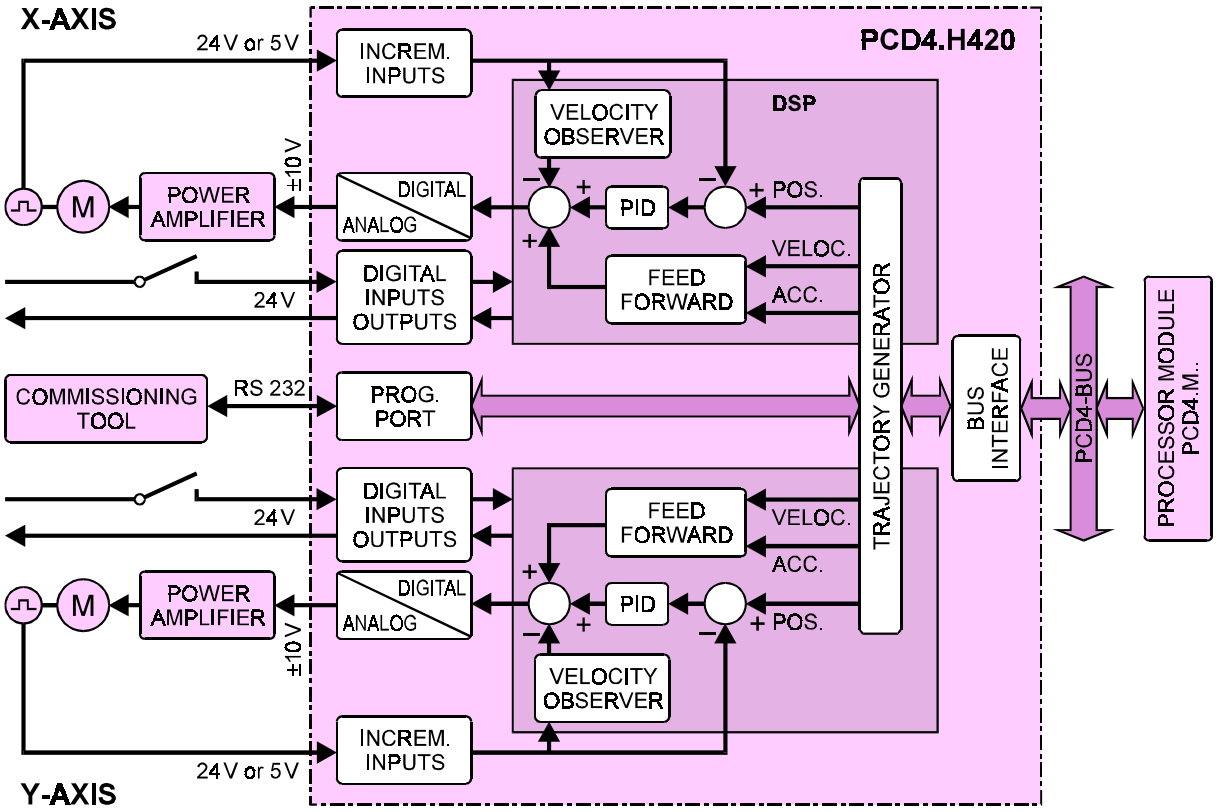
The plant engineer must in turn also contribute his share to the reliable operation of an installation. He is therefore responsible for ensuring that controller use conforms to the technical data and that no excessive stresses are placed on it, e.g. with regard to temperature ranges, overvoltages and noise fields or mechanical stresses.

In addition, the plant engineer is also responsible for ensuring that a faulty product in no case leads to personal injury or even death, nor to the damage or destruction of property. The relevant safety regulations should always be observed. Dangerous faults must be recognized by additional measures and any consequences prevented. For example, outputs which are important for safety should lead back to inputs and be monitored from software. Consistent use should be made of the diagnostic elements of the PCD, such as the watchdog, exception organization blocks (XOB) and test or diagnostic instructions.

If all these points are taken into consideration, the SAIA PCD will provide you with a modern, safe programmable controller to control, regulate and monitor your installation with reliability for many years.

1. Introduction

Block diagram of a servo drive for 2 axes

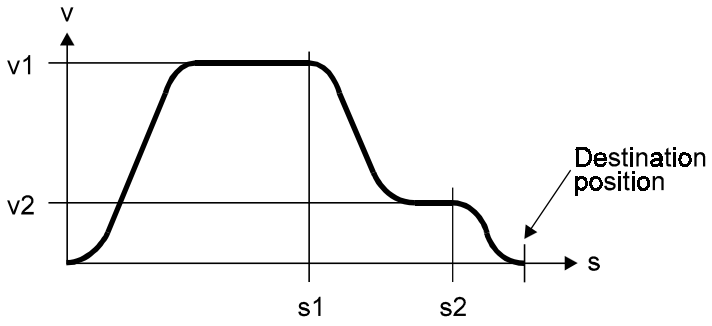


Function and application

The ..H4.. module is the most powerful of the axis control modules for the SAIA® PCD4. By using the latest DSP (digital signal processor) technology, the ..H4.. module is able to control 2 or 4 servo motor axes either independently or with linear or circular interpolation. The resulting S-shaped velocity profile produces motion which is both rapid and smooth.

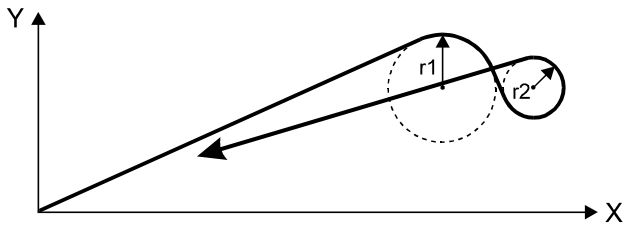
With its own memory and a high level of integrated intelligence, the ..H4.. module can be used in ways which take almost all the load from the PCD4's CPU, leaving it completely free for actual process control. Useful function blocks and a powerful software package make programming and commissioning extremely simple. The novice programmer is supported by readily comprehensible test and diagnostic information with appropriate help functions, all of which make the processes transparent.

Velocity/course profile of an axis following an S-shaped course and approaching the destination position in slow-feed motion



Main characteristics

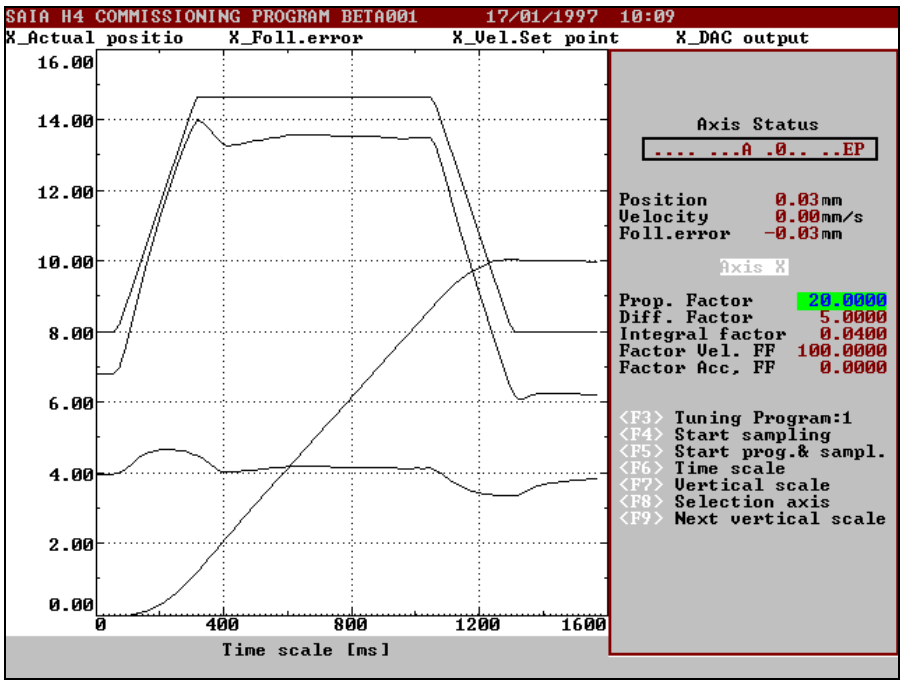
- PID control of 2 to 4 axes, independently of each other or with linear interpolation
- Circular interpolation of any 2 axes on the same module
- Smooth motion due to selectable velocity profile with a trapezoidal or S-shaped form
- High computing speed (40 MIPS)
- The ..H4.. module's autonomous axis functions place almost no load on the controller CPU, which is therefore fully available for process control
- The motion parameters can be stored permanently in EEPROM
- Incremental encoders can be used in 5 V or 24 V versions
- Hardware or software limit switches are monitored and processed independently by the module
- Each axis has an analogue ±10 V output with 16 bit resolution to the external power amplifier
- Programming is made simple by powerful instructions and a useful software library with function blocks
- Convenient programming and commissioning tool, which can be used to monitor or modify any motion, load individual programs directly into the ..H4.. module, and then run them.



Travel of two axes with linear and circular interpolation

Typical areas of application

- Automatic palletizing machines
- Automatic placement and assembly machines
- Packaging machines
- NC controlled cutting machines
- Machines for applying sealants and adhesives
- Pipe bending machines
- Tool changers
- Stock handling
- Handling robots
- Polishing machines and many others



Printed from the existing CP (commissioning and programming tool)

Programming and commissioning tool

With this software package the user has access to all the functions of the powerful ..H4.. module, i.e. writing and testing motion programs and optimizing the control parameters. The following menu-driven programs are available:

- **Configure:** Parameter entry for communications and axes
- **Motion:** Syntax driven editor for writing and commissioning motion programs
- **Graphics:** Graphical representation of motion (see figure above) which enables the regulation parameters to be checked and optimized
- **Utility:** Downloading or saving programs and parameters

Notes

2. Technical data

2.1 PCD4.H4xx

Displacement control	(Incremental, 2 quadrature pulses A and B plus reference mark R)
5 V inputs	5 V differential RS 422 inputs
Isolated	No
Frequency	max. 150 kHz (internal 600 kHz with x4 mode)
24 V inputs	
Signal ranges	Low = 0…4 V High = 19…32 V
Input current	10 mA
Isolated	No
Frequency	max. 100 kHz (internal 400 kHz with x4 mode)
Operating mode	Source

Digital inputs

Common for all axes	- Stop - Start
Per axis	- Limit switch LS1| Can also be replaced with - Limit switch LS2| software limit switches - Reference switch - Position capture - Error in power amplifier
Signal level	Low = 0…4 V, High = 19…32 V
Input current	10 mA
Input filter	30 µs
Isolated	No
Operating mode	Source

Digital outputs

Common for all axes	- ..H4.. ready
Per axis	- position trigger output - power amplifier enable/disable
Isolated	No
Short-circuit protected	No
Output current	1 ... 100 mA (min. load = 240Ω by 24V)
Operating mode	Source

Control unit output (to drive the power-amplifier)

Per axis	±10 V, short-circuit protected, 15-bit resolution plus sign bit, load resistance ≥ 3 kΩ. Offset max. ±100 mV

Programming and commissioning tool
(PC with MS-DOS)

Connection	RS 232 (with standard cable PCD8.K110/111)

Motion parameters (unit of entry selectable as mm, inches, angular degrees or encoder pulses)

Position	-2 147 483 648 to +2 147 483 647 units Range: $-2^{31}\ldots+(2^{31}-1)$ pulses
Velocity	-16 384 to + 16 383 units/servo cycle Range: $-2^{14}\ldots+(2^{14}-1)$ pulses (limited by the input filter of 100 kHz rsp. 150 kHz)
Acceleration	-16 384 to + 16 383 units/servo cycle Range: $-2^{14}\ldots+(2^{14}-1)$ impulses
Duration of S-shape	0.01 to 99.99 s
PID controller	Proportional, integral and derivative factors are programmable. Cycle time 200 µs for 2 axes, 400 µs for 4 axes
Electronic drive	For transmission ratios 0.0001 to 10000
Programming	with Function Blocks supplied as PCD source code or with 'Programming/Commissioning' Tool.

Memory (on ..H4.. module)

- permanent EEPROM for all motion parameters of 4 axes
- buffered RAM (super capacitor for min. 2 weeks)

approx 3000..4000 program lines divisible into 9 separate programs with max. 1000 lines per program.

Supply

External (user)	+24 VDC (19 V…32 V) smoothed, residual ripple max. 10%, max. 0.2 A plus encoder supply
For 5 V encoder	I_{max} = 300 mA/axis
For 24 V encoder	I_{max} = 200 mA/axis
Internal from PCD4 bus +5V	typ. 550 mA + 100 mA per axis

Operating conditions

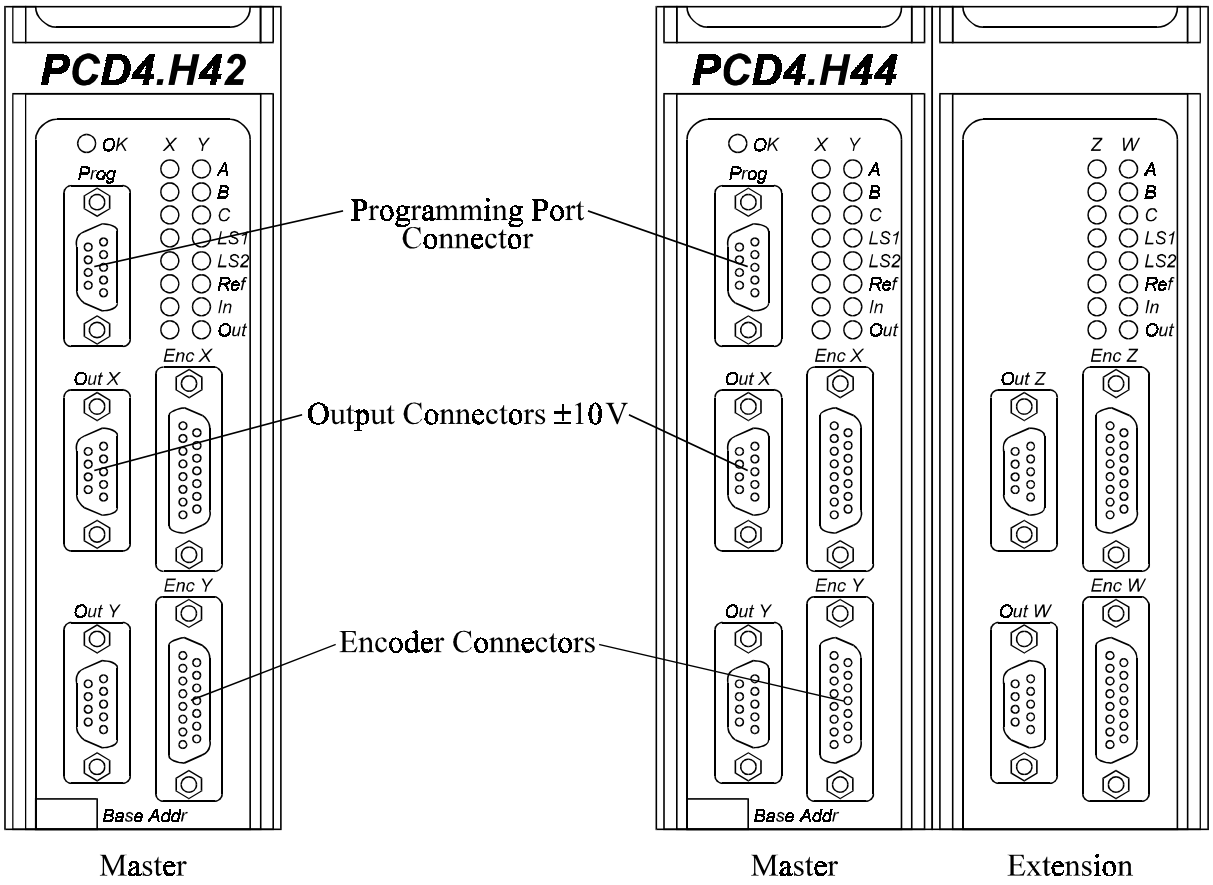
Ambient temperature	Operation: 0 °C…+55 °C without ventilation Storage: -20 °C…+85 °C Humidity: 5…95%
EMC	CE marking in accordance with requirements: Immunity according to EN 50 082-2, 1995 Emission according to EN 50 081-2, 1993
Mechanical resistance	According to IEC 1131-2
Interference resistance	1 kV in capacitive coupling according to IEC 801-4

2.2 PCD4 Configuration

CPU	Any PCD4 CPU can be used.
Supply	PCD4.N210 must be used, because of the H4 module's requirement for ±15 VDC. The 5V current consumption of H4 modules limits the number of modules to 4 x H120 or 3 x H440.
Memory	PCD.R1.. is adequate if no data is stored in the CPU's DB s. In all other cases: PCD7.R3.. (see section 7.7)

3. Presentation

3.1 Front panels and LED description

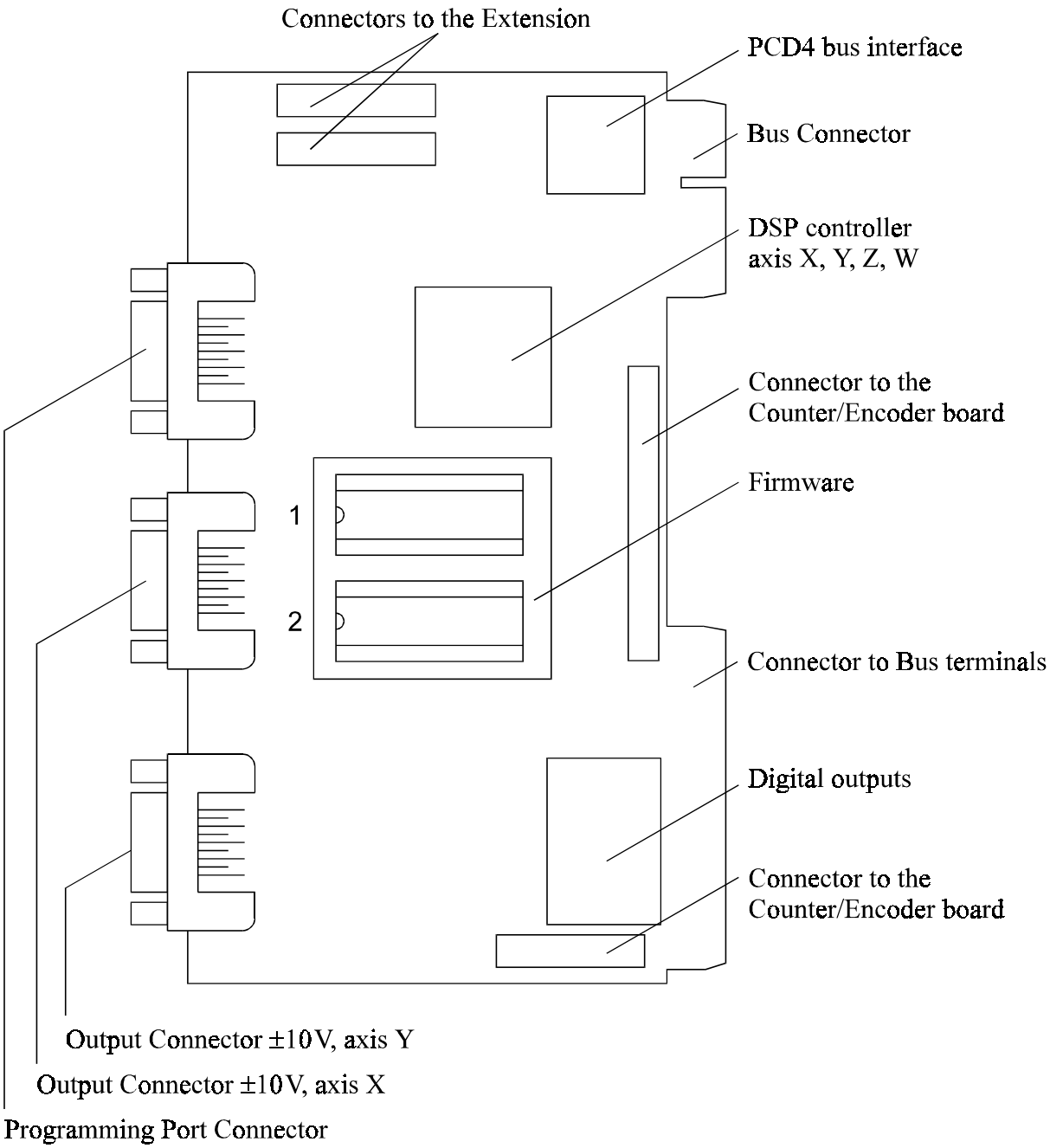


Group	LED	Meaning
System control	**OK**	on – DSP running – checksum OK – no error in user motion program
		flashing – abnormal condition (e.g. limit switch reached) but system still under control
		off – system blocked, major error
Encoder contro l	**A**	encoder signal A
	B	encoder signal B
	C	encoder signal C (reference)
		These LEDs show the status of the relevant encoder inputs

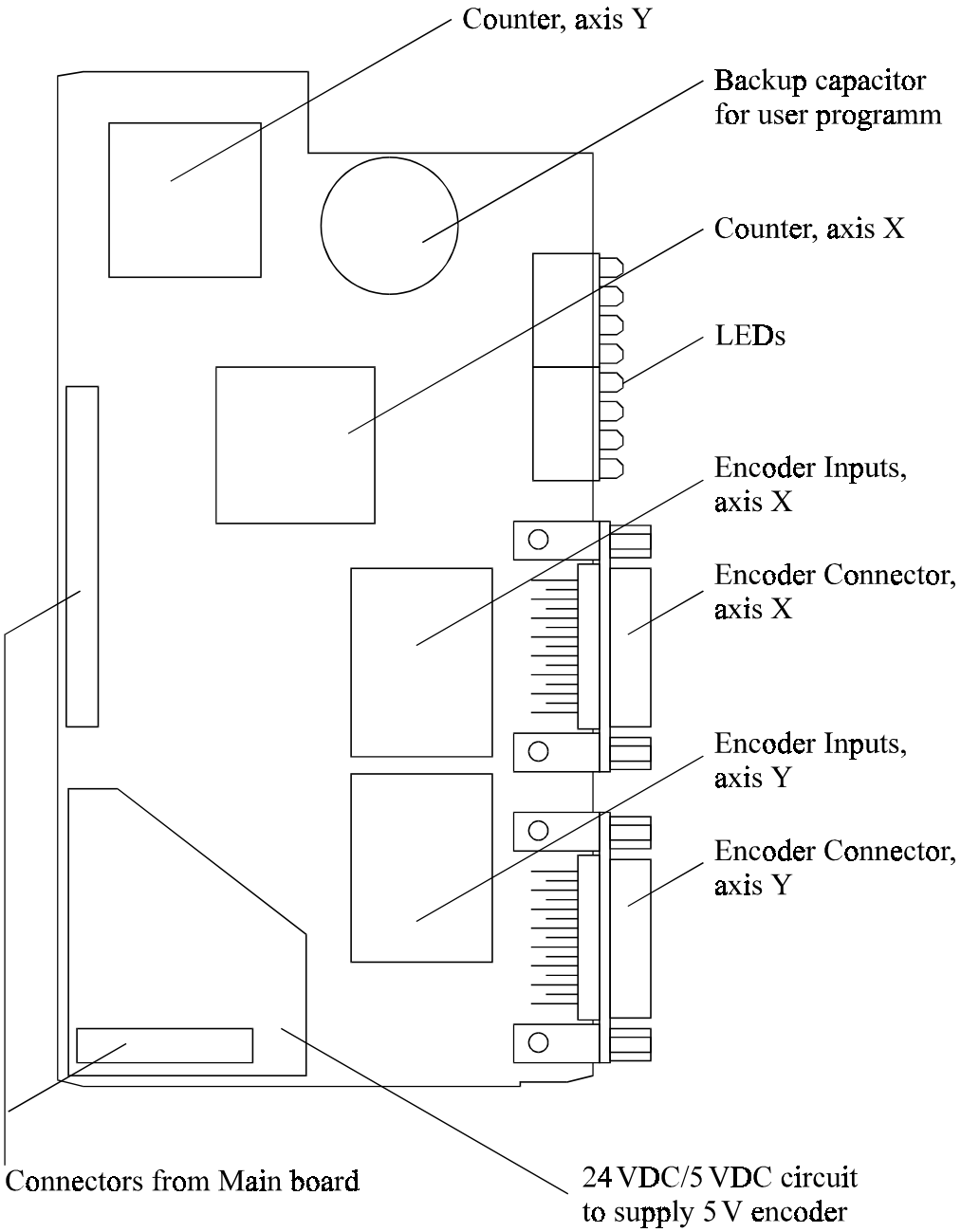
Group	LED	Meaning	
Digital inputs	**LS1**	on	Negative limit switch reached
		off	Negative limit switch not active
	LS2	on	Positive limit switch reached
		off	Positive limit switch not active
	Ref	on	Reference switch reached
		off	Reference switch not active
			These inputs are Low active, **thus 'normally closed' switches are required for security reasons.**
Amplifier control	**Out**	on	The digital output 'Amplifier Enable' is set high by the ..H4.. (command 'enable')
	In	on	The digital input 'Amplifier OK' is set high by the amplifier (as response to the 'enable')

3.2 Printed circuits

Main board, Master

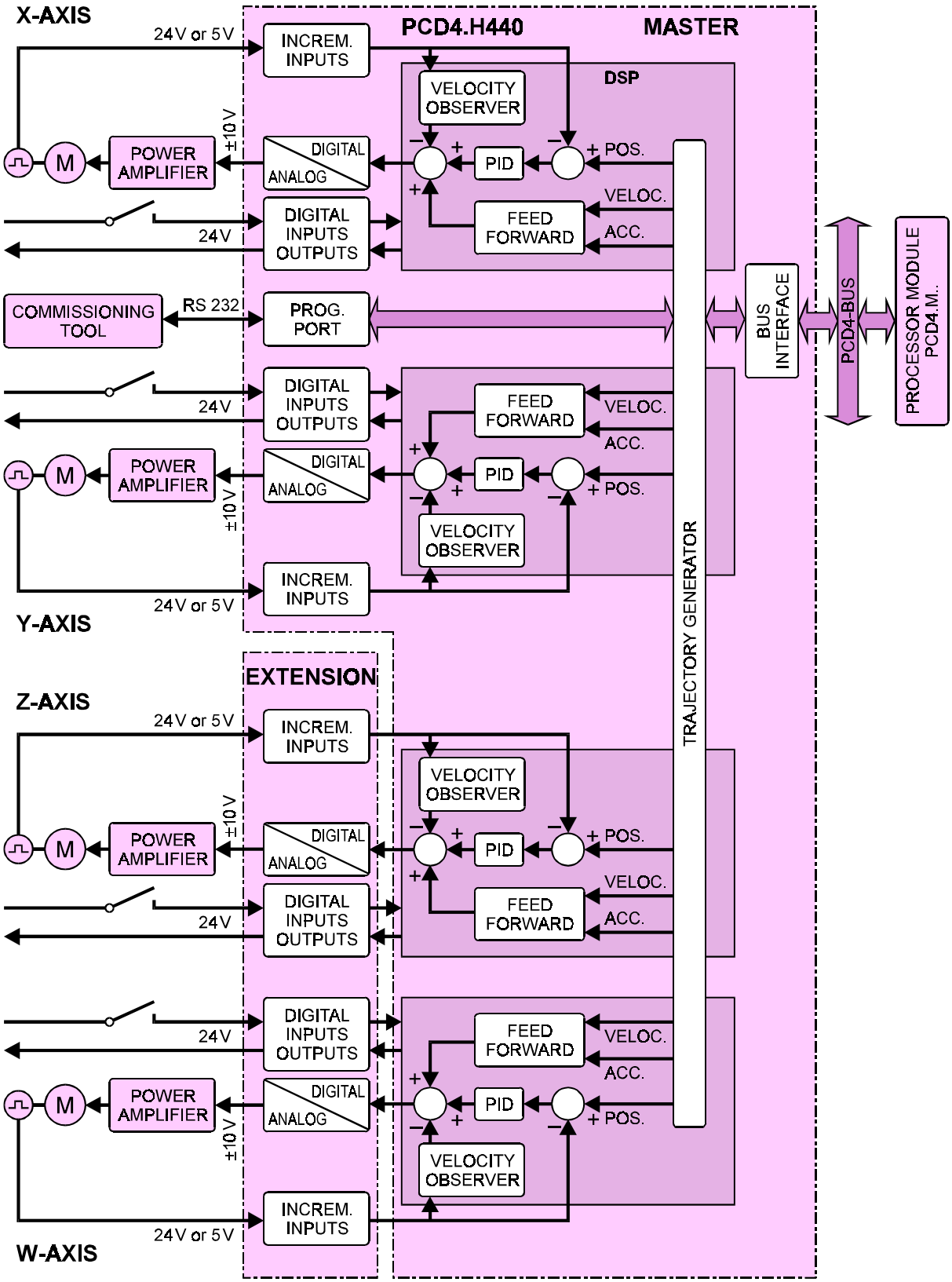


Counter/Encoder board



4. Logic diagram

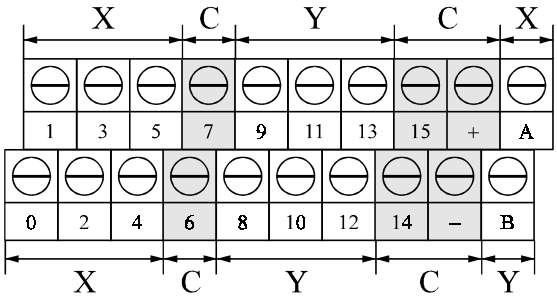
Block diagram of a servo drive for 4 axes



Notes

5. Connections

5.1 Bus module terminals (Overview)



MASTER

X-AXIS (Master)

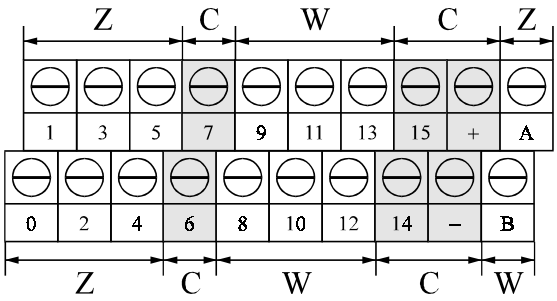
TERMINAL	DESCRIPTION	TYPE
0	AMPLIFIER ENABLE/DISABLE	OUTPUT
1	TRIGGER OUTPUT	OUTPUT
2	AMPLIFIER OK/FAULT INPUT	INPUT
3	LS1: LIMIT SWITCH START (NEG.)	INPUT
4	LS2: LIMIT SWITCH END (POS.)	INPUT
5	Ref: REFERENCE POINT SWITCH	INPUT
A	POSITION CAPTURE INPUT	INPUT

Y-AXIS (Master)

TERMINAL	DESCRIPTION	TYPE
8	AMPLIFIER ENABLE/DISABLE	OUTPUT
9	TRIGGER OUTPUT	OUTPUT
10	AMPLIFIER OK/FAULT INPUT	INPUT
11	LS1: LIMIT SWITCH START (NEG.)	INPUT
12	LS2: LIMIT SWITCH END (POS.)	INPUT
13	Ref: REFERENCE POINT SWITCH	INPUT
B	POSITION CAPTURE INPUT	INPUT

COMMON TERMINALS (Master)

TERMINAL	DESCRIPTION	TYPE
6	..H4.. READY	OUTPUT
7	STOP PROGRAM	INPUT
14		
15	START PROGRAM	INPUT
–	GND	
+	+24 V	



EXTENSION

Z-AXIS (Extension)

TERMINAL	DESCRIPTION	TYPE
0	AMPLIFIER ENABLE/DISABLE	OUTPUT
1	TRIGGER OUTPUT	OUTPUT
2	AMPLIFIER OK/FAULT INPUT	INPUT
3	LS1: LIMIT SWITCH START (NEG.)	INPUT
4	LS2: LIMIT SWITCH END (POS.)	INPUT
5	Ref: REFERENCE POINT SWITCH	INPUT
A	POSITION CAPTURE INPUT	INPUT

W-AXIS (Extension)

TERMINAL	DESCRIPTION	TYPE
8	AMPLIFIER ENABLE/DISABLE	OUTPUT
9	TRIGGER OUTPUT	OUTPUT
10	AMPLIFIER OK/FAULT INPUT	INPUT
11	LS1: LIMIT SWITCH START (NEG.)	INPUT
12	LS2: LIMIT SWITCH END (POS.)	INPUT
13	Ref: REFERENCE POINT SWITCH	INPUT
B	POSITION CAPTURE INPUT	INPUT

COMMON (Extension)

TERMINAL	DESCRIPTION	TYPE
6	NOT USED	
7	NOT USED	
14	NOT USED	
15	NOT USED	
–	GND	
+	+24 V	

5.2. Digital I/Os on the Bus module terminals

Master

TERMINAL	DESCRIPTION	TYPE
0	AMPLIFIER ENABLE/DISABLE X AXIS	OUTPUT
8	AMPLIFIER ENABLE/DISABLE Y AXIS	OUTPUT

Extension

TERMINAL	DESCRIPTION	TYPE
0	AMPLIFIER ENABLE/DISABLE Z AXIS	OUTPUT
8	AMPLIFIER ENABLE/DISABLE W AXIS	OUTPUT

Most amplifiers have an enable/disable input that permits complete shut-down of the amplifier regardless of the voltage of the command signal.

