**Stögra Antriebstechnik GmbH**

Machtlfinger Straße 24
D-81379 München

Tel.: (089)15904000
Fax.: (089)15904009

# SERS 02, SERS 06 and SERS 12

# Version V02

Stepping motor power amplifier board
with position control and RS 232C/RS485 interface

# Installation and programming manual

Edition April 2003

# Index

page

1. General notes

1.1 Safety rules..... 4

1.2 General notices SERS 01/06/12..... 4

1.2 Installation..... 6

2. Power amplifier

2.1 with 32 pole connector

2.1.1 Connections..... 7

2.1.2 Limit-/ home-/ stop-/ and external service switch...……..... 8

2.1.3 Ready Signal - relay to control a brake..... 9

2.1.4 Power supply..... 10

2.1.5 Dimensions SERS 02/06/12..... 11

2.2 Optionally panel mount system ELK

2.2.1 Dimensions ELK..... 12

2.2.2 Connections ELK - screw terminal..... 12

2.3 Panel mount system series ELK…S...…….... 13

2.4 19-inch rack series ELR ....…….... 13

3 Positioning control

3.1 Configurations via DIP-switches..... 14

3.1.1 Drive number..... 14

3.1.2 Automatic program start..... 14

3.1.3 Baud rate..... 15

3.1.4 Enable manual drive functions/ service switch.....……..... 15

3.1.5 Signal level and polarity of inputs..... 15

3.2 Interface RS232C/V24 and RS485

3.2.1 Configurations..... 16

3.2.2 9-pole D-Sub-connector..... 16

3.2.3 Connection PC-SERS and SERS-SERS..... 16

3.3 I/O-Port with digital inputs and outputs

3.3.1 Standard I/O-port.……..... 17

3.3.2 Optional I/O-port extension....……. 18

3.3.3 I/O-port SERS…R1 and SERS…R2....……... 19

3.4 Optional encoder input……... 20

page

3.5 Status-/error indication - 7-Segment indication…. 21
3.6 Operation modes - Overview
3.6.1 Serial mode..... 22
3.6.2 Master mode..... 22
3.6.3 Parallel mode..... 22
3.6.4 Standalone mode..... 22
4. Programming and operation of the SERS
4.1 Syntax general notes..... 23
4.2 Syntax definitions..... 24
4.3 Program lines numbering..... 28
4.4 Limitations for $E^2$PROM-programs..... 28
4.5 SERS functions and modes
4.5.1 Manual operation - manual drive..... 29
4.5.2 Operation in master mode, parallel mode and standalone mode..... 29
4.5.3 Programming mode..... 32
4.5.4 Trace mode..... 32
4.6 Programming commands
4.6.1 IF : conditional execution..... 32
4.6.2 Labels..... 33
4.6.3 GOTO, GT : jump..... 34
4.6.4 GOSUB : subroutine call..... 34
4.6.5 RETURN : subroutine termination..... 34
4.6.6 Programming of positioning jobs..... 35
4.6.7 Positioning with velocity profiles - polynom function..... 36
4.6.8 WAIT instruction..... 37
4.6.9 " : " instruction - manual driving until STOP.. 37
4.6.10 Arithmetic / Calculating in the SERS..... 38
4.6.11 Operation with the SERS-Programmer..... 38
4.6.12 Program examples..... 39
4.7 Parameter overview..... 41
4.8 SERS parameters..... 43
5. Technical specifications..... 71

# 1. General notices
# 1.1 Safety rules

The stepping motor drives SERS 01, SERS 06 and SERS 12 (from now on named **SERS**) are for the installation into other devices or machines or for mounting together with other devices or machines.

The operator of the SERS has to ensure that all installation-, maintenance-, montage- and inspection works at the SERS are carried out **only from qualified and authorised professionals**, who informed themselves by a detailed study of this manual.

The operator is responsible, that all installation-, montage-, maintenance - and inspection works are carried out correctly.
The manufacturer rejects any liability for mistakes or damages because of not correct installation or not proper handling.

The SERS may not be commissioned, as long as it is not built into a device, which corresponds to the specifications of the norm VDE 0100 part 410 – protection against dangerous currents through human bodies – and VDE 0100 part 420 – protection against thermal influences.

At any installation-, maintenance-, montage- and inspection works the SERS first has to be separated **from all electric circuits**. There may not be any electricity at the device **(to be checked !!!**) !

# 1.2 General notes SERS

The SERS is a **stepping motor power amplifier board** with **integrated control of position** and **RS232C (COM – V24) or optionally with RS485, Profibus-DP or CANopen – Interface** (SERS must be ordered with one of the interfaces).

The unit controls **2-phase stepping motors** with phase currents from 0 to 2,8 ampere / phase (SERS 02) or 0 to 8.4 ampere / phase (SERS 06) or 0 to 14,5 ampere 7 phase (SERS 12).

SERS is designed in a modular way with three levels (realised with three PCB´s connected via PCB-connectors).
The lowest level is the power amplifier with a 32 pole male connector according to DIN 41612 (type D). Here the phase currents are created within two H-bridges.
The second level controls the phase currents (control of the microstepping and commutation of the phase currents).
The upper level consists of the position control and an interface (see options above).

The **power amplifier** of the SERS features :

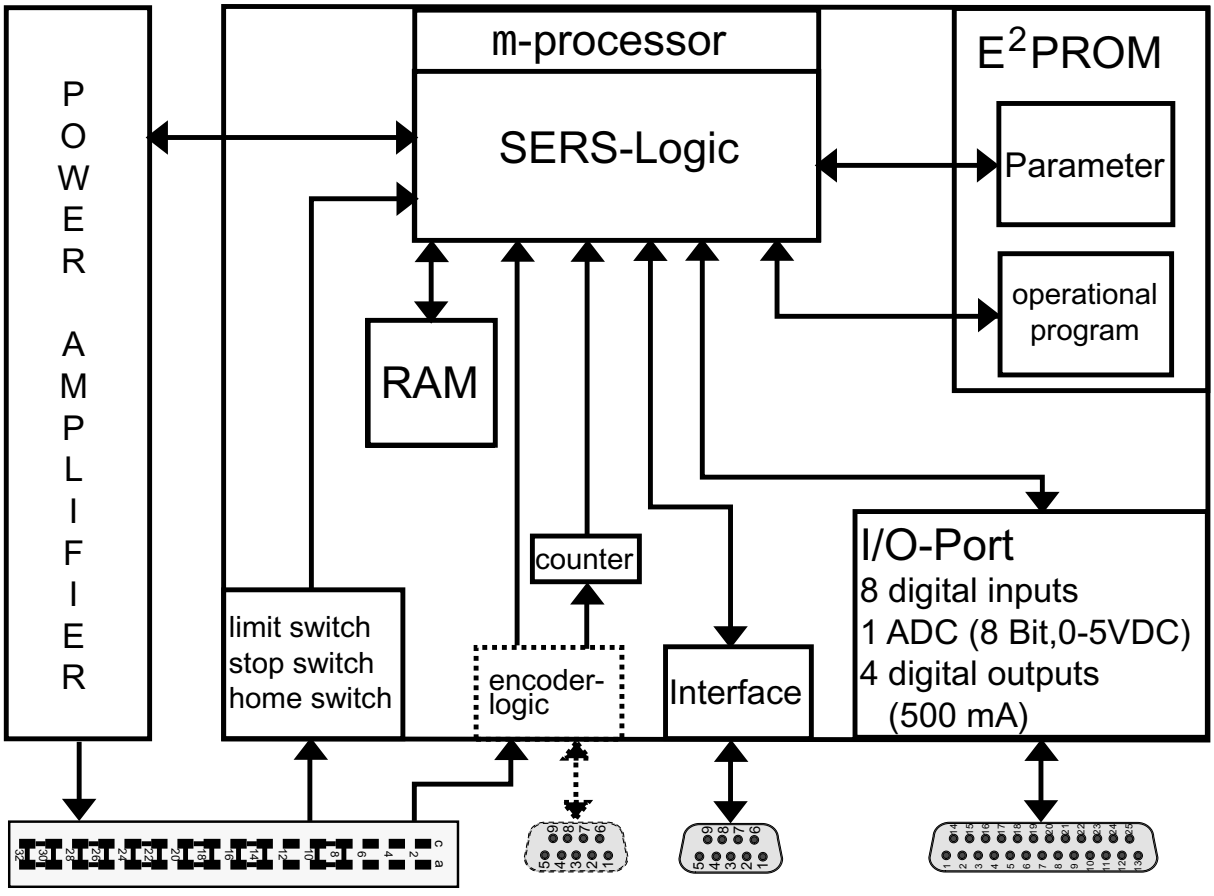
- control of 2-phase stepping motors by using the bipolar chopper-principle.
- microstepping with a resolution of 12800 steps per revolution
- protection against short circuit (in the motor : phase to phase and phase to GND-earth)
- protection against over temperature and under voltage
- phase currents from 0 to 2,8 ampere and a power supply of 24 VDC (SERS 02 – optionally with 1,4 ampere and 85 VDC) , 0 to 8,4 ampere and 85 VDC (SERS 06 – optionally with 24VDC or 120 VDC) or 0 to 14,5 ampere and 120 VDC (SERS 12 - optionally with 240 VDC)

The **control of motor velocity and position** features :
- acceleration : 2 rad/s$^2$ to 15600 rad/s$^2$
- velocity : 0,12 rev/min to 12000 rev/min (motor – depending on size – can be operated until 4000 rev/min)
- position : - $2^{31}$ increments to + $2^{31}$ increments
- control of limit switches and homing

Die **RS232/RS485 - interface** features:
- selectable baud rates from 1200 to 38400 Baud.
- 8 data bits / no parity / 1 stop bit
- handshake : V24 - RTS/CTS – hardware handshake (can be disabled via software)

The **functional principle of the position control** is as follows:
- internal operation-cycle 2ms – that means independent of an existing operational program every 2ms all external inputs are controled, feedback signals of the power stage are controled, various parameters are checked, received characters at the serial input buffer are evaluated, … – similar to working principle of a PLC

- an operational program in the E$^2$Prom is executed line by line. (functional principle as a CNC-control). The control stays at every command in the program lines until the command is finished. Execept the commands ´start positioning´ (E), 'start homing' and ´delay´ (D) all commands are being executed within one internal operation cycle (2ms).




