Universal Position Control Card

PS 30

9013.0179 / 13.07.2017

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

1. General Information

The OWIS® PS 30 is a universal position control for installation in a PC on the basis of a PCI plug-in card.

It consists of a plug-in card and an output stage module without PCI connector, mounted on a second slot panel in 106 x 168 mm format (short PCI format). Both cards are flexibly connected to each other by flat ribbon cables.

The PS 30 is a high-performance device and can operate up to maximum 3 axes with step or DC servo motors. Different combinations of both motor types are possible.

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

Point-to-point, trapezoidal or S-curve velocity profiles, as well as complex continuous path controls such as linear interpolation or circular interpolation are possible.

The software tool OWISoft is included in delivery, too. Thus, the PS 30 can be configured and operated comfortably. OWIS® standard positioning units are stored in OWISoft and can be assigned to the corresponding motor easily. Foreign motors can also be actuated.

2. Setup and Scope of Delivery

The PS 30 consists of a PCI plug-in card and of an output stage module without PCI connector, mounted on a second slot panel in short PCI format.

By default, the PS 30 output stages are supplied with + 12 V from the PC power supply. In case of high requirements to the power output and the positioning velocity, an external 24 V power supply can be attached.

The valid firmware for operation is installed. It can be updated, if necessary, through the PCI or RS-232 interface.

Following parts are included in delivery:

- PS 30 PCI plug-in card
- PS 30 output stage module in the required motor configuration
- 3 flat ribbon cables 50-pin for the connection of the PCI plug-in card and the output stage module
- 3-way motor adapter cable
- CD with software tool OWISoft and documentation in English/German
- printed version of the user manual in English/German
- data sheet in English/German

2.1 Standard

The position control comes with:

- PCI port
- RS-232 port
- 2 inputs for reference and limit switches per axis
- 8 TTL or analog inputs
- 5 TTL outputs
- 2 PWM outputs (e. g. to actuate motor holding brakes)
- connection for enabling the motor output stage

2.2 Accessories

Following accessories are available:

- external desktop power supply AC 100 - 240 V, DC 24 V, 90 W
- connecting cable with plug for different positioning systems
- joystick for 3 axes, analog, with 3 m cable

2.3 Option

The following optioncan be provided:

- stand-alone compiler with USB dongle

3. Safety

As soon as an output stage temperature of about 85°C has been exceeded, the corresponding motor output is switched off and an error status message for the axis concerned is noted in the system. In order to enable the motor output stages jumper JP14 on the PCI plug-in card must be plugged (see 7.3). If the jumper is not plugged, the galvanically separated external release input can be used. For that purpose, the input must be supplied with a voltage of 5 V. If neither the jumper is plugged nor the external voltage is supplied no activating of the output stages is possible.
Furthermore, each type of motor is identified to the motor power stage through a coding resistor. This helps to avoid motor damage if the wrong type of motor has been connected (e.g. a DC motor to a stepper motor output stage).

4. Standards and Directives

The universal position control card PS 30 complies with following standards and regulations:

- RoHS conform
- CE Directive
- EMV Directive 2014/30/EU

Interference immunity according to generic standard EN 61000-6-1:

- Electrostatic discharge immunity test
  Basic standard: EN 61000-4-2 (ESD)
- Radiated, radio-frequency, electromagnetic field immunity test
  Basic standard: EN 61000-4-3 (radiated RF)

Conducted RF emission according to generic standard EN 61000-6-3:

- Radiated RF according to
  Basic standard: EN 55022
  (information technology equipment/ITE devices)

5. Technical Overview

power supply:	+ 12 V through PC power supply or max. 24 V external
power consumption:	max. 24 W (12 V) or max. 90 W (24 V)
drive groups:	3 axes
motor type:	2-phase step motors Open Loop (OL), 2-phase step motors Closed-Loop (CL), DC servo motors
communication:	PCI-COM jumper (max. 115 200 Baud), RS-232 optionally
installation:	PC plug-in card
protection class:	IP 00
encoder:	quadrature signals A/B and index, RS-422 or TTL level, with quad evaluation, max. counting frequency 2 MHz (signal) respectively 8 MHz (quadrature)
functions:	parametrizable acceleration and deceleration ramp trapezoidal velocity profiles or S-curve
motion profiles:	point-to-point positioning operation, linear and circular interpolation

6. Setup of the Control Unit

The PS 30 consists of a PCI plug-in card and an output stage module without PCI plug connector. Both cards have a short PCI format (106 x 168 mm) corresponding to the PCI local bus Rev. 3.0 specification and are installed on separate slot panels.

The cards are flexibly connected to each other by three 50-pin ribbon cables with 1.27 mm pitch (standard length: 280 mm). The length of the ribbon cables enables the connection of the output stage module at a certain distance from the PCI plug-in card (up to three slots). This is important if neighbouring slots should be allocated or if the output stage module cannot be plugged directly next to the PCI card.

The in- and outputs (TTL/analog, PWM, enable input) are accessible over the 25-pin D-Sub connector of the PCI plug-in card. Signals which are relevant for the motor connection (limit switch, encoder etc.) are led out through the 62-pin HD socket of the output stage module.

The PS 30 motor output stages can be supplied alternatively either through the standardized PC power supply plug (hard disk supply) at the upper edge of the printed circuit board of the output stage module with + 12 V from the PC, or externally through the low-voltage socket below the 62-pin motor connector with max 24 V.

6.1 Connections

The PCI bus connector and the input and output plugs are located on the PS 30 PCI plug-in card. The sockets for the connection of the motor output stages and the motor connector are located on the output stage module.

An additional RS-232 communication interface, which can also be used to execute a firmware update, is located on the PCI plug-in board, too.

connection	functions	socket
PCI	communication with a PC	PCI bus connector
motor	motor supply, limit switches and encoder	HD 62-pin socket
in-/outputs	interaction with external hardware	D-Sub 25-pin male connector
+ 12 V internal	operating voltage of the motor output stages	standardized PC power supply
24 V external	optional: operating voltage of the motor output stages	DC circular connector 5.5 x 2.1 x 11 mm
RS-232	communication with the PC (optional), firmware update	10-pin IDC

PCI-Bus Interface

The PCI interface of the PS 30 is implemented as a so-called PCI-COM bridge. The Windows device driver recognizes the PS 30 as "PCI serial port" and assigns a COM port number to it. This number can be changed by the user, if necessary. After successful installation, the PCI interface is addressed as virtual RS-232 interface.

The PS 30 can operate with transfer rates of 9 600, 19 200, 38 400, 57 600 or 115 200 baud. Please make sure that the transfer rate of the PS 30 corresponds to the transfer rate defined in the device driver, otherwise no communication is possible. Preset is 9 600 baud.

If the communication speed of the PS 30 is to be changed, then the required value for the baud rate of the PS 30 should first be set by command. Afterwards, the same value should be set for the transfer

rate of the device driver. Then, Windows should be started again. When all this has been done, the PS 30 should communicate at the changed rate.

Power Supply
The power supply of the PCI plug-in card generally takes place through the PCI bus. Should the host PC not supply the 3.3 V to the PCI bus, this necessary working voltage can be supplied internally from + 5 V. A presetting is possible by a jumper (see also chapter "Operating Elements and Settings").

The operating voltage of the PCI-COM bridge can be set by jumper as well. + 3.3 V, + 5 V and automatic adjustment are possible. Automatic adjustment should work with all modern PCs without problems (see also chapter "Operating Elements and Settings").

Motor Connector of 3-Way Motor Adapter
The positioning units are connected using the suitable OWIS® connecting cable. The universal motor connector enables the current supply of the motor, control of the motor holding brake, where applicable, and the transfer of the encoder or limit-switch signals.

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

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

When being switched on, the PS 30 measures the resistance value and reports an error message if the measured value does not match the type of the motor power stage. The error message of the output stage can be read out using the command "?ASTAT" (see command set, page 60).

The pin assignment can be seen in attachment. The signals of the 62-pin motor connector which are converted by the 3-way adapter are presented there. The pin assignment matches the OWIS® standard.

Limit and Reference Switches
Maximum two switches can be connected for each axis. They can be micro switches, TTL Hall switches or TTL light barriers with + 5 V voltage. Various n.c. or n.o. contacts, switching towards GND, can be attached to the inputs.
One of the two switches is defined as reference switch, if necessary. The active level and the switch assignment are configured by software.

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

Following input signals are defined:

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

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

6.2 Inputs and Outputs

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

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

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

features	level	current	others
TTL/analog inputs	0 - 5 VDC		can be used as 10 bits analog or digital inputs, alternatively
TTL outputs	0 - 5 V	10 mA	
power outputs	0 - 24 VDC	1.0 A	PWM

The analog inputs can measure voltages between 0 V and 5.0 V directly and convert them with a resolution of 10 bits (reference voltage: 5.0 V). The inputs are not galvanically separated.

The query commands "?ANIN<uv>" and "?INPUTS" correspond to the same inputs of the PS 30 (see command set, page 60). The evaluation of the inputs takes place either analog or digital.

The two power outputs are PWM-type and switching towards GND. They are designed to drive inductive loads which need a high actuating current for a short time and a low stand-by current afterwards, such as holding brakes or solenoids.

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

7. Control Architecture and Function

7.1 Assembly

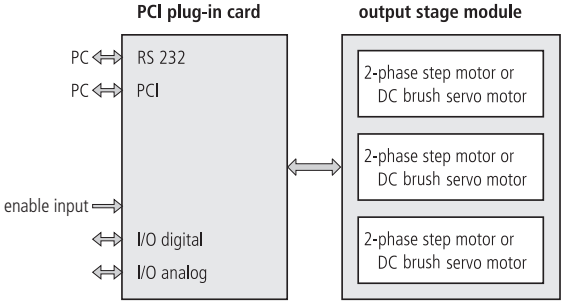

fig. 1: Control Architecture

PCI Plug-in Card

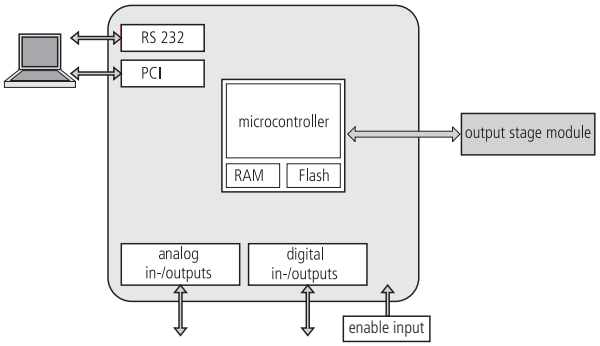

fig. 2: Main Board Control Architecture

The PCI plug-in card (main board) is the core of the PS 30. It takes over the control of the main process flow, communicates with the PC and with the output stages and governs the digital and analog inputs and outputs.

A flash memory with 512 kB and a RAM memory with 128 kB belong to the microcontroller. The flash memory is used as program memory for the microcontroller, the RAM memory as data storage. The RAM contents are buffered by a replaceable lithium cell.

The PCI plug-in card communicates with the PC through the PCI-COM bridge. Through the PCI bus an update of the firmware is possible as well. A further serial interface (RS-232) is implemented as alternative command and update interface to the PC.

The PCI plug-in card contains a motion processor which can control respectively actuate three axes. The motion processor executes the commands received from the microcontroller and generates the corresponding control signals for the output stage modules. The interface to the output stage modules is galvanically separated by optoelectronic couplers.

With the motor type DC, the motion processor generates a feed-back control loop (closed-loop operation) including the output stage, the motor itself and the encoder. Thereby, the motion processor generates direction and PWM signals. Size and direction of the DC motor torque will be given by a PWM and a direction signal. A power H-bridge on the motor driver board applies appropriate motor current to the motor coils.

Step motors are usually operated open-loop i.e., the motor controller generates the field vector via PWM signals, whereby only in the angle of the field vector, not however its length will be varied.