This control function is very important for safety reasons to make sure the amplifier can be completely shutdown when needed (in an error condition, and also to control the power-up sequence of the system). It is not secure to rely on a "zero" analog output voltage because offsets can easily build up.

The logic active level of this output is high.

Master

TERMINAL	DESCRIPTION	TYPE
2	AMPLIFIER OK/FAULT INPUT X AXIS	INPUT
10	AMPLIFIER OK/FAULT INPUT Y AXIS	INPUT

Extension

TERMINAL	DESCRIPTION	TYPE
2	AMPLIFIER OK/FAULT INPUT Z AXIS	INPUT
10	AMPLIFIER OK/FAULT INPUT W AXIS	INPUT

This input (HW signal from amplifier) indicates that the amplifier is ready. In error situation, if this signal is low, the H4xx module will perform a 'Kill' command and stop all motions.

The logic active level of this signal is high.

Master

TERMINAL	DESCRIPTION	TYPE
3	LS1: LIMIT SWITCH NEGATIVE X AXIS	INPUT
11	LS1: LIMIT SWITCH NEGATIVE Y AXIS	INPUT

Extension

TERMINAL	DESCRIPTION	TYPE
3	LS1: LIMIT SWITCH NEGATIVE Z AXIS	INPUT
11	LS1: LIMIT SWITCH NEGATIVE W AXIS	INPUT

To this input the HW limit switch of the negative direction must be connected.

The logic active level of these inputs is low, i.e. in normal situations (axis not at limit switch) +24 V should be present.

This requires the use of a normally closed (or normally conducting, if solid state) limit switch.

Master

TERMINAL	DESCRIPTION	TYPE
4	LS2: LIMIT SWITCH POSITIVE X AXIS	INPUT
12	LS2: LIMIT SWITCH POSITIVE Y AXIS	INPUT

Extension

TERMINAL	DESCRIPTION	TYPE
4	LS2: LIMIT SWITCH POSITIVE Z AXIS	INPUT
12	LS2: LIMIT SWITCH POSITIVE W AXIS	INPUT

To this input the HW limit switch of the positive direction must be connected.

The logic active level of these inputs is low, i.e. in normal situations (axis not at limit switch) +24 V should be present.

This requires the use of a normally closed (or normally conducting, if solid state) limit switch.

Master

TERMINAL	DESCRIPTION	TYPE
5	Ref: REFERENCE POINT SWITCH X AXIS	INPUT
13	Ref: REFERENCE POINT SWITCH Y AXIS	INPUT

Extension

TERMINAL	DESCRIPTION	TYPE
5	Ref: REFERENCE POINT SWITCH Z AXIS	INPUT
13	Ref: REFERENCE POINT SWITCH W AXIS	INPUT

This input is intended for use in the homing routine.

This requires the use of a normally closed (or normally conducting, if solid state) reference switch.

Master

TERMINAL	DESCRIPTION	TYPE
6	..H4.. READY	OUTPUT

The logic active level of this output is high when the system is ready for operation and no abnormal condition is present (e.g. limit switch reached).

The output is switched low when the OK LED flashes or goes off completely.

Master

TERMINAL	DESCRIPTION	TYPE
15	START INPUT	INPUT

With this input it is possible to start the program selected with P95.
The logic active level of this input is programmable. (P90)

Master

TERMINAL	DESCRIPTION	TYPE
7	STOP INPUT	INPUT

With this input it is possible to stop the program action of the program selected with P95 at the next wait instruction.

The logic active level of this input is programmable. (P91)

Master

TERMINAL	DESCRIPTION	TYPE
1	TRIGGER OUTPUT X AXIS	OUTPUT
9	TRIGGER OUTPUT Y AXIS	OUTPUT

Extension

TERMINAL	DESCRIPTION	TYPE
1	TRIGGER OUTPUT Z AXIS	OUTPUT
9	TRIGGER OUTPUT W AXIS	OUTPUT

If enabled by software, the trigger outputs generate a signal edge when an encoder position reaches a preloaded value.

This is very useful for the practically instantaneous triggering of an external action or event when the system arrives in a certain position.

The logic active level of this output is programmable. (P'x'62)

Master

TERMINAL	DESCRIPTION	TYPE
A	POSITION CAPTURE X AXIS	INPUT
B	POSITION CAPTURE Y AXIS	INPUT

Extension

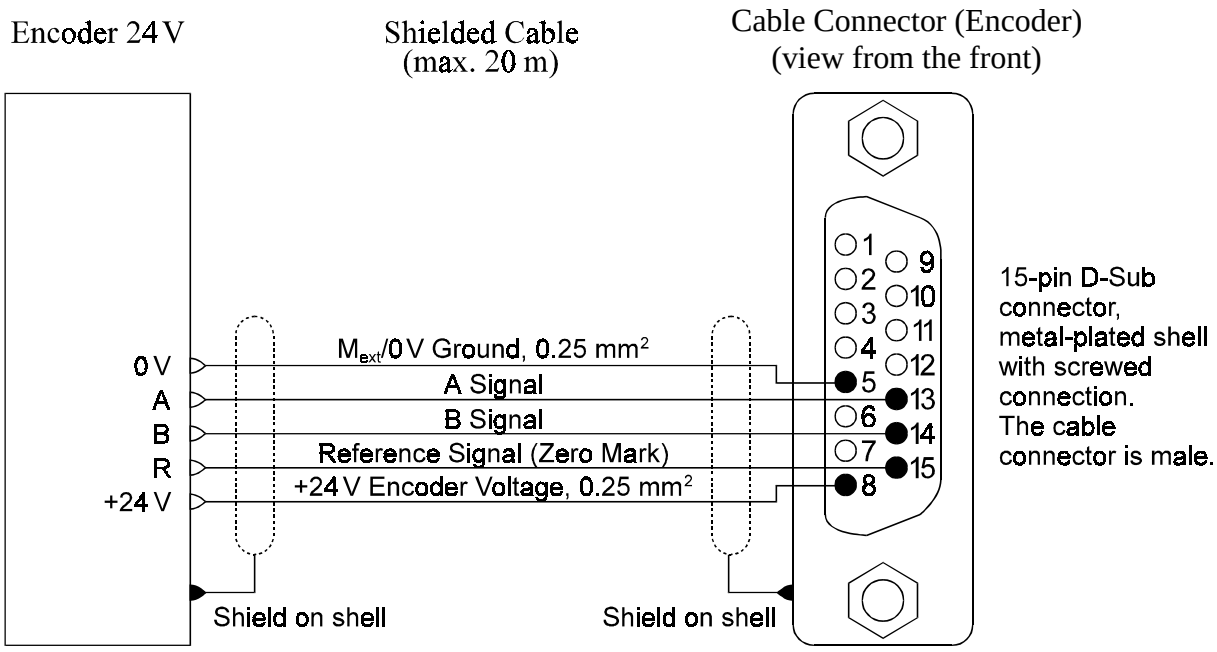
TERMINAL	DESCRIPTION	TYPE
A	POSITION CAPTURE Z AXIS	INPUT
B	POSITION CAPTURE W AXIS	INPUT

With this signal, when enabled by parameters or programs, you can store the actual position of the axis in real-time for subsequent analysis (position capture).

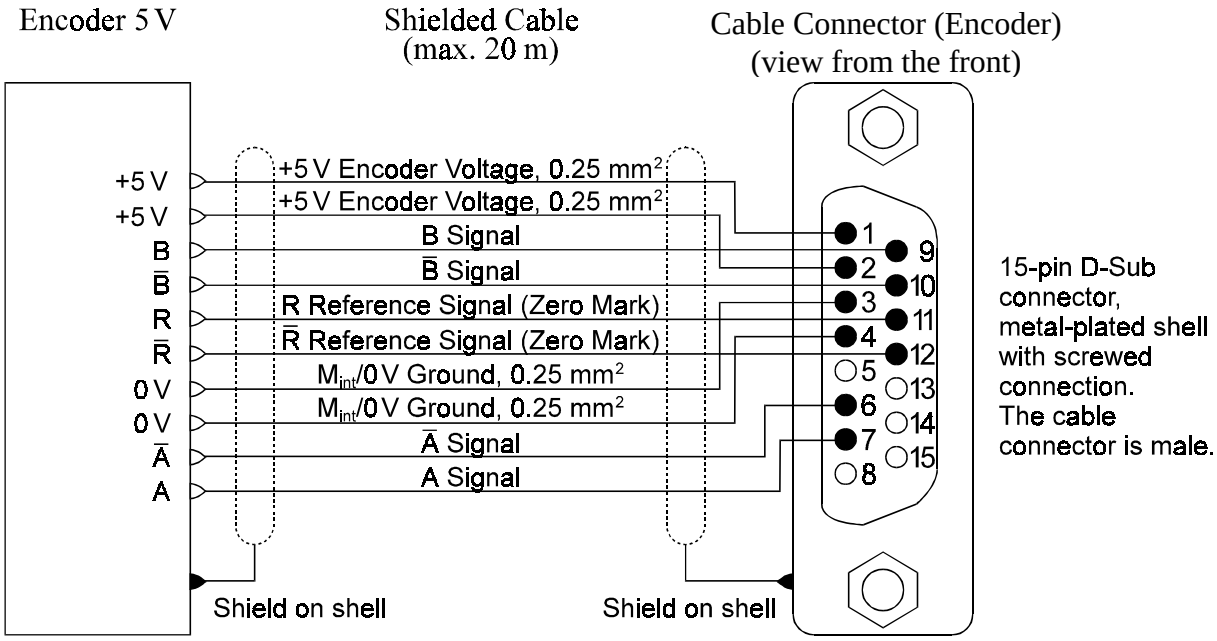
The logic active level of this input is high.

5.3. Front connectors and connecting cables

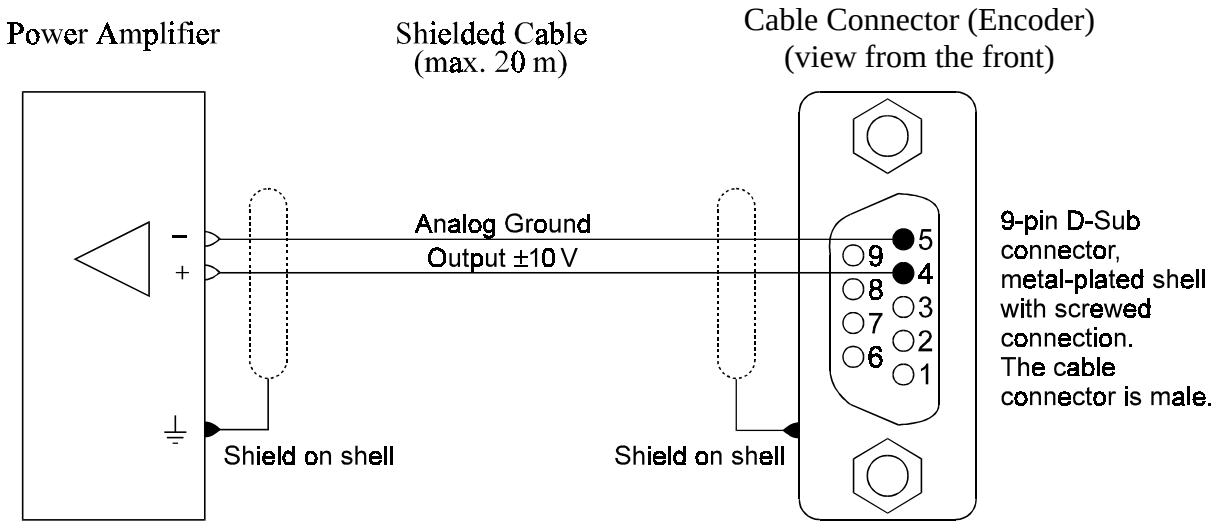
Connection diagram for the 24 V encoder connecting cable



Connection diagram for the 5 V/RS 422 encoder connecting cable



Connection diagram for the analogue output cable (±10 V)



Programming port (PROG)

with standard cable PCD8.K110/111

(for pin configuration, see PCD4 manual)

Notes

6. Function specifications

6.1 Introduction

The module

The PCD4.H4.. motion control module can control 2 to 4 axes and apply linear or circular interpolation.

The module is plugged onto the PCD4 bus, where it occupies 16 addresses for communication with PCD4 user software. The 5V current consumption limits the number of H4 modules. It would theoretically be possible to plug in up to 8 modules (32 axes) with each module able to interpolate its axes (see section 2.2, Supply). Cross-module interpolation is not possible. However, with one H4 module it is possible to control and monitor the autonomous motion of 4 axes independently or the single (interpolated) motion of 4 axes. Any intervening combination is also possible.

Independence

The module operates independently. It controls the axes, travels along them precisely, communicates with the CP tool (commissioning and programming tool) and/or with the PCD CPU via standard SAIA® function blocks and has its own program memory.

Integral to the PCD

Depending on the complexity of the application(e.g. variable process-dependent data: velocity, position etc.) a PCD CPU must be called on to control the H4 modules.

User friendly

Programming and parameter definition is possible with the CP tool on the PC screen, but also via FBs from the PCD4 CPU. This gives the user complete freedom to exploit all the capabilities for his project.

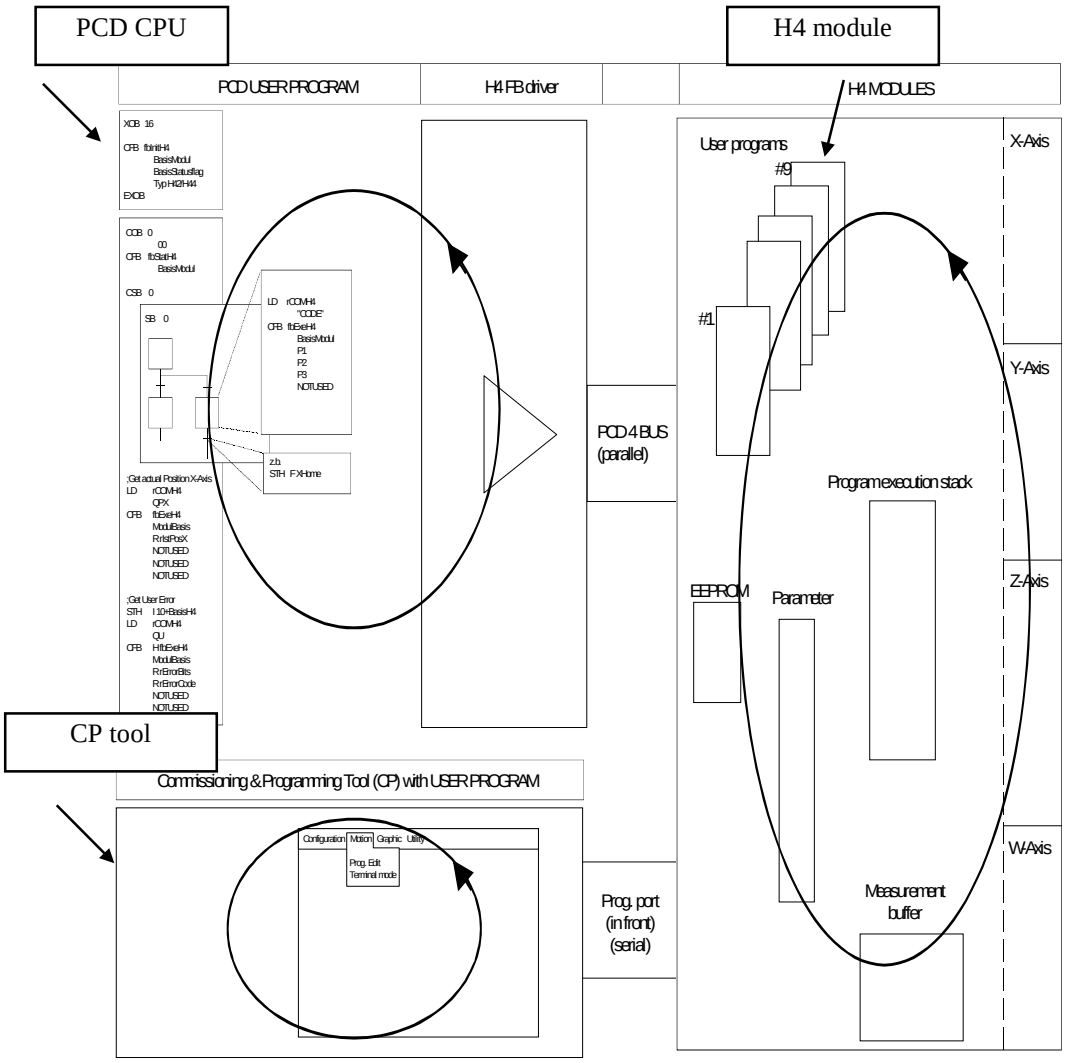


Figure 6.01

This figure shows that each module (PCD, H4 module, CP tool) operates independently and places no significant load on the others.

6.2 Block diagram, functional method

6.2.1 Overview

A hardware logic diagram is shown in chapter 4. The H4 module is depicted here in its resources and functions. It can therefore be seen, for example, that there is one parameter range (block). All functions or instructions (indicated with arrows) always affect the relevant block.

PCD4.H4xx

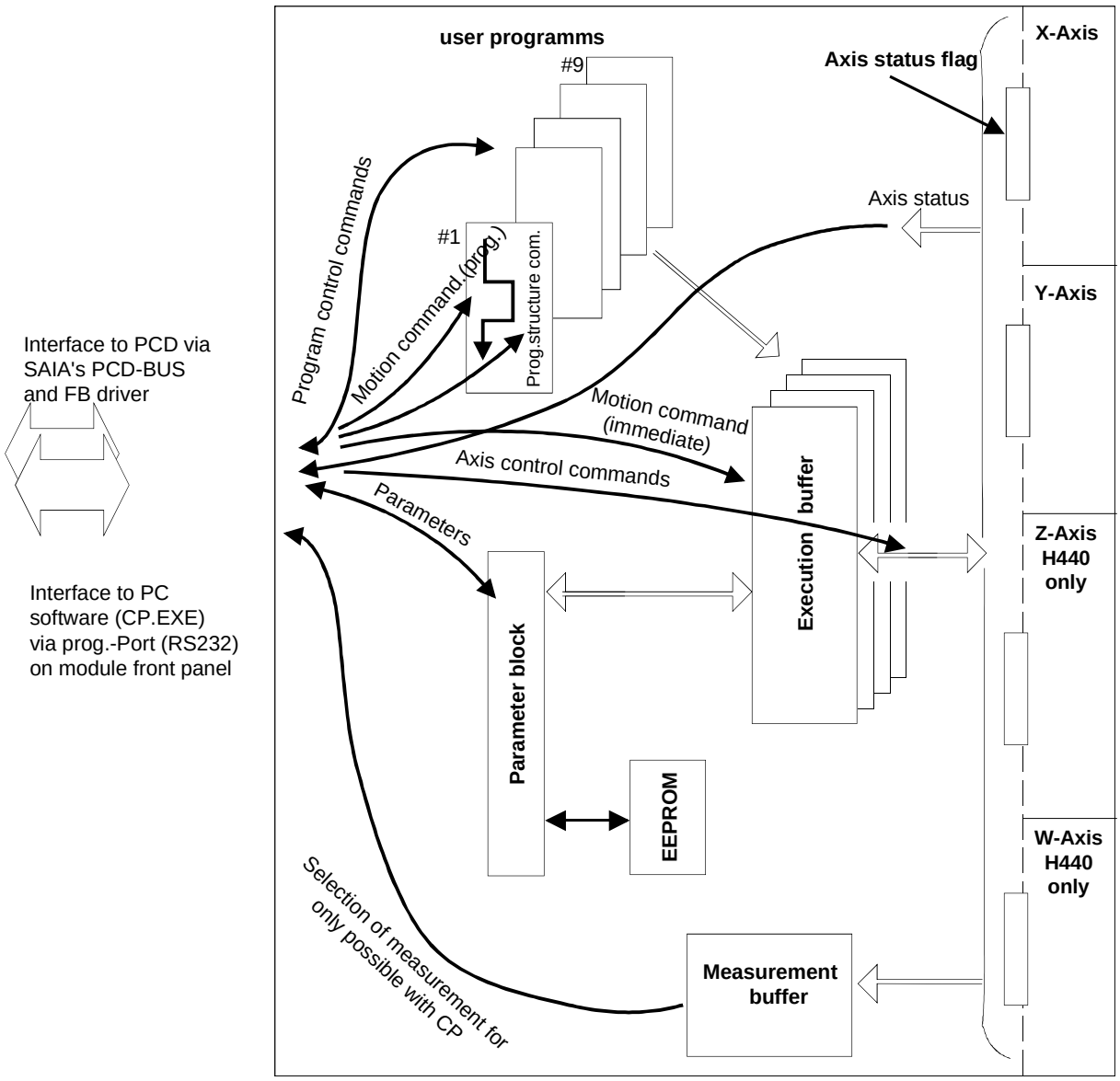


Figure 6.02

The above diagram shows that commands to the H4 module can be divided into various groups. It does not matter here whether these commands originate from the PCD CPU (via FBs) or from the PC (CP.EXE).

6.2.2 H4 program memory

The H4 module has a separate memory for user programs. Commands from the H4 module command set can be stored in this memory. A program can be written with the CP tool and then transferred to the H4 module. This is also possible from the PCD4 CPU by means of FBs.

Programs
Programs are assigned a number on loading into the H4 module. 9 programs are available. 4 programs can run simultaneous.

Program lines
For each program, a maximum of 1000 lines can be stored in the H4 module. In total, and depending on the commands used, it is possible to store approx. 3000 - 4000 program lines in the H4 module.

Storage
Programs in the H4 module are protected with a supercap against loss from power failure and are held for at least two weeks. Parameters are saved in EEPROM and are not lost..

6.2.3 Parameters

Around 80 parameters are stored in the H4 module. At power-up they are copied from EEPROM into the 'Parameter block' which the H4 operating system uses. These parameters are divided into function groups and summarized in the list of parameters, section 7.6.

Modifications/EEPROM
Parameter modifications are volatile under conditions of power failure. However, settings for a specific application can be stored in the H4 module by copying the parameters into EEPROM. This happens automatically with the CP tool when parameters are written to the H4 module from the parameter menu. When working with FBs, the parameters must be deliberately saved with a special command. The maximum number of write cycles is limited to 100,000. For this reason, saving cannot be executed cyclically.

6.2.4 Execution mode (Immediate / Program)

FBs:
Figure 6.02 shows that there are two types of motion command. 'Immediate' motion commands are transferred directly to the execution buffer and carried out consecutively. 'Program' motion commands are not executed directly, but are written to an H4 program (no. 1 - 9). In the command list (section 7.5) the 'ip Parameter' column indicates the execution mode in which each command operates. (I = immediate only, P = program only, IP = immediate + program)

CP tool: (CP = Commissioning / Programming)
Basically, the two execution modes (immediate / program) also apply when using the CP tool. However, the user is only confronted with this indirectly. If the CP tool is used to work from its 'Motion/Program Edit' menu, where programs are written and then downloaded into the H4 module, only the 'Motion Prog. Cmd' commands are accepted. In the 'Motion/ Terminal' window, however, the 'Motion immediate' commands are used.

6.2.5 Execution buffer

The H4 module is autonomous and can therefore execute programs to their conclusion without further support. It is then only necessary, by means of the CP tool or using FBs from the PCD CPU, to control the program flow (e.g. Run 5).

For this autonomous program execution process, and for the execution of 'Motion Immediate' commands, the H4 uses 4 internal execution buffers.

With immediate commands, only one buffer is used. This execution buffer can store 50 'Immediate' commands. If this total is exceeded, the 'User Error' input and error bit 9 are set. This error message is reset when the number of commands in the buffer is dropped to 45.

Buffer overload: If further immediate commands are sent to the H4 despite the 'Buffer full' (bit 9) error message, they will be lost.

The execution buffer is processed sequentially, i.e. a new command is only executed once the preceding one has terminated.

If a number of axes are to be moved simultaneously (not interpolated) it is necessary to work with different programs (1 program/axis) which can be started in parallel.

Each program uses one buffer, i.e. max. 4 programs respectively 3 programs and immediate commands can be executed at the same time.

6.2.6 Axis status flag

Figure 6.02 shows that each axis includes an axis status flag. This can be used, for example, to determine whether an axis has reached its end-of-travel switch, whether position control is active, or if the home process has finished. These axis status flags are queried with the 'Query status x' instruction. Please refer to the command list (section 7.5) for individual flags and their meanings.

The status flags are divided into groups. Flags 0-7 are occupied by standard FBs and cannot be employed by the user. Flags 8-23 are reserved for the X axis, 24-39 for Y, 40-55 for Z and 56-71 for the W axis. When programming, the user is free to work either with numbers or can assign a symbol to the appropriate flag. To all flags, the flag base address is added witch can be defined in the initialization FB. (see section 7.4.6). The following list is for Flag base address = 0.

Flag 0-6: Flags 0-6 are occupied by standard FBs and cannot be employed by the user.
7: Fatal Error (see Chapter 8).

	X	Y	Z	W:	Axis
Flag	8	24	40	56:	Axis in position
	9	25	41	57:	Axis running in 'immediate' mode
	10	26	42	58:	Axis in hardware LS
	11	27	43	59:	Axis in software LS
	12	28	44	60:	Following error
	13	29	45	61:	Following error warning
	14	30	46	62:	Theoretical velocity of axis = 0
	15	31	47	63:	Capture position detected
	16	32	48	64:	Drive OK (status input AOK)
	17	33	49	65:	Negative LS input activated (LSS)
	18	34	50	66:	Positive LS input activated (LSE)
	19	35	51	67:	Reference switch activated (RPS)
	20	36	52	68:	'Position Capture Input' activated (PCI)
	21	37	53	69:	Trigger position reached
	22	38	54	70:	Position overrun
	23	39	55	71:	Home function successfully performed

6.2.7 Measurement buffer (see diagram on page 1-3 and figure 6-02)

The measurement buffer can only be read with the CP tool in its graphic menu. This buffer serves to store motion data previously selected by the user. This data can be displayed graphically (oscilloscope function) and used to adjust the axis control parameters on machines. The operation and use of this function is described in section 7.3, CP tool.

6.3 Function overview

Functions:	
Positioning a linear axis	Yes
Positioning a rotational axis	Yes
Linear interpolation up to 4 axes	Yes
Circular interpolation	Yes
Spline interpolation	No
Position control (positioning mode)	Yes
Speed control	No
Electronic drive (of two or more axes)	Yes
Blended moves	Yes
S-curve acceleration profile	Yes
Feed-forward for velocity and acceleration	Yes
Adjustment of axis control parameters	Yes, with software (CP tool)
Save motion program of H4 module outside the module	Yes, on PC or PLC
Spindle pitch error compensation	No
Compensation of lost motion on return	Yes
Teach-in	No
ISO code (CNC)	No
Parameter modification during operation	Yes (see ‘on the fly’)
M commands as in CNC	No
Jog: manual control	Yes

The majority of functions marked 'No' can be resolved with the CPU.