SERS shown with function blocks

# 1.3 Installation

**Before doing any installation-, maintenance-, mounting- or inspection work pay attention to the safety rules (this manual chapter 1.1) !!!**

**All relevant standards of the low voltage-, EMC- and machine directives must be fulfilled before commissioning the SERS !!!**

## Leads to the stepping motor:

**Shielding:**
The leads of the SERS to the stepping motor should be shielded, for preventing noise radiation.
A shield made of metal braiding achieves better results, than a shield made of metal foil.
The shield should be mounted on a large surface at both sides (motor and case or cabinet in where the SERS is installed).

**Leads cross section:**
The following table shows reference values for choosing the right cross sections.
The values in the table refer to a standard leads type, e.g. flexible PVC control leads with copper-shield braiding.
By using different types of leads, we reference to the standard VDE 0298 part 4.

| motor current [A] | cross section [$mm^2$] |
|---|---|
| until 4 | 0,75 |
| until 6 | 0,75 to 1,0 |
| until 10 | 1,0 to 1,5 |
| until 16 | 1,5 to 2,5 |

## Ground conductor system:

The ground conductor system must be installed correctly (VDE 0113).
Here fore it is important e.g.:

- The ground clamp inside the motor must be connected.
- The GND-connection of the power supply for the SERS must be connected to ground direct at the power supply (in case of long leads between the power supply and the supplied device – SERS).

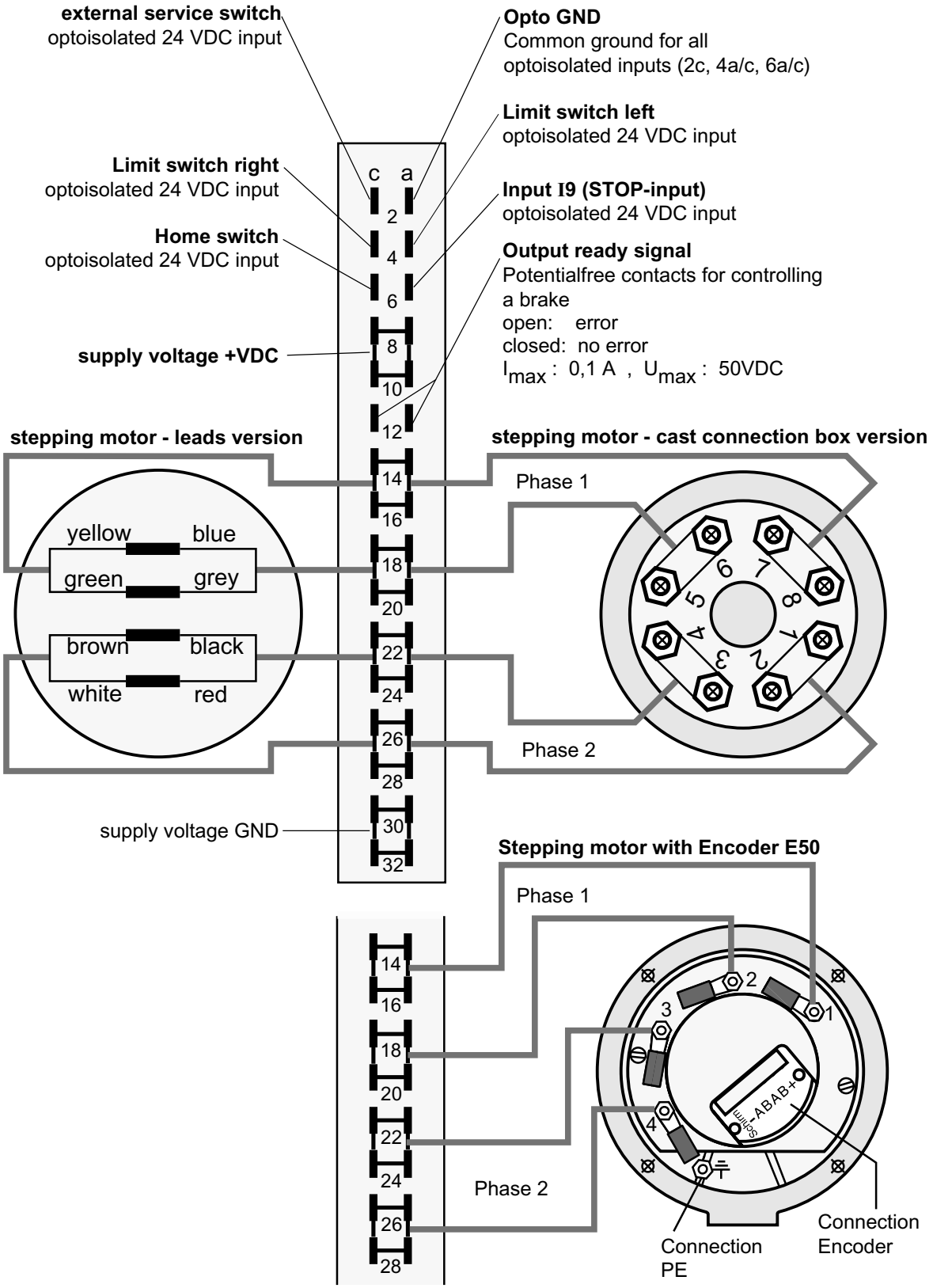
## Ground conductor (PE) connection at ELK – panel mount systems:

Connect the connectors ´**PE**´, ´**GND (of 24 VDC)**´ and ´**GND (of VCC)**´ – three connectors direct beside each other , located direct beside the mains connection (PE, L1, N) – to your ground conductor (PE) bar inside your switch cabinet.
If there is no switch cabinet respective no PE bar, then connect the 3 connectors ´**PE**´, ´**GND (of 24 VDC)**´ and ´**GND (of VCC)**´ to each other with short 1,5$mm^2$ leads.

## Protection against touching:

The SERS must be installed in a way, that there is no danger (electrical shock) when being touched (VDE 0113).

# 2. Power amplifier
# 2.1 with 32-pole male connector
# 2.1.1 Connections



# 2.1.2 Limit-/home-/reset-/stop-/ and external service switch

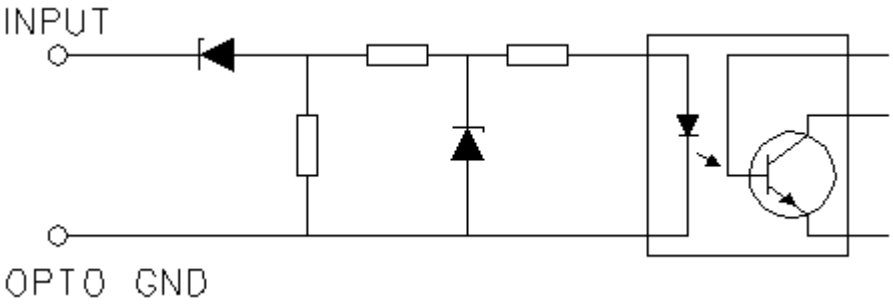
The two limit switch-inputs, the home switch-input, the reset-input and the stop-input are optoisolated 24 VDC inputs.

The two **limit switch-inputs** and **the stop-input must** be connected for the operation of the SERS ! Meaning that for the operation there must be 24 VDC at both limit switch inputs, in case the motor is not on a limit switch position or in case there is there is no stop. Otherwise the unit returns the message 'position overflow' and the 7-segment display indicates a '**C**' (open limit-switch) or it returns the message 'stop input open'.

The **home switch-input** is used for the drive controlled homing procedure. A 24 VDC voltage level at the home switch-input means, that the home position is reached (the motor is on the home position).