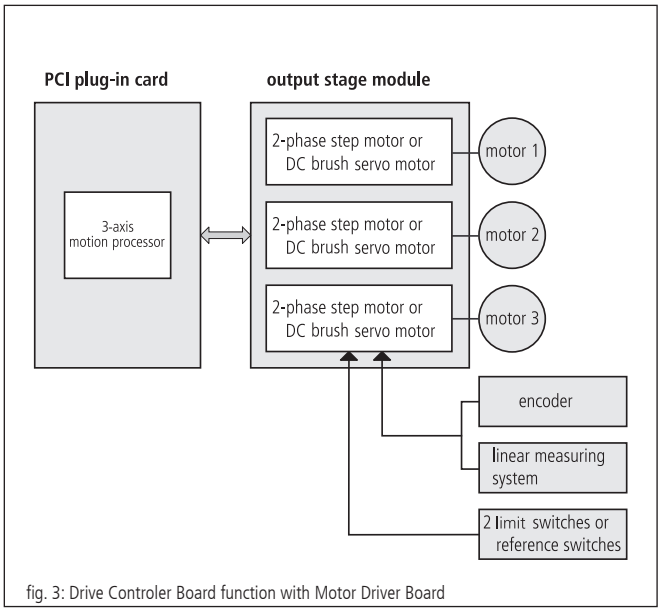

fig. 3: Drive Controler Board function with Motor Driver Board

Output Stage Module

The PS 30 can actuate maximum three motors. The interface between the PCI plug-in card and the output stage module is galvanically separated.

On the output stage module a universal motor connector is fitted. On this connector, all necessary motor signals such as motor coils, encoder and limit switches can be found.

Safety Fuse Concept

The different power and auxiliary supplies and auxiliary voltages of the PS 30 are protected by plug-in SMD fuses respectively self-resetting safety devices (Polyswitch), in order to avoid serious damage in case of a hardware defect.

PCI Plug-in Card

Auxiliary supply to the 25-pin D-Sub female connector:

- + 12 V respectively + 24 V PWM: 1 A (fast-blow)
- + 5 V: 300 mA, self-resetting (Polyswitch)

Output Stage Module

- + 12 V (PC): 2 A (slow-blow)
- + 5 V: 500 mA (slow-blow)
- + 24 V external: 5 A (slow-blow)
- + 5 V auxiliary power supply: 500 mA (fast-blow)

7.2 Operation of Different Motor Types

The PS 30 can operate both 2-phase step motors as well as brushed servo motors.

The output stage module comes already configured for various combinations according to customers' requirements. In the DC motor configuration current limiting for each axis can be set by means of a potentiometer (see "Operating Elements and Settings").

7.3 Operating Elements and Settings

Jumper Settings of Main Board

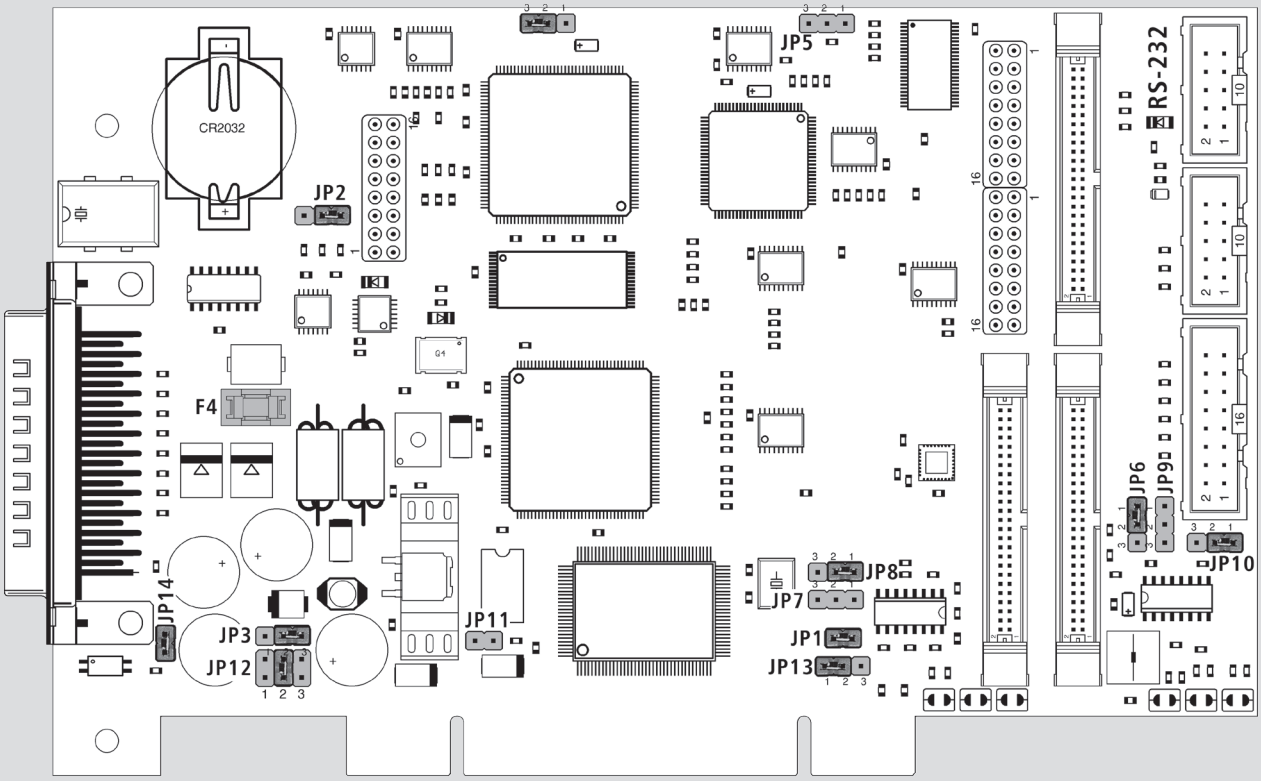

fig. 4: Jumper presets for the PCI plug-in card, defined by OWIS®

jumper	function	setting possibilities	presetting defined by OWIS®	note
JP1	flash external/internal	jumper plugged = external jumper open = internal	plugged	value preset by the manufacturer should not be changed.
JP2	boot external/internal flash	1-2: ext. flash; 2-3: int. flash	1-2 (right)	value preset by the manufacturer should not be changed.
JP3	+ 3.3 V external/internal	1-2: internal; 2-3: external	2-3 (right)	preset value can be changed with very old PC mainboard, if necessary.
JP5	reserved		no jumper plugged	
JP6	polarity Index axis 1	1-2: normal; 2-3: inverted	1-2 (oben)	
JP7	reserved		no jumper plugged	
JP8	polarity Index axis 2	1-2: normal; 2-3: inverted	1-2 (right)	
JP9	reserved		no jumper plugged	
JP10	polarity Index axis 3	1-2: normal; 2-3: inverted	1-2 (right)	
JP11	enable or disable EEPROM	jumper plugged = EEPROM disabled jumper open = EEPROM enabled	open	value preset by the manufacturer should not be changed.
JP12	operation voltage of the PCI-COM bridge	1: + 5 V; 2: automatical; 3: + 3.3 V	2 (mid)	manual preset with very old PC-mainboard, if necessary.
JP13	PCI-bus frequency	1-2: 33 MHz; 2-3: 66 MHz	1-2 (left)	value preset by the manufacturer should not be changed.
JP14	output stage release	jumper plugged = release; jumper open = release with external release input possible	plugged	

Safety Fuse of Main Board

F4: 1 A fast-blow for the protection of the + 12 V- bzw. + 24 V auxiliary power supply for the PWM outputs.

Jumper and Potentiometer Settings for PS 30 Output Stage Module

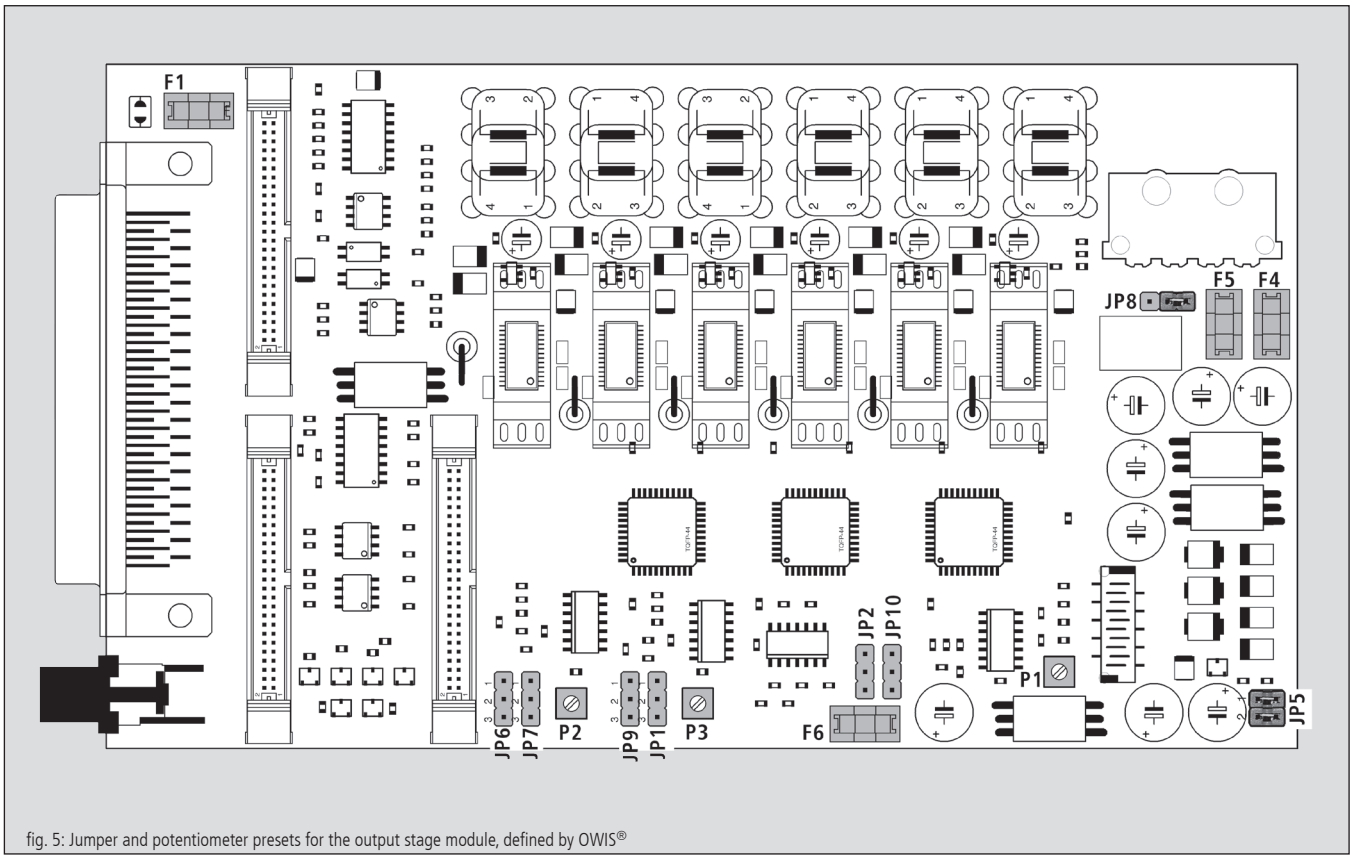

fig. 5: Jumper and potentiometer presets for the output stage module, defined by OWIS®

jumper/ potentiometer	function	setting possibilities	presetting defined by OWIS®	note
JP2/JP10	motor type axis 1	1-2 (top): 2-phase step motor Open Loop 2-3 (bottom): DC servo motor	dependent on configuration; both jumpers must always have the same position.	value preset by the manufacturer cannot be changed.
JP6/JP7	motor type axis 2			
JP1/JP9	motor type axis 3			
JP8	auxiliary power supply + 5 V	1-2: internal 2-3: external (from PC power supply connector)	2-3 (left)	value preset by the manufacturer should not be changed.
JP5	supply + 12 V respectiv. + 24 V auxiliary power supply for PCI plug-in card	two jumpers: 1 (top) / 2 (bottom)	both jumpers plugged	set or remove both jumpers together.
P1	current limiting DC motor, axis 1	left stop (rotation counterclockwise): 2.4 A (current range 1) 6.6 A (current range 2) *) right stop (rotation clockwise): current zero *) the max. allowed current is 3.5 A	depending on configuration	setting of the current limiting is done by the manufacturer during the configuration procedure.
P2	current limiting DC motor, axis 2			
P3	current limiting DC motor, axis 3			

Safety Fuses for PS 30 Output Stage Module

F4: 3 A slow-blow for the protection of the + 12 V power-supply of the hard-disk power supply connector
F5: 500 mA fast-blow for the protection of the internal + 5 V auxiliary power supply
F6: 5 A slow-blow for the protection of the external 24 V power input for the motor power stages
F1: 500 mA fast-blow for the protection of the 5 V auxiliary power supply

7.4 Selection of the Current Range for the Motor Power Stage

The PS 30 motor power stage has two configurable current ranges in order to obtain a high precision in the current setting respectively a micro step resolution at its best.