Examples of uses: (see also Chapter 1)	
Electronic cam programmer	No
Flying cut possibilities	No
Cartesian robots	Yes
Handling device	Yes
Special machines	Yes

6.4 Differences between the H3 and H4 modules

Differences	**PCD4.H3xx**	**PCD4.H4xx**
Operating modes	Position and velocity modes	Position mode only
Program editor for IL instructions	Any ASCII editor, **but not** SEDIT (symbol offset not possible).	Any ASCII editor possible, **including** SEDIT.
Motion program	Cannot be stored in the module. All data and axis information are stored in the PCD CPU.	Up to 9 different programs can be stored in the module's RAM. This reduces the load on the PCD CPU.
Parameters	Are not stored permanently in the module. This means that they are lost when the module is powered off.	Are saved in EEPROM in the module. Therefore, they are not lost when the module is powered off.
Axis Init and Axis Handling	Each axis is controlled and monitored by two FBs: "AxInit" and "AxHndlg". A function is triggered by setting a flag.	Each function is executed directly with one command by calling the FB "fbH4.exe". For initialization and module status, two additional FBs are available.
Reference procedure	Must be solved by the user.	Executed automatically by the H4 on request
FB nesting level	3 levels	1 level (Prog. Up/Download has 2 levels)
Synchronisation between axes	By user program in PCD.	By the module (multi-axis linear or circular interpolation)
Motion Control Factor : - Units - Conversion of pulses to mm / inches or vice versa	 - Encoder pulses or mm - by PCD CPU	 - Encoder pulses, mm, angular degrees or inches - By H4
Commissioning & Progr. Tool : - Connection - Programming - Commissioning	 - PGU connector on PCD-CPU - not possible - Only one axis can be programmed and executed. A minimal PCD program must be present in the PCD CPU.	 - Programming connector on the H4 - Entire motion cycles can be written, executed and saved on diskette. This does not require any user program in the PCD-CPU. - Possibility for direct execution of motion - Online optimization of parameters with graphical support.
Limit switches and reference switch	Must be monitored by the user.	Monitored by the H4 module.
Addresses	The initial address and the number of modules must be defined. When more than one module is used, they must all be adjacent (no spaces).	More flexible. The address of each module must be defined and transferred to the FBs as parameters. This allows freedom when positioning modules on the PCD4 bus.
I/O for motion control module	All I/Os must be controlled by the PCD CPU.	Integral to and controlled by the H4.

6.5 Generator for the velocity profile

Profile generator

TheH4 module can generate either trapezoidal or S-curved velocity profiles. These can be selected and defined for each axis with P 'x' 42. The generator produces the specified curve for each axis. The servo-position controller then regulates the actual position as closely as possible to the desired position.

6.5.1 Trapezoidal velocity profile

This is the simplest velocity profile. The axis travels at a defined velocity towards a target, accelerating and decelerating with a constant ramp. These velocities are defined in the following parameters:

maximum acceleration/deceleration rate:	P 'x' 33
acceleration rate:	P 'x' 43
deceleration rate:	P 'x' 44
velocity with the command	SS 'x'
acceleration mode	P 'x' 42 = 0 (trapezoidal)

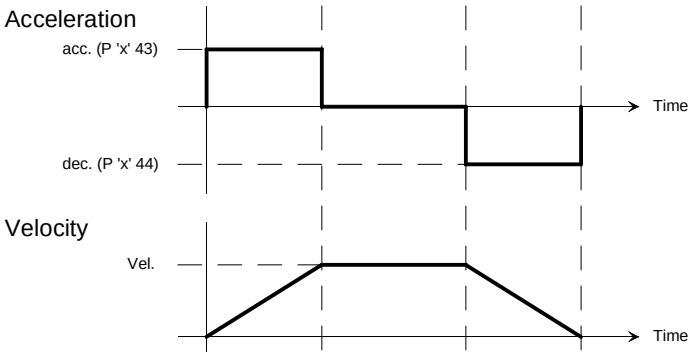


Figure 6-03

When using a high velocity or travelling a very short distance, it is possible for the desired velocity not to be reached. In this case the velocity profile is triangular.

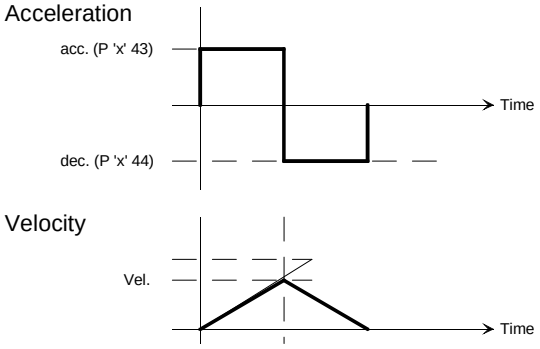


Figure 6-

6.5.2 S-curve velocity profile

A trapezoidal velocity profile with constant acceleration results in a sudden change when acceleration starts, which can cause oscillation of the axis. For a smoother transition and to overcome static friction at V = 0 an S-curve profile is used. This profile is achieved by altering acceleration during acceleration. The duration of the S-curve t_s is user-definable and stored as a parameter.

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

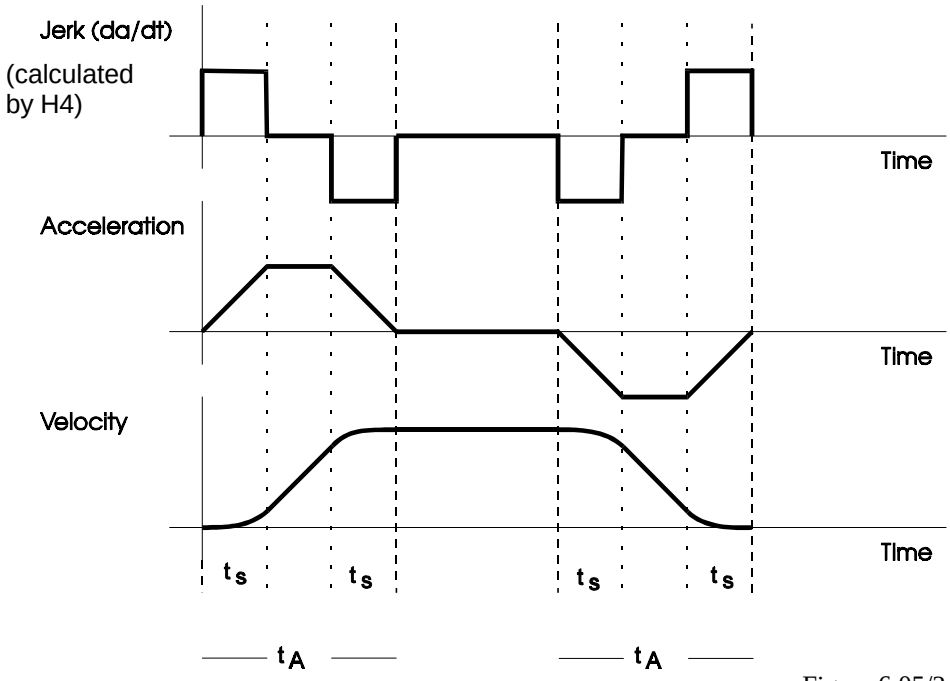


Figure 6.05/2

If an S-curve is used, parameters 43 and 44 apply as mean acceleration and deceleration.

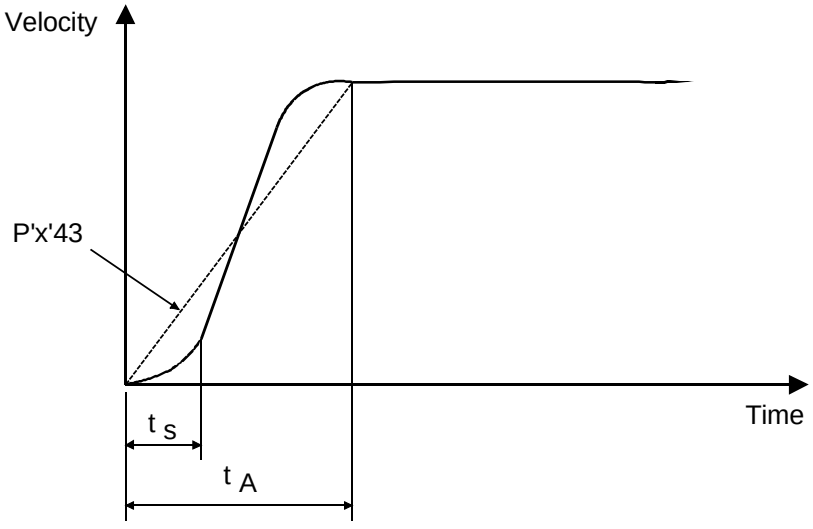


Figure 6.05/3

If an S-curve duration time of $t_s = 0$ is specified, the resultant velocity profile is fully trapezoidal. If t_s is specified as greater than half the (calculated) acceleration time (t_A), ts is limited to $t_{A/2}$. This produces no linear element within the acceleration. It results in a purely S-shaped acceleration with a maximum which corresponds to 2 times the mean acceleration (P 'x' 43). Therefore, when a pure S-curve is used the maximum acceleration, defined in P 'x' 33, can also be exceeded 2 times, which would entail quite a large following error during acceleration.

In practice, a reasonable S-curve duration time would be 5 ... 30% of acceleration time t_a .

Combination of two axes with different ramp profiles

If, for example, the X axis is defined with a trapezoidal ramp profile while the Y axis has an S-curve and a motion path is started, both axes are accelerated with an S-curve.

If a different duration time t_s has been defined for each axis, the greater time value is used.

The following commands apply for interpolated motion:

SV instead of SS 'x' (velocity)
SA instead of P 'x' 43 (acceleration)
SD instead of P 'x' 44 (delay)

max. Velocity	P 'x' 30	¦ are also
max. Acc./Dec.	P 'x' 33	¦ considered

Path velocity SV is divided among the individual axes according to the trajectories.

$SV = \sqrt{V_x^2 + V_y^2}$ for 2 axis interpolation

$SV = \sqrt{V_x^2 + V_y^2 + V_z^2}$ for 3 axis interpolation

$SV = \sqrt{V_x^2 + V_y^2 + V_z^2 + V_w^2}$ for 4 axis interpolation

Only one path velocity SV can be used for each module. With immediate commands it is not possible with an H440 to carry out linear interpolation of x/y and z/w with different path velocities at the same time, because the H4xx module can only execute one immediate command at the time. However, as SV is also a program command, it can be set in different programs for independent axes interpolation, e.g. x/y and z/w separately.

6.6 Blended move

The H4 module can be instructed to execute complete cycles. This means a number of separate motion sequences which together produce a cycle. If an individual motion in a cycle is over, all axes concerned would reduce velocity to zero, so that they can then accelerate again for the next motion. With the 'blended move' function, the new velocity is adopted and the only change made is from the first to the second velocity. This change starts where the deceleration ramp of the first motion would start without 'Blended move'. The blending from one velocity to a new one always takes place trapezoidally.

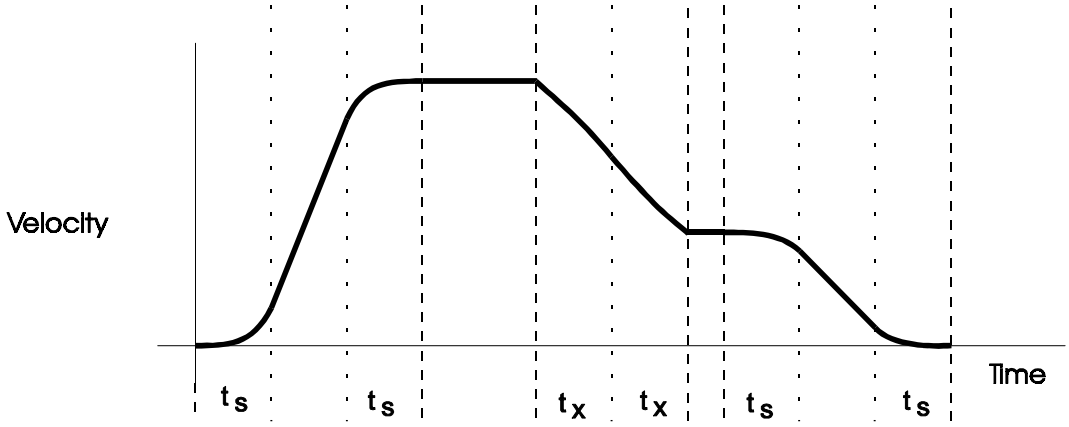


Figure 6.06

Examples:

without blended move:	with blended move:
XR500,YR0 WAIT0 XR200YR100	XR500,YR0 XR200YR100
SV100 XR50,YR0 WAIT0 SV120 XR200,YR100	SV100 XR50,YR0 SV120 XR200,YR100
XR500 YR200	XR500,YR0 XR0,YR200

Parameter P97 "Blended move angle"

Parameter 97 sets an angle α from which the H4 module applies 'blended move'.

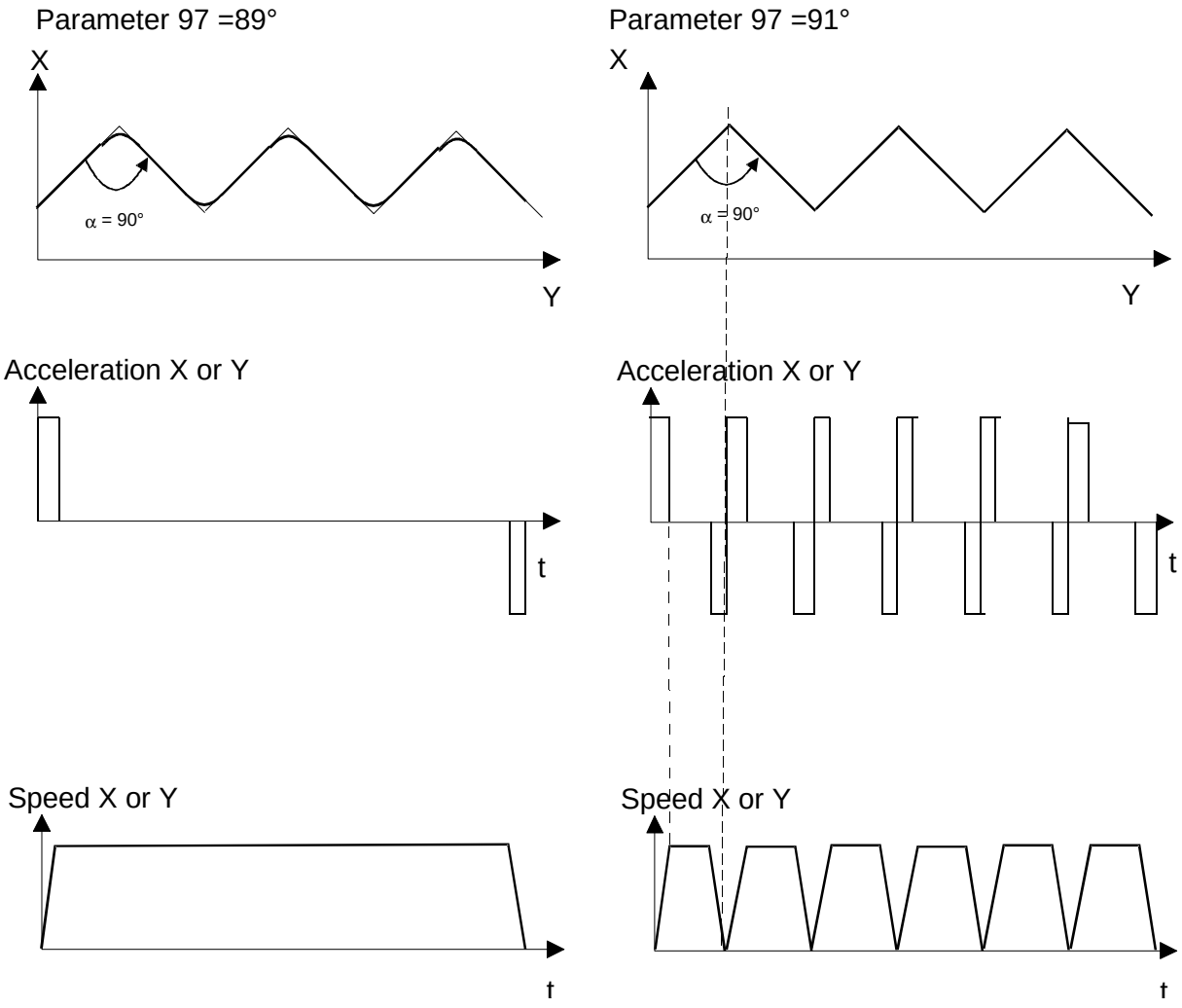


Figure 6.06.1

If P97 is smaller than the angle arising when the axes are travelled, 'blended move' applies. This makes sense if, for example, a handling device must travel back some distance without pausing in between. However, if specific points are to be approached exactly, the angle can be set so high the blended move is not used. (with P97 = 181° the function is switched off entirely). It should be noted though that, with blended move, intermediate positions are not completely reached: this is to maintain a constant path velocity.

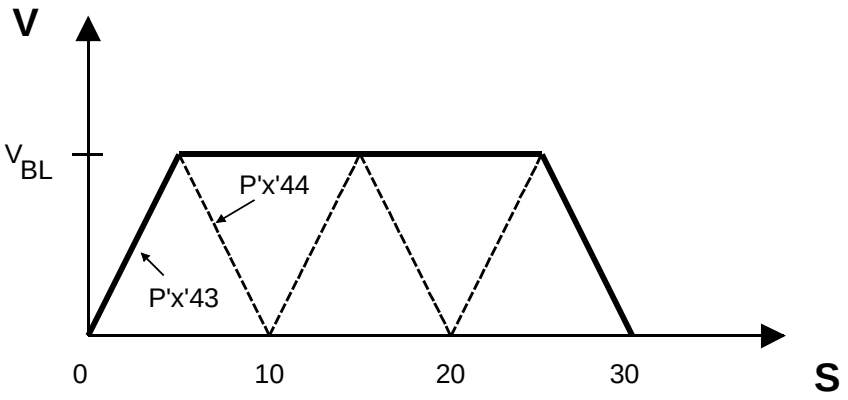
Velocity with 'Blended move'

If the path of each single movement is short comparing to the selected velocity, the velocity set point cannot be reached.

The velocity V_{BL} with blended move equals to the max. velocity which would be reached in a single movement and depends on P'x'43 (acceleration), P'x'44 (deceleration) and the paths.

There are 3 possibilities to get the selected velocity:

- set low the blended move velocity set point or
- set as high as possible P'x'43 and P'x'44 or
- increase the path for each single movement



With P'x'43 = P'x'44 = a for example, the velocity is calculated as:

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

where: s = path

6.7 Home Function

The H4 module can execute the home function independently. This function is itself a small program cycle and must therefore also be defined. The definitions are located in parameters 20-24 and are listed in section 7.6.3.

The axis to be homed must be active (ENABLE).
(The following description refers to figure 6.07)

1. The search for the reference switch takes place at the velocity defined in parameter 22. Search direction is defined in parameter 20. If the reference switch is not found and the axis encounters a limit switch (HW or SW), the search direction is changed.

2. If the reference switch has been found, freewheely commences (exits from reference switch). The direction of freewheeling is defined in parameter 23 and the velocity in P 'x' 24.

3. When the reference switch has fallen, the axis moves further until the encoder index signal (channel C) is detected. The repeating accuracy of the reference switch is not relevant, since homing refers to the encoder's C signal. Repeating accuracy is therefore incrementally precise.

4. When the C signal is recognized, the axis is loaded with the value defined in parameter 23 (frequently '0') and then decelerates. After stopping, the axis is not located precisely on the reference point.

   The home function has been successfully concluded and the status flag 'H' is set (see description of the command QS 'x' in the chapter 7.5.5).

- If the reference switch or the encoder's C signal is not found, the OK LED flashes and error bit 7 is set (see section 7.5.5, cell 2.13). If the encoder delivers no C signal, this input is to put on logical H (see section 5.3).
- The reference position P 'x' 23 can, for example, also be used to align the axes of a number of similar machines.
- Execution of the home function is not PID regulated; only controlled. The controle voltage is:

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

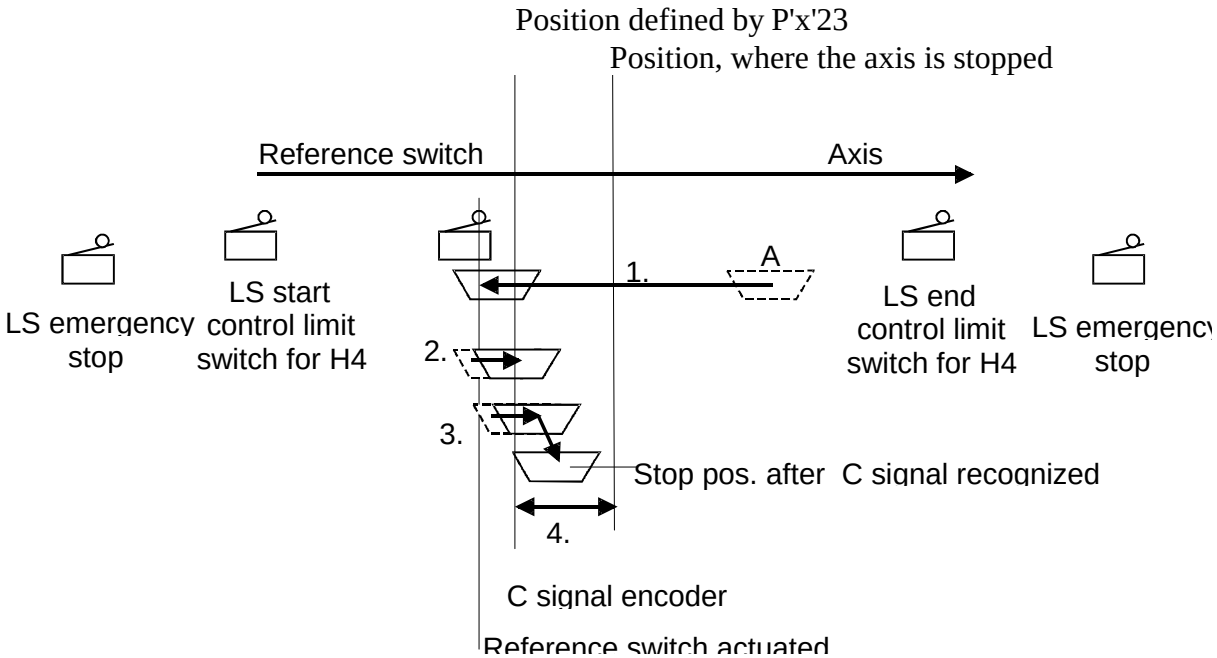


Figure 6.07

6.8 PID controller

All the regulating parameters of the digital position controller with 'velocity feedforward' and 'acceleration feedforward' can be modified during the movement ('on the fly').

Controller parameters:

Proportional factor	Parameter 50
Differential factor	Parameter 51
Sampling factor for D	Parameter 56
Integral factor	Parameter 52
Integral limit	Parameter 53
Integral mode	Parameter 16
Velocity feedforward	Parameter 54
Acceleration feedforward	Parameter 55
Dead band	Parameter 10

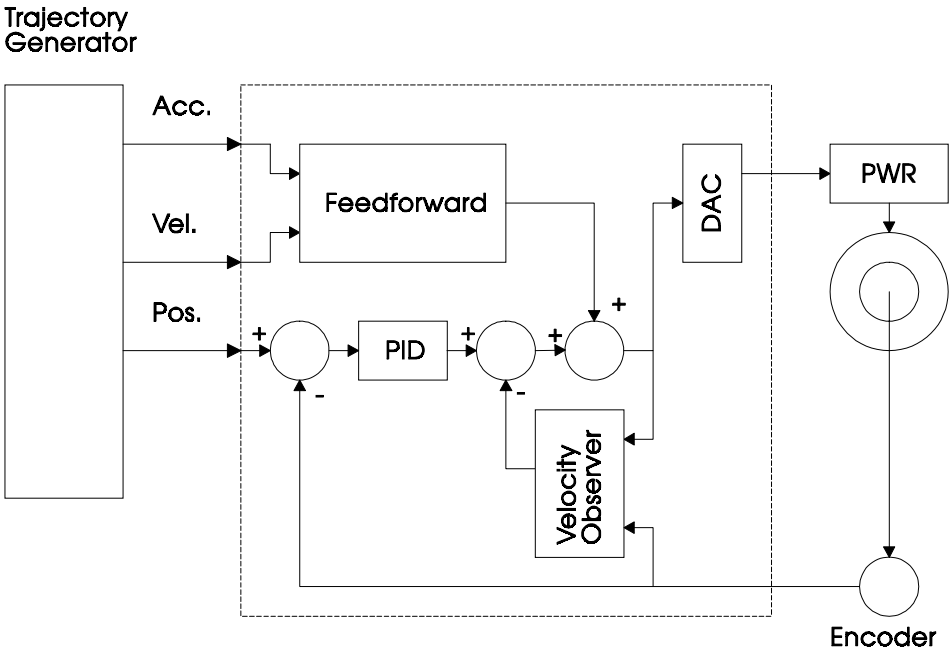


Figure 6.08

The trajectory generator produces all digital information needed to control the system (destination position, velocity, acceleration). The 'jerk' function is also generated (change in acceleration).

6.9 Encoder

The precise position of the axis is detected with an incremental shaft encoder or a linear encoder The position is counted in the "quadruple count" mode. In this way the four-way resolution of encoder partition is obtained.

6.9.1. Encoder type

Incremental encoders can produce different signals. The H4 module can process 24 VDC or 5 VDC encoders (for hardware details see Chapter 2 "Technical data").

Encoder type can be selected in parameter 92 and defined. This definition always applies to a pair of axes: X and Y or Z and W. Since LEDs A, B and C are activated according to the choice of encoder, they can be used to check that the setting is correct.

6.9.2. Direction of rotation

The sequence of signals A and B determines the direction of rotation, so that the current position is incremented or decremented accordingly. The count direction is 'positive' or 'negative' depends on which shaft end of axis the incremental encoder is fitted to and on its mounting position. It is possible to change the count direction by reversing signals A and B. The H4 module allows this, even without wiring alterations, by modifying parameter 08. In the diagram too, this allows for uniform wiring, regardless of the encoder position.

6.9.3. Format / units

The H4 module works directly in physical units. Parameter 01 selects the desired unit of operation (mm / inches / angular degrees / pulses). In this way it is possible to give for example a path of 233,56 mm directly to the H4 module. To optimize the way in which the PCD CPU and the H4 work together, while still achieving a high resolution, the 'virtual integer' format was chosen for floating point values, i.e. all values are transferred as integers with a virtual decimal place, which is defined in parameter 96.

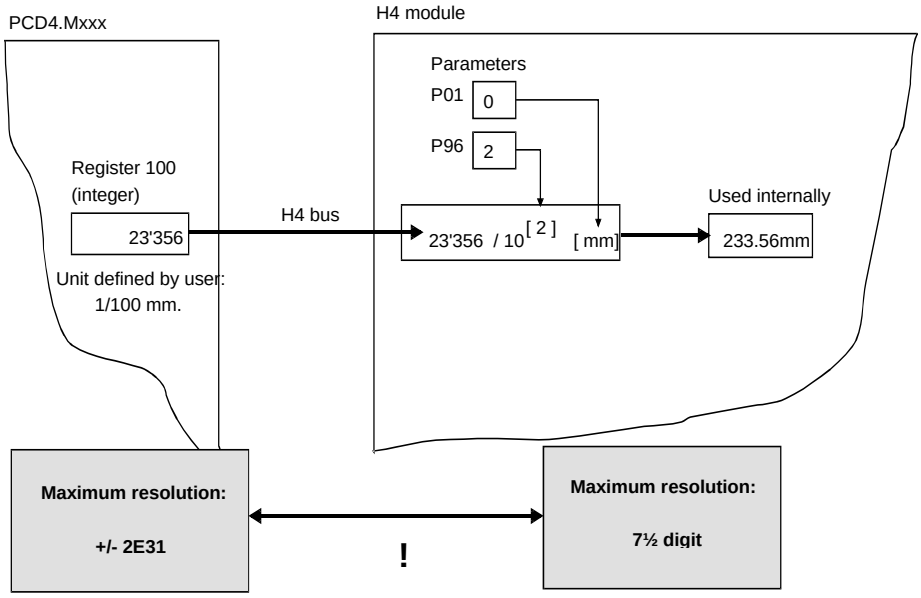


Figure 6.09

Integer values are not concerned from this conversion. In the command list all the instructions using the "virtual integer" format are marked with "VI".

The standard accuracy for floating point values is 7½ digits. That means, all values using this format, e.g. the position, are accurate to the 7th decimal. (Example: 0.001 mm accurate on 10 meters).

To read the position in Encoder-steps (without any conversion and accuracy lost) there are special instructions available (QPI 'x')

6.10 Backlash

Any axis with a spindle drive usually has a backlash. If the path direction is altered, the axis travels the required path minus the backlash: i.e. not exactly the path required.

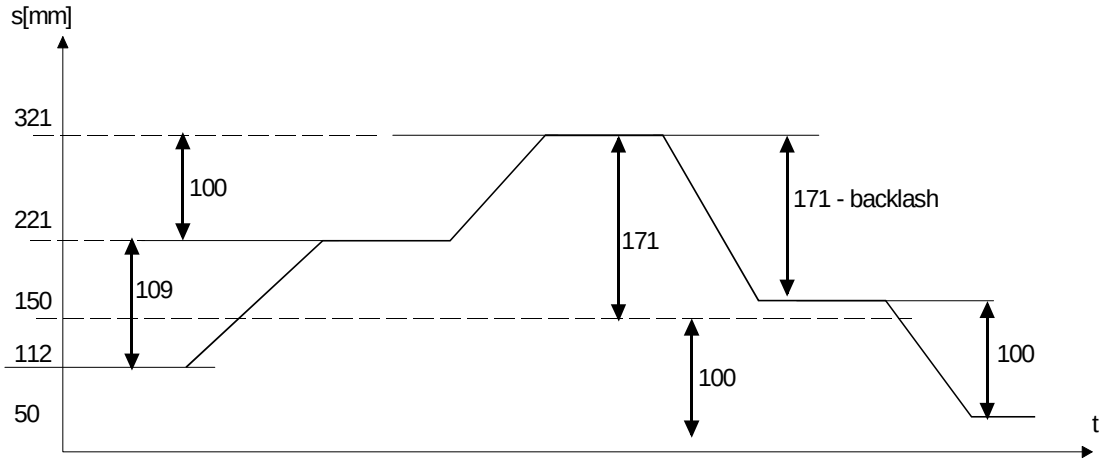


Figure 6.10

Backlash considerably reduces the accuracy of a machine. The mechanism used must either be free from backlash (which is often very expensive) or backlash compensation is required to set it to zero.

The H4 module is capable of this. The backlash can be defined in parameter 14 (see parameter list 7.6.4) and is added to the travel when there is a change of direction. If there is no change of direction, the backlash has already been compensated and no correction is made.

Velocity for backlash compensation

Parameter 63 can be used to set the velocity at which the backlash is compensated. In certain circumstances, this may be higher than the normal velocity, since it is only necessary to move the motor and spindle.

Range = 10... 100% of max. velocity (parameter 30).