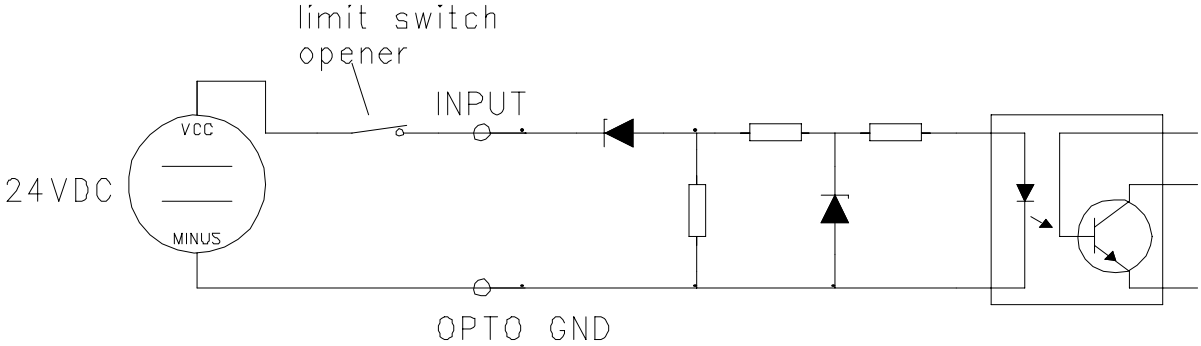
The **input external service switch** is for enabling the manual drive functions at the digital inputs I 1 to I 8 (the parameter P1092 must be set "1" additionally). A 24 VDC level at the external service switch input means, that the external services witch is active and the manual drive functions are enabled.

The following electrical schematics shows the input circuits at the SERS.




## Voltage range inputs: 13 VDC – 30 VDC !

With a connected limit switch the electrical schematics is as follows (motor not on limit switch position) :




**Both limit switches and the stop switch must be break contacts (opener) !!!**
**The home switch and the reset switch must be make contacts (closer) !!!**

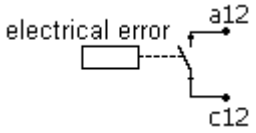
# 2.1.3 Ready signal - relay to control a brake

The output „ready signal“ is e.g. for controlling an emergency brake at the motor (especially for Z-axis).
The signal indicates that the motor phase current is switched on.
**Attention !** The output can not control the brake directly. There must be used an additional power relay! (Alternatively the brake can be controlled directly from one of the SERS outputs O1 – O4 – see parameter 1036)

Stepping motors with integrated brake usually have permanent magnet brakes, which will be activated in case of loss of power.

Between the pins 12a and 12c of the 32-pole male connector there is a **relay**, which will be **closed**, when the **motor current** is switched **on** (command “ON” or in manual mode via the inputs I1 until I6).

In case of switching **off** the **motor current** by a command (“OFF”), or in case of an **error** (electrical error, mechanical error - encoder feedback error / only with installed option encoder - , or other errors like a limit switch error - see parameter P11) - which will switch off the motor current automatically, the **relay** will **open**.




**Maximum load** of the relay : 50 VDC – 100mA

**<u>Attention :</u> When connecting the outputs 12/ac with an external circuit the maximum load of the relay has to be followed (see above).**
**Especially when connecting an inductive load (e.g. small brake), because of the working inductance of the load there must be installed a recovery diode!!!**

**In non disturbed condition** (ready signal active – motor phase current is ON) the r**elay contact is closed**.

Following conditions cause a disabled ready signal and an **opened relay :**
1. a drive error (see parameter P11) is active
2. the SERS did not receive the command ON or the last received command is OFF

In the case of any error - except short circuit - the drive will be stopped controlled, meaning that the motor will be decelerated until motor stand still. Then the relay will be opened. After wards the phase current will be switched off.

In case of a short circuit the phase current will be switched off immediately and at the same time the relay will be opened.

# 2.1.4 Power supply

The SERS requires a DC-voltage supply.
The unit includes an electrolytic capacitor, which is calculated for buffering a **non controlled DC-voltage** with a **Ripple of maximum 5%**.

In case of **power supply leads > 0,5 m** (distance between power supply and SERS), at the input of SERS (Pins 8-10 ac and Pins 30-32 ac - see connections 2.1.1) there has to be installed an **additional electrolytic-capacitor with at least 1000µF**.

Following voltage values are defined :
1. $U_V$ = Maximum voltage supply = nominal voltage supply
2. $U_W$ = Voltage level for the indication of 'pre-warning undervoltage'
3. $U_B$ = Voltage level for the indication of 'error undervoltage'
4. $U_L$ = Voltage level for switching off the power amplifier internally

| | $U_V$ [VDC] | $U_W$ [VDC] | $U_B$ [VDC] | $U_L$ [VDC] |
|---|---|---|---|---|
| SERS XX.24 | 24 | 19 | 18 | 16 |
| SERS XX.85 | 85 | 48 | 44 | 32 |
| SERS XX.120 | 120 | 58 | 50 | 36 |

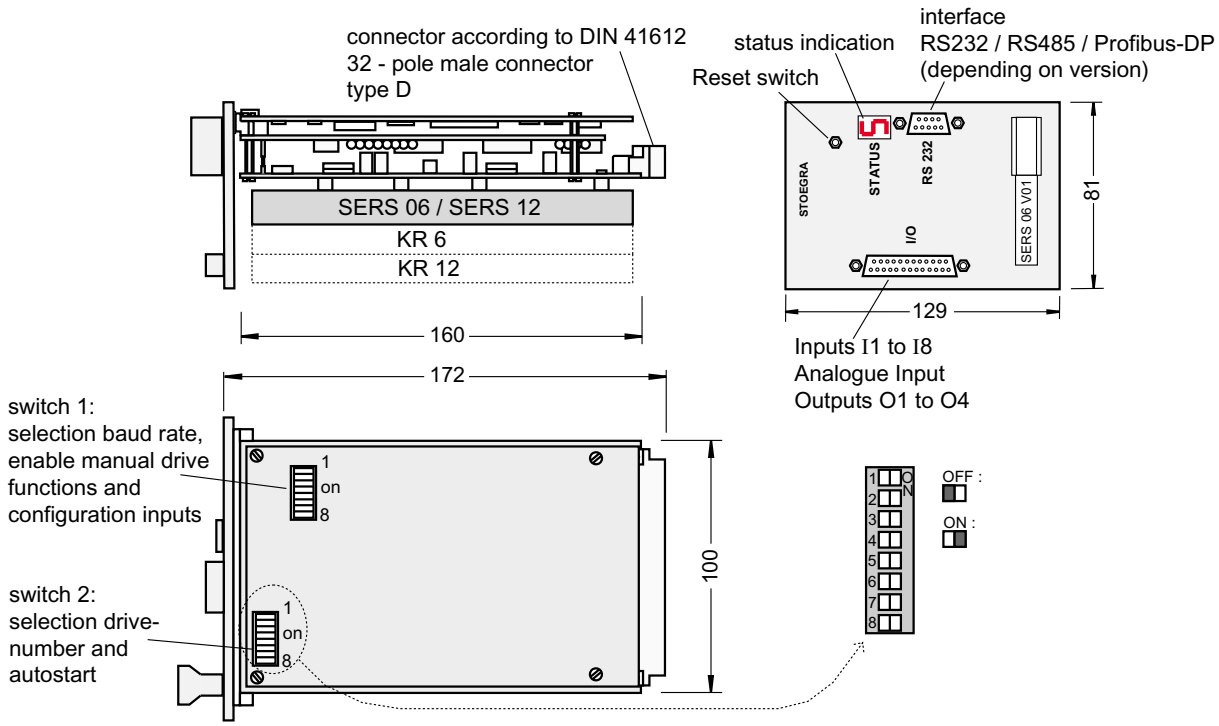
**$U_V$ :**
The maximum voltage supply **$U_V$** is calculated for a maximum mains tolerance of + 15%. That means, that the non controlled DC-voltage output of the power supply, which is used for the **voltage-supply** for the SERS, **may have nominally maximum 85 VDC and not 85 VDC + 15% !**

**$U_W$**
If the supplied voltage becomes lower than **$U_W$**, then a message 'pre-warning undervoltage' will be sent, Bit 15 of P12 (warning parameter) will be set and a blinking '9' will be indicated at the 7-segment display.

**$U_B$**
If the supplied voltage becomes lower than **$U_B$** then the drive will be stopped, meaning that the stepping motor will be decelerated until stand still.
Then the brake signal will be activated (output-relay for the control of the brake will be opened) and 50ms afterwards the stepping motor phases will be switched off (switching off the motor torque).
A message 'error undervoltage' will be sent, Bit 9 of P11 (error parameter) will be set and the 7-Segment display will indicate '9'.