After switching on the control unit, the current range selected is stored in the static RAM. In order to activate a new current range, it is necessary to reinitialize the axis <n> after the preset has been done.

Preselection of the current range 2 for axis <n> takes place after the following command sequence:

AMPSHNT<n>=1

INIT<n>

In order to switch back to current range 1 (low current), use the following command sequence:

AMPSHNT<n>=0

INIT<n>

Phase Current Setting for 2-Phase Step Motors

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

driving current: DRICUR<n>=<uv>

holding current: HOLCUR<n>=<uv>

maximum phase current, current range 1 (corresponding to 100%): 1.2 A

maximum phase current, current range 2 (corresponding to 100%): 3.3 A

With the standard motor driver board, a phase current of 1.8 A max., corresponding to 54.5% of the maximum value that can be preset, should not be exceeded.

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

Note:
The maximum thermally-limited continuous current of the motor power stages (automatic shutdown when exceeding a printed circuit board temperature of approx. 85°C) is heavily dependent on the connected motor types, the number of actuated motors, their mechanical load, the motion velocities, the PC internal temperature and ventilation conditions, the ambient temperature, etc.

Therefore, no exact value for a guaranteed phase continuous current can be indicated. With normal ambient temperature and average installation conditions, it can be assumed that a phase current of approx. 1.8 A with three attached motors can be reached.

Current Range Setting for DC Servo Motors

The suitable current range for the DC servo motors has to be set in accordance with the thermally admissible continuous current of the corresponding motor type. A current limiting can be configured on the corresponding board. (For further information please see chapter "Operating Elements and Settings".)

7.5 Output Stage Configuration by Software

Various characteristics of the motor power stages, which in the most applications do not need to be set or changed, can be configured before the initialization of a power stage ("INIT…" command).

Four output bits are available, which can be set by means of "ETTLOUTS<n> = <uv>" respectively reset (deleted) by means of "ETTLOUTC<n> = <uv>". Axis <n>, where the setting has been changed, as well as a four-digit bit mask <uv> will be transferred. A cleared bit (0) means that the status of the respective output may not be changed. A set bit (1) means that the output bit concerned has to be set (ETTLOUTS) respectively deleted (ETTLOUTC).

Standard settings are marked grey in the following tables.

Bit 1: Timing

bit no. 4-3-2-1	blanking time
x-x-x-1	1.3 µs
x-x-x-0	0.6 µs

Bits 3 and 4: Decay of the Motor Coils

bit no. 4-3-2-1	decay with 2-phase step motor *
0-0-x-x	fast decay, 100%
0-1-x-x	mixed decay B, 48%
1-0-x-x	mixed decay A, 15%
1-1-x-x	slow decay, 0%

*) decay is active with falling curent.

bit no. 4-3-2-1	decay with DC servo motor
x-0-x-x	fast, if PWM = 0
x-1-x-x	slow, if PWM = 0
0-x-x-x	fast, if current chopper active
1-x-x-x	slow, if current chopper active

8. Control Functions

8.1 Trapezoidal Point-to-Point Profile

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

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

For this profile, them host specifies an initial acceleration and deceleration, a velocity and a destination position. The profile is named after the curve shape (fig. 6): the axis accelerates linearly (on the basis of the programmed acceleration value), until it reaches the programmed speed. Afterwards, the axis slows down linearly (using the deceleration value), until it stops at the defined position. If the programmed travelling distance is so short that deceleration must begin before the axis reaches the programmed velocity, the profile will not have a constant-velocity range, and the trapeze becomes a triangle (fig. 7).

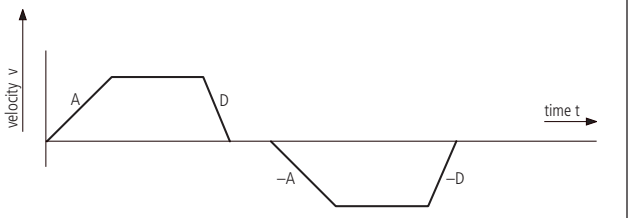

fig. 6: A = acceleration, D = deceleration

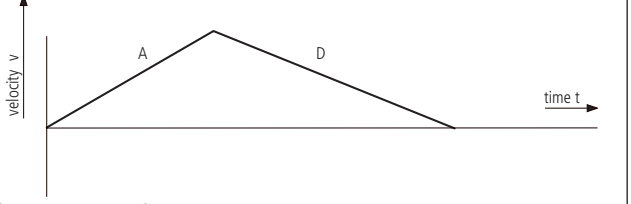

fig. 7: Trapezoidal profile, A = acceleration, D = deceleration

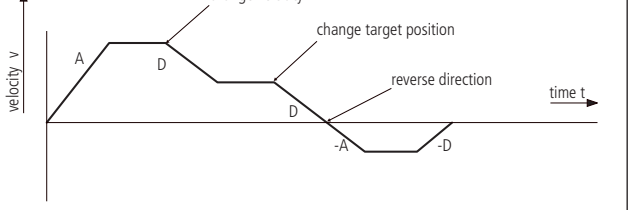

fig. 8: A = acceleration, D = deceleration

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

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

It is possible to change one of the profile parameters while the axis is in this profile mode. The profile generator will always try to execute the movement within the set conditions given by the parameters. If the end position is changed during the movement so that the remaining travel distance changes sign, the PS 30 will decelerate to stop and then accelerate in reverse direction to move to the specified target position.

8.2 S-Curve Point-to-Point Profile

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

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

Compared to the trapezoidal point-to-point mode, the S-curve point-to-point profile adds one more parameter ("jerk"). This defines the rate with which the acceleration changes.

If a positioning is performed in this mode, first the acceleration increases linearly with the set value for jerk until it reaches its programmed value. The change from constant acceleration to constant velocity is performed with a linear rise in deceleration as well. At the end of the movement profile the behaviour is analogous to this.

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

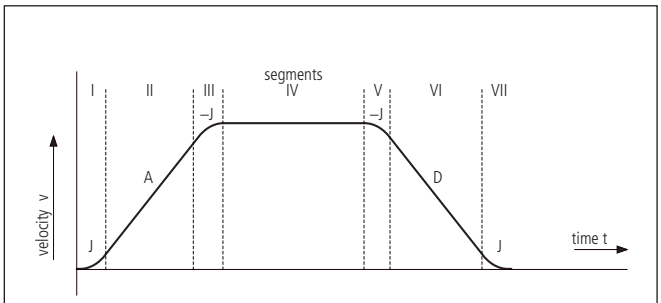

fig. 9: S-curve profile, A = acceleration, D = deceleration J = jerk

Fig. 9 shows a typical S-curve profile. In segment I the acceleration value increases by the value set by the jerk, until the maximum acceleration is reached. The axis continues accelerating linearly (jerk = 0) within segment II. The profile uses then the negative value of the jerk in segment III in order to reduce acceleration. In segment IV the axis moves with maximum speed (V). Then, the profile slows down similarly to the acceleration value, by using the negative jerk in opposite direction, in order to first reach the maximum acceleration (A) and then to halt the axis at the end position.

It is possible that a S-curve profile only contains some of the segments shown in fig. 9. This can e.g. be the case, if the maximum acceleration cannot be reached before "half a way" in direction end velocity or end position. This profile does not contain segments II and VI (see fig. 10).

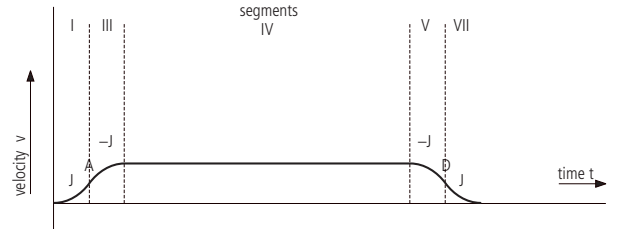

fig. 10: The S-curve does not reach the max. acceleration.
A = acceleration, D = deceleration, J = jerk

If a position is defined in such a way that the end acceleration cannot be reached, then there is no segment IV (see fig. 11).

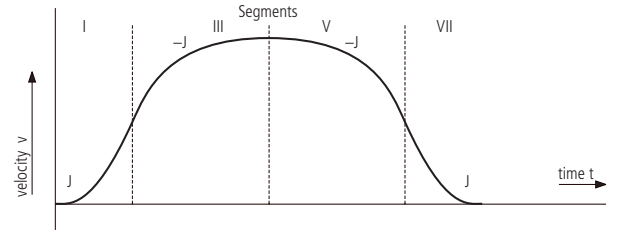

fig. 11: The S-curve without max. acceleration segment,
A = acceleration, D = deceleration, J = jerk

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

8.3 Velocity Mode

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

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

Unlike in trapezoidal and S-curve profiling modes, where the final position determines whether positive or negative speed is defined, it is the sign of the velocity value transmitted within the velocity mode that determines whether the axis moves in positive or negative direction. Therefore, the velocity value sent to the PS 30 can take positive values (for positive direction of motion) or negative values (for reverse direction of motion). For this profile no destination position is specified.

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

A simple velocity profile looks like a simple trapezoidal point-to-point profile as shown in fig. 6.

Fig.12 shows a more complicated profile, in which both speed and direction of motion change twice.

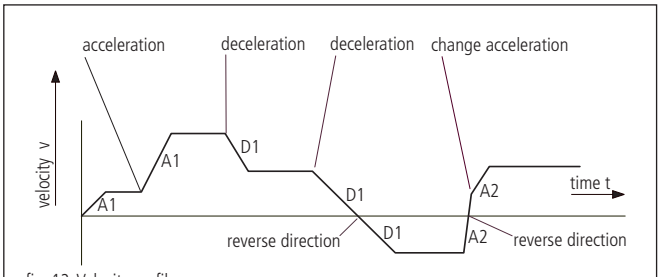

fig. 12: Velocity profile

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

8.4 Reference run

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

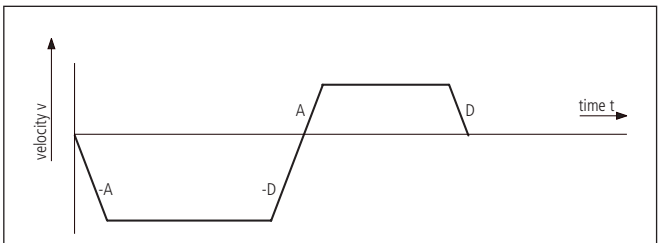

fig. 13: Reference drive, A = acceleration, D = deceleration

8.5 Operating Mode of Linear Interpolation

Definition

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

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

Functional Principle

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

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

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

The linear interpolation axes are designated in the following by (i). The following diagram shows the general shape of the speed profile of the guiding axis vf (t) and any linear interpolation axis vn (t) by an example:

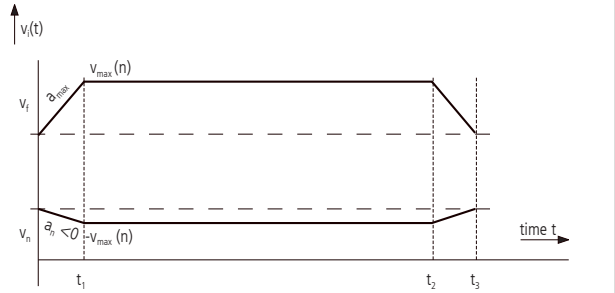

fig. 14: Linear interpolation

In the example the driving distance of the axis (n) is negative, the driving distance of the guiding axis (f) is positive. At time t1 the acceleration phase is completed. The deceleration starts with at t2, and all axes stop together at time t3.

8.6 Operating Mode of the General Continuous Path Control

Definition

The PS 30 enables the approximation of any paths by chains of single vectors which are passed to the control in a vector table. Therefore, the general continuous path control is realised by a vector mode.

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

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

Realisation of Vector Mode

Vector table

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

n	$\Delta\vec{x}$	Δt	F	E	T
1	$\Delta x_{a1}, \Delta x_{b1}, \Delta x_{c1}, 0, 0, 0, 0, 0$	t_1	t_1	e_1	t_1
...	...	...	...	...	...
N	$\Delta x_{aN}, \Delta x_{bN}, \Delta x_{cN}, 0, 0, 0, 0, 0$	Δt_N	f_N	e_N	t_N

Maximum 2000 vectors can be defined ($N_{max} = 2000$).

The elements of the motion vector (single distances) are represented as integral signed values (integer 16 bit). The maximum path distance for a time interval Δt_n is 32760 increments, i.e. for the range of values of a position entry numerical values from - 32760 to + 32760 are permissible.