6.11 Electronic gearing

When the Y axis is linked to the X axis, all path commands to the X axis are also executed on the Y axis. The transmission ratio can be defined in parameter 07, in this case X/Y. This link is only effective in one direction (Y accepts X commands and not vice versa). If a two-way link is required, these links must be set in both directions: link Y to X and link X to Y. When doing this, ensure that the second link has the reverse transmission ratio to the first (1/2 and 2/1). This electronic gearing is calculated as a linear interpolation, so that vector velocity SV and vector acceleration SA or SD apply.

Example:

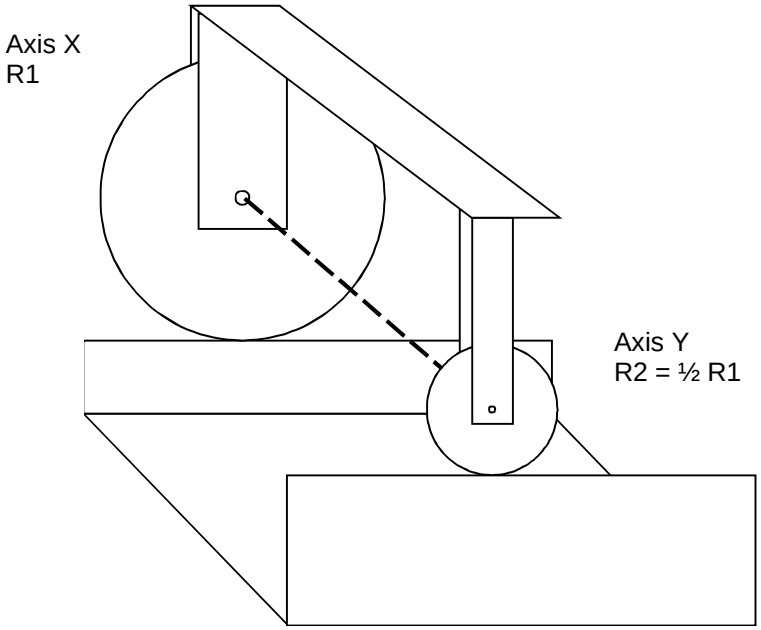


Figure 6.11

A machine has two axis drives which are located far apart and which must travel synchronously with each other. A mechanical link is difficult or very expensive. The answer is therefore two individual drives which travel synchronously with each other and which are connected as if by gears. The term used is therefore ‘electronic gearing’.

Figure 6.11 shows a virtual axis connection of both wheels. If it existed, the vehicle could not travel in a straight line, since the wheels are of different sizes. With 'electronic gearing' it is possible to equip these wheels, whose diameters differ, with individual drives and cause them to travel in the correct ratio synchronously with each other, so that the vehicle travels in a straight line. If X is selected as master, the necessary settings are P 'x' 06 = 2 (Y axis) and P 'x' 07 = Y/X.

6.12 "Trigger-out" signal function

Each axis has a "Trigger-out" signal available. This output allows a very accurate and fast reaction to be achieved, depending of axis position. If the axis exceeds the P value set by the SOx command (section 7.5.5, cell 2.14) the "Trigger-out" signal is activated. (This signal is available to the user at bus terminal 1 or 9 on the H4 module).

The position value which enables the "Trigger-out" signal is written to the H4 module with the SOx command. The value is used in the unit of measurement selected for the H4. If this unit is inadequate, the SOIx command can be used to load (or set) the position value to the exact pulse. When executing the SO.. command, the status bit (21 on the X axis) is deleted and the "Trigger out" signal is disabled. QSx (section 7.5.5, cell 2.12)

If the axis then overtravels the set position value in either direction, the "Trigger out" signal is activated and the status bit is set.

The trigger output polarity can be selected with parameter 62.

When the trigger position is reached and P62 = 0, the output is set high (positive logic). The inactive state is = L.

When the trigger position is reached and P62 = 1, the output is set low (negative logic). The inactive state is = H.

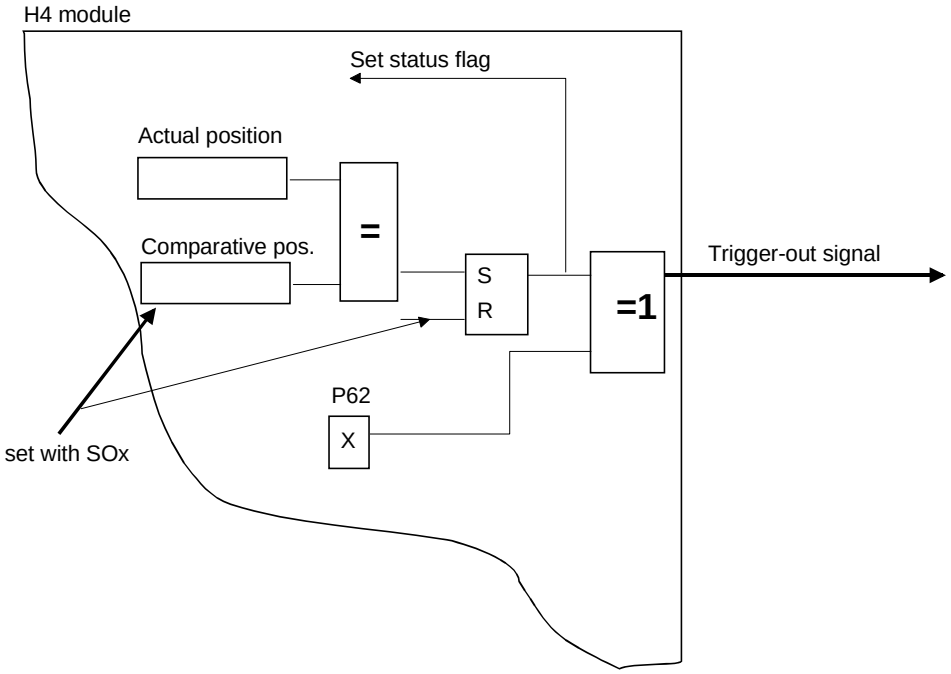


Figure 6.12

6.13 "Position capture input" signal function

There is one of these hardware inputs for each axis on the H4 module. If this function is enabled (SC 'x'), when the 'PCI' input is set High (bus terminal A or B) the value for the actual axis position is stored in the "Position Capture" register. This value can then be read from the H4 (QC 'x'). If a capture position has been recorded, the status flag 'C' is set for the corresponding axis (section 7.5.5, cell 2.12).

The following commands, parameters and flags belong to the "Position Capture" function:

SC 'x' / QC "x" Command
'PCI entered' flag (for X axis = flag 15)

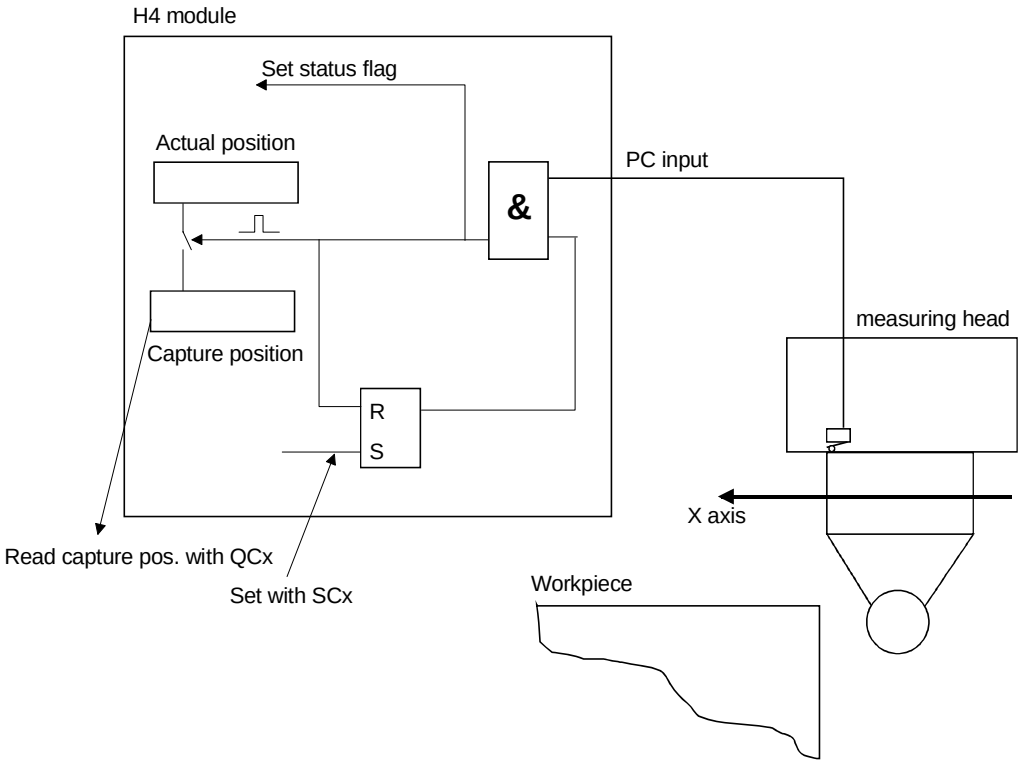


Figure 6.13

Example:

A measuring head on a cartesian robot is used to measure the edge of a workpiece. If the measuring head responds, the actual position is immediately stored. This can then be read from the H4 module.

6.14 "Change on the fly" function

Various parameters are calculated or taken into consideration during the execution of commands. If the command has been interpreted in the H4 module and transferred to the execution process, parameters can be modified, but will only take effect from the next command.

In the parameter list (section 7.6) commands marked ‘YES’ under the heading "change on the fly" affect the axis at any time. If they are modified during motion, they influence the axis immediately.

It is therefore possible, for example, to modify control parameters during motion and for them to influence the control system immediately. In jog mode, it is even possible to modify the jog speed and the velocity then changes “on the fly”.

6.15 Description of circular axis (position rollover)

Parameter Px05 is used to indicate whether an axis is linear (**n=0**) or rotating (**n>0**). A linear axis is characterised by limited motion, which is controlled by hardware limit switches. There are no limits for a rotating axis.

This parameter defines the position rollover (the period). The value 'n' defines the limits of the display range (for the 'query axis position' command). The start and end of the traversing range are at the same physical position. At the range limits, the display changes from the start value to the end value, or from the end value to the start value, depending on the direction of rotation. This means that the absolute or relative motion commands cause a maximum displacement of the axis equal to Px05 / 2 (half the period) in the direction of the shortest path (negative or positive direction).

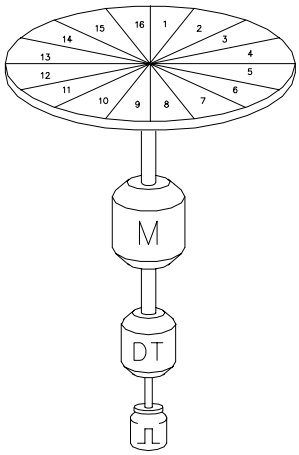
When working in mm or inches (see Px01 and Px03), the position rollover Px05 defines the circumference of the axis. When working in degrees, the position rollover Px05 is 360°. When working in steps, see example 1.

Example 1: A rotating plate is activated by a motor in 16 steps.

The encoder, with 2000 impulses/rev., is connected to the motor's axis. The motion is to be divided into 16 steps.

The parameters for axis 'x' will be:

Px01 = 2 (degrees)
Px02 = 2000 (impulses/rev. from encoder)
Px03 = 16 (16 steps from 0 to 360°)
Px05 = 16 (steps/ rev. of rotating plate



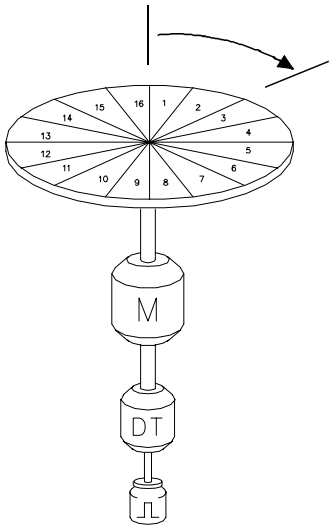
The motion commands will be expressed in steps:

xA = 2 axis 'x' will move to the 2nd section of the table (direction of motion is clockwise)

with :

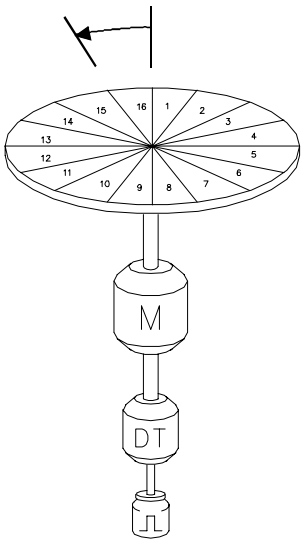
xA = 18 axis 'x' will move exactly to the same position as for xA = 2 (18 - 16 = 2) with the same motion direction.

So that the position will be indicated the same when using the command QPx (query position of axis 'x') in both cases.



But with :

xA = 15 axis 'x' will move to the 15th section of the table but with direction of motion is counter clockwise, i.e. via the shortest path.



Example 2: Command with steps a translation system formed by two pinions.

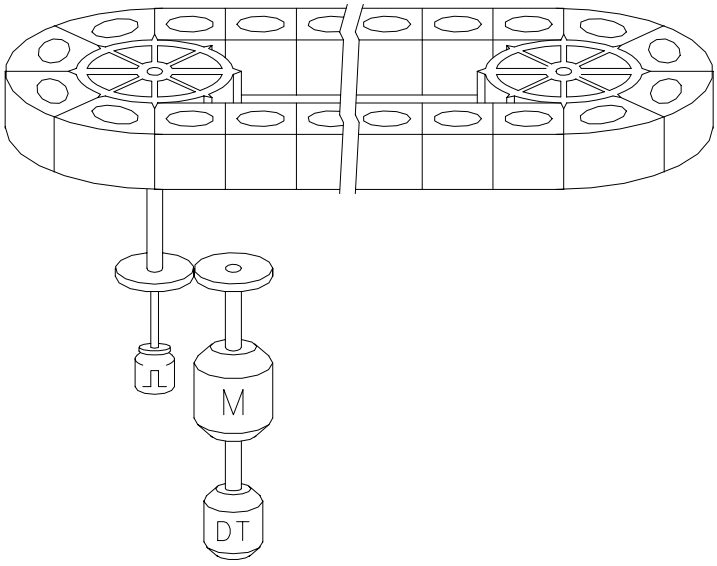
The pinions have 8 teeth.
The 2000 impulses/rev. Is on the axis of a pinion.
The chain has a development equal to 48 interacts-tooth.

The parameters for the axis will be.

Px01 = 2	(units are degrees)
Px02 = 2000	(impulses/rev. from encoder)
Px03 = 48	(48 degrees/rev.)
Px05 = 48	(steps/ rev.of translator)

The motion commands will be expressed in steps:

xA = 36
xA = 56.7
xR = 49
xR = -3.67



Notes

7. Programming

7.1. Introduction

The "PCD4.H4..Motion Control" unit has a large number of commands and parameters. The parameters for module, system or axis settings can be modified in various operating situations. Apart from the motion control commands, there are also special commands for system control and program control. All commands and parameters can be modified or read by accessing in two different ways.

a.) from the PCD CPU, using standard FBs via PCD bus.
b.) from the PC via the front panel connector on the H4.

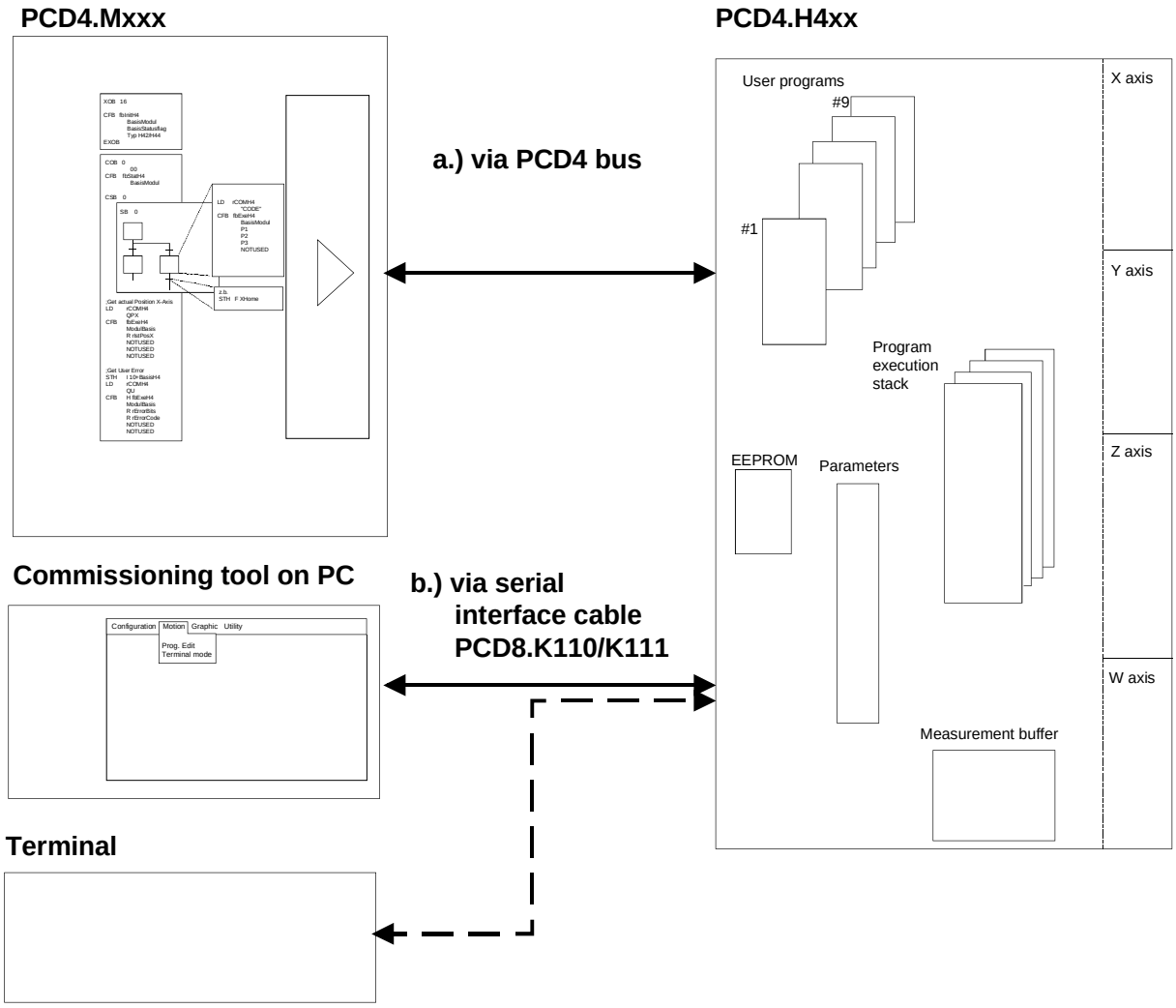


Figure 7.01

7.2 Programming concept

Programming the H4 module can be divided into two basic methods:

a) Programming the H4 module with the CP tool

b) Programming the H4 module via PCD user software

In both cases, however, the following applies:

- The user can adjust and fine-tune the module with various commands or parameters.

- Entire motion programs can be loaded into the H4 module or transmitted to the module for execution. (Program Mode). It is, of course, also possible to execute individual commands or modify individual parameters (Immediate Mode). Various user motion programs can be stored in the H4 module and then recalled for execution as needed.

a) Programming the H4 module with the CP tool

"CP" refers to the commissioning/programming tool, which runs on a personal computer (PC) and is connected with the PCD4.H4 via a serial interface. This tool has been specially developed for the H4 module and can execute all the functions of the H4. The CP tool not only edits motion programs and loads them into the H4 module, it can also upload from H4 module and store them on the harddisk. Apart from the program, parameters can also be handled by the same way.

b) Programming the H4 module via PCD user software.

SAIA-Burgess Electronics has provided standard FBs to drive the H4 module from PCD user software. This allows the user to execute any functions required in the H4 module simply by calling the relevant FBs. In this way, complex motion programs can be created in PCD application software. Parameter modification is also very easily solved by calling FBs in the application software. Special functions also allows entire travel programs to be loaded into the H4 module. Travel programs which have been edited with the CP tool can be safeguarded by uploading them from the H4 module, storing them in the PCD, and then loading them again into the H4 module.

Synchronization of programs

Loaded programs can be started directly (and independently of each other) with RUN'p' (p = program no. 1…9).

BREAK'p' can be used to stop them after execution of the current command. If several commands are executed in 'Blended mode', they are treated as a single command and BREAK cannot interrupt them individually.

The RUN function can also be executed from hardware with the start input (terminal 15). This is also possible for the BREAK function by means of the stop input (terminal 7).

Parameter 95 selects the program to be affected by the start and stop inputs.

If programs are to be nested (max. up to 4 levels), a program can be called and executed from another program as follows:

e.g.
```
1 - XA20
2 - RUN3
3 - XA40
4 - END
```

If a separate program must start between two motions with blended execution, note that the RUN command is executed at the moment when blending begins, i.e. just before position 20 (depending on P'x'44 and the velocity. See also chapter 6.6 for 'Blended move).

For position-dependent actions, the trigger output can be used.

If programs only need to be processed up to a specific line, a breakpoint can be set with the STOP command:

e.g.
```
1 - XA20
2 - STOP
3 - XA40
4 - END
```

A program stop can cancelled directly with RUN'p', or from another nested program, or with the hardware start input.

To view a running program, the command QL'p' (p = program no.) can be used to query the program line currently executing. If the program has finished (or not yet started) the value 0 is returned.

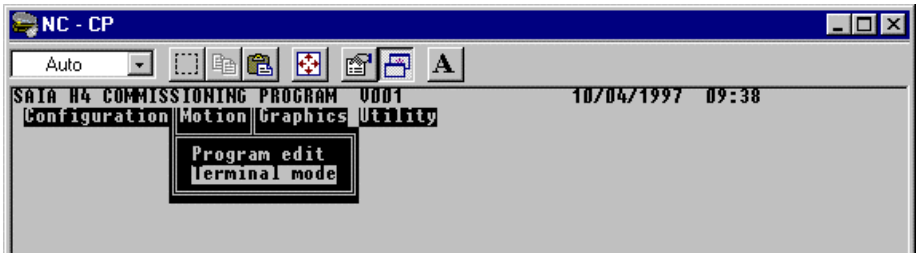
7.3 Programming with the CP tool (Commissioning / Programming tool)

The 'Commissioning / Programming Tool' (CP) is a software tool for IBM compatible PCs. It runs under the DOS operating system. Installation and programming are very simple.

7.3.1 Installation

Installation consists of copying all files from the diskette to the hard disk, e.g. into a directory \SAIA-H4. This requires approx. 2 MByte free memory on the harddisk and 400 kBytes RAM. The program is started by calling it from the DOS prompt: 'CP' <CR>. On starting CP, the SAIA logo appears.

Pressing any key displays the following pull-down menu.



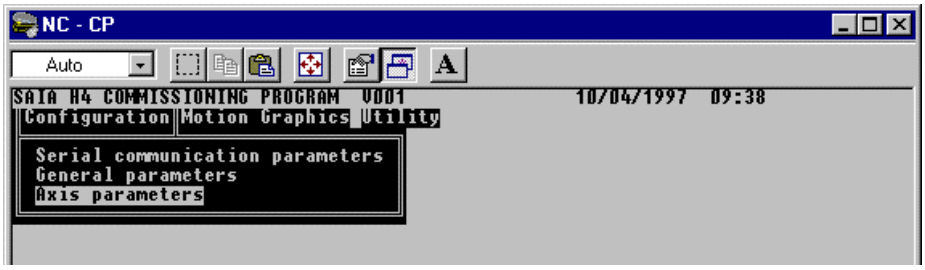
The cursor can be moved with the arrow keys <↓>, <↑>, <←> and <→> to select the required menu.

The menu is then opened with <CR>. See section 7.3.3 Menu explanation, for clarification of menu items.

<ESC> = Menu
Pressing the <ESC> key activates the pulldown menu.

The **F1** key is not shown. It is used to call help.

7.3.2 Menu overview




```
- 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
```

If a menu is selected, additional function keys will be shown.

For example, with 'Program Edit' the following keys are available:

The following function keys can be selected:

<F4> = From/To H4 mod
Function key 4 defines the destination and source of data for the two functions F2 and F3 (PC or H4).

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

- Reads the .cnf (config) file into the CP working memory

when <F4> = H4:

- Reads the configuration from the H4 into the CP working memory

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

- Saves the configuration from the CP working memory to the hard disk.

when <F4> = H4:

- Loads the configuration from the CP working memory into the H4.

7.3.3 Menu explanation

The functions described below have been divided into groups according to the main menu.

Configuration

'Serial communication parameters'
The PC COM port can be selected. (COM1: or COM2:). The communications parameters are fixed. (e.g.: 9600,8,E,1)

'General parameters'
The relevant general parameters for the H4 module can be set. These are parameters 90 ... 98.

Function key description:

If all parameters have been set correctly, they can be saved with <F3>. The destination to which they are saved must previously have been defined with the <F4> key (H4 or PC). These parameters can be read back with <F2> (<F4> selects whether from H4 or PC).

'Axis parameters'
All axis parameters are displayed. Some parameters depend on how the others have been set. It is therefore advisable to complete the parameters from top to bottom. The limits are displayed in the edit window.

Pressing function key <F8> selects the next axis. To save, see 'General Parameters'.

Motion

'Program edit'
The program editor works offline, i.e. programs can be written, viewed, modified and saved on the harddisk without being connected to the H4 module. If a connection is established with the H4 and selected with the <F4> key, programs can be read from the H4 or loaded into the H4. Max. 9 programs can be loaded in the H4xx module. (See also chapter 6.2.2 for H4 program memory). The commands are stored in the H4 module, but not yet executed. Note that max. 1000 lines per program can be inserted.

To enter a command, press the <Enter> key (<↵>). An edit window opens at the lower margin of the screen and allows the required command to be entered. See section 7.5 ‘Command list’ and section 7.6 ‘Parameter list’ for possible commands or parameters and their function.

The following function keys can be selected:

<F4> = From/To H4 mod
Function key F4 defines the destination and source of data for the two functions F2 and F3 (from/to PC dis or from/to H4 Mod).

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

- Reads the .prg (program) file into the CP working memory

when <F4> = from/to H4 Mod

- Reads a program from the H4 into the CP working memory

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

- Saves a program from the CP working memory to the hard disk.

when <F4> = from/to H4 Mod

- Loads a program from the CP working memory into the H4.

<F8> = Analyse

- Checks the lines of code entered for syntax errors and indicates in a window which lines have errors.

<F5> = Delete line

- Deletes the current program line and move the following lines up and store the deleted line in the buffer (Cut function).

<F6> = Insert line

- Inserts an empty line at the current line position and moves the following lines down.

<F7> = Paste line

- Inserts the last deleted line in the buffer, provoked by F5, at the current line position and moves the following lines down.

Before loading a program into the H4 module, the syntax is checked and any lines containing errors are indicated. If it was not possible to execute the loading correctly, an error message appears. A correctly loaded program can be started in terminal mode.

<F4> function key must always be set correctly to ensure the down/up loading procedure in the correct places.

Commands which, according to the command list, can only be executed in immediate mode, are not accepted in the program.

Remark: If P94 is set for axes X and Y only, all parameters or program commands for axes Z and W will not be accepted in the 'Program Edit' mode.

'**Terminal mode**'
In contrast to the program editor, commands or parameters entered here are executed immediately (Immediate: see also section 7.5.1 Syntax explanation, box 'Execution mode'). Further function keys appear in this window.

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

'ESC Menu': Exists terminal mode, returns to menu.

F9 Previous commands reverse

F10 Previous commands forward

The entry line has a 10-line ring buffer. This can be scrolled backwards <F9> or forewards <F10> as a simple means of reusing previously entered command lines. The last entry in the command line can also be retrieved with <Enter> (<↵>).

Several instructions can be entered on the same time in terminal mode and will be executed after confirming by <CR>.

Example: QPX QVX QEX <CR>

Execute program:
Having loaded a program into the H4, it can be started with the command "RUN" + program number. However, make sure that parameters have been set and loaded and that the relevant axes have been correctly initialized (check 'Enable' and 'Home').

Concerning the axis status window which appears on top, see section 7.5.5 cell 2.12. The active bits are indicated with a letter or character (e.g. 'Positive limit switch reached', '+' and 'h').

Graphics

'Graphics'
This mode allows the user to observe various functions of the axis. The user can employ these graphics when setting up the axis. If the user changes the PID parameters, the graphics allow him to verify the effects directly. The modified control parameters are only transmitted to the H4 when data capture started by <F4>/<F5>. After a test motion, the CP tool needs a short time to transmit the captured measurements from the module to the PC and to represent them. A maximum of 4 curves can be sampled for the axis selected by <F8>. The following possibilities are available: 'Actual position', 'Destination position', 'Following error', 'Set point Velocity', 'Acceleration', 'Analogue output voltage DAC'.

The optimal set point parameters discovered from this are not copied automatically into EPROM. The user has to perform the command 'WRITE' to do that.

Most of the function keys and other information on this screen are self-explanatory. Here briefly are a few additional notes:

Function by <F5> performs the following Steps:

- send PID parameters to H4 module
- start the program selected by <F3>
- start data acquisition

Concerning the axis status window which appears top right, see section 7.5.5 cell 2.12. The active bits are indicated with a letter or character (e.g. 'Positive limit switch reached', '+' and 'h').

Utility

'Language change'
The language of the CP tool can be selected here. (ENG, ITA, [GER, FR])

'Colour change'
The foreground and background colours of different windows can be selected. Monochrome can also be chosen. By pressing the arrow keys <←> or <→> the different window areas can be selected. Pressing <CR> and then <Space> displays the colours available; <F4> monochrome, <F5> default setting.

'Graphics colour change'
The foreground and background colours of the graphics window can be selected.

'Select colour for Text in graphic'
Colours can be selected for texts in the graphical display to achieve the optimum graphical representation. See above for colour change.

'DOS Shell'
Temporarily exits CP without losing data from working memory. (Type 'Exit' to return)

'Quit'
Exits CP. On termination, data in working memory is lost. 'Quit' can also be achieved with <Ctrl> + <z>.

Notes

7.4 Programming with FBs

This part of the manual explains how to use the standard FBs provided by SAIA-Burgess Electronics for the H4 module. It describes the FBs and how to call thems. Individual H4 commands and parameters are explained in sections 7.5 'Command list' and 7.6 'Parameter list'.

7.4.1 Introduction

The user creates an individual program (see Figure 7.04 "User program") by using the three basic FBs provided for the H4 module (see Figure 7.4 "H4 FBs"). First, the user has to call 'fbInitH4' (in XOB16 for example) to initialise the H4 module. In the COB, the 'fbStatH4' must be tested cyclically to get status of axes. (Status Flags for each axis can be used further in the user program). For programming motions or setting parameters, the user has to load first the corresponding instruction to the command code register (rComH4) and call the 'fbExecH4' to execute the instruction. A simple overview of this can be obtained by considering figure 7.04. Two FBs, 'fbUpLdH4' and 'fbDnLdH4', have been provided for transferring whole programs (section 7.7).