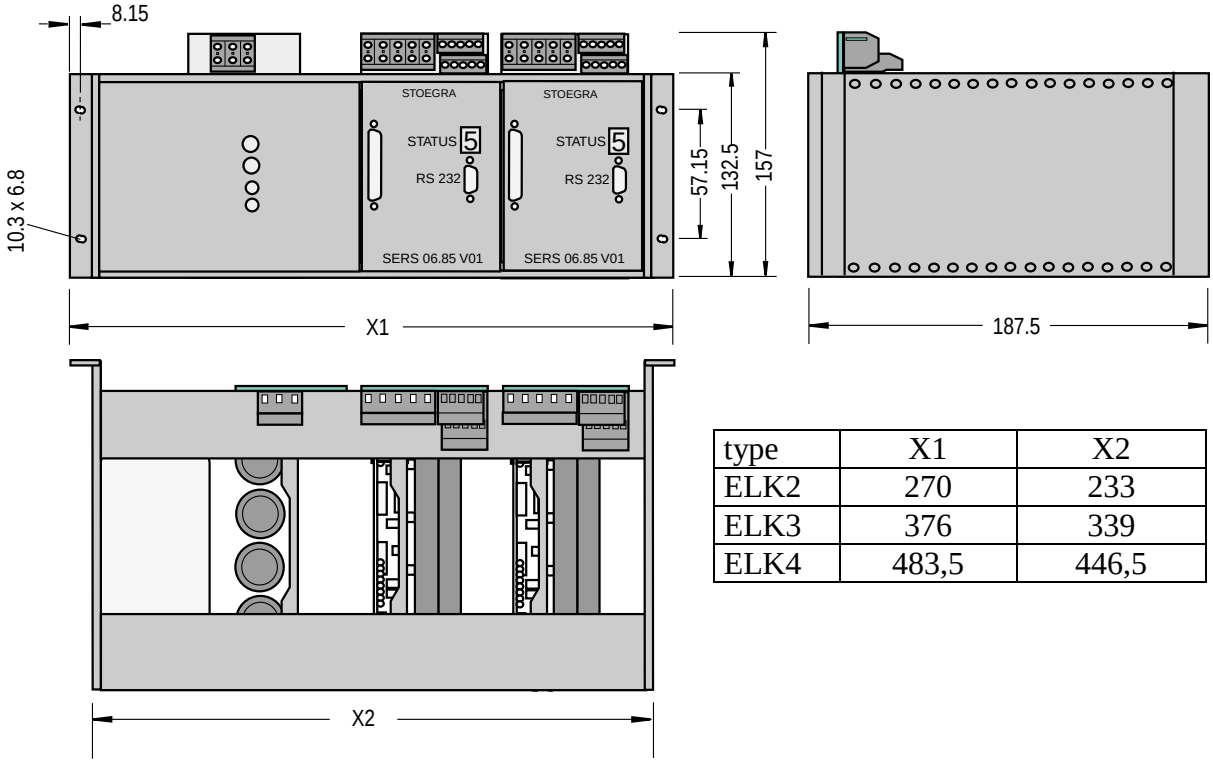
**$U_L$**
If the supplied voltage becomes lower than **$U_L$** then the motor torque will be switched off immediately without decelerating the motor for protecting the electronics of the power amplifier.
The 7-Segment display is as described in **$U_B$**.

# 2.1.5 Dimensions SERS



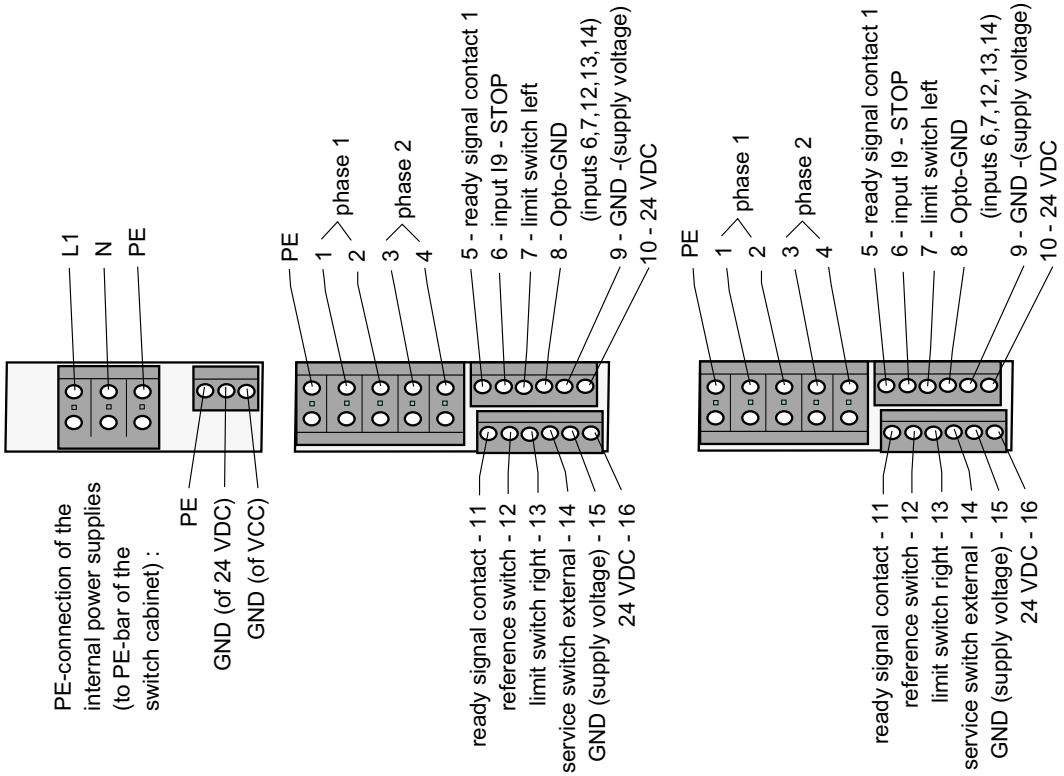
# 2.2 Optionally as panel mount system ELK
# 2.2.1 Dimensions ELK

Following schematics show a ELK-panel mount unit type ELK3 with integrated power supply and 2 stepping motor drivers.
The dimensions depend on the ELK-type – see table below with ELK2, ELK3 and ELK4.




| type | X1 | X2 |
|---|---|---|
| ELK2 | 270 | 233 |
| ELK3 | 376 | 339 |
| ELK4 | 483,5 | 446,5 |

# 2.2.2 Connections ELK

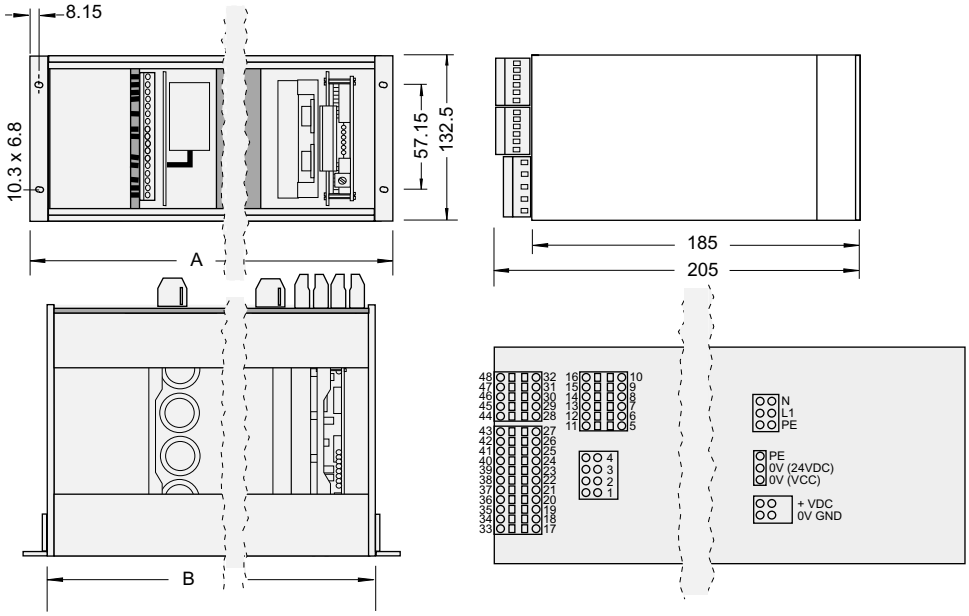


# 2.3 Panel mount system series ELK..S

1 : PE / shield
2 : motor phase 1
3 : motor phase 2
4 : ready signal
5 : STOP
6 : limit switch left
7 : Opto-GND
8 : GND (24VDC)
9 : +24VDC

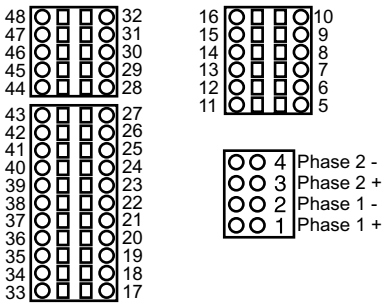
11 : PE / shield
12 : motor phase 1
13 : motor phase 2
14 : ready signal
15 : home switch
16 : limit switch right
17 : Service switch ext.
18 : GND (24VDC)
19 : +24VDC

# 2.4 19-inch rack series ELR




17 : input I2
18 : input I3
19 : input I4
20 : input I5
21 : input I6
22 : input I7
23 : input I8
24 : output O1
25 : output O2
26 : output O3
27 : output O4
28 : RS232 / HSO
29 : RS232 / 8VDC
30 : RS232 / GND
31 : RS232 / RXD
32 : RS232 / TXD

33 : input I1
34 : input ADC pos
35 : input ADC neg
36 : GND (24VDC)
37 : GND IN/ input
38 : not connected
39 : +24VDC
40 : GND (24VDC)
41 : GND OUT.
42 : VCC OUT
43 : +24VDC
44 : not connected
45 : RS232 / HSI
46 : RS232 / Pin 6
47 : RS232 / Pin 4
48 : RS232 / Pin 1




5 : ready signal
6 : GND (VCC)
7 : STOP
8 : limt switch left
9 : Opto-GND
10 : GND (24V)

11 : ready signal
12 : GND (VCC)
13 : home switch
14 : limit switch right
15 : service switch ext.
16 : +24VDC

# 3. Position control
# 3.1 configurations via DIP-switches

All adjustments must be made during power-off of the SERS !
Changes of switches during power on of the SERS are be ignored – exceptionally bits 4 to 6 from switch 1.