Segment duration

The time interval Δt_n for the driving segment <n> is indicated as integral multiple of 1.024 ms. The values facet range from 20 to 1638, out of this a definable segment time of minimum 20.48 ms to maximum 1.677312 s in steps of 1.024 ms results:

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

$\Delta t_{n_{max}} = 1638 \cdot 1.024\,ms = 1.677312\,s$

Control codes

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

The function code F is represented as 16 bit value. In the current firmware version bit 15 is used to preselect the mode of operation, i.e. "constant velocity" (v=const, bit 15 deleted) or "constant acceleration" (a=const, bit 15 set). The remaining bits are reserved for possible extensions in future. Thus, the "constant speed" f = 0 is to be set for the "constant acceleration" and correspondingly f = 32768, for the standard mode of operation.

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

The 8-bit enable code T defines which of the axes 1 to 3 is active in the vector mode. The allocation of the single bits to the axis number corresponds to the error code E, i.e. a set bit 0 stands for axis 1 is active etc.

Operating methods

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

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

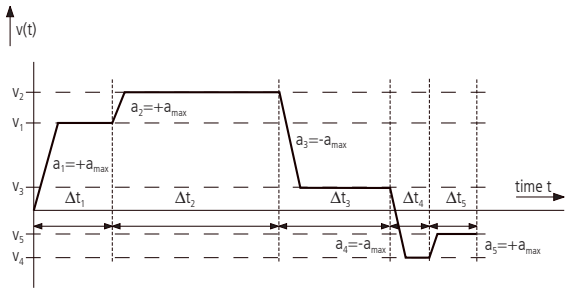

fig. 15: Velocity-time-diagram for mode of operation v=const.

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

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

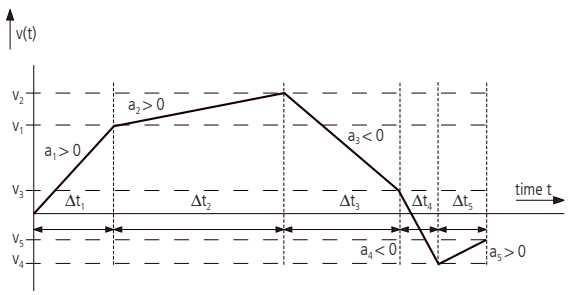

fig. 16: Velocity-time-diagram for mode of operation a=const.

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

Maximum velocity and acceleration

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

Plausibility check

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

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

Axis enabling

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

Syntax

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

POSTAB <n> = Δx_{an}, Δx_{bn}, Δx_{cn}, 0, 0, 0, 0, 0,

Δt_n,f_n, e_n, t_n

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

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

PTABPLAUS <n>.

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

?POSTAB <n>
returns as answer:

Δx_{an}, Δx_{bn}, Δx_{cn}, 0, 0, 0, 0, 0, Δt_n,f_n, e_n, t_n Vel_i, Acc_i

Example:

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

Segment time about 100 ms

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

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

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

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

Operation mode a=const.

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

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

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

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

IVEL1=800000

IVEL2=500000

IVEL3=300000

IACC1=2000

IACC2=4000

IACC3=10000

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

Plausibility check using

?PTABPLAUS0

and read out the table elements by

?POSTAB0

returns the answer:

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

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

End of motion

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

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

Circular Interpolation

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

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

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

Definitions:

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

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

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

Radius of the segments of a circle: r

Illustration by the diagram:

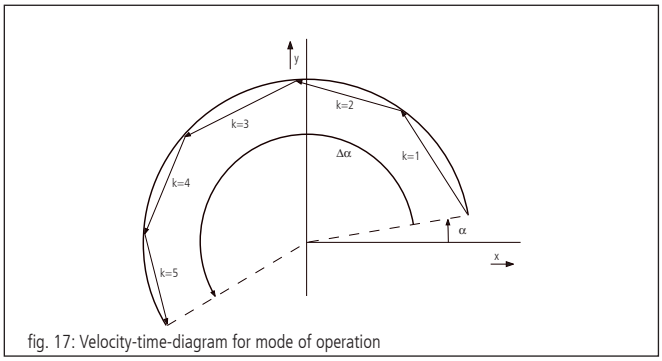

fig. 17: Velocity-time-diagram for mode of operation

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

Calculation

The approximated segment of a circle is defined by the radius, the number of secants, the angle offset and the angle range.
The direction of rotation is fixed by the sign of the angle range specification. Here a positive angle corresponds to a counterclockwise turn during appropriate arrangement of the axes (see also the position of the coordinate system in aforementioned diagram).

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

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

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

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

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

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

Scaling factor

The scaling factor represented by counters and denominators is meant to adapt different resolution values of the two circle interpolation axes and to realize ellipses respectively. They can be set by two separate commands. The denominator is designated with N, the counter with Z.
If N > Z, the y-axis leads and the path data of x are divided by (N/Z). If Z > N, the x axis leads and the path data of y are divided by (Z / N). The default value is Z = N = 1, if no data are given by the user.

Syntax

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

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

Example:

PTABCIRCLE0=1,2,326,0,5,1000,10,190,1,1
generates a pitch circle starting from table element 0 with axis 1 as x and axis 2 as y-axis, segment time 1/3 second, mode of operation v=const., 5 secants, radius 1000 increments, starting angle 10°, angle range 190° and scaling 1/1.

9. Travel Measuring

Encoder

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

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

Encoders are mounted stationary on the motor and directly connected with the rotor. The encoder output signals are named A and B (CHA and CHB) with a phase-shift of 90 degrees (quadrature signals), and, if necessary one Index pulse per revolution. The PS 30 can process TTL-level or antivalent signals (line-driver outputs). After level transformation and filtering, the signals are transmitted directly to the motion processor.

Linear Measuring System

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

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

Evaluation of Linear Measuring Systems

Optionally, the PS 30 can be equipped with an additional quadrature-encoder counter board.

The signals of the linear measuring system correspond to the encoder signals specified before (quadrature A and B as well as Index I). On the quadrature-encoder counter board there is a 32-bit counter for each axis. The counter values are read out by the main processor. The maximum counting frequency is 5.5 MHz (signal) or 22 MHz (quadrature), respectively.

Position Feedback Control

Two encoder inputs are provided to operate servo motors. The first encoder signal serves the data acquisition for the closed-loop position control (PID type), the second, optional dual-loop encoder, serves the optional follow-up position control.

Function of the Follow Up Control

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

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

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

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

A positioning run with a follow-up control corresponds to the following 3-phase scheme. Depending on the settings not all phases must occur:

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

break ramp will be triggered. If the actuator runs beyond the limit, the direction will be reversed, etc.

- It can also be set by a system parameter whether or not the follow-up control in the velocity mode should always be active or switched off upon reaching the target window.

Calculation of the conversion factor F:

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

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

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

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

calculation of r_s:

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

with:

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

example:

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

it follows:

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

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

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

Thus, we have

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

and therefore:

Z = 5

N = 1.

10. PID Servo Loop Algorithm

The servo filter used in the PS 30 operates according to a PID algorithm. An integration limit provides an upper bound for the accumulated error.

The PID formula is as follows:

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

Meaning of following abbreviations:

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

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

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

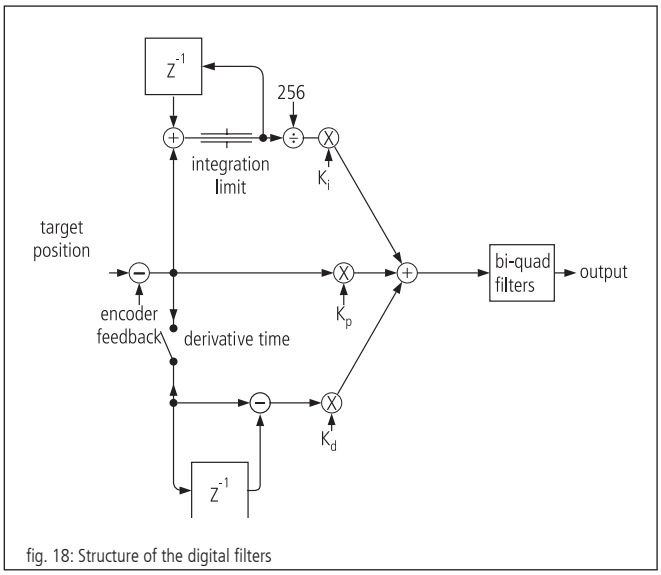

fig. 18: Structure of the digital filters

11. Positioning Velocity and Acceleration, Calculation

11.1 2-Phase Step Motor (Open Loop)

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

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

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

The PS 30 has a digital profile generator. The speed profiles are periodically calculated and sent to the 2-phase step motor.

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

$T_P = 256\,\mu s$

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

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

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

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

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

resp.

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

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

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

resp.

$$n_{RPS} = \frac{10}{s} \cdot \frac{1}{Mcstp \cdot R \cdot T_P} \cdot \frac{V}{65536} \qquad \text{(revolutions/second)}$$

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

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

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

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

Travelled distance during the trapezoidal profile acceleration/deceleration :

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

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

General Information
The PS 30 has a digital position/speed controller. Output and control variables are periodically calculated. The acquisition of the actual position value is done in the simplest case by means of a rotary encoder, which is attached to the 2nd shaft extension of the motor. The most important parameter of the encoder is the number of encoder lines R. This is the number of the light/dark periods on the encoder disc for each motor shaft revolution. The signals go through a quad evaluation, which results in a generally 4-fold higher resolution than the number of encoder lines.

Servo Loop Cycle Time
The cycle duration of the digital profile generator is defined by hardware. It is preset to 256 µs. It is defined by hardware. The minimum cycle time is 204.8 µs. If necessary, it can be increased by an integer multiple of 51.2 µs:

$T_s = 204.8\,\mu s + n \cdot 51.2\,\mu s;\ n \in [0, 1, ..., 386]$

corresponding to a cycle time of

$T_s = [204.8\,\mu s, 256\,\mu s, ..., 19986\,\mu s]$

Only integer values can be handed over to the PS 30. The value is rounded internally to the next valid value.
Default value (presetting): $T_s = 256\,\mu s$.

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

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

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

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

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

resp.

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

resp.

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

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

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

Example:

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

Solution:

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

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

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

Acceleration for Trapezoidal Velocity Profiling

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

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

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

Travelled distance during the trapezoidal acceleration/ deceleration ramp:

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

12. Initial Operation of the PS 30

12.1 Installation and Preparation

Note:
The installation of PCI plug-in card and output stage module into the control PC may only be done when the PC is switched off, i.e., the power supply plug is disconnected!

Before the installation is done, corresponding slots for PCI card and output stage card must be selected, and both cards must be interconnected through the three 50-pin flat ribbon cables provided. The length of the ribbon cables enables the connection of the output stage module at a certain distance from the PCI plug-in card (up to three slots).

The flat cables are to be installed in such a way that they do not cross each other. If, by mistake, the cables from the PCI are cross-connected to the output stage card, no damage is caused. However, the axis assignment X-Y-Z, 1-2-3 respectively, is interchanged and does not match the tables in the manual or the labelling of the 3-way adapter.

Note:
Please take care that all work on the cards is accomplished exclusively in a working environment free of electrostatic discharge (ESD protected area). Furthermore, please check particularly the correct placement of the PCI plug-in card after the installation. The PCI connector may, under no circumstances, be separated during the operation, otherwise the PC and the PCI plug-in card might be destroyed.

12.2 Connection of Peripherals and Devices

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

This is:

- the positioning unit
- the power supply
- the computer

The controller is connected via the PCI interface to the computer.

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

For the installation please start "setup.exe".

Note:
Any equipment and peripherals must be connected before the system starts, as otherwise it will not be will not be recognized by the controller and initialized.

12.3 Getting Started

After successful installation, all peripheral devices have to be connected, before the current supply is activated and the PC is switched on.

When Windows is first started after the PS 30 has been installed, the operating system should recognize the new hardware. The drivers can then be installed, too. In order to do this, administrator rights are necessary.

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

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

Software
Following tools are part of delivery: the software tool OWISoft, the PCI driver (PCI-COM bridge) and the software interface (SDK/API) for C, C++, C#, LabView (V 8.2 and higher) and additional programming languages (32/64 bit). Thus, the PS 30 can be configured and operated comfortably.

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

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

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

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

For adjusting, please read the manual OWISoft.

If the control shall be used by a user´s own software, please read the chapter "Instructions Concerning the Setup of User Application Software". There you will also find the command table for the PS 30 as well.

13. Malfunction Monitoring

13.1 Limit Switches

The PS 30 has two limit switch inputs for limit switches (MINSTOP, MAXSTOP) as well as evaluation possibility for a reference switch for each axis. One of the two limit switches is defined as reference switch.