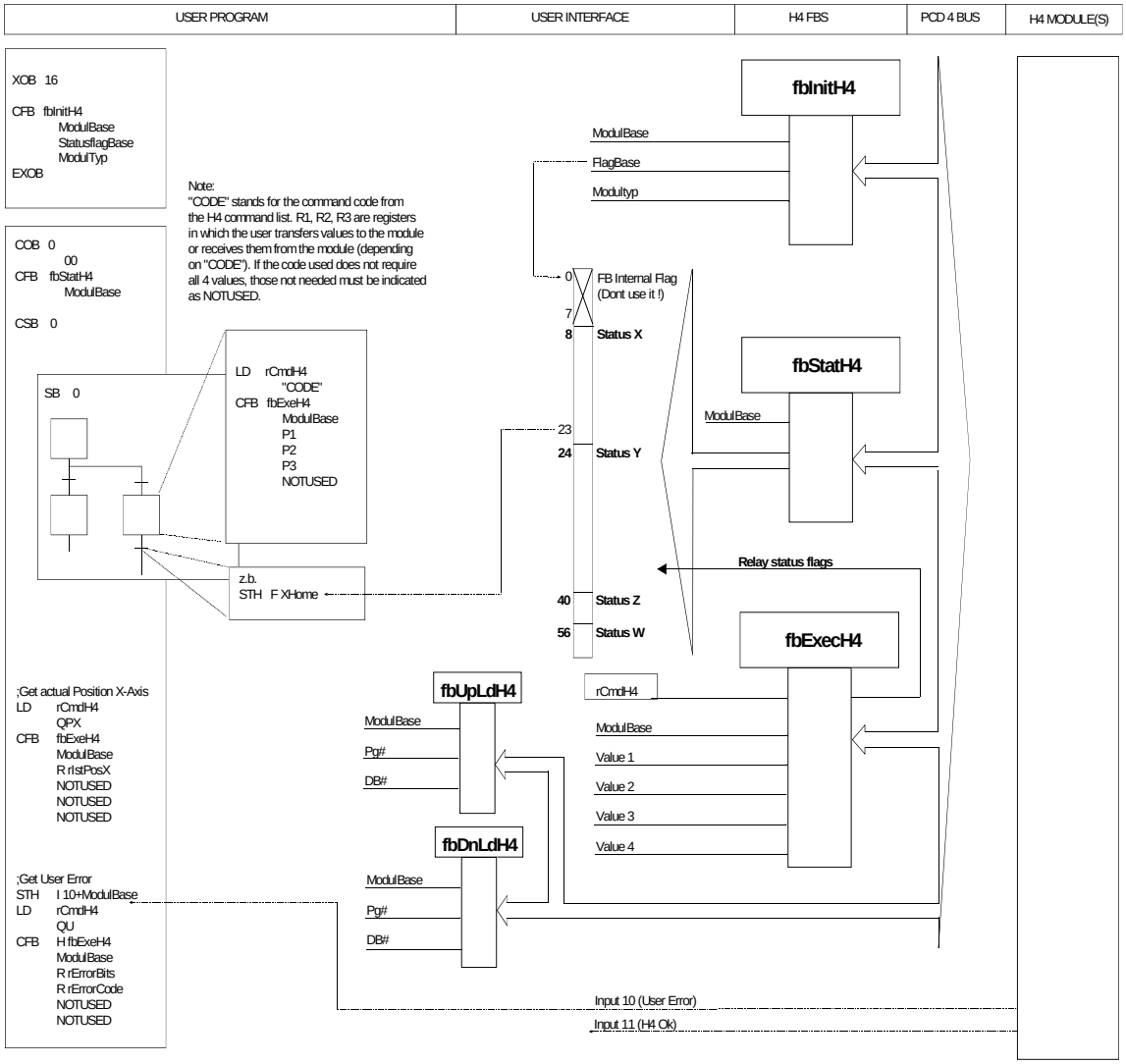


Figure 7.04

A user program must be assembled, linked and then downloaded into the CPU. (See explanations in section 7.4.4).

7.4.2 Addressing the H4 module

This module, like all other PCD4 modules, needs 16 bus addresses. The lowest module address is the base address. All other addresses are used accordingly. The user must therefore define the module base address and the base address of flags (see 'fbInitH4'-FB).

What individual addresses signify: (+ base addresses)

	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"	

The direct input signals from H4 bus 'Axis in position' are normally used in the debugger for control purpose. For programming with FBs, we recommend the same signals (flags) delivered by the function block 'fbStatH4' because of timing reasons.

7.4.3 Preset status flags

(Status flags: see section 7.5.5 cell 2.12)

When certain commands are executed with function block (FB) 'fbExecH4', some status flags are updated immediately (preset), before FB 'fbStatH4' refreshes the status flags. This makes the FBs easier to use, it is not necessary to call FB 'fbStatH4' between two calls of 'fbExecH4'. Cyclical asynchronous calling of 'fbStatH4' is sufficient. The status flags are refreshed only by the FbStatH4. For this reason, before using the status flags in the user program, make sure that they are previously refreshed by the 'fbStatH4'.

The following status flags are preset:

- Axis in pos. (8)
- Immediate command in execution (9)
- Capture pos. reached (15)
- Trigger position reached (21)
- Homing successfully concluded (23)

7.4.4 Software library with function blocks (PCD9.H4..)

The standard FB package for the H4 module comprises 3 files:

H4DEF.SRC This file concerns only the base addresses of resources (Flags, Reg., FBs). The user can define here the base addresses. This file will be called with the command $INCLUDE in the file H4FB.SRC

H4EXTN.DEF This file contains all necessary symbols which can be used in the user program. This file must be included in the user program with the command $INCLUDE.

H4FB.SRC This file contains the FBs source code and all symbols which are only used for the FBs. This file must not be modified or edited with SEDIT.

FB nesting levels required:	1
Size of H4FB software (lines):	< 4200

7.4.5 Assembling and linking files

The following picture shows how the FBs of the H4 module can be integrated into the user software.

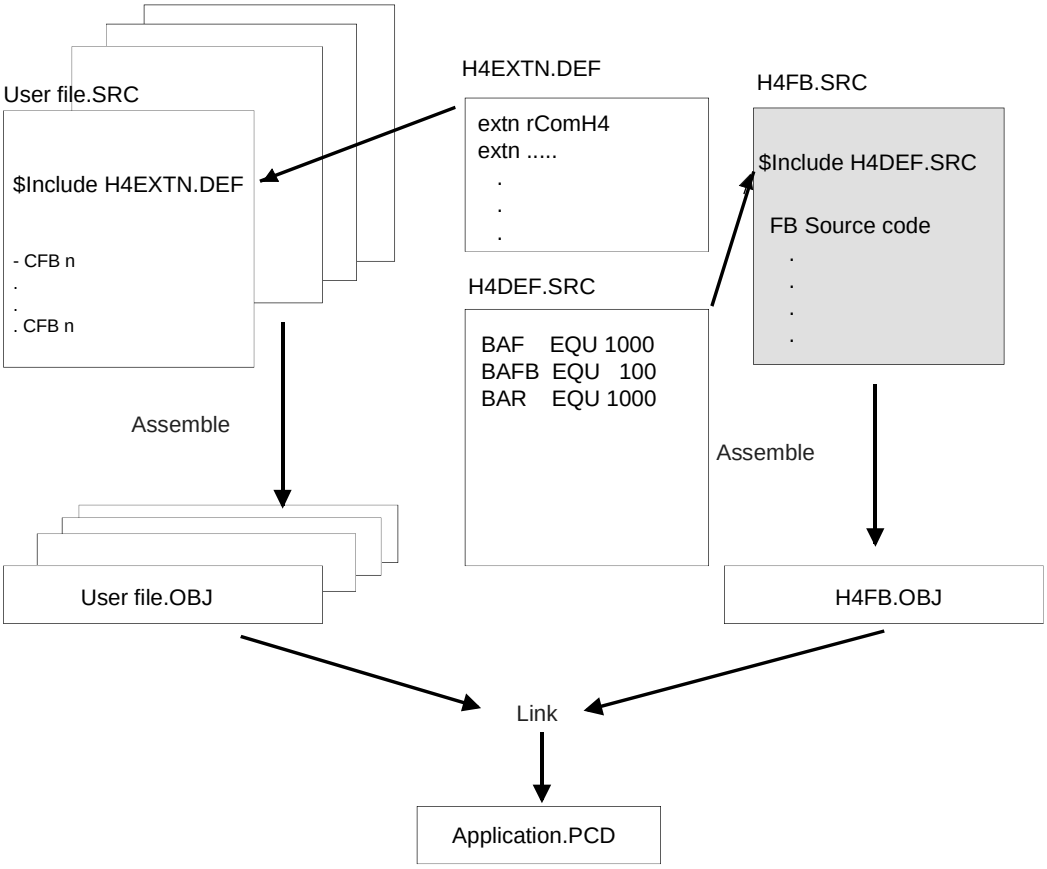


Figure 7.4.4

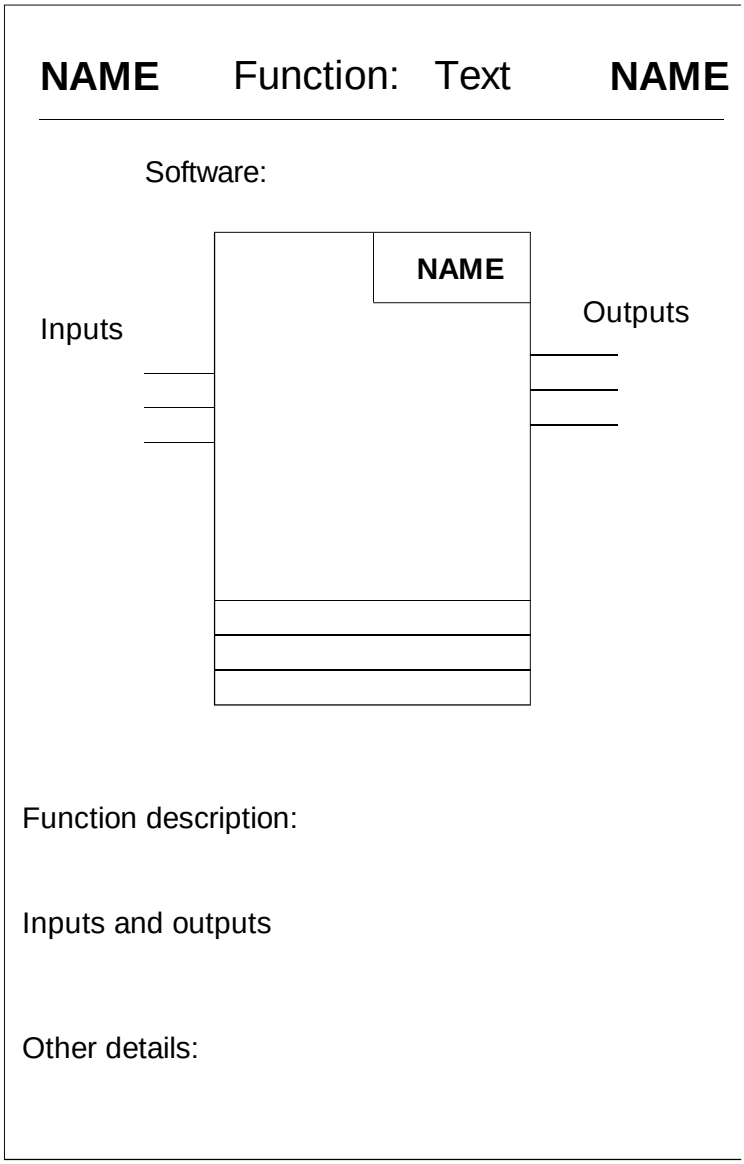
The base addresses of the H4 FBs are defined in file H4DEF.SRC by the user, the user must therefore modify this file. Files H4FB.SRC and H4EXTN.DEF should **never** be modified by the user.

To call the various H4 functions from the user program, it must also have access to the definition file H4EXTN.DEF. The H4 definition file is therefore included in each user file (with $include H4EXTN.DEF).

The user can define his project definitions in a separate definition file which also must be included in his program. If he wants to use also the H4 definitions (BAF, BAR and BAFB) in his program, he has to include the file H4DEF.SRC either in his project definition file or directly in the user program.

7.4.6 Description of FBs

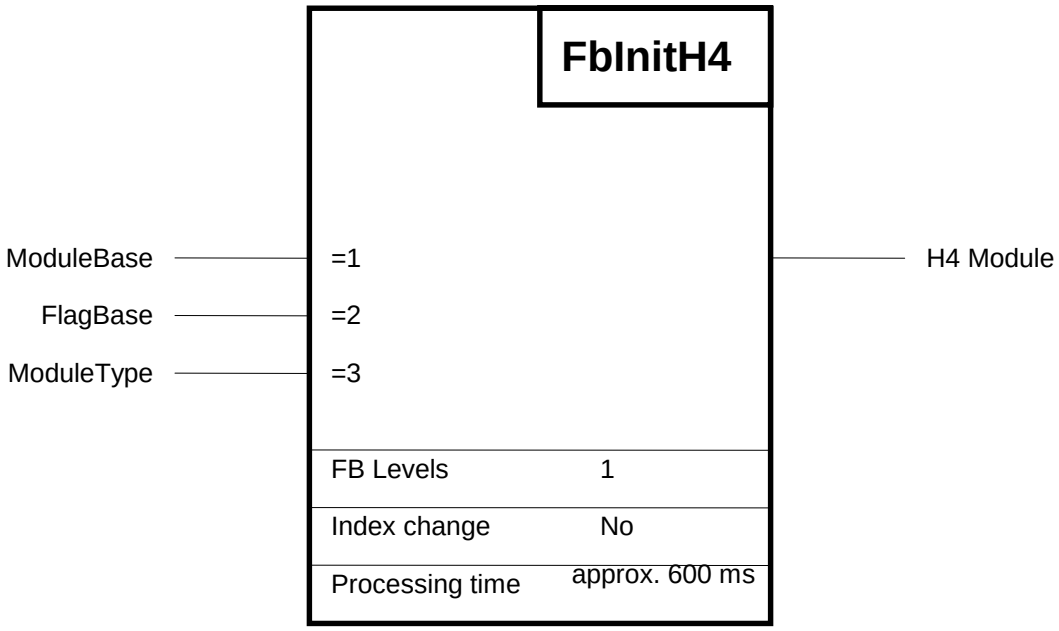
For the sake of simplicity, all FBs are described in the same way. The figure below clarifies the layout of the following pages.



The format and unit of values entered are according to parameters P01 and P96 (see section 6.9.3)

FbInitH4 Function: - Initialize H4 module FbInitH4

Caution:
This FB measures the CPU performance. It must therefore only be called in the coldstart routine (XOB16).



Function description:

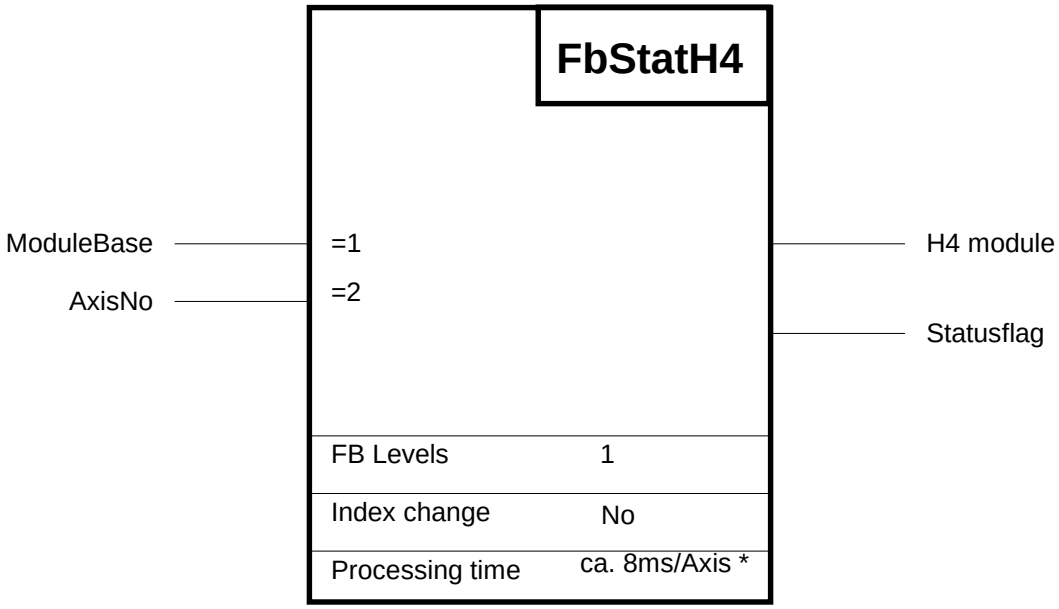
This function is used to initialize the H4 module and the PCD resources used by it. During 'PowerUp' (XOB16) one 'fbInitH4'-FB must be called for each H4 module.

If several H4 modules are used in one PCD4 system, flag base addresses must be checked to ensure that the flag ranges of each module do not overlap.

Description of I/Os:

Symbol	Description	Para-meter	Media		
			Type	Format	Addr. range
ModuleBase	Module base address	yes	K	Integer	0 -512
FlagBase	Flag base address	yes		Integer	0 -8192
ModuleType	Select module H420/H440	yes	K	Integer	2 or 4

FbStatH4 Function: - Read status of H4 module FbStatH4



* dependent on CPU power

Function description:

This function reads the status register for each axis from the H4 module. The status register is then output to the status flags and can therefore be queried by the user. The base address for the status flags is taken from the function 'fbInitH4'. The offset and meaning of the individual status flags are described in the command list, section 7.5.3 (2.11) or 6.2.6. The status flags are refreshed only by this FB. For this reason, before using them in the user program, make sure that they are previously refreshed.

Description of I/Os:

Symbol	Description	Para-meter	Media		
			Type	Format	Addr. Range
ModuleBase	Module base address	yes	K	Integer	0 -512
AxisNo	Axis number	yes	-	Integer	0 - 15 (0-F)

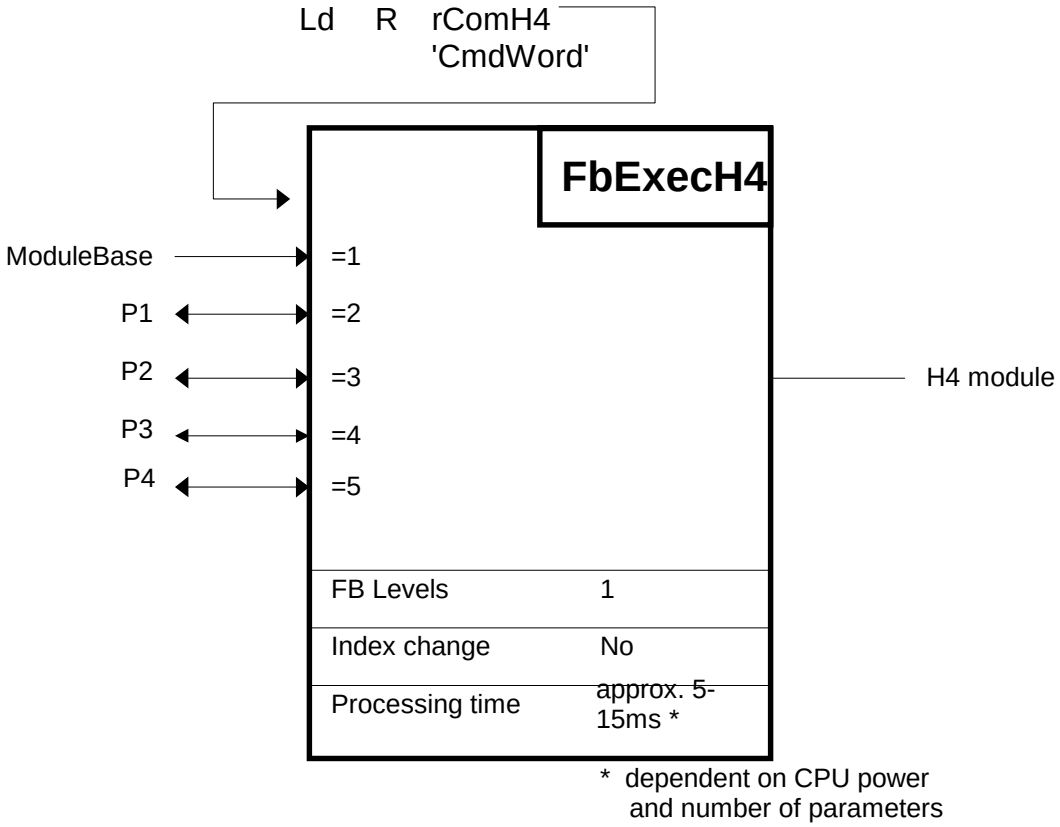
Axis number:

00h : one axis/cycle
01h : axis X
02h : axis Y
04h : axis Z
08h : axis W
0Fh : all available axes in one cycle

Combinations are also possible:
e.g. Ch or 12d = : The status flags of axes Z and W are refreshed. (h = hex; d= decimal)

If an axis is addressed which does not exist, no function is executed.

FbExecH4 Function: - Execute Command to H4 FbExecH4



Function description:

The function 'fbExecH4' is the command execution function for all commands and H4 parameters. This function can read or write H4 module data. Before calling this FB, the register 'rComH4' must be loaded with the appropriate instruction word. The FB itself reads all the information required for execution from the instruction word. Section 7.5 lists all possible instruction words.

Description of I/Os:

Symbol	Description	Parameter	Media		
			Type	Format	Addr. range
ModuleBase	Module base address	yes	K	Integer♥	0 -512
P1*	Parameter 1	yes	R	Integer♥	0 - 4095
P2*	Parameter 2	yes	R	Integer♥	0 - 4095
P3*	Parameter 3	yes	R	Integer♥	0 - 4095
P4*	Parameter 4	yes	R	Integer♥	0 - 4095

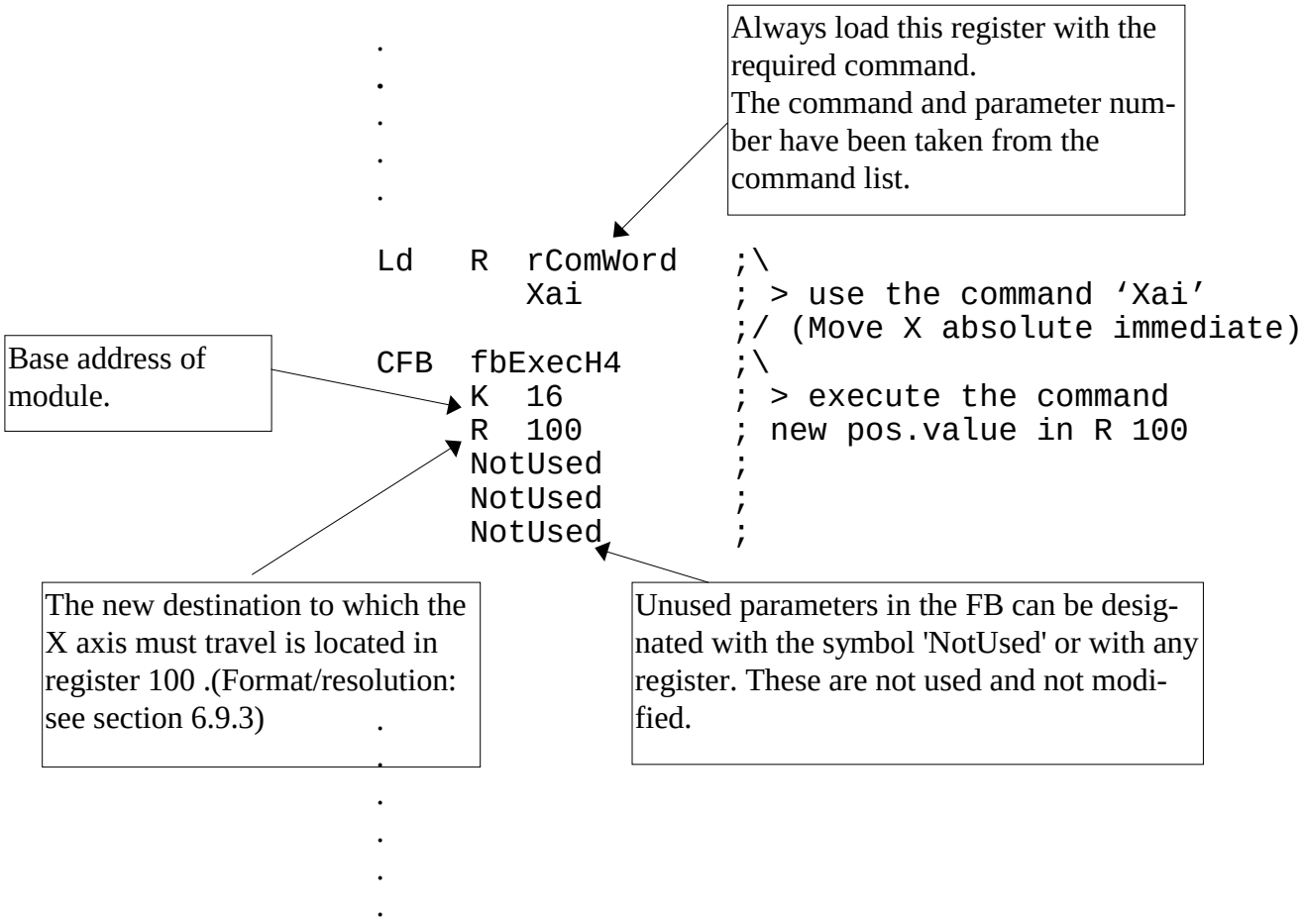
* Refer to command list for the number of parameters.

♥ Integer or "virtual integer", depending on command (see section 6.9.3 and command list section 7.5)

Example: see next page

Example for FbExecH4:

This example shows an absolute motion (immediate) of axis X (see section 7.5.4 cell 1.10)



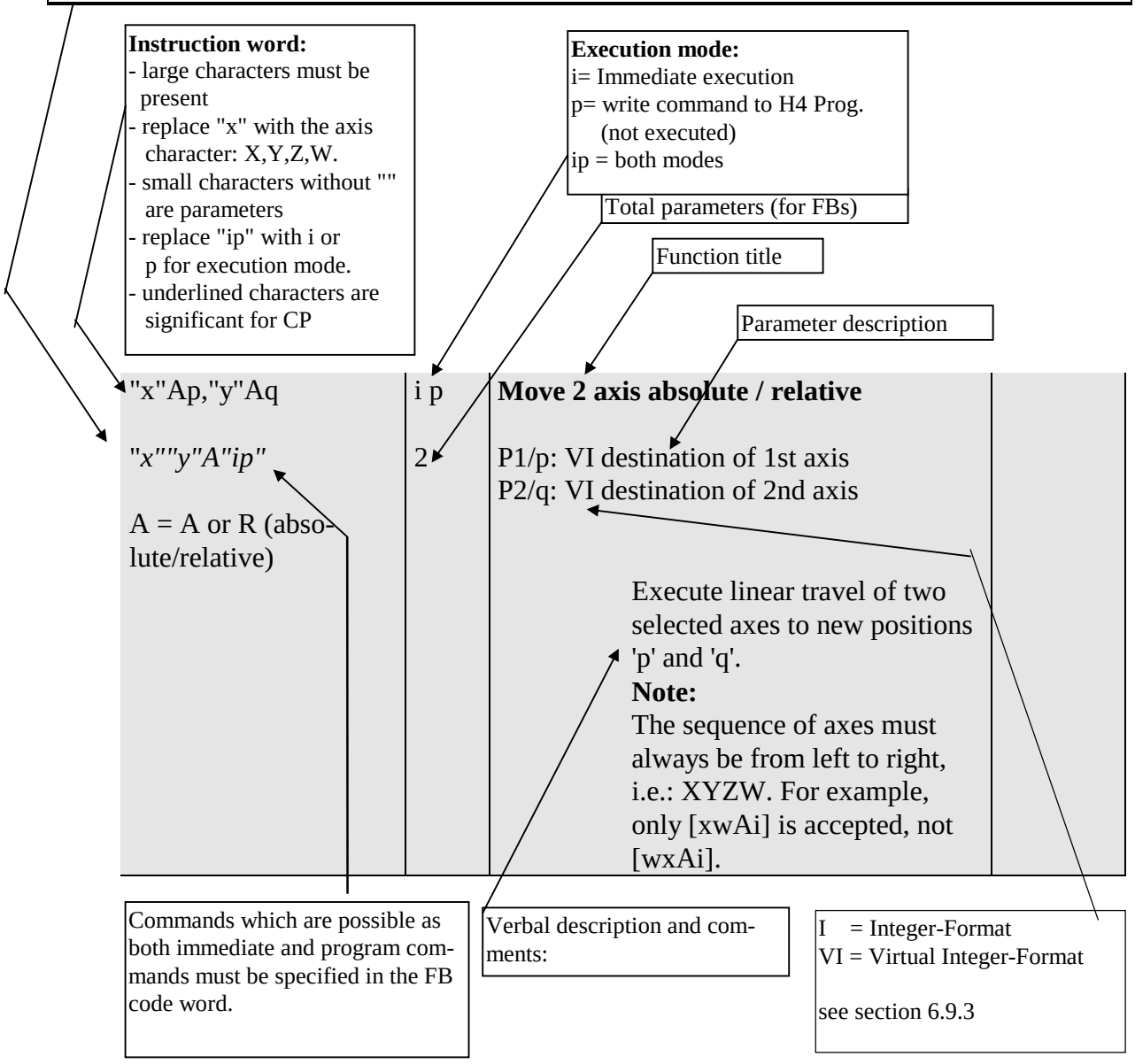

```
.
.
.
.
.

Ld   R  rComWord   ;\
        Xai        ; > use the command 'Xai'
                   ;/ (Move X absolute immediate)

CFB  fbExecH4      ;\
     K  16         ; > execute the command
     R  100        ; new pos.value in R 100
     NotUsed       ;
     NotUsed       ;
     NotUsed       ;

.
.
.
.
.
.
```

Notes

7.5. Command list

7.5.1 Syntax explanation of command list

The command list has been designed for programming with the CP tool or with FBs. Instruction words in Roman type apply to the CP tool, those in italics are for FBs.