## 3.1.1 Drive address

The **drive address** (= address of a SERS slave within a ring of drives) can be selected at the 8 pole DIP-switch 2 (see 2.1.5 dimensions) **bits 1 to 7**.
The selected drive address is build with a binary code of the 7 switch positions.

| bit | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| value | 1 | 2 | 4 | 8 | 16 | 32 | 64 |

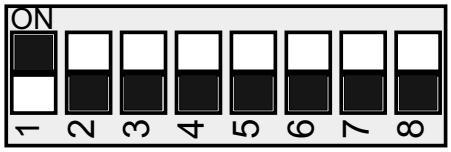
e.g. bit 1 = ON → address = 1
bit 1 = ON and Bit 2 = ON → address = 3 (= 1 + 2)
bit 2 = ON and Bit 4 = ON → address = 10 ( = 2 + 8)

Possible values for the **drive address** are **1 to 127 for SERS-Slaves** and the **address 0 for configuring a SERS as master** (if the master is called with the address 200, then it behaves like a slave and can be programmed).
A ON-position of a switch means a set bit (=1).

## 3.1.2 Automatic program start (autostart)

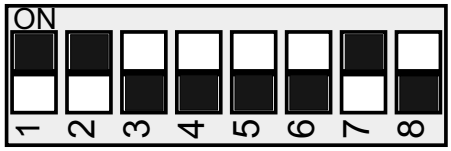
An **automatic program start** can be selected by switching on **bit 8** of DIP-switch 2.
In case there is a valid executable program in the $E^2$Prom of the SERS, then with an ON-position of this switch the program will be started automatically after a power-on-reset (= connecting SERS with power supply).

Bit 8 : ON : autostart enabled
OFF : autostart disabled



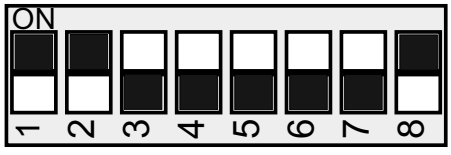

ON
OFF

Selected drive address: 1
autostart : disabled




ON
OFF

Selected drive address : 67
autostart : disabled




ON

Selected drive address : 3
autostart : enabled

## 3.1.3 Baud rate

The baud rate can be selected via bits 1 to 3 of the 8-pole DIP-switch 1 – (see 2.1.5 dimensions).

| Baud rate (baud) | 1200 | 2400 | 4800 | 9600 | 19200 | 38400 |
|---|---|---|---|---|---|---|
| bit 1 | OFF | ON | OFF | ON | OFF | ON |
| bit 2 | OFF | OFF | ON | ON | OFF | OFF |
| bit 3 | OFF | OFF | OFF | OFF | ON | ON |

## 3.1.4 Enable manual drive functions

Via **bit 6** of the 8-pole DIP-switch 1 - (see 2.1.5 dimensions) – s**ervice-switch** – can be selected, that only the manual drive functions (manual drive right/left/ slow/ fast and start homing) via the digital inputs are enabled. The auto start function and the program start via the serial interface will be disabled.

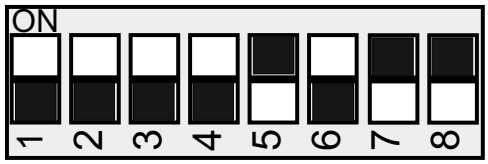
bit 6 : ON : all manual drive functions at the inputs enabled
no auto start or start program via serial interface possible
OFF : manual drive functions at the inputs enabled depending on parameter P1021
auto start and start program via serial interface enabled

For enabling or disabling manual drive functions via parameter P1021 see page 43.

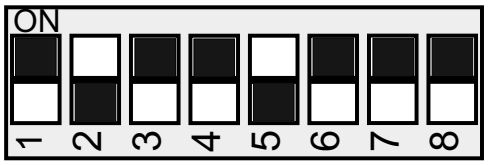
## 3.1.5 Signal levels and polarity of the inputs

Via bits 4 and 5 of the 6-pole DIP-switch 1 (see 2.1.5 dimensions) the signal levels and the polarity of the digital inputs I1-I8 can be configured.

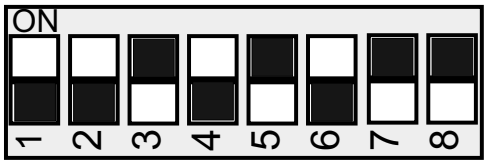
bit 4 : OFF and bit 5 : OFF : high-active with PLC/SPS-level (13,5 V signal level)
bit 4 : ON and bit 5 : OFF : high-active with TTL-level (3,5 V signal level)
bit 4 : ON and bit 5 : ON : low-active (1 V signal level)
bit 4 : OFF and bit 5 : ON : not valid (may not be selected)




selected baud rate : 1200 baud
inputs : not valid
manual drive via dig. inputs depend on P1021




selected baud rate : 38400 baud
inputs high-active – TTL-level
manual drive via digital inputs always enabled




selected baud rate : 19200 baud
inputs : not valid
manual drive via dig. inputs depend on P1021

**Bit 7 and 8 of the 8-pole DIL-switch 1 are for the option ´input pulse´ (see chapter 3.4) of the inputs I 3 and I 4 at the I/O-port.**
**In standard adjustment both switches are set to ON !**

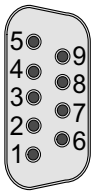
# 3.2 Interface RS232C/V24
## 3.2.1 Configurations

Interface configurations for the SERS:

**baud rate** : selectable via DIP-switch 1 (see 3.1.3 page 15)
**data bits** : 8
**stop bit** : 1
**parity** : no
**handshake** : selectable in parameter P1017 – see page 43.
- standard : hardware – pins 7 and 8 at 9-pole D-sub connector must be connected

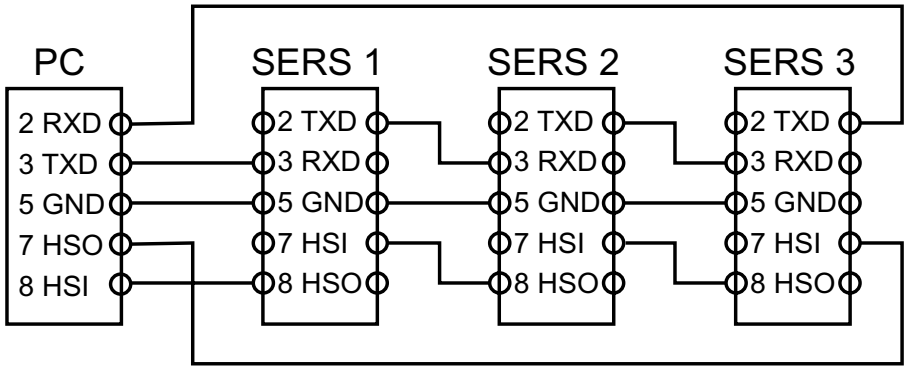
## 3.2.2 Assignment 9-pole D-sub-connector

pin
1 : not connected
2 : TXD
3 : RXD
4 : not connected
5 : GND



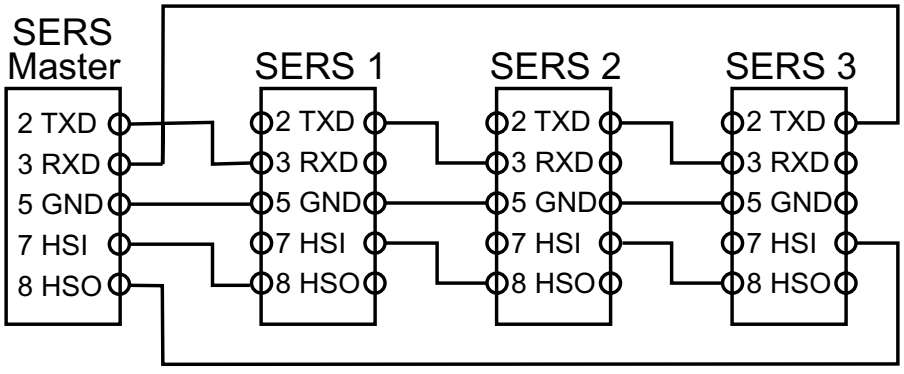

6 : not connected
7 : HSI (hand shake IN)
8 : HSO (hand shake Out)
9 : 9 VDC (power supply terminal)

## 3.2.3 Connection PC-SERS and SERS-SERS - RS232




If the hardware hand shake is disabled in parameter P1017 then the pins 7 and 8 are not used !