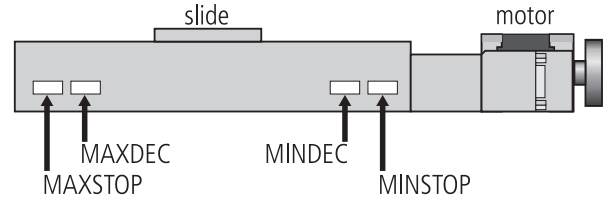

fig. 19: Identification of limit switches

OWIS® positioning units are provided with maximum four limit switches. The limit switches working in negative direction (motion of the slide towards the motor) are named MINDEC and MINSTOP. The limit switches working in positive direction (motion of the slide away from the motor) are similarly named MAXDEC and MAXSTOP. MINDEC and MAXDEC cannot be evaluated by the PS 30.

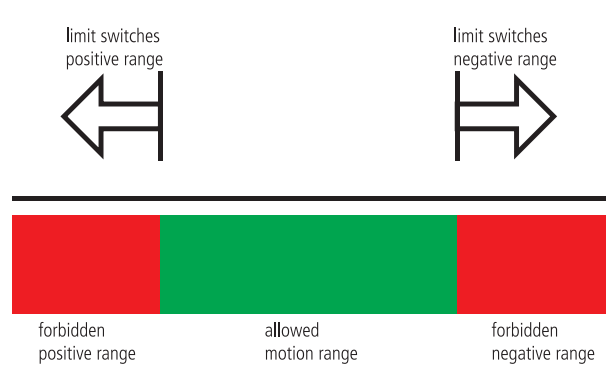

fig. 20: Function of limit switches

Working Principle of the Limit-Switch Monitoring

1. MINSTOP: Actuation of this limit switch with motion in negative direction results in immediate disable of the motor power, after a certain reaction time which can be some milliseconds.
   DC servo motor: The motor is disabled. However, the residual kinetic energy leads to some remaining movement until it is used up by friction or stoppers.
   Step motor open loop: If the current travel frequency with which it is stopped was higher than the system start-stop frequency, the kinetic energy in the system leads to a remaining motion. This motion cannot be detected by the control unit, thus resulting in a wrongly indicated potision. A reference travel is necessary to match the current position with the motor steps.
2. MAXSTOP: The reaction is similar to the MINSTOP limit switch, but the effect is in positive direction.

Configuration of Limit and Reference Switches

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

Generally, MINDEC and MAXDEC have to be deactivated by software. That means that the bits of the limit and reference switch mask ("SMK…"/"RMK…"), representing MINDEC and MAXDEC, must be set to 0, when configuring the axes. The signals are not evaluated by the hardware.

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

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

Reconnection after Axis Error

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

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

13.2 Output-Stage Error Monitoring

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

13.3 Motion-Controller Error Monitoring

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

13.4 Time-Out Monitoring

Additionally, a timeout value (in ms, 32-bit range) can be defined as parameter for each axis. The monitoring can be switched off by setting the timeout to 0. This timeout is monitored periodically, while a motion is executed (PGO, REF, EFREE, PWMSGO, LIGO). If the motion lasts longer than this time, then the motor is shut off (?ASTAT → "Z", see comand table p. 61), that means that the control loop is opened and the power stage is disabled. This function is useful, if, for instance, during the reference motion one of the reference switches cannot be found.

14. Instructions Concerning the Setup of an Own Application Software

Generally, a PS 30 application consists of an initialization part which sets the necessary axis parameters for all for axes <n> to be used and which switches on the axes, too. Furthermore, it consists of a loop which executes a reference motion for all axes and of the actual user program with all the functions required.

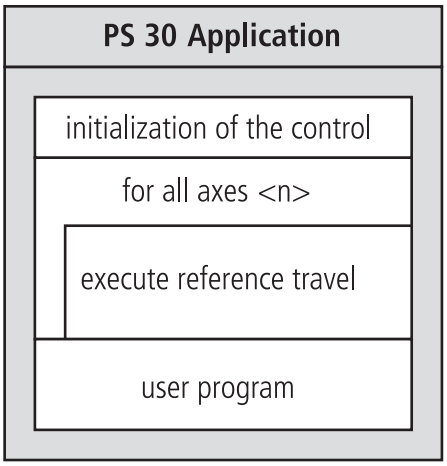

fig. 21: PS 30 application architecture

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

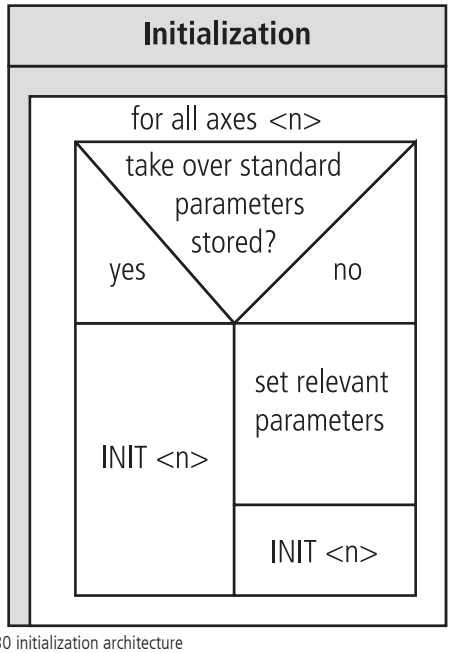

fig. 22: PS 30 initialization architecture

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

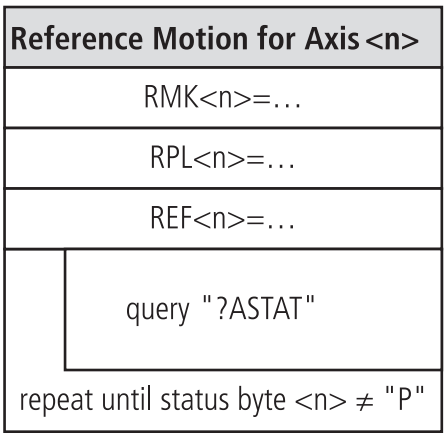

fig. 23: PS 30 reference motion for axis

A command processing time (interpretation time) of about 20 to 40 milliseconds has to be considered between two individual commands sent to the PS 30. The control unit signals received can be e.g. retrieved character by character every millisecond, until the defined end-of-string identifier is received.

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

15. Command Set for the PS 30

General information concerning the command format:

Each command is transferred over the interface (PCI or RS-232) in ASCII format. The individual characters of a command are converted automatically into capital letters. Each command ends with CR or CR+LF or LF (adjustable). Furthermore, the response mode can be preset (TERM). For this purpose, there are three settings available:

1) When reading out the message buffer, only a two digit number is returned (error code). This setting is especially selected when a control takes place via software through a host PC, since the message strings are here at the shortest, and therefore the command throughput is optimized.

2) Reading the message buffer returns a two digit number (error code) and an additional plain text string explaining the error code.

3) Similar to 2) and, additionally, each executed command giving no return value, will be acknowledged by "OK".

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

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

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

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

<n> = axis number 1…3 (respectively, highest possible axis number)
<uv> = unsigned integer value
<sv> = signed integer value
<v> = signed way indication

Attachment
I Command Table

command group	commando	function description	example	response
general status request	?ASTAT	Axis status inquiry, a character each axis is returned to describe the current axis mode: "I" = axis is initialized "O" = axis is disabled "R" = axis initialised and ready "T" = axis is positioning in trapezoidal profile "S" = axis is positioning in S-curve profile "V" = axis is operating in velocity mode "P" = reference motion is in progress "F" = axis is releasing a limit switch "J" = axis is operating in joystick mode "L" = axis has been disabled after approaching a hardware limit switch (MINSTOP, MAXSTOP) "B" = axis has been stopped after approaching a brake switch (MINDEC, MAXDEC) "A" = axis has been disabled after limit switch error "M" = axis has been disabled after motion controller error "Z" = axis has been disabled after timeout error "H" = phase initialization activ (step motor axis) "U" = axis is not released. "E" = axis has been disabled after motion error "W" = axis is positioning in trapezoidal profile with WMS "X" = axis is positioning in S-curve profile with WMS "Y" = axis is operating in velocity mode with WMS "C" = axis is operating in velocity mode with continuous-path control "?" = error, unknown status of axis	?ASTAT	IOR
	?MSG	Read out the message exit buffer, the message exit buffer is used only for error messages, which concern the command interface (wrong command, missing parameters, invalid value). Possible messages are: "00 NO MESSAGE AVAILABLE" (will be returned after the attempt to read out the message buffer, even though no message is currently present) "01 PARAMETER BEFORE EQUAL WRONG" (will be written into the message buffer, if the command interpreter has failed to convert the parameter before the equals sign into a number correctly) "02 AXIS NUMBER WRONG" (will be written into the message buffer, if the command interpreter has failed to evaluate the given axis number correctly; valid: 1...9, e.g.) "03 PARAMETER AFTER EQUAL WRONG" (will be written into the message buffer, if the command interpreter has failed to convert the parameter after the equals sign into a number correctly) "04 PARAMETER AFTER EQUAL RANGE" (will be written into the message buffer, if the command interpreter has recognized that the parameter after the equals sign is beyond its valid range) "05 WRONG COMMAND ERROR" (will be written into the message buffer, if a syntax error has occurred, i.e., the command interpreter has not been able to recognize the given command) "06 REPLY IMPOSSIBLE"(will be returned, if the reply could not be transferred to the host, because the output buffer is not yet empty, e.g.) "07 AXIS IS IN WRONG STATE" (will be written into the message buffer, if a positioning command or a configuration parameter has been sent that could not be recognized because the axis is currently in a different motion state)	?MSG	00 NO MESSAGE...
	?ERR	Error queries from the error memory with a memory depth of 20. The error number is always returned as number with four digits. On the basis of the error code the cause can be determined. If the returned value is 0, there are no further errors stored.	?ERR	1211
	ERRCLEAR	Clear error memory.	ERRCLEAR	
	?EMERGINP	Returns current state of emergency-stop.	?EMERGINP	1

command group	commando	function description	example	response
general status request	?ESTAT<n>	Read out current logical state of the limit switches and power stage feedback for an axis: bit 0 = MINSTOP bit 1 = MINDEC bit 2 = MAXDEC bit 3 = MAXSTOP bit 4 = motor power-stage error	?ESTAT	10001
	?AXSIGNALS<n>	Query hardware axis signals for an axis: bit 0 = encoder CHA, bit 1 = encoder CHB, bit 2 = encoder Index, bit 3 = encoder Home, bit 4 = MAXSTOP, bit 5 = MINSTOP, bit 6 = reserved, bit 7 = reserved, bit 8 = reserved, bit 9 = reserved, bit 10 = reserved, bit 11-15 = reserved.	?AXSIGNALS1	0000011101101001
	?READOWID<n>	Read out memory contents of ONE-Wire-Chip from positioning unit until 0x00 limit detection and data transfer to the PC. As parameter the initial address 0x00 up to 0x70 will be passed in the ONE-Wire-Chip. From this Address max. 16 bytes will be detected or it will be read until the end recognition.	?READOWID1=0	INFO1|INFO2|....
	?READOWUB<n>	Read out memory contents of ONE-Wire-Chip from address 0x86 and 0x87 (=UserBytes) in the positioning unit and transmission of data to the PC.	?READOWUB1	10
base configuration	AXIS<n>=<uv>	Release or unreleased a certain axis. With this command one can release (1) or unrelease (0) an axis.	AXIS3=1	
	?AXIS<n>	Read out release status for an axis. If an axis is released, a "1" is sent back to the host, otherwise a "0" is returned.	?AXIS3	1
	MOTYPE<n>=<uv>	0 = DC brush 2 = Step motor Open Loop 3 = Step motor Closed-Loop	MOTYPE1=0	
	?MOTYPE<n>	Read out motor type for an axis.	?MOTYPE1	0
	AMPSHNT<n>=<uv>	Set current range for one axis: 0 = current range 1 (low) 1 = current range 2 (high)	AMPSHNT1=0	
	?AMPSHNT<n>	preselected current range for one axis	?AMPSHNT1	1
	TERM=<uv>	Set terminal mode: mode 0 = short response mode 1 = response with plain text mode 2 = response with plain text and "OK" handshake after each command without feedback.	TERM=2	
	?TERM	Query terminal mode.	?TERM	2
	BAUDRATE	Set baud rate for the serial interface, allowed values : 9 600, 19 200, 38 400, 57 600, 115 200. This setting becomes active only after the next reset or power-on.	BAUDRATE=9600	
	?BAUDRATE	Read out current baud rate for the serial interface.	?BAUDRATE	9600
	COMEND	Set command end identification: 0 = CR, 1 = CR+LF, 2 = LF	COMEND=0	
	?COMEND	Read out command end identification.	?COMEND	0
	SAVEGLOB	Store global parameters in serial FRAM.	SAVEGLOB	
	LOADGLOB	Recall global parameters out of the serial FRAM.	LOADGLOB	
	SAVEAXPA<n>	Store axis parameters for an axis in the serial FRAM.	SAVEAXPA1	
	LOADAXPA<n>	Recall axis parameters for an axis in the serial FRAM.	LOADAXPA1	
	?SERNUM	Query serial number of the PS 30.	?SERNUM	09080145
	?VERSION	Read out software version of the firmware installed on the main board.	?VERSION	PS30-V5.0-240512
	?MCTRVER	Return version data for the motion controller chips.		
	?PCHECK	Calculate and read out checksum of the program memory.	?PCHECK	12227
	JZONE=<uv>	Set inactive joystick zone (0 -256).	JZONE=25	
	?JZONE	Read out inactive joystick zone.	?JZONE	25
	JZEROX=<uv>	Set center point for the X joystick.	JZEROX=505	
	?JZEROX	Read out center point for the X joystick.	?JZEROX	505