7.5.2 Summary of command groups:

Group:		Examples:
7.5.4	Motion commands	(XA100 / cir... / Home / Zero / ...)
7.5.5	Axis control commands:	(Enable / Kill / Query Pos. Acc. / Decel.)
7.5.6	Special commands:	(Out / Lock / Unlock)
7.5.7	Parameter commands:	(read & write Param.)
7.5.8	Program control commands:	(run / step / ...
7.5.9	Program structure commands:	(for / next / ...)
7.5.10	Program list commands for terminal (CP only)	(list)
7.5.11	Program build commands	(open / close / erase)

7.5.3 Alphabetical command and parameter list with cell reference of the following tables

Query capture position	2.10-11
Query current execution line	5.7
Query memory lines free	8.6
Query position	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
Travel one axis	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 Motion commands

	Instruction word for CP *for FBs*	**i p** Mode	**Code designation** **Brief explanation**	
1.1	**ZERO "x"** *ZERO"x""ip"*	i p 0	**Set axis to zero** Set position of axis to zero. Regardless of where the axis is currently located, the position value is lost and set to '0'.	
1.2	**HOME "x"** *HOME"x"*	i 0	**Synchronize axis** Synchronizes the axis with the mechanism. The method for locating the Home Pos. is defined in parameters 20-24. This small executive routine takes place in the H4, placing no load on the PCD CPU. By querying the 'Home' flag from the H4 status flags, it is possible to see when the 'Home' command is terminated. The ref. position (param. 23) is then loaded as the actual position. Home and Jog moves are not PID controlled (see Chapter 7.6.3)	Relay: The status flag (23 for X axis) is deleted.
1.3	**J+ "x"** *JUP"x"*	i 0	**Jog positive** (manual move) Jog mode: In this operating mode, the axis can travel in controlled mode (without PID). The axis accelerates with P'x'43 to the defined velocity and moves until a Jog stop command is sent or until the pos. Limit switch (HW or SW) is reached. For the velocity in jog mode, see commands 'RAPID' and 'NORMAL'.	

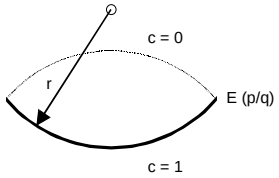
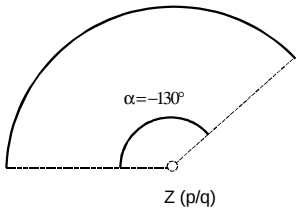
7.5.4

1.4	**J- "x"** ***JDN"x"***	i 0	**Jog negative** Travels until the negative LS (HW/SW) is reached. See 'J+ command	
1.5	**JS "x"** ***JS"x"***	i 0	**Jog stop** Jog mode operation of the relevant axis is stopped with the deceleration. P'x'44.	
1.6	**SS "x",s** ***SS"x""ip"***	i p 1	**Set motion speed** P1/s: VI [unit/sec] Never 0 max = V_{max} (P 'x' 30) Default = 1/10 V_{max} Sets the motion speed of the axis. Only applies for the travel of one axis.	
1.7	**SV s** ***SV"ip"***	i p 1	**Set vector speed** P1/s: VI [unit/sec] Never 0 Sets the vector speed of the motion. The axis velocities are calculated vectorially. Only applies for the interpolated travel of several axes. 2 axes: $V = \sqrt{V_x^2 + V_y^2}$ 3 axes: $V = \sqrt{V_x^2 + V_y^2 + V_z^2}$ 4 axes: $V = \sqrt{V_x^2 + V_y^2 + V_z^2 + V_w^2}$	
1.8	**SA a** ***SA"ip"***	i p 1	**Set motion acceleration** P1/a: VI [unit/sec^2] Never 0 max = A_{max} (P 'x' 33) per axis Loads the acceleration for interpolated motion.	
1.9	**SD d** ***SD"ip"***	i p 1	**Set motion deceleration** P1/d: VI [unit/sec^2] Never 0 max = A_{max} (P 'x' 33) per axis Loads the delay for interpolated motion.	

7.5.4

1.10	**"x"Ap** ***"x"A"ip"*** A = A or R (absolute/relative)	i p 1	**Move axis absolute / relative** P1/p: VI [unit] The max. positioning range is -2^{31} ... $+2^{31}$ pulses ($-2{,}147*10^9$ ... $+2{,}147*10^9$) However, resolution is limited by the floating point format to 7½ places. For details see section: 6.9.3. Travel with the selected axis to position 'p'. If a LS is reached, the axis is killed and the error bit 2 as well as the corresponding axis statusflag h rsp. s is set. The move out of the LS is only possible with the Jog comd.	Relay: The 'In Pos' flags are deleted
1.11	**"x"Ap,"y"Aq** ***"x""y"A"ip"*** A = A or R (absolute/relative) same for all axes e.g. XAp, ZAq XRp, WRq	i p 2	**Move 2 axes absolute / relative** P1/p: VI position/path of X axis [unit] P2/q: VI position/path of Y axis [unit] The linear interpolated travel of two selected axes to new positions 'p' and 'q'. (Applies to absolute motion only; for relative motion 'path' applies). **Note:** Axes must always be used in the order x,y,z,w. Therefore, for example, only [xwAi] is accepted, not [wxAi]	Relay: The 'In Pos' flags are deleted
1.12	**"x"Ap,"y"Aq,"z"Ar** ***"x""y""z"A"ip"*** A = A or R (absolute/relative) same for all axes	i p 3	**Move 3 axis absolute / relative** P1/p: VI position/path of X axis [unit] P2/q: VI position/path of Y axis [unit] P3/r : VI position/path of Z axis [unit] The linear interpolated travel of three selected axes to new positions 'p', 'q' and 'r'.	Relay: The 'In Pos' flags are deleted.
1.13	**XAp,YAq,ZAr,WAs a** **XYZWA"ip"** A = A or R (absolute/relative) same for all axes	i p 4	**Move 4 axis absolute / relative** P1/p: VI position/path of X axis [unit] P2/q: VI position/path of Y axis [unit] P3/r : VI position/path of Z axis [unit] P4/s : VI position/path of W axis [unit] The linear interpolated travel of four axes to new absolute positions 'p', 'q', 'r' and 's'.	Relay: The 'In Pos' flags are deleted.

7.5.4

1.14	**CIRr,c,"x"Ap,"y"Aq** **CIR"x""y"A"ip"** A = A or R (absolute/relative)  	i p 4	**Move circle with radius (abs. / rel.)** P1/r: VI radius [unit] P2/c: I direction: 0 = clockwise 1 = counter-clockwise (when pos. coordinate system has been defined) P3/p : VI end position/path of 1st axis P4/q : VI end position/path of 2nd axis [unit] Travel in an arc from actual pos. to new positions 'p', 'q', with a radius 'r' and direction 'c'. Note: If the arc is >100° a point is calculated every 5°. If the arc is <100° the number of points calculated is always 20. Caution: The radius must total at least half the distance (to the end position) otherwise the command is not executed.	Relay: The 'In Pos' flags are deleted. End position abs./relative
1.15	**CIRa,2,"x"Ap,"y"Aq** **CIR"x""y"A"ip"** A = A or R (absolute/relative)  	i p 4	**Move circle with angle (abs. / rel.)** (Centre mode) P1/a: I angle [unit] P2/c: 2 centre mode P3/p : VI centre of circle on 1st axis P4/q : VI centre of circle on 2nd axis [unit] Travel in an arc from the actual pos. with angle 'a', whose centre is at pos. 'p', 'q'. If the angle is negative, the direction of the arc turns Note: (see CIR command with radius)	Relay: The status flag (8 for X axis) is deleted.

7.5.5 Axis control commands

7.5.5	**Instruction word for CP** *for FBs*	**i p** Mode	**Code designation** **Brief explanation**	
2.1	**FO=x** *FO*	i 1	**Feed Override** P1/x: I in % from 1 to 120 Executes the motion velocities of the whole module in percent. After power-up, this value is set at 100%. This function does not affect jog mode operation. The value must be selected before any motion (i.e. not 'on the fly')	
2.2	**ENABLE "x"** *ENABLE"x"*	i p 0	**Enable axis** 'Enable..' switches on the position controller for the selected axis and maintains the position as active. It also sets the output 'Ampl. Ena' (Terminal 0/8). The controller and output are switched off again with 'Kill..'	
2.3	**KILL "x"** *KILL"x"*	i 0	**Disable axis** 'Kill..' switches off the controller and the output (Terminal 0/8), sets the analogue output to 0V and sets the program pointer to 0. See also 'Enable..'.	
2.4	**QP "x"** *QP"x"*	i 1	**Query actual position** P1: VI actual position [unit] Reads the current position of the selected axis.Actual value of axis. (e.g. for display on MMI)	

7.5.5				
2.5	**QPI "x"** ***QP"x"***	i 1	**Query actual position in impulses** P1: I +/- 2E31 (regardless of P96) [Imp x 4] Reads the current encoder position of the selected axis. The actual value is resolved in encoder impulses. (Mode x4) + Explanation under 2.10.	
2.6	**QV "x"** ***QV"x"***	i 1	**Query actual velocity** P1: VI velocity of axis [unit/s] Read current velocity of axis.	
2.7	**QE "x"** ***QE"x"***	i 1	**Query actual position error** P1: VI position error of axis [unit] Read current following/ position error in state.	
2.8	**SC "x"** **SC"x""ip"**	i p 0	**Set capture position** Activate function: 'Store position' when module input 'PCI' (Kl.A/B) has responded. The value can be queried with the QC command. Setting the capture function deletes the status flag ‘PCI captured’.	Relay: The status flag (8 for X axis) is deleted
2.9	**QC "x"** **QC"x"**	i 1	**Query capture position** P1: VI stored position [unit] Read the value recorded with the 'Capture' function. Querying the status flag (flag 15 on X axis) can be used to identify whether the value has already been captured. (see also SC command)	

7.5.5				
2.10	**QCI "x"** **QCI"x"**	i 1	**Query capture position in impulses** P1: VI stored position in impulses $\pm 2^{31}$ (regardless of P96) Read the value recorded with the 'Capture' function directly from the encoder. Note: In one revolution, an encoder with 1000 pulses generates 4000 impulses in the H4 module (in 'x 4' mode). (see also SC and QC commands)	
2.11	**SP "x", p** **SP"x" "ip"**	ip 1	**Set position** P1/p: VI [unit] Position that is loaded as actual position max. Pos. range -2^{31} .. $+2^{31}$ impulses. The axis position is lost and overwritten with the position p	

7.5.5

2.12	**QS "x"** **special** ***CFB FbStatH4***	i 0	**Query status** The axis status is displayed online in the terminal mode, the QS'x' instruction is therefor not especially used. (The format is Hex Code Decimal) 16 status flag / axis The flag base address is defined in FBInitH4. Read status flag of each axis. Flag : 0-6 internal FB use only +BAF 7: Fatal Error H4 (no link to H4 module). Read access only. The following applies for the X axis: per axis 8: P axis in Pos. 9: E immediate command in execution 10: h axis in hardware LS 11: s axis in software LS 12: F Following error 13: W Following error warning 14: 0 theoretical speed = 0 15: C Capture pos. reached 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 reached 22: V Position overflow 23: H homing successfully concluded *for Y axis 'Flag+16'* *for Z axis 'Flag+32'* *for W axis 'Flag+48'* FB: Querying of status flags is executed by a special FB which must be called cyclically in the program.	

7.5.5

2.13	**QU** ***QU***	i 2	**Query user error code** The user error is displayed online in the terminal menu. With QU this error message and the user error input I 10 on PCD4 bus will be cleared. P1: I 16 error bits P2: I error code Read the word 'User error code' : Error bits: bit 0: EEPROM not ready 1: EEPROM checksum error 2: LS triggered 3: (Reserved internal) 4: Max. Following error with stop 5: Calculated pos. overrun 6. Max. Following error 7: Error in home routine 8: LS active 9: Execution buff. overflow 10: Wrong command in prog. prog. no. see bit 12-15 11: Checksum error in prog. 12: 20 ¦ LSB 13: 21 ¦ Program has an 14: 22 ¦ error 15: 23 ¦ MSB for prog. no. Error codes: 0: no error 1 Start prog. with err. bit 11 2 Start prog. with err. bit 10 3 More than 4 progs. in RUN 4 Start prog. whichdoes not exist 5 Start prog. on line which does not exist 6: A prog. in RUN cannot be deleted 7: Move an axis which is disabled. 8: 'Home' instructed for a disabled axis. 50: Data overflow in PCD interface 100: Impossible command from PCD 200: No H440 identified 255: GENERAL ERROR	L = stored Errors deleted by: L L L online L ENA L ENA online online L Run 'p' L

7.5.5

2.14	**SO "x",p** **SO"x""ip"**	i p 1	**Set output compare** P1/p: VI position [unit] The H4 module has a 'Trigger Out' output, which is activated when the position set with this command is exceeded. On execution of this command, the output is deleted.	Relay: The status flag (21 for X axis) is deleted.
2.15	**SOI "x",p** **SOI"x""ip"**	i p 1	**Set output compare in impulses** P1/p: I position $\pm 2^{31}$ (regardless of P96) See SO command for description.But here the position is indicated in impulses. Caution: Note that encoder resolution is multiplied by 4. (See chapter 6.9 Encoder)	Relay: The status flag (21 for X axis) is deleted
2.16	**RAPID** ***Rapid***	i 0	**Use rapid speed for jog** Set the velocity for Jog move to P'x'32. If the axis is already moving with 'Normal speed', the velocity is changed 'on the fly' with the acceleration P'x'43 to 'Rapid speed'. 'Rapid' and 'Normal' affect all axes, although they can have different speeds (P'x'31 and P'x'32) for each axis..	
2.17	**NORMAL** ***NORMAL***	i 0	**Use normal speed for jog** Sets jog speed to normal (parameter 31). If the axis is already moving with 'Rapid speed', the velocity is changed 'on the fly' with the deceleration P'x'44 to 'Normal speed'. See 'Rapid' command for additional information.	

7.5.5

2.18	**DRIFT "x"** ***DRIFT"x"***	i 0	**Drift compensation** (offset adjustment) Executes drift compensation for the selected axis. Adds an offset voltage to the analogue output. This calculated compensation offset is depending on the position error (in stand) and the proportional factor P'x'50. Therefore no integral factor should be used that a position error can be measured and the compensation is effective.	

7.5.6 Special commands

	Instruction word for CP ***for FBs***	**i p** Mode	**Code designation** **Brief description**	
3.1	**OUT "x",v** ***VOUT"x"***	i 1	**Output to DAC** P1: VI v: ±0.00 ... 10V The analogue output which the control circuit needs to drive the amplifier can be set, controlled directly with the command 'Out..'. In this case the controller of the corresponding axis must be disabled ('Kill'). Note: This command is used mostly for tests and commissioning purposes only.	
3.2	- ***SPLOCK***	(i) 0	**Disable serial port (for FBs only)** Locks the CP port.	
3.3	- ***SPUNLOCK***	(i) 0	**Enable serial port (forFBs only)** Unlocks the CP port again.	

7.5.7 Parameter commands

	Instruction word for CP *for FBs*	**i p** Mode	**Code designation** **Brief description**	
4.1	**Pxn=y**	ip 3	**Set parameter** x: axis (for FBs: x=1,....W=4) n: parameter number y: I/VI parameter value (see parameter list) Load parameter 'n' of selected axis with value 'yyy'	

	P"n" (0-89,92)
	Use for H4 prog.
		p
		
		3
			Set axis parameter in 'Program' mode
			P1: axis
			(for FBs: x = 1,....W = 4)
			P2: parameter number 'n' = 0 - 89,92
			P3: I/VI parameter value (see parameter list)
			Load parameter ('n' and P2) of selected axis (P1) with value from P3
			Note:
			For nn the instruction word (e.g. P01) must agree with parameter 2, i.e. they must be the same.
				Example:
				
				`Ld rComH4 P01`
				`CFB fbExecH4 K 32 R rr (1) R rr (01) R rr(0)`

	P"n" (90-99)
	Use for H4 prog.
		p
		
		3
			Set general parameters in 'Program' mode
			
			P1: parameter number 'n'=90-99
			P2: I parameter value (see parameter list)
			Load parameter ('n' and P2) of selected axis (P1) with value from P2
			Note:
			For nn the instruction word (e.g. P01) must agree with parameter 1, i.e. they must be the same.
				Example:
				
				`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 in 'Immediate' mode**
			
			P1: parameter value (see parameter list)
				Example:
				
				`Ld rComH4`
				`Py01W`
				`CFB fbExecH4`
				`K 32`
				`R rr(2)`

	(n= 0-98)
	for general parameters (90-98) without axes, 'x' is omitted.
		1	Load parameter 'n' of selected axis with value from P1	`Ld rComH4`
				`P96W`
				`CFB fbExecH4`
				`K32`
				`R rr (3)`

4.2	**Pxnn?**
	
	P"x""nn"R
	
	(see parameter list).
		i
		
		1
			Get parameter
			
			P1/nn: parameter value
			(Destination register)
			Read parameter 'nn' of selected axis. See parameter list for content of P1.
				`Ld rComH4`
				`Px50R`
				`CFB fbExecH4`
				`K32`
				`R rr`

4.3	**EREAD** ***EREAD***	i 0	**Read from EEPROM** Read parameters from EEPROM into H4 RAM. This is also executed during power-up.	
4.4	**EWRITE** ***EWRITE***	i 0	**Store in EEPROM** Store parameters from H4 RAM in EEPROM. This is executed automatically by CP during a 'Parameter Download'.	

7.5.8 Program control commands

	Instruction word for CP *for FBs*	i p Mode	Code designation Brief description	
5.1	**RUN p** ***RUN"ip"***	i p 1	**Run program** P1: I program number 1-9 Starts the selected program at line 1. When the 'Run' command is started from a program, remember that 4 is the maximum number of programs which can run simultaneously (see also P98).	
5.2	**Gpn** ***G"ip"***	i p 2	**Run program on line N** P1/p: I program number (1-9) P2/n: I program line (1-1000) Starts the selected program at line n.	
5.3	**STEP p** ***STEP***	i 1	**Step program** (Inactive with run) P1/p: I program number (1-9) Executes an individual program command.	
5.4	**BREAK p** ***BREAK"ip"***	i p 1	**Break program** P1/p:I program number (1-9) Stops the pogram after the current command. (If several 'blended' commands are being executed, they count as a single command). 'Run' makes the program continue.	
5.5	**HALT** ***HALTALL***	i 0	**Stop all program and motion** Stops all motion immediately (all axes) with the max. deceleration. The control system remains active and positions are retained. This command can only be reversed with the 'Resume' (or 'Kill') command.	

5.6	**RESUME** ***RESUME***	i 0	**Resume command Halt** Releases any module halt state initiated by the 'HALT' command. A motion interrupted by the halt command is brought to its conclusion.	
5.7	**QL"p"** ***QL"p"*** replace „p" with the prog. number (1-9)	i 1	**Query current execution line** P1: I program line (1-1000) Outputs the current program line. If the program is at an end (or not yet started) the value returned is 0.	

7.5.9 Program structure commands

	Instruction word for CP *for FBs*	i p Mode	Code designation Brief description	
6.1	**FOR n** ***FOR***	p 1	**Starts a repeat block** P1/n: I number of loops (0 - 32767) Marks the start of a repeat block, which will be repeated n times. This command can be nested to a depth of 8 levels.	
6.2	**NEXT** ***NEXT***	p 0	**Ends a repeat block** Marks the end of a repeat block.	
6.3	**GOTO n** ***GOTO***	p 1	**Jump to program line** P1/n: I program line (1 - 1000) Jumps to a specific line in the program.	
6.4	**GOSUB n** ***GOSUB***	p 1	**Jump to subroutine** P1/n: I program line (1 - 1000) Calls a subroutine at line n. This command can be nested to a depth of 8 levels.	
6.5	**RETURN** ***RETURN***	p 0	**End of the subroutine** Marks the end of the subroutine.	
6.6	**STOP** ***STOP***	p 0	**Stop program** Stops the own program on the line where the stop instruction is written (wait endless) until a new RUN or STEP command is received. Acts like a break point.	

6.7	**WAIT n** ***WAIT***	p 1	**Wait** P1/n: I wait time 0 - 65535 [msec] While the program is being processed, WAIT executes a pause lasting n msec. Note: Insertion of a WAIT 0 between two motion commands suppresses the blended move function.	
6.8	**END** ***END***	p 0	**Identifies the end of program** Every H4 program must end with this command.	

7.5.10 Program list commands for terminal (CP only)

	Instruction word for CP *for FBs*	**i p** Mode	**Code designation** **Brief description**	
7.1	**Lpn=.....**	i	**Function of list: Set program line** Overwrite line n of program p with the following command character. The changes are stored with the 'close' command. For FBs see 'OPEN'	
7.2	**Lpn?**	i	**Function of list: Get program line** Display line n of program p in command character.	
7.3	**Lpn,m?**	i	**Function of list: Get area of program line** Display line n to m (max. 20 lines) of program p in command character.	

7.5.11 Program build commands

	Instruction word for CP *for FBs*	**i p** Mode	**Code designation** **Brief description**	
8.1	- ***OPEN"p"*** Replace "p" with prog. number (1-9).	i 1	**Open program for edit** P1: I program line (1 - 1000) Opens the selected program, so that commands can then be written in the program. Note: This happens automatically in CP with the Lpn = ... command and therefore requires no special instruction for CP.	Example: `Ld rComH4 OPEN5 CFB fbExecH4 K32 Rrr(1)`
8.2	**CLOSE** ***CLOSE***	i 0	**Close and save program under edit** Close program currently open and save it. Also required when a program has been modified with Lpm = ... If the edited program is currently being executed, the CLOSE command is not executed.	
8.3	**Ep** ***EP***	i 1	**Erase program** P1/p: I program number (1-9) Erases the selected program. If this program is running, the command is not executed. See also bits in 'User Error'.	

8.4	**Epn** -	i 2	**Erase program line** p: program number (1-9) n: program line (1-1000) Erase a line in the selected program. The erased line remains empty. Subsequent lines don't move up. This command cannot be executed with FBs.	
8.5	**Epn,m** -	i 3	**Erase area of program lines** p: program number (1 - 9) n: program line start (1 - 1000) m: Program line end (n - 1000) Erase an area of lines n to m in the selected program. This command cannot be executed with FBs.	
8.6	**QM** ***QM***	i 1	**Query memory lines free** P1: I number of free program lines (0 - 1000) Query how many program lines remain free in the edit buffer. P2: I free memory in the H4 module in words (max. approx. 11K words, representing approx. 3000..4000 program lines) depending of the used commands.	

7.6. Parameter list

7.6.1 Module parameters (general)

Abbreviations used in the list:

U = base Unit
s = second
V = Volt
mV = milli Volt

(See also explanations in the section 7.5.1, Syntax explanation of command list)

Param. no. per axis	Description	Unit	changed on the fly	default	For-mat	Values :	ip mode
94	Module type	-		2 4	I	2: for H420 4: for H440	i
95	Program number to be executed with the inputs 'Start' and 'Stop'.	-		1	I	1-9: for program numbers	i
96	Number of decimal places; only for virtual integer values via FBs	-		3	I	0-6: number of decimal places	i
90	Edge of the 'Start' signal at which the program starts	-		positive	I	0: positive 1: negative	i
91	Edge of the 'Stop' signal at which the program stops	-		positive	I	0: positive 1: negative	i
92	Encoder voltage P'x'92 sets X and Y; p'z'92 sets axes Z and W.	-		5V	I	0: 5V 1: 24V	i
97	Angle from which blended move is executed.	degrees		0°	I	0-181°	i
98	Check whether several programs are accessing to one axis.	-		1		0: no check 1: check	i

Per default, the check is active and when different programs are started proceeding on the same axis, only the first program is executed and from the second program an error message 'Axis locked' will appear.
When selecting 'No check', no more control will be done and the user is responsible to execute the program in correct order.

ip mode: i 'Immediate' only (cannot be used in a program)
ip 'Immediate' + 'Program' mode

7.6.2 Machine parameters

Param. no. per axis	Description	Unit	change on the fly	default	For-mat	Values :	ip mode
01	Axis unit	U		mm	I	0: mm 1: inch 2: ang. degree 3: impulse	i
02	Encoder impulses/revolution	-		1000	I	0..65535	i
03	Path/encoder revolution (in units according to P01)	U		5	VI	7 digits 0 - 100’000	i
04	Encoder count direction	-		positive	I	0: positive 1: negative	i
08	Analogue output polarity	-		positive	I	0: positive 1: negative	i
30	Maximum velocity with V_{out} = 10V	U/s		200	VI	7 digits 0 - 150kHz*P03/P02	ip
33	Maximum acceleration (positive & negative)	U/s^2		1000	VI	7 digits 0 - 1'000'000	ip
40	Positive position limit software limit *)	U	yes	0	VI	7 digits $\pm 2^{31}$ steps 0 = no limit	ip
41	Negative position limit software limit *)	U	yes	0	VI	7 digits $\pm 2^{31}$ steps 0 = no limit	ip
11	Following error-limit error signal	U	yes	2	VI	7 digits 0 - 8192*P03/P02	ip
12	Following error-limit warning signal	U	yes	0,5	VI	7 digits 0 - 8192*P03/P02	ip
13	Following error action	-	yes	stop	I	0: no stop 1: stop	ip

*) If no software limit has to be define, set P40 and P41 to 0 to deactivate software limit function.
It is possible to define one software limit to 0 and the other to a certain position. In that case, the defined position 0 is taken in account.

Note: The software limits are related to the counting position. This means, that the SW LS are displaced if the position value is modified with the commands Zero'x' or SP'x'. The 'home routine' (search for reference position) handles the SW LS in the same way as the HW LS. If the reference switch is placed out of the SW LS, the software limit function has to be deactivate to execute the home routine.

7.6.3 Jog and homing

Param. no. per axis	Description	Unit	changed on the fly	default	For-mat	Values :	ip mode
31	Jog speed (normal)	U/s	yes	20	VI	7 digits 0 - P30 *)	ip
32	Jog speed (rapid)	U/s	yes	40	VI	7 digits 0 - P30 *)	ip
22	Reference search speed	U/s		20	VI	7 digits 0 - P30 *)	ip
24	Encoder C signal search speed	U/s		10	VI	7 digits 0 - P30 *)	ip
20	Reference switch search direction	-		posi-tive	I	0: positive 1: negative	ip
21	Reference switch exit direction	-		posi-tive	I	0: positive 1: negative	ip
23	Axis position value after homing (preset)	U		0	VI	7 digits $\pm 2^{31}$ steps	ip

*) Jog and homing speeds are not PID regulated, but only controlled.

The control voltage is calculated as follows:

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

It is therefore important to define max. speed P 'x' 30 in advance.

The control voltage and the velocity is reached after execution of the acceleration ramp (defined by P'x'43)

7.6.4 Control parameters

Param. no. per axis	Description	Unit	changed on the fly	default	For-mat	Values :	ip mode
50	Kp Controller proportional factor	V/U		1	VI	0 - 40*P02/P03	ip
51	Kd Controller differential factor	DAC step/ cycle/ impulse		0	VI	0 - 32767	ip
56	Sampling time of Kd	Servo-cycles		100	I	1 - 1000	ip
52	Ki Controller integral factor	DAC step/ impulse/ cycle		0	VI	0 - 32767	ip
53	Integration limit of Ki (anti-windup protection)	V		2	VI	0 - 10	ip
16	Integral mode The controller uses the integral factor only according to the setting	-		always	I	0: always 1: only if stationary	ip
54	Velocity feedforward	mV/U/s		0	VI	0 - 15000/P30	ip
55	Acceleration feedfor-ward	$mV/U/s^2$		0	VI	0 - 10000/P33	ip
10	Dead band	U		0	VI	7 digits 0 - 2^{31} steps	ip
14	Backlash	U		0	VI	0 - 8192 0 - 8192*P03/P02	ip
63	Backlash compensation speed	%		10	I	10-100	ip
15	In-position zone (for 'in-position' flag)	E		0,2	VI		ip

7.6.5 Acceleration parameters

Param. no. per axis	Description	Unit	changed on the fly	default	For-mat	Values :	ip mode
42	Acceleration mode (for interpolations, the gentlest shape of the involved axis is used)	-		Trape-zoid	I	0: Trape-zoid 1: S-curve	ip
43	Acceleration (for interpolations, the lowest of the involved axes is used)	U/s^2	yes	100	VI	0 - P33	ip
44	Deceleration (for interpolations, the lowest of the involved axes is used)	U/s^2	yes	100	VI	0 - P33	ip
45	Duration of S-curve acceleration	s		0	VI	0 - 99.99	ip

7.6.6 Axis mode parameters

Param. no. per axis	Description	Unit	changed on the fly	default	For-mat	Values :	ip mode
05	Circular axis period (overrun position)	U		0	VI	0: linear > 0 - 9999.99	ip
06	Electronic gearing (Couples the selected axis with the master axis) Note: Coupling only applies from the slave axis to the master axis, but not vice versa. To achieve both, P06 must also be set for the slave axis.	-		0	I	0: not coupled 1: X is coupled 2: Y is coupled 3: Z is coupled 4: W is coupled	ip
07	Transmission of elec-tronic gearing (slave axis/master axis)	-		0	VI	0 - 9999.9999	ip

7.6.7 Special parameters

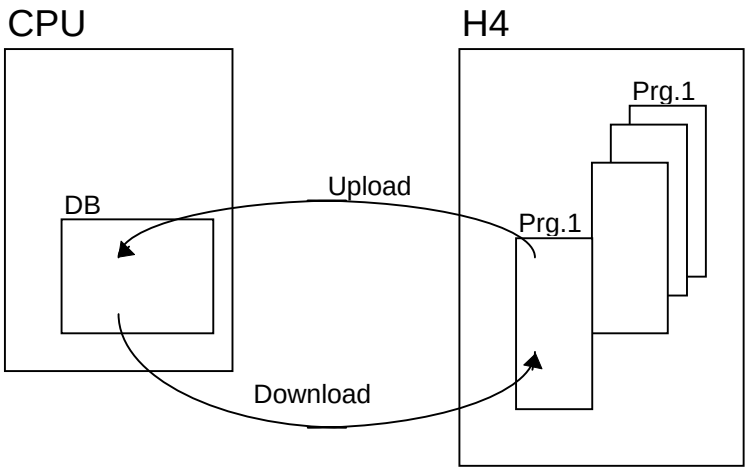
Param. no. per axis	Description	Unit	changed on the fly	default	For-mat	Values :	ip mode
62	Polarity of trigger-output signal	-		0	I	0: positive 1: negative	ip

7.7. FBs for writing and reading H4 programs

Description

The H4 module can manage up to 9 programs. These programs are buffered in the H4 by a large capacitor. If the module is left without power for more than two weeks, the user program may be lost. The 'Upload' and 'Download' functions can be used to transfer programs between the H4 and the PCD.