In standard adjustment the hardware hand shake is disabled !




**Attention:**

PC – SERS-Slave:
1:1 connection

SERS-Master-SERS- Slave:
Zero modem - connection

# 2.3 I/O-Port with digital inputs and outputs

## 3.3.1 Standard I/O-port

All SERS units include in standard version:

- 8 free programmable digital inputs (**I1** until **I8**)
- 1 analogue input (**ADC**) and
- 4 outputs (**O1** until **O4**)

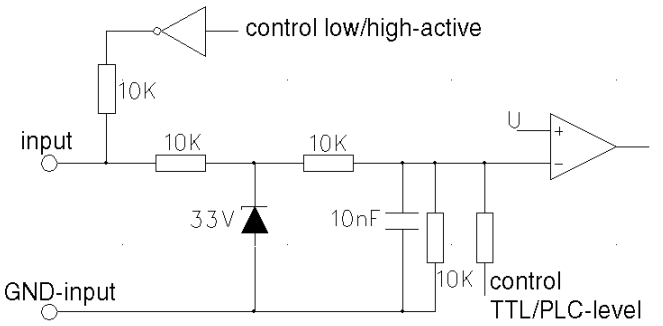
The I/Os are connected to the 25-pole D-Sub-female connector, located on the front panel of the SERS (see page 11), or for versions SERS ... R1 and SERS ... R2 at the additional 32-pol. Connector at the boards rear side.

The inputs can be configured low or high-active and with TTL or PLC/SPS- signal level (see chapter 3.1.5).

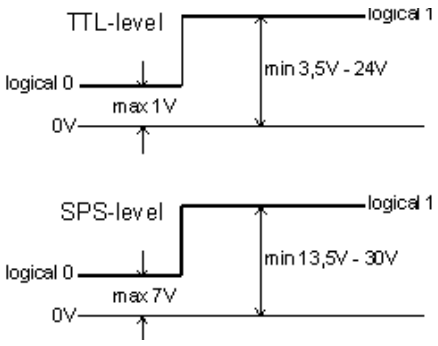
**The GND-potential of the inputs is connected with the GND-potential of the power supply for the SERS** (pins 30-32a/c of the 32-pole VG-connector - see page 7) and when using a ELK-rack with the internal common GND.

In case of an active service switch (extern and P1092=1 or DIP-switch 1 Bit 6 = ON) the inputs I1 to I8 are assigned to the functions manual drive (jog), start homing, ON/OFF, STOP and START executable program. Else the inputs may be used as free programmable inputs for an executable program (or they may be requested in the serial mode via parameter P1300). Via parameter **P1021** it is possible to assign just some of the predefined functions to the inputs. The not assigned inputs may be used as free programmable inputs - see also input parameters page 43

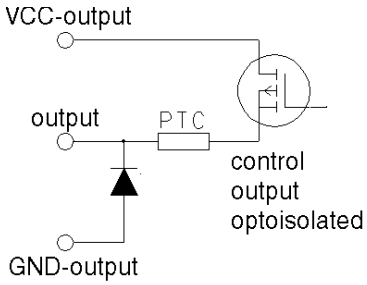
**Digital Inputs - schematics**




**Digital Inputs – signal level**




**Digital Outputs - schematics**

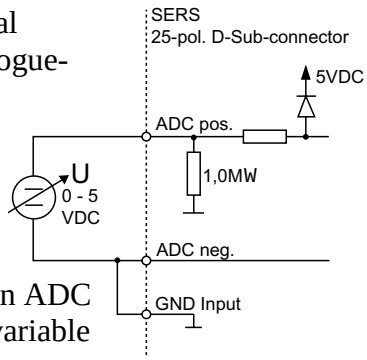



The outputs must be supplied externally by DC-voltage (5VDC - 30 VDC).
When an output is set (e.g. O1=1) then a P-FET will switch through the external supply voltage, a not set output means tri-state.

**ADC analog Input - schematics**

Differential 8-Bit analogue-Input

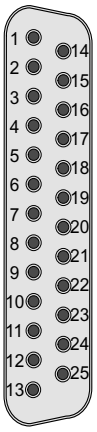
0 until 5VDC




Connection ADC e.g. with variable power source

**Connections standard I/O-port**

| Pin | connection |
|---|---|
| 1 | VCC outputs |
| 2 | VCC outputs |
| 3 | output O4 |
| 4 | output O2 |
| 5 | GND output |
| 6 | GND output |
| 7 | do not connect |
| 8 | analogue input ADC pos |
| 9 | input I8 - Start program |
| 10 | input I6 - Motor ON/OFF |
| 11 | input I4 - Right fast |
| 12 | input I2 - Right slow |
| 13 | GND input |




**25 pol. D-Sub connector**

| Pin | connection |
|---|---|
| 14 | VCC outputs |
| 15 | output O3 |
| 16 | output O1 |
| 17 | GND output |
| 18 | GND output |
| 19 | do not connect |
| 20 | analogue input ADC neg. |
| 21 | input I7 - Stop |
| 22 | input I5 - Start homing |
| 23 | input I3 - Left fast |
| 24 | input I1 - Left slow |
| 25 | GND input |

# 3.3.2 Optional I/O-port extension

The optional I/O-extension (type designation with extension “I/O“) includes additional 8 digital inputs and 12 outputs (additional to the standard 8 digital inputs and 4 outputs).

In the **serial mode** the inputs I9 until I16 can be questioned via parameter **P1301**.
In the other modes the syntax for the additional inputs is **I9** until **I16**.
The additional 12 outputs are programmed via **O5** until **O16**.

The inputs I9 until I16 and outputs O5 until O16 are connected to the 25-pole female D-Sub-connector “**I/O-2**“ at the SERS-front panel.

**Pin connections at the 25-pol. D-Sub-connector “I/O-2“:**

| | | | | |
|---|---|---|---|---|
| 1: VCC (OUT) | 6: O9 | 11: I11 | 16: O14 | 21: I16 |
| 2: VCC (OUT) | 7: O7 | 12: I9 | 17: O12 | 22: I14 |
| 3: O15 | 8: O5 | 13: GND (IN/OUT) | 18: O10 | 23: I12 |
| 4: O13 | 9: I15 | 14: VCC (OUT) | 19: O8 | 24: I10 |
| 5: O11 | 10: I13 | 15: O16 | 20: O6 | 25: GND (IN/OUT) |

**Ouputs:**
VCC (OUT) max: 24VDC (+ 20%)
Max. load outputs O5 – O16 : 100mA per output (protected against short circuit)
Set output (e.g. O5=1) → VCC (OUT) is connected to the set output (PNP-output)
Not set output (e.g. O5=0) → output is in tri state
The outputs O5 until O16 are not galvanically isolated (in contrary to O1 until O4). The output logic must be supplied via the pins 1,2,14 (VCC OUT) and 13,25 (DND IN/OUT) with 5 - 24 VDC.

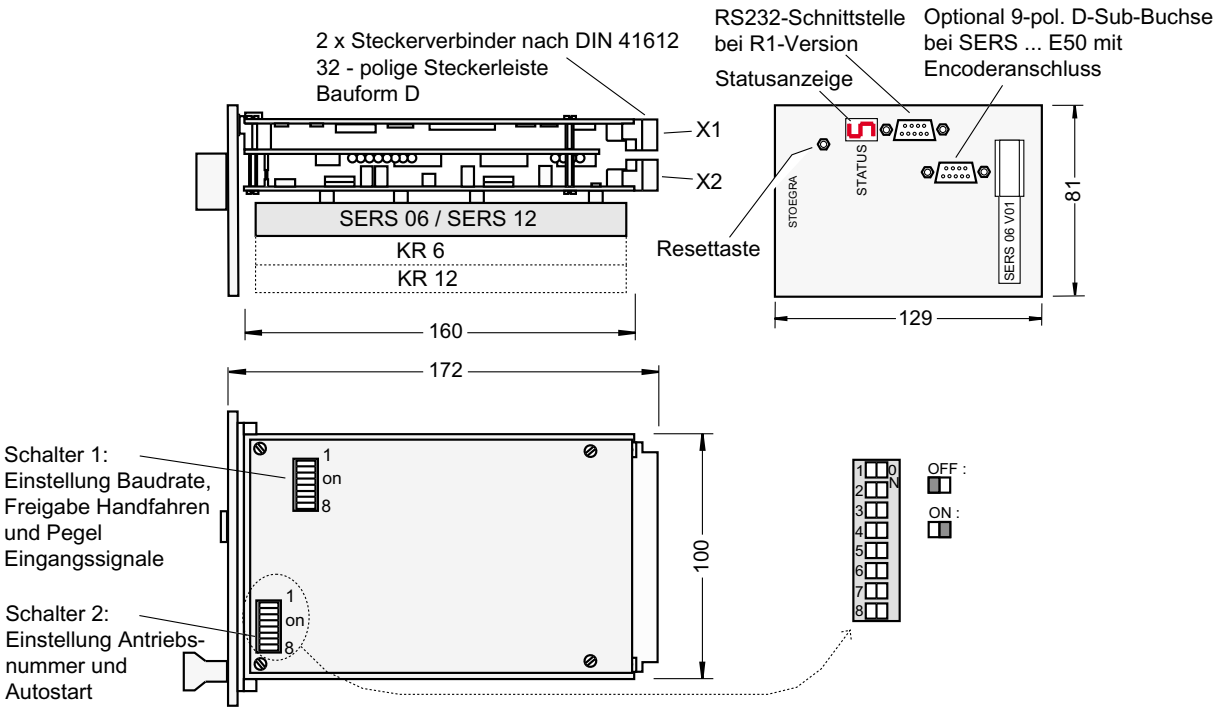
**Inputs**:
Signal level: High-active 24VDC
(signal >= 12VDC → input = “1“ , signal < 12VDC → signal = “0“)

Input resistance approximately 120kOhm (internal pull-down resistor).