command group	commando	function description	example	response
base configuration	JZEROY=<uv>	Set center point for the Y joystick.	JZEROY=515	
	?JZEROY	Read out center point for the Y joystick.	?JZEROY	515
	JZEROZ=<uv>	Set center point for the Z joystick.	JZEROZ=508	
	?JZEROZ	Read out center point for the Z joystick.	?JZEROZ	508
	JBUTTON=<uv>	Switch on/off the evaluation of joystick button.	JBUTTON=1	
	?JBUTTON	Read out whether the joystick button will be evaluated or not.	?JBUTTON	1
positioning mode	INIT<n>	Enable the motor power stage, release and activate position control loop. With this command, the axis is initialized completely, the motor is powered and the position control feedback loop is active. **This command must be transferred after switching-on, so that the axis can be taken into operation, using the commands REF, PGO, VGO, etc.** Before this, the following parameters must have been preset: motor type, limit switch mask and polarity, feedback control loop parameters, current range of the motor output stage.	INIT1	
	PSET<n>=<sv>	Set target position respectively relative travel distance (ABSOL/RELAT) for an axis. If absolute position format is switched on, then the parameter is interpreted as signed absolute position; if relative position indication is chosen, then the parameter is interpreted as signed travel distance. The new absolute target position is the sum of last absolute target position and transferred travel.	PSET2=100000	
	?PSET<n>	Read out target position resp. relative travel distance for an axis.	?PVEL1	10000
	PCHANGE<n>=<sv>	Change target position or distance of an axis, while this axis goes in the trapezoidal profile.	PCHANGE2= 50000	
	?CMDPOS	Read out current target position of the PID regulator.	?CMDPOS1	5000
	VVEL<n>=<sv>	Set target speed for velocity mode for an axis. With this command, the start speed and maybe also a new speed are transmitted, while the axis moves in the velocity mode.	VVEL1= -20000	
	?VVEL<n>	Read out target speed for velocity mode.	?VVEL1	-20000
	PGO<n>	Start positioning for an axis. The axis approaches the new target position in trapezoidal or S-curve profiling mode (see "PMOD").	PGO2	
	VGO<n>	Start velocity mode for an axis.	VGO2	
	STOP<n>	Stop motion of an axis. Any active motion command for an axis is interrupted. The drive decelerates with the preset ramp parameters and halts.		
	VSTP<n>	Stop velocity mode for an axis. If an axis is in the velocity mode, this command will terminate this mode and stop the axis.	VSTP2	
	EFREE<n>	Release limit switch(es) of an axis. After a drive has moved onto a limit switch (MINSTOP, MAXSTOP) or brake switch (MINDEC, MAXDEC), the active switch(es) can be released using this command. The direction of the movement is automatically decided according to whether a positive or negative limit or break switch is activated.		
	MON<n>	Enable the motor power stage and activate position control feedback loop. With this command, an axis that has been switched off previously (by means of the "MOFF" command) can be switched on again. Position control loop and the enable input for the power stage are activated.	MON1	
	MOFF<n>	Disable the motor power stage and deactivate position control feedback loop. With this command, position control loop and the enable input for the power stage are deactivated. The motor is switched off.	MOFF1	
	JOYON	Activate joystick mode for the predefined joystick axes. Thereafter, one up to three axes move in velocity mode. The velocity and the direction are given by joystick.	JOYON	
	JOYOFF	Terminate the joystick mode.	JOYOFF	
	CNT<n>=<sv>	Set current position counter for an axis.	CNT1=5000	
	?CNT<n>	Read out current position counter for an axis.	?CNT1	5000
	CRES<n>	Reset current position counter for an axis.	CRES1	
	?POSERR<n>	Read out the current position error for an axis. The difference between encoder position and default position is returned. May be used "on the fly" as well for read out of the contouring error.	?POSERR1	-15

command group	commando	function description	example	response
positioning mode	?VACT<n>	Read out current speed for an axis. This is returned in 16.16 format signed, shifted left by 16 bits. The fractional part is "0" however, because the current speed is calculated on the basis of the position deviation between two samples. The output value will be a integer multiple of 65536, therefore.	?VACT2	1000
	?ENCPOS<n>	Read out current encoder position counter for an axis. Do not use this command with open-loop step-motor axes.	?ENCPOS1	5000
	?MXSTROKE<n>	Read out measured travel. This command reads out the travel ascertained by referencing in mode 6 and 7.	?MXSTROKE1	340000
positioning parameters	RELAT<n>	Set entry mode of coordinates for an axis to "relative" (= indication of the signed travel distance).	RELAT1	
	ABSOL<n>	Set entry mode of coordinates for an axis to "absolute" (= indication of the signed target position).	ABSOL2	
	?MODE<n>	Query the active/current entry mode of coordinates for one axis.	?MODE2	ABSOL
	PMOD<n>=<uv>	Set positioning mode trapezoidal/S-curve for an axis (0 = trapezoidal profiling mode, 1 = S-curve profiling mode).	PMOD1=0	
	?PMOD<n>	Read out positioning mode trapezoidal/S-curve for an axis.	?PMOD1	1
	PVEL<n>=<uv>	Set max. positioning velocity for an axis; used for the trapezoidal and S-curve profile.	PVEL1=10000	
	?PVEL<n>	Read out max. positioning velocity for an axis.	?PVEL1	10000
	FVEL<n>=<uv>	Set limit switch release speed for an axis (unsigned value).	FVEL1=1000	
	?FVEL<n>	Read out limit switch release speed for an axis.	?FVEL1	1000
	ACC<n>=<uv>	Set acceleration (= run-up ramp) for an axis, is used for all modi (trapezoidal, S-curve, velocity mode, etc.).	ACC1=100	
	?ACC<n>	Read out acceleration for an axis.	?ACC1	100
	DACC<n>=<uv>	Set deceleration (= slow-down ramp) for an axis, is used for all modi except S-curve.	DACC2=68	
	?DACC<n>	Read out deceleration for an axis.	?DACC2	68
	JACC<n>=<uv>	Set maximum "jerk" for an axis, is used only with S-curve profile.	JACC3=5	
	?JACC<n>	Read out maximum "jerk" for an axis.	?JACC3	5
	EDACC<n>=<uv>	Set emergency-stop deceleration for an axis. This is used when a brake switch has responded.	EDACC1=1000	
	?EDACC<n>	Read out emergency-stop deceleration for an axis.	?EDACC1	1000
	JVEL<n>=<sv>	Set maximum axis velocity for "joystick travel". With this command, the maximum velocity at maximum joystick deflection is defined.	JVEL3=1000	
	?JVEL<n>	Read out maximum axis velocity for "joystick travel".	?JVEL3	1000
	JOYACC<n>=<uv>	Set axis acceleration and deceleration for "joystick travel".		
	?JOYACC<n>	Read out axis acceleration and deceleration for "joystick travel".	?JOYACC3	100
	JPLAX=<n>	The joystick X axis (first axis of the joystick plane) is assigned to a certain axis number. If "0" is transferred, the X axis of the joystick is disabled.	JPLAX=2	
	?JPLAX	Read out joystick plane X axis assignment.	?PLAX	2
	JPLAY=<n>	The joystick Y axis (second axis of the joystick plane) is assigned to a certain axis number. If "0" is transferred, the Y axis of the joystick is disabled.	JPLAY=3	
	?JPLAY	Read out joystick plane Y axis assignment.	?JPLAY	3
	JPLAZ=<n>	The joystick Z axis (third axis of the joystick plane) is assigned to a certain axis number. If "0" is transferred, the Z axis of the joystick is disabled.	JPLAZ=1	
	?JPLAZ	Read out joystick plane Z axis assignment.	?JPLAZ	1
	LIGO=<uv>	Start positioning with linear interpolation for a group of axis (binary definition mask). Bit-order: <axis 3, axis 2, axis 1>.	LIGO=111	
	IVEL<n>=<uv>	Set maximum velocity <uv> for axis<n> used for linear interpolation.	IVEL1=50000	
	?IVEL<n>	Read out maximum velocity <uv> for axis<n> used for linear interpolation.	?IVEL1	50000

command group	commando	function description	example	response
positioning parameters	IACC<n>=<uv>	Set maximum acceleration <uv> for axis<n> used for linear interpolation.	IACC3=2000	
	?IACC<n>	Read out maximum acceleration <uv> for axis<n> used for linear interpolation.	?IACC3	2000
	POSTAB<uv>=<v>, <v>,...	Load table line in the path table; the value before the "="-sign indicates the table line number (0 to …), the value behind the "=" sign is for the table column follow separated by commas. Parameter list: 1. travel (signed) for axis 1 (-32760 bis 32760) 2. travel (signed) for axis 2 (-32760 bis 32760) 3. travel (signed) for axis 3 (-32760 bis 32760) 4. reserved 5. reserved 6. reserved 7. reserved 8. reserved 9. segment time in ms 10. function code Bit 15 of function code is set: move with a=const. Bit 15 of function code is deleted: move with v=const. 11. error byte 12. enable byte (always as numerical value)	POSTAB0=1000, 2000, 0, 0, 0, 0, 0, 0, 100, 0, 0, 3	
	?POSTAB<uv>	Load a table line out of the path table. The table line number (0 to …) is transferred as parameter. The table values are listed separated by commas. Parameter-list: 1. travel (signed) for axis 1 (-32760 bis 32760) 2. travel (signed) for axis 2 (-32760 bis 32760) 3. travel (signed) for axis 3 (-32760 bis 32760) 4. reserved 5. reserved 6. reserved 7. reserved 8. reserved 9. segment time in ms (16 Bit) 10. function code (16 Bit) 11. error byte (8 Bit) 12. enable byte (8 Bit) 13. velocity (32 bit) calculated by the plausibility check 14. acceleration (32 bit) calculated by the plausibility check	?POSTAB0	1000, 2000, 0, 0, 0, 0, 0, 0, 100, 0, 0, 3, 2100, 50,
	PTABPLAUS<uv>	Execute plausibility check for path table. The limits for velocity and acceleration are checked and the error bytes set, accordingly. Velocity and acceleration are calculated for each single table lines and ist values registered for the active axis with the highes axis number.	PTABPLAUS0	
	PTABGO<uv>	Start path control at a certain table entry.	PTABGO0	
	PTABSTP	Cancel a current path control; the participating axes behave like at the end of the path table.	PTABSTP	
	PTABCLR	Delete table for path control.		
	PTABCIRCLE <uv>=<uv>, ...	Calculates circular interpolation and registers in the positioning table starting at the denoted start line. The parameters are passed as a list, separated by comma. The enable-bits of the axes are disjuncted bit by bit with the possibly already existing entries of the current table. 1. number of axis for X (0 for no X axis) 2. number of axis for Y (0 for no Y axis) 3. segment time in ms (16 bit) 4. function code (OR-template) for the segments (16 bit) 5. number of segment of the circle (16 bit) 6. signed radius (32 bit) 7. start angle in degree with signed value (16 bit) 8. angle range in degree with signed value (16 bit) 9. optionally scaling counter with signed value (16 bit) 10. optionally scaling denominator with signed value (16 bit)	PTABCIRLCE0= 1,2,10000,0,12, 50000,0,360,1,1	
	PTABCPY <uv>=<uv>,<uv>	Copy an area of table for path control. The value before the "=" sign specifies the target index in the positioning table, the value behind the "=" sign specifies the source index and the value behind the comma the amount of lines that are to be copied.	PTABCPY 50=10, 20	
	PTABCLR<uv>=<uv>	Delete an area of the table for path control. The value before the "=" sign indicates the index of lines wherefrom it is to be deleted, the value behind the "=" sign indicates the amount of lines which are to be deleted.	PTABCLR50=20	