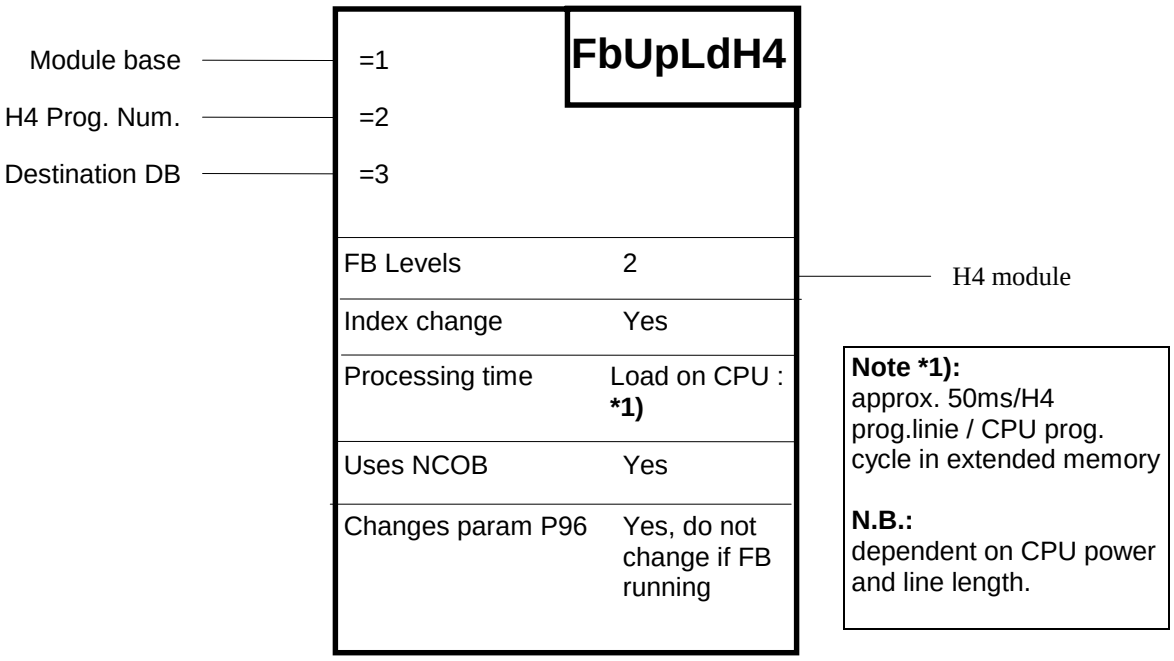
The graphical representation of FBs is explained in section 7.4.6.



FbUpLdH4 Function: - Read H4 program FbUpLdH4

```
Ld   R  rHelp      ; e.g.3 decimal places
     3             ;
Ld   R  rComH4
     P96W
CFB  fbExecH4
     ModulBase
     rHelp
     NotUsed
     NotUsed
     NotUsed
```

The number of decimal places with which the program is to be stored in the DB must be set in advance. The value under P96 is also stored in the DB.



Function description:

This function reads a program from the H4 module and stores it in a DB owned by the PCD's CPU. See following page for detailed functions.

Description of inputs and outputs:

Symbol	Description	Para-meter	Media		
			Type	Format	Addr. range
ModuleBase	Module base address	yes	K	Integer	0 - 496
H4 Prog.Num.	Program number in H4	yes	K	Integer	1 -9
Destination DB	Destination / DB memory	yes	DB	Integer	(0 -) 4000-7999 ***2)**

Note *2):
It is preferable to use a DB >4000, as access to these DBs is faster and the DBs can be larger. DB >4000 are stored in extension memory, which requires a PCD7.R3.. memory module.

Function description:

After reading, the H4 program is stored in a single DB. This DB must be defined by the user (for size, see calculation). If the size defined is too small, uploading stops and the error flag is set. This DB cannot be loaded back into the H4.

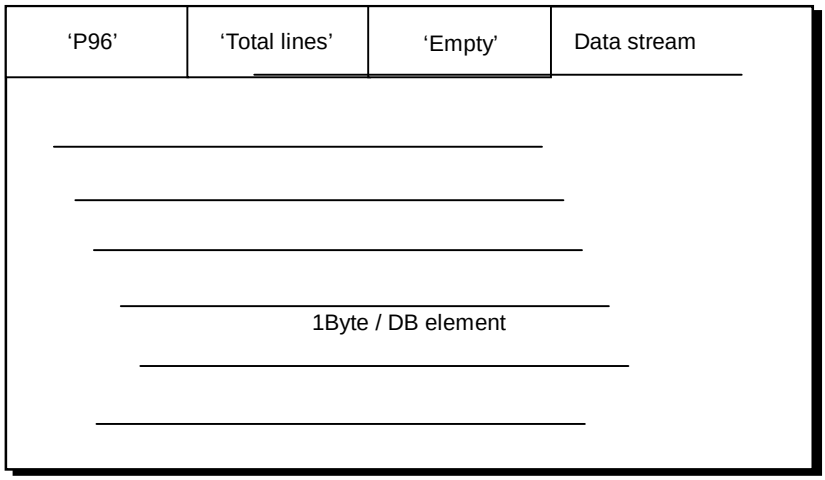
While an Upload or Download FB is running, all Immediate commands can still be called. For example, it is still possible to read the status flags or the actual position of an axis.

Commands which need to access the program memory of the H4 module must not be called.

The 'FbUpldH4' only returns when the whole H4 program has been up-loaded into the DB. This FB uses the Next COB (NCOB) instruction, which allows other tasks in other COBs to continue to run.

DB structure:

DB nnnn:



Calculation of DB size: (approximation)

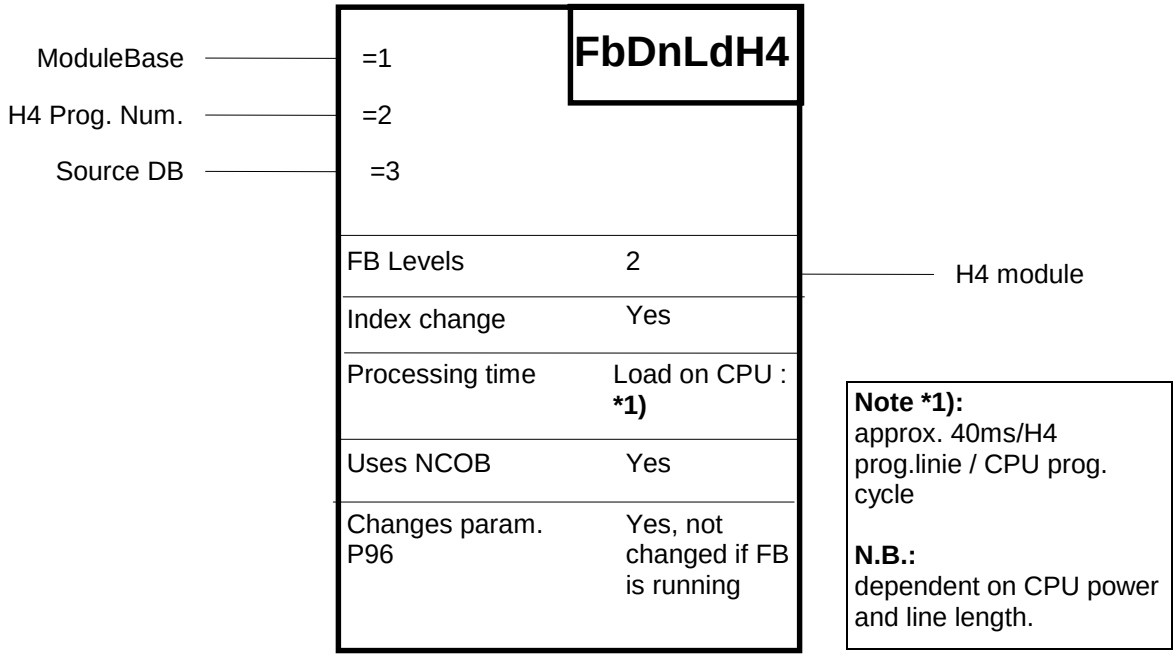
DB size = number of H4 program lines * 9

Example:

For a program with 120 lines:

```
  :
DB      3600     [120*9] ; max. length = 16384
  :
```

FbDnLdH4 Function: - write H4 program FbDnLdH4



Function description:

This function writes a program to the H4 module. The program is taken from a DB in the PCD. See following page for detailed functions.

Description of inputs and outputs:

Symbol	Description	Para-meter	Media		
			Type	Format	Addr. range
ModuleBase	Module base address	Yes	K	Integer	0 -512
H4 Prog.Num.	Program number in H4	Yes	K	Integer	1 -9
Source DB	Source DB	Yes	DB	Integer	(0 -) 4000-7999 ***2)**

Note *2):
It is preferable to use a DB >4000, as access to these DBs is faster and the DBs can be larger. DB >4000 are stored in extension memory, which requires a PCD7.R3.. memory module.

Function description:

While an Upload or Download FB is running, all Immediate commands can still be called. For example, it is still possible to read the status flags or the actual position of an axis. Commands which need to access the program memory of the H4 module must not be called.

If the program from the DB cannot find space in the H4, downloading stops and no program is written to the H4.

FbUpldH4 only returns when the whole H4 program has been uploaded into the DB. This FB uses the Next COB (NCOB) instruction, which allows other tasks in other COBs to continue to run.

8. Error handling / prevention

8.1 Installation

- To prevent positioning errors in perturbed environments, please observe the following points:
- The PCD4 system should be properly grounded by a short connection from the GND terminal to the grounding bar.
- Use shielded cables between the H4 and the encoders and power amplifiers and connect to ground both sides of the shield. (max. approx. 20m)
- Use D-type connectors with full metal housing (shielding connected to housing)
- If you have a differential voltage between the PCD4 ground and the machine ground, connect the shield on the machine side through a parallel RC filter.
- Do not install the H4 cables (for encoder and DAC output) in parallel to high voltage or high current cables (e.g. motors, contactors, soldering heads)
- Use amplifiers with differential input. (voltage ≈ velocity)

8.2 Checklist for error detection

1. Is the PCD4 system powered with 24V and does it have a proper ground?
2. Is the H4 powered with a **smoothed** voltage of 19 to 32V?
3. Is the cabling of the axis correct:
   - Limit switches / ref. switch working correctly? neg. logic (seeLEDs on H4)
   - Encoders working ? (powered from H4)
4. Set the general parameters correctly.
5. Is the correct encoder type selected? 5V/24V (see LEDs A, B, C on H4)
6. Set the machine parameters correctly. (max. velocity, max. acceleration, encoder resolution, mm/revolution, following error, following action = stop etc.)
7. Is the counting direction correct? (move axis by hand or with jog commands of slow speed)
8. Is the counting position correct? (QP'x')
9. Set the parameters for Jog move. (not PID controlled)
10. Switch on the power amplifier (one hand on the emergency stop!). OK signal from the amplifier? (see LED IN on H4)
11. Enable the axis from the H4 ENA'x' (one hand on the emergency stop!)
12. Is the DAC polarity correct? (jog pos. moves in pos. direction)
13. Set the parameters for the home routine. (search speed, search direction, home pos.)
14. Execute the home command. Is the position correct?
15. Execute a move command ex. XR10 (PID controlled)
16. Find the correct regulation parameters with a small motion program (graphic mode)
17. Store the optimal parameters in the EEPROM with EW.

Troubleshooting

If the axis does not move, check the axis status displayed in Terminal menu and compare the flags with the table no.2.12 in chapter 7.5.5.

lf the H4 does not execute the commands, check the user error code and compare with table no.2.13 in chapter 7.5.5.

lf there is no connection from the CP to the H4 (communication error) check that:

- the right COM port is selected
- the PLC does not set the RESET DSP (output BA + 11)
- the H4 is working (OK led on)
- the correct cable is used

8.3 Error handling with FBs

The standard FBs do not contain a structure which stops or disables a machine if a fault arises in the H4 module. This has to be thought about and solved by the user individually for his machine or installation. However, the standard FBs support the user and still allow him the greatest possible freedom to find a solution.

A difference is drawn between two distinct types of error: fatal errors and user errors.

How FBs behave when there is a user error:

A "User Error" is generated by the H4 module. The FBs continue to function. The operator recognizes a user error from the state of input I 10 on the H4 module (see section 7.4.2). By calling the function block 'fbExecH4' with the 'QU' command, the operator obtains more detailed information about the user error. Interpretation of the error code is described in section 7.5.5 cell 2.13. Using this information, he can then program the behaviour of his machine or installation.

Fatal Error

The "Fatal Error" fault (F7 + Flag BA, see section 7.5.5 cell 2.12) is generated by the 'StatH4' FB, which must be called cyclically. This error is only set when it is impossible to access the H4 module or if the H4 reacts wrongly. This can arise when the H4 module is faulty or when there is an error on the PCD bus. A fatal error can also arise if the H4 module is incorrectlly addressed, as the module does not respond, producing the wrong behaviour on the bus.

The 'Fatal Error' flag cannot be erased by the user. There would be little sense in being able to continue running a faulty module by means of a simple acknowledgement.

DSP-Reset

To restart the H4 after a 'Fatal Error' (produced as example by a strong perturbation impulse) there are two possibilities:

- Either the PCD-system can be switched off and on or
- only the H4 can be restarted with the output 'DSP-Reset' (O 11), which can be done in the user-program.

After a reset, the H4 needs about 500 ms to restart, which is signalled with the input 'DSP-Ready' (I 11). Afterwards the handshake is to set in the initial state with the output 'Clear channel' (O10)

Example:

```
       SET   O 11       ;DSP-Reset
       CFB     wait10   ;reset-pulse min. 10ms
       RES   O 11
ready: STH   I 11       ;DSP ready?
       JR    L ready
       SET   O 10       ;clear channel for handshake
       RES   O 10
```

After a DSP-Reset all the parameters are read from the EEPROM.
All parameters modified in the H4 working memory only (and not stored in the EEPROM) will be lost.

How FBs behave when there is a fatal error:

FB 'fbExecH4':
This FB ceases to be executed. Flag presetting, however, continues to be handled. If a fatal error arises while the FB is working with the module, the FB is exited via a timeout. However, in this case the fatal error flag is not set. (The timeout of approx. 200 ms is fixed and generated without a timer).

FB 'fbStatH4':
If this FB identifies a fatal error, the fatal error flag is set. This is the only FB which can set 'Fatal Error-Flag 7' from among the status flags. Further access to the H4 module is blocked. The status flags are no more refreshed and are no longer valid.

FB 'fbInitH4':
If a fatal error occurs when the Initial-FB is called, a QIO error is generated (XOB 5) and the PCD CPU blocks. This happens, for example, when the H4 module is missing. Further operation is impossible.

FB 'fbUpLdH4':
The Upload FB is aborted and the destination DB is invalidated. A valid DB is present if the second element (no. 1) in the actual DB does not equal 0 (zero).

FB 'fbDnLdH4':
The download is aborted. The selected program in the H4 module is not modified.

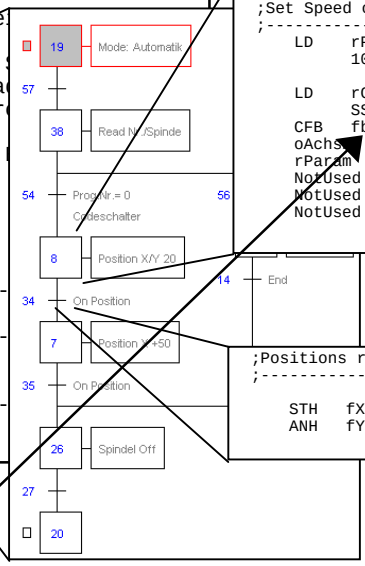
Example with the different situations clarified:

```
;=============
;Lathe
;=============

;-------------------------------------
;Integration of definitions
;-------------------------------------
$INCLUDE        DREHDEF.SRC

;-------------------------------------
;Cold start
;-------------------------------------

 XOB    16

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

 EXOB

;-------------------------------------
;Cycle / Monitoring
;-------------------------------------

 COB    0
        0

 COM    O 255           ;refre

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

 CSB    0               ;Call

 ECOB


;-------------------------------
;Integration of SAIA standard
;-------------------------------
$INCLUDE        H4FB.SRC

;---------End Source Code-----
```

Fatal Error:

The fatal error flag is set. Referring to it enables motion processes to be terminated (e.g. with RSB).

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

    LD    rComH4
          SSXi    ;Command Set Speed X
    CFB   fbExeH4 ; Execute
    oAchs         ;Base address H4
    rParam        ;Parameter
    NotUsed
    NotUsed
    NotUsed
```




```
;Positions reached
;---------------------------------

    STH   fXPos
    ANH   fYPos
```

Querying the "User Error" could be incorporated here and the register could be read with QU.
Depending on the error, it is then possible to program the continues operation of the machine or installation.

Fatal Error:

The FB aborts and sets the relay flags. The SB is exited. The next transition is not executed. If it had been executed, the SB would not run on, because the status flags would have been relayed. The user can now stop the SB and thereby prevent faulty operation.

Notes

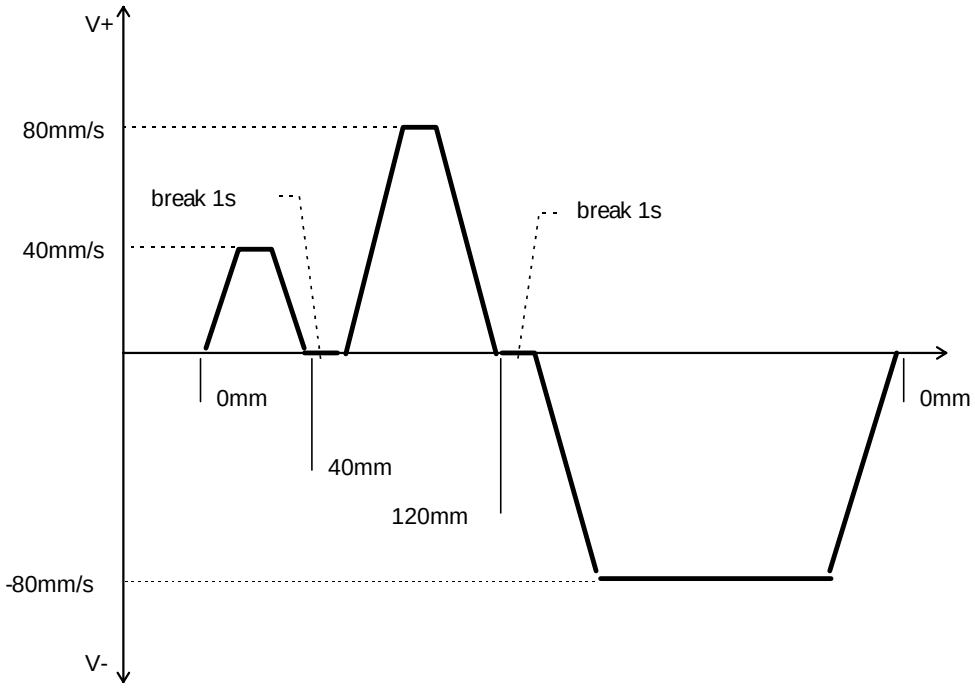
9. Application examples

9.1 Travelling a simple path

9.1.1 Example

This example requires one axis (X) only. It demonstrates how a simple motion process can be produced. Two alternative methods are contrasted: control by an H4 sequential program or control by a PCD program.

The following motion sequence is to be produced:



a) Install software

It is assumed that SAIA tools (PG3 or PG 4) have already been installed on your PC. In addition, the H4 CP tool must also be installed. For installation and operation, see section 7.3 'Programming with the CP tool'.

b) Setting up

Set up the X axis to be ready for operation. See chapter 5 for the electrical connections and section 8.2 "Checklist". You require a PCD4 controller with an M1.. CPU, an R... memory module, a PCD4.H4.. motion control module, a PCD4.N210 power supply and bus modules. A second axis is not necessary for this example.

The starting position of the axis should be such that a movement of 120mm can be executed without exceeding the limit switches.

9.1.2 Alternative using CP tool

Programming via the CP tool is the simplest method. It requires neither PCD program nor PCD4 CPU. First adjust the H4 module parameters in accordance with the axis used. See section 7.6 'Parameter list'.

Caution:

Set the encoder voltage (parameter 92) correctly at 5V or 24V. Parameter 30 must be adjusted according to the maximum possible velocity of the axis used. Ensure that the parameters for count direction and DAC polarity have been set correctly, otherwise the axis can accelerate out of control.

a) Programming

Afterwards, enter these commands in the CP tool program editor. Here you write your first H4 program. (To enter the commands, first open the edit window with <CR>)

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

Transfers the program to the H4 module as program no. 1 (F4: From/To H4 module). If a communication error is displayed, check the cable, the COM port selected in CP tool and the PCD supply voltage. (The program is already stored in the CP tool as example 1)

b) Operation

Switch on the X axis and start up the program. Enter the following commands in CP's terminal mode:

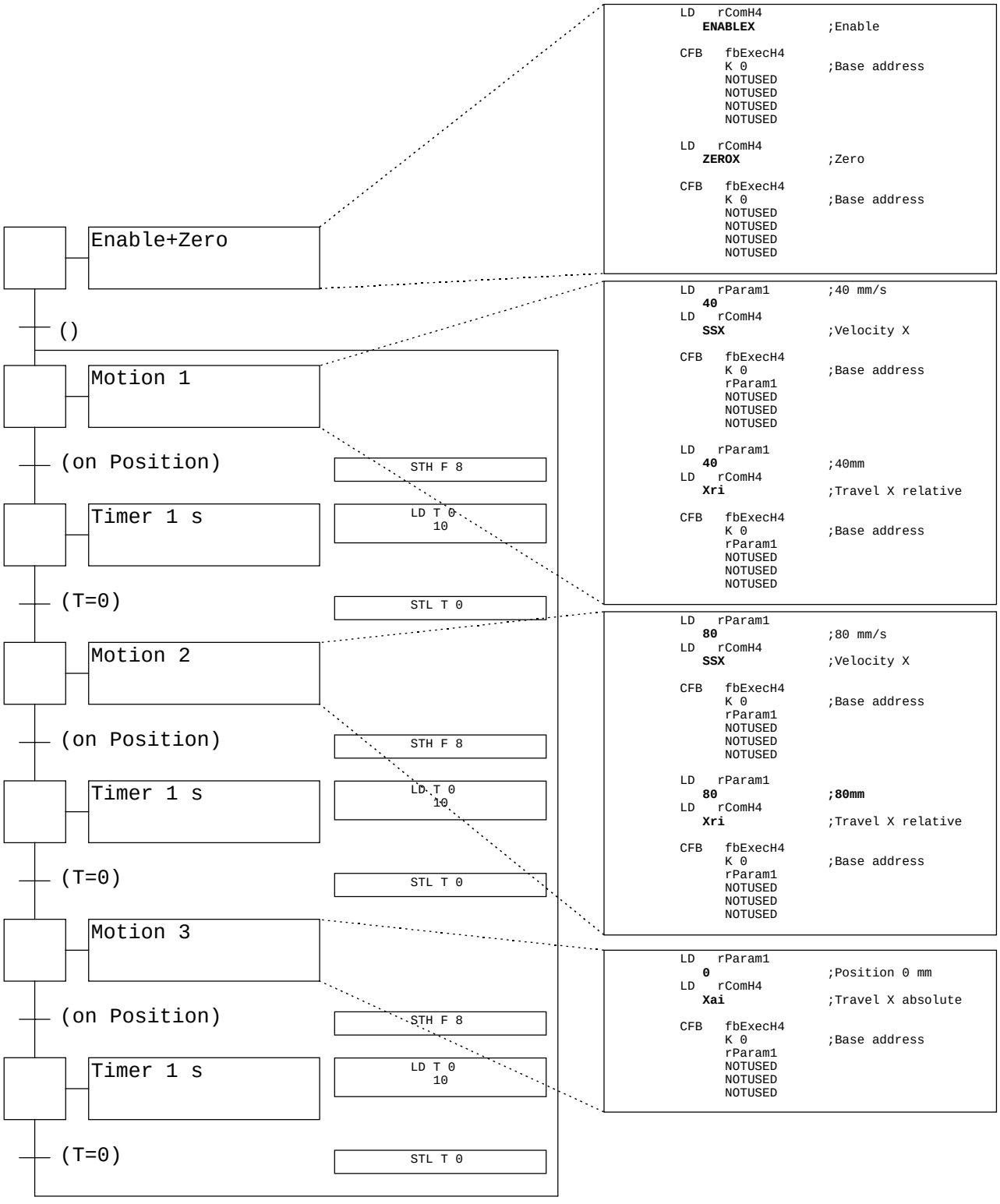
```
ENABLEX
RUN1
```

9.1.3 Alternative using PCD program

The program is written in GRAFTEC.

a) Programming

SB 0



XOB 16

```
XOB     16          ;Cold start

CFB     fbInitH4    ;Initialize H4 module
      K 0           ;Base address
        0           ;Base flags
      K 2           ;Module type 2 axes

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

EXOB
```

COB 0

```
COB     0           ;Cyclical block
        0

CFB     fbStatH4    ;Read status H4
      K 0           ;Base address
        0           ;Read cyclical