# 3.3.3 I/O-Port versions SERS ... R1 and SERS … R2

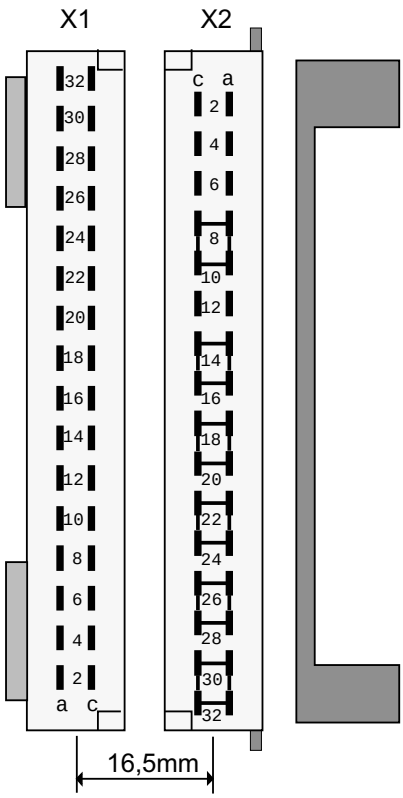
The SERS versions “R1“ and “R2“ include a second 32-pole connector at the boards rear side (connector X1 in drawing below) for the connection of the I/O-ports and the RS232 interface. (connections X2 – see chapter 2.1.1)

The version “R1“ includes an additional 9-pole D-Sub-connector at the front panel for the RS232 interface.




**Connections:**

| X1 | signal | X1 | signal |
|---|---|---|---|
| 2 a | not connected | 2 c | not connected |
| 4 a | input I 5 | 4 c | input I 2 |
| 6 a | input I 1 | 6 c | input I 4 |
| 8 a | GND input | 8 c | input I 6 |
| 10 a | input I 3 | 10 c | input I 8 |
| 12 a | input I 7 | 12 c | ADC neg |
| 14 a | not connected | 14 c | ADC pos |
| 16 a | GND outputs | 16 c | output O1 |
| 18 a | output O3 | 18 c | output O2 |
| 20 a | VCC outputs | 20 c | output O4 |
| 22 a | not connected | 22 c | Handshake Out |
| 24 a | not connected | 24 c | not connected |
| 26 a | Handshake In | 26 c | 9 VDC (50mA) |
| 28 a | not connected | 28 c | GND (RS232) |
| 30 a | not connected | 30 c | RXD |
| 32 a | not connected | 32 c | TXD |



# 3.4 Optional Encoder input

Encoder signals (e.g. from an encoder mounted at the stepping motor – for control of the steps - or from a hand wheel with an encoder) and pulse signals (e.g. from a pulse generator) can be connected at a the 9-pole D-Sub-connector ´ENC´ or at the inputs I 3 and I 4 at the I/O-Port (25-polige D-Sub-connector – see page 18).

## Encoder input at 9-pole D-Sub-connector:

**- exisists only at versions SERS ... V02 E50 ... !!!**

The 9-pole encoder-connector is located at the SERS-frontpanel (indication ´ENC´).

Encoder signals from 2-chanel-encoders (signal A and B and inverted signals /A and /B) with 1:1 duty cycle can be evaluated The evaluation of the signals includes a 4-times evaluation (every edge of A and B creates an internal pulse) , the recognition of the motor direction and analogue and digital signal filters. This connection can be used for a control of steps / load angle, for a hand wheel or for a function ´electrical shaft´.

**5VDC encoder**

At Pin 2 of the 9-pole D-Sub-connector ´ENC´ there is provided a 5VDC voltrage (max. 100mA) for the supply of 5VDC-encoders.

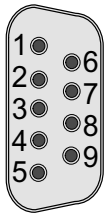
**24VDC encoder**

The SERS does not provide a 24VDC voltage for 24VDC-encoders. The 24 VDC must be supplied externally.

- 24 VDC encoders are allowed to be connected only to types ´SERS ... V02 **E50/24** ...´ (SERS-type designation must include ´**/24**´ after ´E50´) !!!
- 24 VDC encoder may **not** be connected to types ´SERS ... V02 **E50** ...´ (SERS-type designation without ´**/24**´ after ´E50´) !!!

Pin connection
5 : B
4 : shield
3 : n.c.
2 : +5VDC
1 : A

ENC




Pin Belegung
9 : /B
8 : GND
7 : n.c.
6 : /A

Encoder cable:
When connecting an encoders for the control of step-/load angle then a shielded cable must be used !

## Pulse input at I/O-port (25-pole D-Sub-connector) – I 3 and I 4:

At the I/O-Port inputs I 3 and I 4 (Pin 23 / Pin 11) there can be connected encoder or pulse-signals for using a handwheel or realizing an electrical shaft function..

A control of step/load angle is not possible with this inputs.

**Version SERS ... V02 E50 ...** :
The signals are evaluated 4-times, and the motor direction is recognised at encoder signals

**Version SERS ... V02 ...** (without ´E50´):
The signals are evaluated 1-time (A and B signals → only rising edge of A).

Via DIL-switch 1 – bits 7 and 8 (see page 15) the max. input frequency at the inputs I 3 and I 4 is specified.
bit 7 → I 3 and bit 8 → I 4

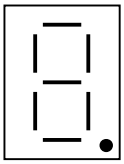
max. input frequency : switch ON : 5 KHz - switch OFF : 100 KHz

**More datas (necessary parameter adjustements) → see parameter P1029, P1053...**

# 3.5 Status- and error indication via 7-segment display

The 7-segment-display above the interface connector (see 2.1.5 dimensions) indicates the actual status of the SERS.

Elements of indication :
7 segments



Error and status indications are indicated by constant illuminating characters.

Warnings are indicated by blinking characters.

Following table shoes all possible indications :

| Indication | description | comment |
|---|---|---|
| - | initialisation phase | Is indicated 1 second after a power-on-reset |
| 4 | ready for switching phase current ON | |
| 5 | phase current of motor is ON positioning jobs can be executed | |
| 7 | checksum error of dates in $E^2$Prom | Overwrite $E^2$Prom-parameters with backed up parameters (with SERS-software) or select standard parameters (with P1004=3 - see page 66) |
| 8 | error over temperature - from 85 °C (± 10%) on the power amplifier | check cooling of SERS – forced draft (fan and 24 VDC in ELK) o.k. ? |
| 9 | error under voltage - see 2.1.4 | check power supply |
| 8 (blinking) | warning over temperature 75 °C (± 10%) at the power amplifier stage | check cooling of SERS – forced draft (fan and 24 VDC in ELK) o.k. ? |
| 9 (blinking) | warning under voltage - see 2.1.4 | check power supply |
| A (blinking) | warning position overflow – positioning job is reaching position limit | - Parameter W (P47) to big<br>- check positioning mode (for endless positioning → P1014=1 see page 48) |
| C | Limit switch is open | check limit switches at machine and limit switch inputs of SERS – see 2.1.2 |
| E | shortcircuit in motor or at power amplifier board | when installing the motor check phase connections |
| F | error step angle control - the motor could not follow the position command value (only with option step angle control) – see P1029 | - check encoder connections<br>- acceleration to high<br>- external load to big (not enough motor torque) |
| F (blinking) | warning step angle control – description as F – see P1029 page 62 | see comment for F |
| H (blinking) | warning program error – executable program stopped because of an error in the program | check parameters and labels in program (use SERS-software for debugging) |
| L (blinking) | Warning software limit switch | The actual position exceeded the limit position stored in P1040 or P1041 |

**Reset of an error with 'P11=0'** ( see P11 page 46)

**Reset of a warning with 'P12=0'** ( see P12 page 46)

# 3.6 Operating modes - overview

## 3.6.1 Serial operation

In the serial mode the SERS stepping motor positioning drives are controlled by a higher ranking master (PC, PLC or NC) via the serial interface RS232C (e.g. COM1 of a IBM-PC) or RS485.
The master controls the connected SERS-drives by sending commands and parameter assignments as ASCII characters via the serial interface, requests actual parameters like position, drive status, inputs ... , or starts executable programs or subroutines in the SERS drives.