command group	commando	function description	example	response
axis parameters	MCSTP<n>=<uv>	Set micro step resolution with the step motor axes.	MCSTP1=50	
	?MCSTP<n>	Read out micro step resolution with the step motor axes.	?MCSTP1	50
	DRICUR<n>=<uv>	Set driving current with step motors in percent of the maximum output value defined by the selected current range of the motor power stage.	DRICUR1=50	
	?DRICUR<n>	Read out driving current with step motors in percent.	?DRICUR1	50
	HOLCUR<n>=<uv>	Set holding current with step motor axes in percent.	HOLCUR1=30	
	?HOLCUR<n>	Read out holding current with step motor axes in percent.	?HOLCUR1	30
	ATOT<n>=<uv>	Set timeout in milliseconds, "0" switches off the timeout monitoring.	ATOT1=20000	
	?ATOT<n>	Query time-out for the axes.	?ATOT1	20000
	FKP<n>=<uv>	Set control parameter KP for an axis.	FKP1=25	
	?FKP<n>	Query control parameter KP for an axis.	?FKP1	25
	FKD<n>=<uv>	Set control parameter KD for an axis.	FKD1=5	
	?FKD<n>	Query control parameter KD for an axis.	?FKD1	5
	FKI<n>=<uv>	Set control parameter KI for an axis.	FKI1=10	
	?FKI<n>	Query control parameter KI for an axis.	?FKI1	10
	FIL<n>=<uv>	Set control parameter integration limit for an axis.	FIL1=100000	
	?FIL<n>	Query control parameter integration limit for an axis.	?FIL1	100000
	FST<n>=<uv>	Set sample time for an axis (in micro seconds).	FST1=500	
	?FST<n>	Query sample time for an axis (in micro seconds).	?FST1	500
	FDT<n>=<uv>	Set delay time of the differential term (KD) for an axis (in sample time cycles).	FDT1=5	
	?FDT<n>	Query delay time of the differential term (KD) for an axis (in sample time cycles).	?FDT1	5
	MXPOSERR<n>=<uv>	Set maximum position error for a servo axis. If the value is exceeded, the axis is deactivated. This works only for the motor types DC brush and step motor Closed-Loop.	MXPOSERR1= 50	
	?MXPOSERR<n>	Read out maximum position error for an axis.	?MXPOSERR1	50
	MAXOUT<n>=<uv>	Set maximum output value for the servo loop in percent. With this command, the maximum value for an axis to be returned at the servo amplifier can be set. **Maximum value allowed: 99%.**	MAXOUT1=95	
	?MAXOUT<n>	Read out maximum output value in percent.	?MAXOUT1	95
	INPOSMOD<n>=<uv>	Set end-of-motion reporting mode: 0 = target position reached, 1 = actual position is within the settling window for a time defined by "INPOSTIM" command.	INPOSMOD1=0	
	?INPOSMOD<n>	Read out end-of-motion reporting mode.	?INPOSMOD1	0
	INPOSTIM<n>=<uv>	Set end-of-motion reporting time in multiples of the cycle time.	INPOSTIM1=1000	
	?INPOSTIM<n>	Read out end-of-motion reporting time.	?INPOSTIM1	1000
	INPOSWND<n>=<uv>	Set end-of-motion settling window in encoder counts.	INPOSWND1=50	
	?INPOSWND<n>	Read out end-of-motion settling window.	?INPOSWND1	50
	AMPPWMF<n>=<uv>	Set PWM output frequency of drive controller board, 20000 or 80000 is possible.	AMPPWMF1= 20000	
	?AMPPWMF<n>	Read out PWM frequency of drive controller board.	?AMPPWMF1	?AMPPWMF1
	ENCLINES<n>=<uv>	Set encoder line number for an axis.	ENCLINES1=500	
	?ENCLINES<n>	Read out encoder line number for an axis.	?ENCLINES1	500
	MOTPOLES<n>=<uv>	Set motor pole number for an axis.	MOTPOLES1=25	
	?MOTPOLES<n>	Read out motor pole number for an axis.	?MOTPOLES1	25
	ELCYCNT<n>=<uv>	Set encoder counts for one electrical commutation cycle.	ELCYCNT1=128	
	?ELCYCNT<n>	Read out encoder counts for one electrical commutation cycle.	?ELCYCNT1	128
	PHINTIM<n>=<uv>	Set phase initialization time in multiples of the cycle time.	PHINTIM1=10	
	?PHINTIM<n>	Read out phase initialization time in multiples of the cycle time.	?PHINTIM1	10
	PHINAMP<n>=<uv>	Set phase initialisation amplitude in %.	PHINAMP1=50	
	?PHINAMP<n>	Read out phase initialisation amplitude in %.	?PHINAMP1	50

command group	commando	function description	example	response
limit switch configuration / reference motion	REF<n>=<uv>	Start reference travel while indicating the reference mode for one axis: mode 0 = search next index impulse and stop mode 1 = approach reference switch and stop mode 2 = approach reference switch, search next index impulse and stop mode 3 = mode 0, additionally set act. position to "0" mode 4 = mode 1, additionally set act. position to "0" mode 5 = mode 2, additionally set act. position to "0" mode 6 = approach maximum reference switch, approach minimum reference switch, set current position to 0 mode 7 = approach minimum reference switch, approach maximum reference switch, set current position to 0	REF2=4	
	RVELS<n>=<sv>	Set reference travel speed "slow" for one axis. Using this speed, the index pulse will be searched or the reference switch will be released, respectively (signed value).	RVELS2=2000	
	?RVELS<n>	Read out reference travel speed "slow" for an axis.	?RVELS2	2000
	RVELF<n>=<sv>	Set reference travel speed "fast" for an axis. The drive moves with this speed towards the limit switch (signed value).	RVELF2=-20000	
	?RVELF<n>	Read out reference travel speed "fast" for an axis.	?RVELF2	-20000
	RDACC<n>=<uv>	Set reference travel deceleration for an axis. This value is used when the reference point is approached.	RDACC1=2000	
	?RDACC<n>	Read out reference travel deceleration for an axis.	?RDACC1	2000
	SMK<n>=<uv>	Set limit switch mask for an axis. This command activates or deactivates the limit and break switches. If a limit switch is approached, the movement is stopped abruptly and the motor is shut off. Bit sequence : <MAXSTOP, MAXDEC, MINDEC, MINSTOP>.	SMK3=1001	
	?SMK<n>	Read out limit switch mask for an axis.	SMK3	1001
	SPL<n>=<uv>	Set limit switch polarity for an axis. With this command, the active level for the limit and brake switches is defined. Bit sequence: <MAXSTOP, MAXDEC, MINDEC, MINSTOP>.	SPL3=1111	
	?SPL<n>	Read out limit switch polarity for an axis.	SPL3	1111
	RMK<n>=<uv>	Set reference switch mask for an axis. With this command, it can be defined which of the 4 limit switches for an axis should be interpreted as reference switch. A mask with exactly one character "1" has to be transferred. Bit sequence: <MAXSTOP, MAXDEC, MINDEC, MINSTOP>.	RMK3=0001	
	?RMK<n>	Read out reference switch mask for one axis.	?RMK3	0001
	RPL<n>=<uv>	Set reference switch polarity for one axis. This command defines the active level of the reference switch. Bit sequence: <MAXSTOP, MAXDEC, MINDEC, MINSTOP>.	RPL3=1111	
	?RPL<n>	Read out reference switch polarity for one axis.	?RPL3	1111
	?HYST<n>	Read out reference switch hysteresis for an axis. After a reference motion has been terminated successfully, the hysteresis of the switch can be read out with this command. (The value is correct only, if none of the reference/limit switches is active any more).	?HYST1	28
	?REFST<n>	Inquiry of reference motion validity. When reference motion successfully completed, the status is set on 1 = "valid". If a motor without encoder is switched off (e.g. step motor Open Loop), then the validity is reset to "0".	?REFST	1
	LMK<n>=<uv>	Set mask for limit-positioning monitoring for the axis. With this command the limit-positioning monitoring for tho lower limit and/or upper limit for position can be set activ and inactive, respectively. The limit-positioning monitoring behaves like the according DEC-switch. Bit sequence: <MAXDEC, MINDEC>.	LMK1=01	
	?LMK<n>	Read out limit-positioning monitoring mask for axis	?LMK1	01
	?LSTAT<n>	Read current, logical state of limit-positioning monitoring for the axis. Bit 0 = MINDEC lower limit is transcanded Bit 1 = MAXDEC upper limit is transcanded	?LSTAT1	01
	SLMIN<n>=<uv>	Set negative limit position for the axis.	SLMIN1=100	
	?SLMIN<n>	Read out negative limit position for the axis.	?SLMIN1	100
	SLMAX<n>=<uv>	Set positve limit position for the axis.	SLMAX1=100000	
	?SLMAX<n>	Read out positive limit position for the axis.	?SLMAX1	100000

command group	command	function description	example	response
in-/outputs	ETTLOUTS<n>=<bin>	Set TTL outputs for the motor power stage of an axis. The axis number and a binary set mask are transferred.	ETTLOUTS1=10	
	ETTLOUTC<n>=<bin>	Reset TTL outputs for the motor power stage of an axis. The axis number and a binary delete mask are transferred.	ETTLOUTC1=01	
	?INPUTS	Read out current state of the inputs (8-bit binary digit).	?INPUTS	10100100
	?INPTTL	Read out current state of all TTL inputs (8-bit binary digit).	?INPTTL	10100100
	OUTPUT<uv>=<uv>	Change current state of an output.	OUTPUT1=0	
	?OUTPUTS	Read out current state of all outputs.	?OUTPUTS	10100
	OUTTTL<uv>=<uv>	Change current state of a TTL output.	OUTTTL1=0	
	?OUTTTL	Read out current state of all TTL outputs.	?OUTTTL	10100
	?ANIN<uv>	Query analog input, the port number from 1 to 8 will be set, and the converted 10-bit value will be returned.	?ANIN3	234
	OPWM<uv>=<uv>	Set PWM output, the port number from 1 to 8 and the level control value are set from 0 to 100 %.	OPWM1=55	
	?OPWM<uv>	Query PWM output, the port number from 1 to 4 is entered and the level control value that has been set is returned from 0 to 100 %.	?OPWM1	55
	AXOUTPUT<n>=<uv>	Set axis out-pin for an axis to high/low.		
follow-up control	?APWMS<n>	Read out current position of linear measuring system.	APWMS4	3000
	WMSRES<n>	Set current position of measuring system of an axis to 0 (is not required and must not be used after reference scan, as the position will be lost).	WMSRES4	
	?MXWMSSTRK<n>	Query maximum travel of linear measuring system.	?MXWMSSTRK2	100000
	WMSFAKZ<n>=<uv>	Set numerator factor for conversion of the resolution of the WMS to the resolution of the actuator for positioning with follow up control.	WMSFAKZ1=1	
	?WMSFAKZ<n>	Query numerator factor for positioning with follow up control.	?WMSFAKZ1	1
	WMSFAKN<n>=<uv>	Set denominator factor for conversion of the resolution of the WMS to the resolution of the actuator for positioning with follow up control.	WMSFAKN1=5	
	?WMSFAKN<n>	Query denominator factor for positioning with follow up control.	?WMSFAKN1	5
	PWMSSET<n>=<sv>	Set target position and relative travel, respectively for an axis (preselection is, by analogy to the normal positioning without follow up control, via the commands ABSOL and RELAT, respectively); if absolute position indication is selected, the parameter is interpreted as absolute position with sign. If relative position indication is chosen, the parameter is interpreted as travel with sign. The new absolute target position is now calculated by the sum of the last target position and travel.	PWMSSET2= 100000	
	?PWMSSET<n>	Read out target position and relative travel, respectively, for an axis.	?PWMSSET2	100000
	PWMSGO<n>	Start positioning with WMS at an axis. The axis moves to the new target position either in trapezidol- or S-shaped profile (see PMOD).	PWMSGO2	
	PWMSWIN<n>=<uv>	Set half target window width for the positioning with WMS (whole width of target position = ±PWMSWIN).	PWMSWIN1=10	
	?PWMSWIN<n>	Query half target window width for positioning with WMS.	?PWMSWIN1	10
	PWMSMODE<n>=<uv>	Set positioning mode for positioning with WMS. Mode 0: coarse positioning (phase 1). Mode 1: coarse positioning (phase 1), iteration (phase 2). Mode 2: coarse positioning (phase 1), iteration (phase 2), position correction (phase 3), phase 3 is active. Mode 3: coarse positioning (phase 1), position correction (phase 3), phase 3 is active. Mode 4: coarse positioning (phase 1), iteration (phase 2), position correction (phase 3), phase 3 is finished in target window. Mode 5: coarse positioning (phase 1), position correction (phase 3), phase 3 is finished in target window.	PWMSMODE1=1	
	?PWMSMODE<n>	Query positioning mode for positioning with WMS.	?PWMSMODE1	
	WMSVEL<n>=<uv>	Set follow up velocity for positioning with WMS (without sign).	WMSVEL1=100	
	?WMSVEL<n>	Query follow up velocity for positioning with WMS.	?WMSVEL1	