CSB     0           ;Call SB

ECOB
```

b) Operation

After the program has been entered and loaded, it can be started. The same motion sequence is executed as for the alternative using the CP tool. However, the sequence is repeated cyclically.

Note the commands printed in bold type and their analogy with the preceding example.

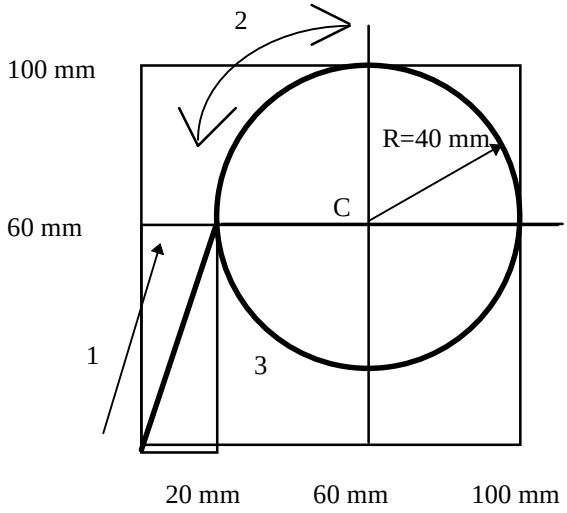
9.2 Application example with circular interpolation

This example describes working with linear and circular interpolation.

To set up the axes and install the software, the same information applies as in example 9.1. However, two axes are provided. Input takes place with the CP tool.

Task description:

The following motion program is to be executed:



The momentary axis position is defined as point zero. From this point, motion should start a linear interpolation with a path velocity of 20mm/s. Afterwards, a circular path is travelled 10 times at 80mm/s. Motion takes place clockwise.

Alternative 1:
The same task, but with the circular path going counter-clockwise and endless repeating.

Alternative 2:
The original task, but the circle is programmed with centre mode. In addition, travel between the two interpolations should take place without "blended move".

Program: Example 2

ZEROX	Zero X axis
ZEROY	Zero Y axis
SV20	Path velocity = 20mm/s
XR20,YR60	Motion relative at 20,60mm
WAIT0	Prevents "blended move"
SV80	Path velocity = 80mm/s
FOR10	Start 10x loop
CIR40,**0**,XR80,YR0	Half circle clockwise with radius 40 and destination +80
CIR40,**0**,XR-80,YR0	Half circle clockwise with radius 40
NEXT	End loop
END	End program

With alternative 1: Example 3

Differences from the original program are in bold type and underlined**.**

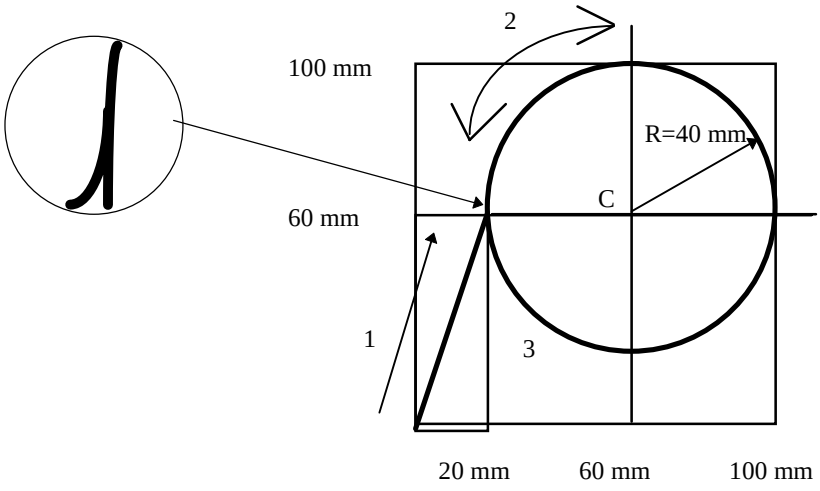
ZEROX	Zero X axis
ZEROY	Zero Y axis
SV20	Path velocity = 20mm/s
XR20,YR60	Motion relative at 20,60mm
WAIT0	Prevents "blended move"
SV80	Path velocity = 80mm/s
FOR**0**	Start endless loop
CIR40,**1**,XR80,YR0	Half circle counter-clockwise with radius 40
CIR40,**1**,XR-80,YR0	Half circle counter clockwise with radius 40
NEXT	End loop
END	End program

With alternative 2: Example 4

Differences from the original program are in bold type and underlined**.**

	ZEROX	Zero X axis
	ZEROY	Zero Y axis
	SV20	Path velocity = 20mm/s
	XR20,YR60	Motion relative at 20,60mm
*)	WAIT0	Prevents "blended move"
	SV80	Path velocity = 80mm/s
	FOR**0**	Start endless loop
	CIR**360**,**2**,XR40,YR0	360° circle with centre specified
	NEXT	End loop
	END	End program

*) Without this instruction, "blended move" is executed. See also drawing with detail and chapter 6.6.



9.3 Application example: automatic lathe

This example describes a simple lathe. The machine homes the axes automatically when it is powered up. From the control console, either manual or automatic operating modes can be selected. In manual mode, the spindle can be switched on or off and the axes can travel forwards or back. Automatic mode starts programs from the H4.

The following hardware is required:

- PCD4.M1.. CPU
- PCD7.R... RAM memory
- PCD4.E1.. input module (BA 0)
- PCD4.A4.. output module (BA 16)
- PCD4.N210 power supply
- PCD4.C1.. CPU bus module
- PCD4.C2.. I/O bus module
- PCD4.H4.. axis module (BA 32)
- Switchbox with at least 12 switches
- Axis model: two axes with incremental shaft encoders, reference switches and two limit switches

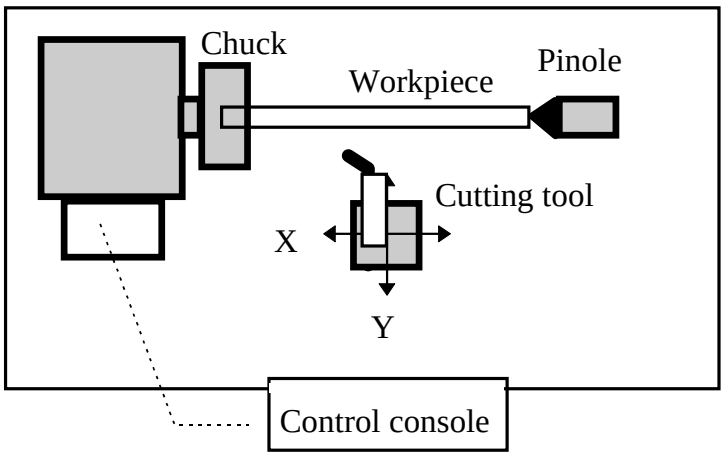
The following software is required:

- SAIA 'PG3' assembler complete with editor
- Commissioning Tool CP.EXE for H4 module
- SAIA's standard function blocks with definition file H4FB.SRC, H4EXTN.DEF and H4DEF.SRC

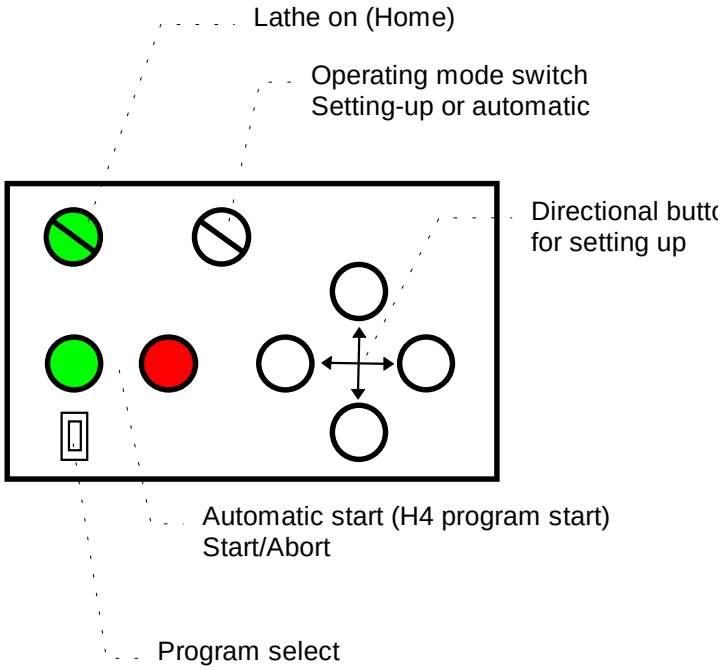
Source code for application example:

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

Sketch of a lathe



Control console:



Description of the function implemented in file 'Dreh_xx.SRC'.

After switching on the lathe, the home command is executed. As soon as the axes have finished homing, the machine is ready for operation. It is possible to choose between the two operating modes 'Setting-up' and 'Automatic'. With 'Setting-up' the axes can be travelled using the directional buttons. In this operating mode, the start button can be used to switch on the spindle drive and the off button to switch it off again. If automatic operation is selected, the same start button can start an H4 program. The start button is illuminated for as long as this program runs.

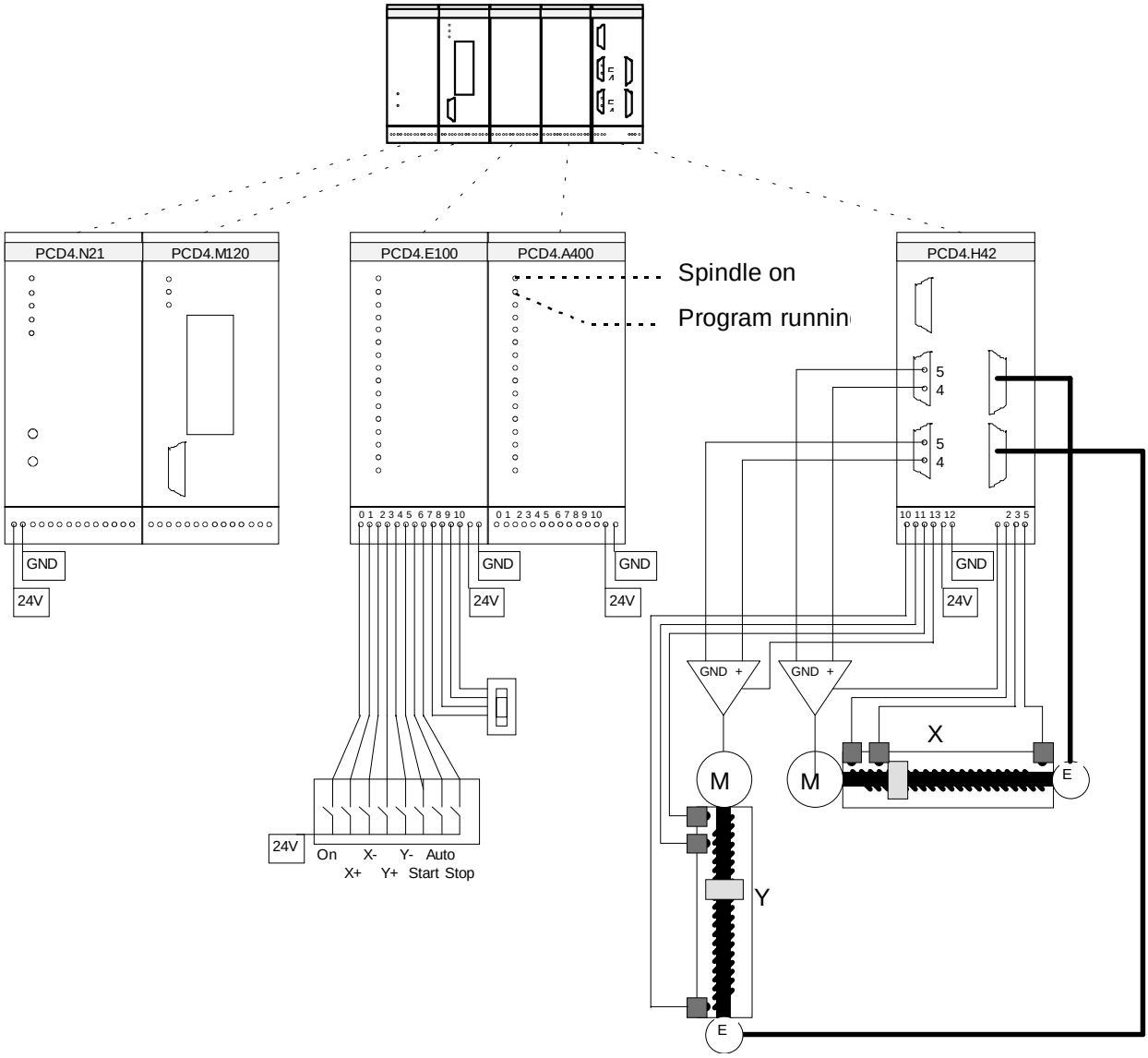
By means of a coding switch, a program number (1-9) can be selected. This must be defined before pressing the start key. These programs are stored in the H4 and can be modified with the commissioning tool. The PCD CPU is only responsible for starting the H4 programs.

If "0" is selected with the coding switch, no H4 program runs. Instead, the PCD CPU generates a sequence in which each individual motion is transferred with the FBs as an immediate command to the H4 module. The H4 module executes these commands directly, i.e. the motion sequence is not stored in the H4 but in the PCD CPU.

Installation of the application example:

The inputs of the control elements are simulated with a switch box. The I/O addresses used are defined in the file DREHDEF.INC.

The status of outputs can be viewed on the A400 module.

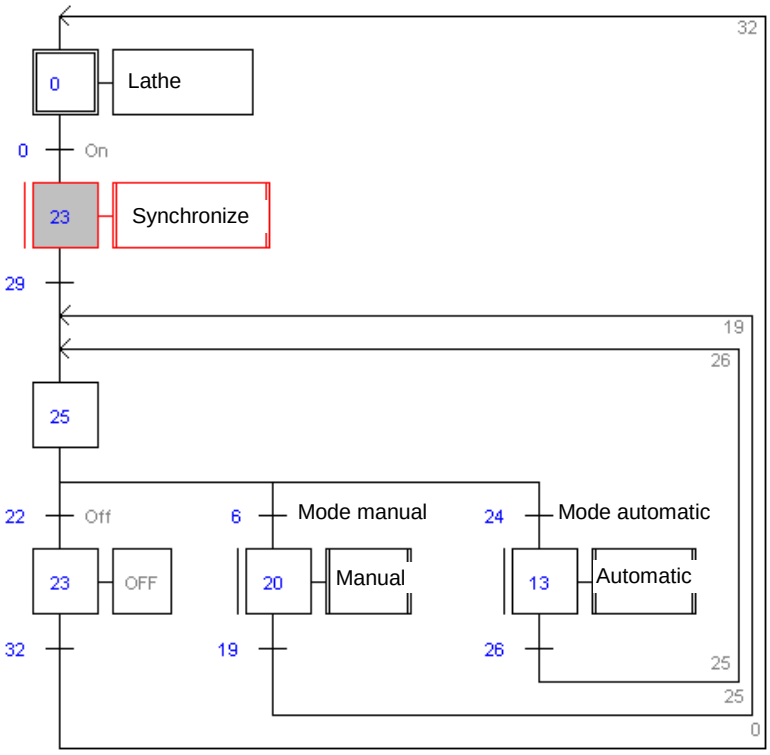


PCD base structure for lathe:

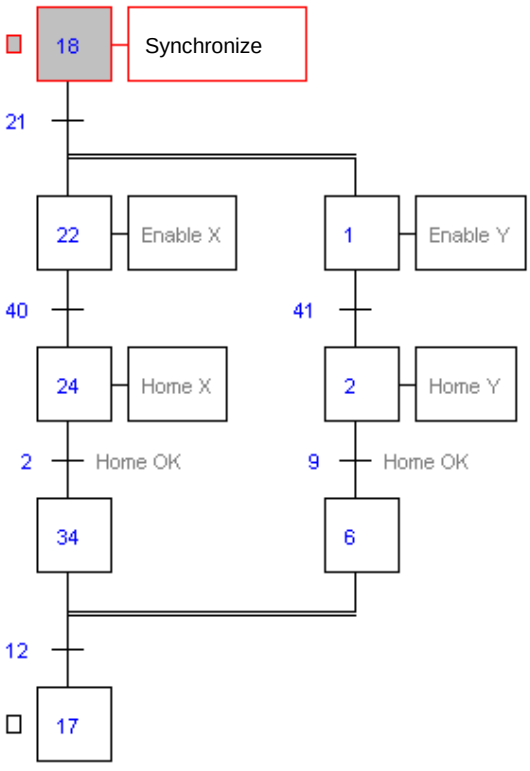
```
;=============
;Lathe
;=============


;-----------------------------------
;Integration of definitions
;-----------------------------------
$INCLUDE           DREHDEF.SRC
$INCLUDE           H4EXTN.DEF

;-----------------------------------
;Cold start
;-----------------------------------

XOB     16

CFB     fbInitH4   ; Init H4
        oAchsH4    ; Base Address Module
        fStatus    ; Base Status flags
        K_2        ; Module type

EXOB


-----------------------------------
Cycle / Monitoring
-----------------------------------

COB     0
        0

COM   O 255        ; refresh watchdog

CFB     fbStatH4   ; Read Status H4
        oAchsH4    ; Base address of H4
        0          ; 0 = one axis per cycle

CSB     0          ;Call basic sequence structure of lathe


ECOB


;-----------------------------------
;Integrate SAIA standard FBs for H4 module
;-----------------------------------
$INCLUDE           H4FB.SRC

;-------- End source code ------------------------
```


Sequence of lathe in SB 0

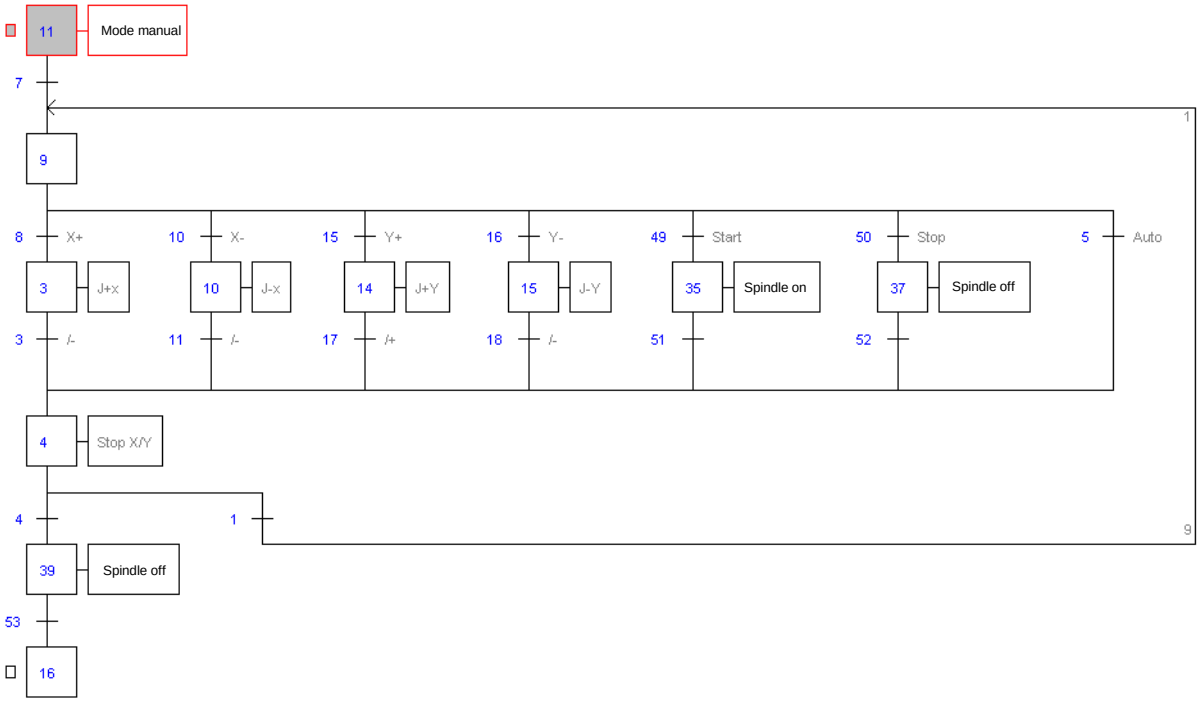
Basic sequence structure



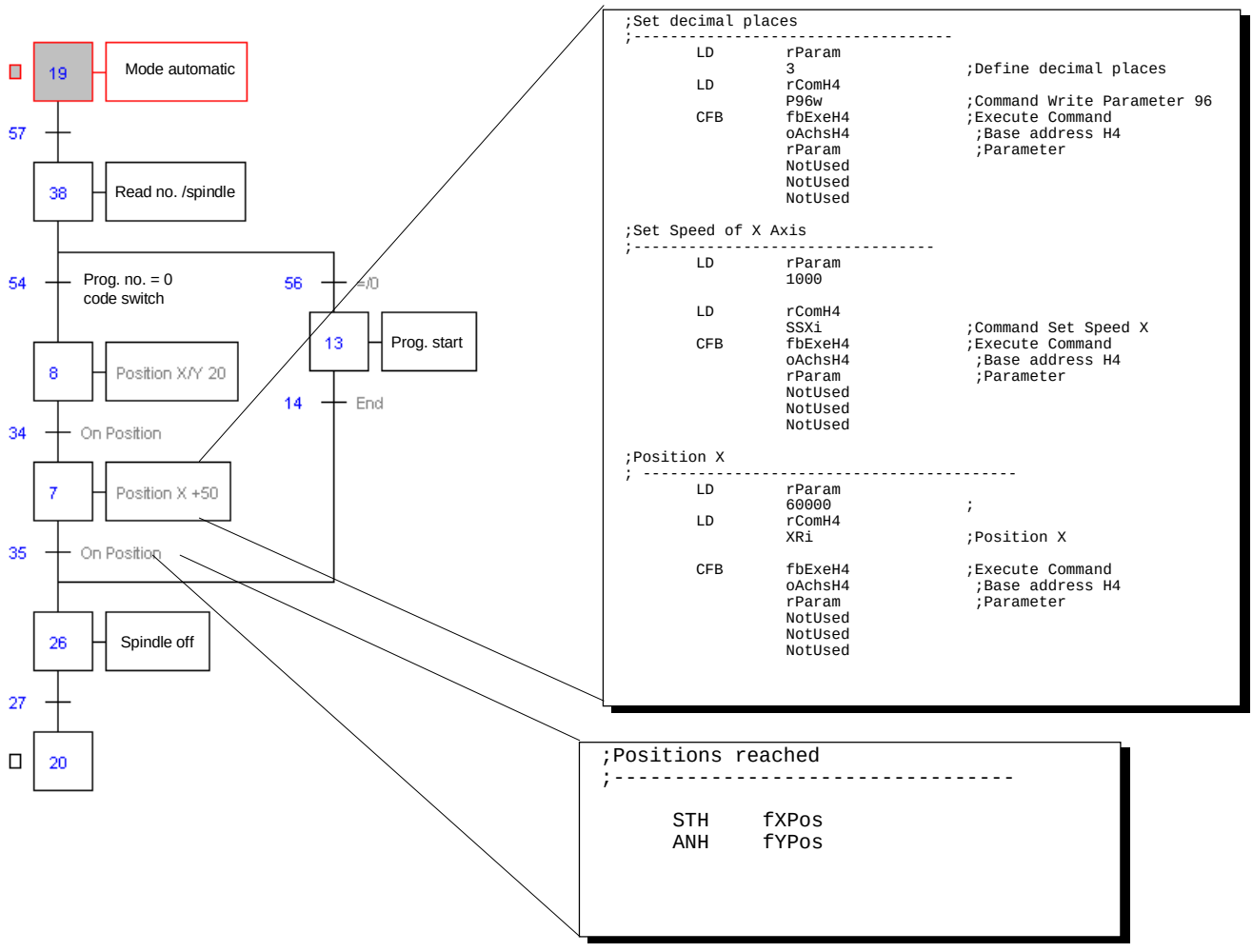
Synchronization:



Manual operating mode



Automatic operating mode



9.4 Application example with independent axes (with OPEN/CLOSE function)

This example describes working with independent axes which have to run simultaneously or with an overlap, but without interpolation.

For this purpose, a program must be written for each axis. These programs can then be started independently (immediate) or dependently of each other (nested program start within a program).

With the FBs, the OPEN and CLOSE commands are used to create a program:

```
LD      R 0
        K 1         ; line no. from which the program will be edited
LD      rComH4
        OPEN5       ; program 5 is opened for editing
                    ;   (at line 1)
CFB     fbExeH4     ; execute command
        K 0         ; base address of H4
        R 0         ; line no.
        R 1         ; not used for this command
        R 2         ;   "
        R 3         ;   "

LD      R 0
        K 20000     ; corresponds to 20 if P96 = 3
LD      rComH4      ; XA20 is written in program 5 on line 1
        XAp         ; the p indicates the program command
                    ;  Xai would be executed directly (immediate)
                    ;  and not written into program memory
CFB     fbExeH4     ; execute command
        K 0         ; base address
        R 0         ; not used for this command
        R 1         ;   "
        R 2         ;   "
        R 3         ;   "

LD      rComH4      ; END is written at line 2
        END         ; (END only exists as a program command and
                    ;  so needs no special identification with p)
```

```
CFB     fbExeH4     ; execute command
        K 0         ; base address
        R 0         ; not used for this command
        R 1         ;   "
        R 2         ;   "
        R 3         ;   "

LD      rComH4      ; With CLOSE, the edited program is saved to
        CLOSE       ; memory location 5.
CFB     fbExeH4     ; execute command
        K 0         ; base address
        R 0         ; not used for this command
        R 1         ;   "
        R 2         ;   "
        R 3         ;   "
```

If the program edited is currently in execution, the CLOSE command is not accepted (error code 6) and the program is not overwritten.

Otherwise, the following program will stand in memory location 5

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

Only the lines edited are overwritten. If a longer program is to be overwritten, it must first be deleted with the EP (erase program) command preceding CLOSE.

Notes

Appendix A: Command code definitions for programming with FBs

```
;
;-----------------------------------------------------------------
;Command code definitions for FBs version V001
;-----------------------------------------------------------------
;
; All codes are in Hex format and will be loaded into the
; register rComH4 (BAR+0) before being used.
; Pay attention to display this register in Hex format in
; the debugger.
;

;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
```

Appendix B: Examples Programming with FBs

Example 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,
;  Example program for PCD4.H4xx module                      BLOCTEC
;  Motion without blended move.
;         ---------------
;  File:        E1EX1.SRC
;
;  Description : This program consists of the following motions :
;                1.- move X from Reference point X to 40mm with 20mm/s
;                2.- move X from actual position X to 80mm with 80mm/s
;                3.- move X back to the reference point
;
;                This program is written in BLOCTEC. The user must set
;                the input I0 to start the motions. The complete motions
;                program (step 1 to 3) is executed only once by each
;                positive edge detection of the input I0.
;                To restart the motion program, activate again the I0.
;
;                If the Input I0 is low during the step 1 to step 3, then
;                the program will finish the current motion and stop at
;                the end of this motion. To continue, I0 must be set high.
;
;                The user can display the actual position and the actual
;                velocity by refreshing in the debugger the registers
;                R 100 (actual position) and R 101 (actual velocity).
;
;                We suppose all machine and module parameters have been
;                downloaded in the PCD4.H4xx module.
;
```

```
;  Remark :      The FB FbStatH4 must be at least called once per cycle,
;                otherwise axes status flags like 'axis in position' or
;                'home procedure executed' are not refreshed.
;
;                - if the signal 'Axis in position' (inputs 12 to 14 of the
;                  H4 address) is not set at the end of a motion, please check
;                  the PID parameter (e.g. set higher factor P) and the
;                  parameter P15 (tolerance Axis in position).
;                - if the flags 'Axis in position' (from FB FbStatH4), please
;                  check the parameter 'Axis No.' of this FB to read the right
;                  axies.
;
;  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
```


Notes

Example 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,
;  Example program for PCD4.H4xx module                      BLOCTEC
;  Motion with blended move.
;         ------------
;  File:        E1EX1BL.SRC
;
;  Description : This program consists of the following motions :
;                1.- move X from Reference point X to 40mm with 20mm/s
;                2.- move X from actual position X to 80mm with 80mm/s
;                3.- move X back to the reference point
;
;                This program is written in BLOCTEC. The user must set
;                the input I0 to start the motions. The complete motions
;                program (step 1 to 3) is executed only once by each
;                positive edge detection of the input I0.
;                To restart the motion program, activate again the I0.
;
;                The user can display the actual position and the actual
;                velocity by refreshing in the debugger the registers
;                R 100 (actual position) and R 101 (actual velocity).
;
;                We suppose all machine and module parameters have been
;                downloaded in the PCD4.H4xx module.
;
;  Remark :     - The FB FbStatH4 must be at least called once per cycle,
;                otherwise axes status flags like 'axis in position' or
;                'home procedure executed' are not refreshed.
;
;                - if the signal 'Axis in position' (inputs 12 to 14 of the
;                  H4 address) is not set at the end of a motion, please check
;                  the PID parameter (e.g. set higher factor P) and the
;                  parameter P15 (tolerance Axis in position).
;                - if the flags 'Axis in position' (from FB FbStatH4), please
;                  check the parameter 'Axis No.' of this FB to read the right
;                  axies.
;
```

```
;  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
```

Example 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,
;  Example program for PCD4.H4xx module                    GRAFTEC
;  Motion without blended move.
;         ---------------
;  File:        E1EX2G.SRC
;
;  Description : This program consists of the following motions :
;                1.- move X from Reference point X to 40mm with 20mm/s
;                2.- move X from actual position X to 80mm with 80mm/s
;                3.- move X back to the reference point
;                This program is written in GRAFTEC. The user must set
;                the input I0 to start the motions. The complete motions
;                program (step 1 to 3) is executed only once by each
;                positive edge detection of the input I0.
;                If the Input I0 is low during the step 1 to step 3, then
;                the program will finish the current motion and stop at
;                the end of this motion. To continue, I0 must be set high
;                We suppose all machine and module parameters have been
;                downloaded in the PCD4.H4xx module.
;
;  Remark :      The FB FbStatH4 must be at least called once per cycle,
;                otherwise axes status flags like 'axis in position' or
;                'home procedure executed' are not refreshed.
;
;                - if the signal 'Axis in position' (inputs 12 to 14 of the
;                  H4 address) is not set at the end of a motion, please check
;                  the PID parameter (e.g. set higher factor P) and the
;                  parameter P15 (tolerance Axis in position).
;                - if the flags 'Axis in position' (from FB FbStatH4), please
;                  check the parameter 'Axis No.' of this FB to read the right
;                  axies.
;
```

```
;  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
```

Notes

Example 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,
;  Example program for PCD4.H4xx module                    GRAFTEC
;  Motion with BLENDED move.
;         ------------
;  File:        E1EX2GB.SRC
;
;  Description : This program consists of the following motions :
;                1.- move X from Reference point X to 40mm with 20mm/s
;                2.- move X from actual position X to 80mm with 80mm/s
;                3.- move X back to the reference point
;
;                This program is written in GRAFTEC. The user must set
;                the input I0 to start the motions. The complete motions
;                program (step 1 to 3) is executed only once by each
;                positive edge detection of the input I0.
;                To restart the motion program, activate again the I0.
;
;                The user can display the actual position and the actual
;                velocity by refreshing in the debugger the registers
;                R 100 (actual position) and R 101 (actual velocity).
;
;                We suppose all machine and module parameters have been
;                downloaded in the PCD4.H4xx module.
;
;  Remark :      The FB FbStatH4 must be at least called once per cycle,
;                otherwise axes status flags like 'axis in position' or
;                'home procedure executed' are not refreshed.
;
;                - if the signal 'Axis in position' (inputs 12 to 14 of the
;                  H4 address) is not set at the end of a motion, please check
;                  the PID parameter (e.g. set higher factor P) and the
;                  parameter P15 (tolerance Axis in position).
;                - if the flags 'Axis in position' (from FB FbStatH4), please
;                  check the parameter 'Axis No.' of this FB to read the right
;                  axies.
;
```

```
;  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
```

Example 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,
;  Example program for PCD4.H4xx module                      GRAFTEC
;  Motion with BLENDED move.
;         ------------
;  File:        E3EX3GB.SRC
;
;  Description : This program consists of the following motions :
;                1.- move X,Y from Ref.point to X=40mm, Y=40mm at 20mm/s
;                2.- move X,Y to X=80mm, Y=80mm at 80mm/s
;                3.- move X,Y back to the reference point
;
;                This program is written in GRAFTEC. The user must set
;                the input I0 to start the motions. The complete motions
;                program (step 1 to 3) is executed only once by each
;                positive edge detection of the input I0.
;                To restart the motion program, activate again the I0.
;
;                The user can display the actual position and the actual
;                velocity by refreshing in the debugger the registers :
;                - Axis X : R100(actual position), R101 (actual velocity).
;                - Axis Y : R102(actual position), R103 (actual velocity).
;
;                We suppose all machine and module parameters have been
;                downloaded in the PCD4.H4xx module.
;
;  Remark :      The FB FbStatH4 must be at least called once per cycle,
;                otherwise axes status flags like 'axis in position' or
;                'home procedure executed' are not refreshed.
;                For reading status of all axes in one cycle, please set
;                the parameter 'Axis No.' of the FB FbStatH4 with OFh.
;
```

```
;                - if the signal 'Axis in position' (inputs 12 to 14 of the
;                  H4 address) is not set at the end of a motion, please check
;                  the PID parameter (e.g. set higher factor P) and the
;                  parameter P15 (tolerance Axis in position).
;                - if the flags 'Axis in position' (from FB FbStatH4), please
;                  check the parameter 'Axis No.' of this FB to read the right
;                  axies.
;
;  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
```

Notes

Example 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,
;  Example program for PCD4.H4xx module                OPEN/CLOSE Progr.
;  Motion with BLENDED move.
;         ------------
;  File:        OPENPR1.SRC
;
;  Description : This program is edited using the instruction OPEN/CLOSE.
;                That means the whole motion program is set together and
;                will be transfered only once into PCD4.H4xx module.
;                The loaded program is run from the CPU by means of the
;                instruction RUNp (run program number) in a COB for
;                example.
;                This program consists of the following motions :
;                1.- move X,Y from Ref.point to X=40mm, Y=40mm at 20mm/s
;                2.- move X,Y to X=80mm, Y=80mm at 80mm/s
;                3.- move X,Y back to the reference point
;
;                This program is written in BLOCTEC. The user must set
;                the input I0 to start the motions.
;                The motion program (step 1 to 3) is executed only once by
;                each positive edge detection of the input I0.
;                To restart the motion program, activate again the I0.
;
;                The user can display the actual position and the actual
;                velocity by refreshing in the debugger the registers
;                R 100 (actual position) and R 101 (actual velocity).
;
;                We suppose all machine and module parameters have been
;                downloaded in the PCD4.H4xx module.
;
;  Remark :      - The FB FbStatH4 must be at least called once per cycle,
;                  otherwise axes status flags like 'axis in position' or
;                  'home procedure executed' are not refreshed.
;                  For reading status of all axes in one cycle, please set
;                  the parameter 'Axis No.' of the FB FbStatH4 with OFh.
;
```

```
;                - If the signal 'Axis in position' (inputs 12 to 14 of the
;                  H4 address) is not set at the end of a motion, please check
;                  the PID parameter (e.g. set higher factor P) and the
;                  parameter P15 (tolerance Axis in position).
;
;                - If the flags 'Axis in position' (from FB FbStatH4), please
;                  check the parameter 'Axis No.' of this FB to read the right
;                  axies.
;
;  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
```

From :

Company :
Department :
Name :
Address :

Tel. :

Date :

Send back to :

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

BA : Electronic Controllers

Manual PCD4.H4x0
Motion control modules for servo drives
with linear and circular interpolation

If you have any suggestions concerning the SAIA® PCD, or have found any errors in this manual, brief details would be appreciated.

Your suggestions :