command group	commando	function description	example	response
follow-up control	PWMSSTP<n>	Stop positioning with WMS at an axis. If the axis at position with WMS in phase 3, the mode has to be stopped by this command before moving the axis with a new command.	PWMSSTP1	
	?PWMSSTATE<n>	Read out state of positioning with WMS of a given axis. Bit 0: axis positions with WMS Bit 1: axis positions with WMS and is in phase 1 Bit 2: axis positions with WMS and is in phase 2 Bit 3: axis positions with WMS and is in phase 3 Bit 4: axis reached the default target window	?PWMSSTATE1	
	?PWMSERR<n>	Read out current position error of an axis when positioning with WMS.	?PWMSERR1	
	WMSINV<n>=<uv>	Reverse count direction of WMS (1=yes/0=no).	WMSINV1=0	
	?WMSINV<n>	Read out whether count direction of WMS is reversed (yes/no).	?WMSINV1	
holding brake control	HBCH<n>=<uv>	Assign PWM output for holding brake to an axis: <AxisNumber> = <PWM port> PWM port = 0 for holding break function off.	HBCH1=1	
	?HBCH<n>	Query holding brake assignment PWM port to an axis.	?HBCH1	1
	HBFV<n>=<uv>	Set first PWM value to activate the holding brake: <AxisNumber> = <Percent Value>.	HBFV1=50	
	?HBFV<n>	Query first PWM value for activation of the holding brake.	?HBFV1	50
	HBSV<n>=<uv>	Set second PWM value for clamping the holding brake: <AxisNumber> = <Percent Value>.	HBSV1=20	
	?HBSV<n>	Query second PWM value for clamping the holding brake.	?HBSV1	20
	HBTI<n>=<uv>	Set settling time for the holding brake. The first PWM value will be set for this amount of time after activation of the holding brake: <AxisNumber> = <Time for first PWM value in ms>.	HBTI1=300	
	?HBTI<n>	Query settling time for the holding brake. The first PWM value will be set for this amount of time after activation of the holding brake.	?HBTI1	300
reset	RESETAC	Activate output stage module Reset.	RESETAC	
	RESETMB	Activate main board Reset.	RESETMB	
stand-alone programming	SAMEM	Set flag value for stand-alone programming.	SAMEM38=50	
	?SAMEM	Query flag value for stand-alone programming.	?SAMEM38	50
	SAEXEC	Start (1) / Stop (0) stand-alone program execution.	SAEXEC0	
	SASTEP	Run a stand-alone program line, the line number is sent and the line number of the next line is returned.	SASTEP1	2
	SALOAD	Download a stand-alone program line, the corresponding line number and the contents of the program line as Hex-Dump (16 byte) in ASCII format are sent.	SALOAD11= 04000015F900...	
	SACHKS	Update checksum of the stand-alone program, after a new stand-alone program has been loaded.		

II Parameter Relevance for different Motor Types

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

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

\+ neccessary
-- not neccessary
(+) optional

1) The command may be used, however, it is important that the value set here is larger than or equal to the maximum PWM value for DRICUR or HOLCUR. In any case, the output is limited to the value defined by MAXOUT. If a too small value is selected, the micro-step operation will not work properly.

III Connecting Tables

In-/Outputs

Pin assignment of the 25-pin D-Sub male connector on the PCI plug-in card

function	pin
input 1 (TTL/analog)	6
input 2 (TTL/analog)	5
input 3 (TTL/analog)	4
input 4 (TTL/analog)	3
input 5 (TTL/analog)	10
input 6 (TTL/analog)	9
input 7 (TTL/analog)	8
input 8 (TTL/analog)	7
TTL output 1	16
TTL output 2	15
TTL output 3	14
TTL output 4	13
TTL output 5	17
+ 5V, max. 300 mA total current	1, 2
PWM output 1, max. 1A *)	20
PWM output 2, max. 1A *)	21
+ 12V/+24V, max. 1A total current **)	18, 19
GND	11, 12, 24, 25
enable input + (5V) ***)	22
enable input - (GND)	23

*) switching towards GND
**)+ 12 V or + 24 V according to the power supply type (internal over PC power supply or external
***) enable motor output stage over optoelectronic coupler (U_B = 5 V) necessary; e.g. jumper pin 2 –> pin 22 and pin 23 –> pin 24

RS-232

Pin assignment of the 10-pin IDC male connector.

RS-232	Pin
n.c.	1
DSR	2
RXD	3
RTS	4
TXD	5
CTS	6
DTR	7
n.c.	8
GND	9
GND	10

Triple Connector on the PCI Plug-in Card

Pin assignment of the 62-pin HD female connector.

pin	DC motor	step motor
1	motor1 +	motor1 phase 1+
2	motor1 +	motor1 phase 1+
3	motor1 -	motor1 phase 1 -
4	motor1 -	motor1 phase 1 -
5	motor1 +	motor1 phase 2+
6	motor1 +	motor1 phase 2+
7	motor1 -	motor1 phase 2 -
8	motor1 -	motor1 phase 2 -
9	motor type encoding	
10	motor1 MAXSTOP	
11	motor1 MINSTOP	
12	encoder1 A	
13	encoder1 $\overline{A}$	
14	encoder1 B	
15	encoder1 $\overline{B}$	
16	encoder1 Index	
17	encoder1 $\overline{Index}$	
18	+ 5V	
19	GND	
20	OWISid	
21	+24V supply for PWM outputs (optional)	
22	motor3 +	motor3 phase 1+
23	motor3 +	motor3 phase 1+
24	motor3 -	motor3 phase 1 -
25	motor3 -	motor3 phase 1 -
26	motor3 +	motor3 phase 2+
27	motor3 +	motor3 phase 2+
28	motor3 -	motor3 phase 2 -
29	motor3 -	motor3 phase 2 -
30	motor type encoding	
31	motor3 MAXSTOP	
32	motor3 MINSTOP	
33	encoder3 A	
34	encoder3 $\overline{A}$	
35	encoder3 B	
36	encoder3 $\overline{B}$	
37	encoder3 Index	
38	encoder3 $\overline{Index}$	
39	+ 5V	
40	GND	
41	OWISid	
42	GND supply for PWM outpus (optional)	
43	motor2 +	motor2 phase 1+
44	motor2 +	motor2 phase 1+
45	motor2 -	motor2 phase 1 -
46	motor2 -	motor2 phase 1 -
47	motor2 +	motor2 phase 2+
48	motor2 +	motor2 phase 2+
49	motor2 -	motor2 phase 2 -
50	motor2 -	motor2 phase 2 -
51	motor type encoding	
52	motor2 MAXSTOP	
53	motor2 MINSTOP	
54	encoder2 A	
55	encoder2 $\overline{A}$	
56	encoder2 B	
57	encoder2 $\overline{B}$	
58	encoder2 Index	
59	encoder2 $\overline{Index}$	
60	+ 5V	
61	GND	
62	OWISid	

3-Way Motor Adapter Cable

Pin assignment of the 3-fold motor adapter cable which is also part of delivery (3x D-Sub 37-pin female connector).

Pin-No. HD 62-polig male	Pin-No. D-Sub 37-polig female Achse X (1)	Pin-No. D-Sub 37-polig female Achse Y (2)	Pin-No. D-Sub 37-polig female Achse Z (3)
1	19		
2			
3	18		
4			
5	17		
6			
7	16		
8			
9	14		
10	2		
11	5		
12	11		
13	10		
14	9		
15	8		
16	7		
17	6		
18	12		
19	13+15		
20	29		

22			19
23			
24			18
25			
26			17
27			
28			16
29			
30			14
31			2
32			5
33			11
34			10
35			9
36			8
37			7
38			6
39			12
40			13+15
41			29

Pin-No. HD 62-polig male	Pin-No. D-Sub 37-polig female Achse X (1)	Pin-No. D-Sub 37-polig female Achse Y (2)	Pin-No. D-Sub 37-polig female Achse Z (3)
43		19	
44			
45		18	
46			
47		17	
48			
49		16	
50			
51		14	
52		2	
53		5	
54		11	
55		10	
56		9	
57		8	
58		7	
59		6	
60		12	
61		13+15	
62		29	

Motor Connector of 3-Way Motor Adapter

The signals of the 62-pin motor connector which are converted by a 3-way adapter are presented in the following table. The pin assignment matches the OWIS® standard.

Pin assignment of the 37-pin D-Sub female connector:

	pin	DC motor	step motor
performance	19	motor +	phase 1 +
	18	motor -	phase 1 -
	17	motor +	phase 2 +
	16	motor -	phase 2 -
	15	motor type encoding	
	14	motor type encoding	

signals	13	GND
	12	+ 5 V
	11	Quad A
	10	Quad $\overline{A}$
	9	Quad B
	8	Quad $\overline{B}$
	7	Index
	6	$\overline{Index}$

switches + signals	5	MINSTOP
	4	(reserved)
	3	(reserved)
	2	MAXSTOP
	1	(reserved)
	37	(reserved)
	36	(reserved)
	35	(reserved)
	34	(reserved)
	33	(reserved)
	32	(reserved)
	31	(reserved)
	30	(reserved)
	29	OWISid
	28	(reserved)
	27	(reserved)
	26	(reserved)
	25	(reserved)
	24	(reserved)
	23	(reserved)
	22	(reserved)
	21	(reserved)
	20	(reserved)

Connecting Cable

1. Signal cable "Twisted Pair" 8 x 2 x 0.15 mm² with overall shielding + star quad core, with additional shield, 4 x 0.25 mm²

pair no.	color wire 1	color wire 2	cross sectional area
1	red	blue	0.15 mm²
2	white	brown	0.15 mm²
3	green	yellow	0.15 mm²
4	grey	pink	0.15 mm²
5	black	violet	0.15 mm²
6	grey/pink	red/blue	0.15 mm²
7	orange	orange/black	0.15 mm²
8	transparent	transparent/red	0.15 mm²
9 a	green/white	green/brown	0.25 mm²
9 b	yellow/white	yellow/brown	0.25 mm²

2. motor cable with overall shielding

core no.	color	cross sectional area
1	red	0.6 mm²
2	blue	0.6 mm²
3	white	0.6 mm²
4	black	0.6 mm²
5	brown	0.6 mm²
6	pink	0.5 mm²
7	grey	0.5 mm²

EU/UE Konformitätserklärung/Declaration of conformity

Wir
We

OWIS GmbH
Im Gaisgraben 7
79219 Staufen / Germany
+49(0)7633/9504-0
+49(0)7633/9504-440
www.owis.eu
info@owis.eu

erklären in alleiniger Verantwortung, dass das Produkt
declare under our sole responsibility that the product

PS 30

auf das sich diese Erklärung bezieht, mit den folgenden Normen oder normativen Dokumenten übereinstimmt.
to which this declaration relates is in conformity with the following standards or other normative documents.

EN 61000-6-1:2007 mit/with EN 61000-4-2:2009, EN 61000-4-3:2011
EN 61000-6-3:2011 mit/with EN 55022:2011

Gemäss den Bestimmungen der Richtlinie:
Following the provisions of directive:

2014/30/EU

Ort und Datum der Ausstellung
Place and date of issue

Staufen, 27.09.2017

Name und Unterschrift
Name and signature

D. J. Schuhen
Leitung Vertrieb

i.A. Dr. Peter Hilgers
Leitung Entwicklung

Aktuelle Ausgabe: 27.09.17 DB / DSCH, 2.01.112 FO Konformitätserklärung