In standard version there can be controlled up to 127 SERS-drives from one e.g. PC-COM-interface – 127 different addresses (1 – 127) can be adjusted at a SERS.

## 3.6.2 Master mode

In the Master mode one SERS drive is configured as higher ranking control (Master).
By selecting the address '**0**' at the DIP-switch, the SERS will function as Master.
If the master (with adjusted address '0') is called with the address '200', then it behaves like a slave and can be programmed.
A Master-SERS sends commands via the serial interface and can control other SERS-drives in that way. So even complex motion applications can be realised without additional control (PC or PLC).
Characteristics of a SERS Master:

- all drive functions for a standalone drive
- assigning parameters of other SERS-drives (e.g. velocity, acceleration, position)
- sending all commands to other SERS-drives (e.g. start, stop, set output)
- start of executable programs and subroutines of other SERS-drives
- request of inputs and drive status of other SERS-drives

## 3.6.3 Parallel mode

In the parallel mode subroutines (stored in the $E^2$Prom) are called in the SERS-drives via the digital inputs. The different subroutines are addressed with 6 inputs (binary address 0 to 63). With another input the address is set (strobe) and the addressed subroutine will be started.

The digital (parallel) inputs can be controlled e.g. by manual switches, or by digital outputs of a PLC.

## 3.6.4 Standalone mode

In the standalone mode the SERS runs independently without a higher ranking control. The $E^2$Prom program is started by a digital input (start input I8) or automatically, after connecting the SERS with a power supply (autostart-selection via DIP-switch, see 3.1.2 page 14).
Events at the inputs can control the execution of the program.

# 4. Programming and operating the SERS
# 4.1 Syntax - general notes

- **each line sent to the SERS must begin with the character '#' !**
  e.g.: **#ON** will switch on the phase current – the line begins with '#'

- **each line must be finished with Carriage Return** (**Return-key** of the keyboard corresponds to the ASCII-character '**#13'** - decimal number 13 in the ASCII-table - when writing a (e.g.) PC-program) **or Linefeed** (ASCII-character **#10**)
  e.g. programming in PASCAL – a string with the assignment 'W=1000' sent to the SERS via the COM-interface has to be programmed as **string_example := '#W=1000' + #13**

- Commands will be executed from the SERS already before receiving a Carriage Return! E.g. when writing **#ON** and a following **Carriage Return** the current in the motor phases will be switched on already before the Carriage Return !

- There may be **multiple commands and parameters in one line** , before writing a Carriage Return – but **maximum 60 characters in one line**.
  e.g.: **#ON V=1000 W=2500 O=1 E** may be written in one line and at the line end finished with a **Carriage Return**.

- **Carriage Return (CR - #13) and Line feed (LF - #10) will be echoed with a 'ok' and a digit** by the SERS, in case the sent commands and parameters are free of errors. The digit can be '0' to '4'.
  '0' : Drive is running (P336=0 )
  '1' : Drive is in position (P336 = 1)
  '2' : Drive is running and a warning was created ( P12 > 0)
  '3' : Drive is in position and a warning was created ( P12 > 0)
  '4' : There is an error at the drive ( P11 > 0)
  The characters, sent by the SERS are :

  **#10 ok1 #10 #13 or #10 ok4 #10 #13** → string with LF, 'ok', digit, LF und CR

  In the programming mode (4.5.3 page 26) there will be sent "pgm" instead of "ok"!

- In the **V24-mode each sent character to the SERS will be sent back** (echoed) from the SERS (mode-selection via parameter P1017).
  **Attention :** In the mode HANDSHAKE_RS485 the SERS will send back only the "ok" ("pgm") after receiving a Carriage Return or Linefeed - there is no character echo !

- If the **commands or parameters are faulty**, then **error messages** will be returned. Error messages start with the character '*****'.
  e.g.: *****limit switch open***** in case the drive is on a limit switch (input limit switch is open) and a start command '**E**' (execute positioning job) was sent to the SERS

  When receiving faulty syntax, the SERS will return immediately an error message. When receiving faulty values (e.g. to big) then depending on the values the SERS will return an error message before or after a Carriage Return.
  **Attention :** In the mode HANDSHAKE_RS485 (see parameter P1017 – Handshake) error messages are returned always only after a finished line (Carriage Return / Line feed) !

- When requesting parameters in the serial mode (e.g. #2P11?) then the SERS returns the result as follows : 'P11=0' #10 #13 . First the parameter number of the requested parameter (P11), then the character '=', then the content of the parameter (e.g. '0' ) and at last the line feed character #10 and a Carriage-Return #13 are returned.
- When assigning parameters, there are **no units** allowed
  e.g. writing '**V=1000 U/min' is wrong, and an error message will be returned !**
  Right is : '**V=1000**' - the unit is defined in the parameter P44 (scaling for velocity dates)
- **Addressing a SERS** is realised by sending the drive address via the interface.
  e.g.: when sending **#2** the drive with the address 2 is called.
  After sending an address, all following commands and assignments of parameters are accepted only by the addressed drive. All other connected drives will ignore the commands and assignments.
  If some commands or parameter assignments are dedicated for another drive, then first the drives address has to be sent – e.g. **#3** .

  e.g.:
  ```
  #2 ON
  #v=1000
  #W=5000 E
  #3 ON
  #V=1500
  #W=4000 E
  ```
  The phase current of the drive with address 2 is switched on with 'ON', the velocity is set to 1000, the way (distance) to be moved is set to 5000 and with 'E' drive 2 is started. Afterwards drive 3 is addressed, the current is switched on with 'ON', and velocity and the way are set and drive 3 is started.

# 4.2 Syntax definitions

Following the general syntax definitions for the SERS.
All expressions and characters, which are not described more detailed must be input directly.
*Cursive* written expressions are defined more detailed in the following lines.
Expressions in [ ] are optionally.
All key words may be written in small or big letters.
Explanations to the syntax are behind the comment characters // .

**Instruction line**

*# [drive address] [instruction list] end of line*

**Drive address**

*decimal constant*
* // addressing all drives – e.g. for synchronically start of all drives – broadcast address

**Instruction list**

*instruction*
*instruction instruction list*

**Instruction**

*" character list "* // Text for displaying in the SERS-Programmer
*assignment*
*command*
*operator operand* // for arithmetic functions
*unary_operator* // for arithmetic functions

IF [ ! ] *address* [*condition decimal constant*] // interrogation see chapter 4.6.1
WAIT [!] *address* [*condition decimal constant*] // Wait for event
: [!] *address* [*condition decimal constant*] // manual driving until input is (not) active
GOTO *decimal constant* // jump to label number decimal constant
GOSUB *decimal constant* // jump to subroutine at label number decimal constant
GT *decimal constant* // equivalent to GOTO
GS *decimal constant* // equivalent to GOSUB
RETURN // return from subroutine – target address is the line after the last GOSUB command
RT // equivalent to RETURN
RS // manual drive right slow (**R**ight **S**low)
RF // manual drive right fast (**R**ight **F**ast)
LS // manual drive left slow (**L**eft **S**low)
LF // manual drive left fast (**L**eft **F**ast)
L *decimal constant* // label number
LIST // list program in $E^2$Prom
LIST **.** // list actual program line – next line to be executed or last listed line
LIST - // list line before – actual position in program is also set to line before
LIST + // list next line – position in program see LIST-
LIST *decimal constant* // list line decimal constant – program position is set to program address decimal constant (in 'PGM'-mode LIST0 enables inserting a line at the program begin
LIST - *decimal constant* // list 20 lines until decimal constant – program-position is set to address decimal constant
// all LIST-commands will change the actual program position !
LIST *decimal constant1.decimal constant2* // list decimal constant2 multiplied 10 lines from program line decimal constant1
LIST P // list all parameters actual values
LISTP *decimal constant* // list parameter decimal constant (internal order – according to P-no. sorted ascending) with actual values
LISTP *decimal constant1.decimal constant2* // list decimal constant2 multiplied 10 parameter from parameter decimal constant2 (internal order - according to P-no. sorted ascending) with actual values
// // comment character – all characters in a line from here will be not interpreted
PE // end of program
PSAVE // parameters in the SERS will be stored permanently – values from RAM of SERS into $E^2$Prom of SERS
POSSAVE // save P51 (actual-position) into $E^2$Prom
POS0 // move to electrical "0"-position (every 7,2°)
POSR // move to position command value (see P1043)
NEW // equivalent to P0=2 – programming mode with erasing of actual program. All characters, sent from now on, will be stored in the $E^2$PROM.

PGM // Turning on the programming mode with editing of the actual program. Input lines will be inserted at the actual program position.
// In the programming mode instructions like LIST, DEL, TR C, will not be stored, but executed !
DEL // erases the actual line in the programming mode
C *decimal constant* // Changes the value of the assignment at the actual programming position
QUIT // terminates programming mode
RUN // equivalent to P0=1 – start program in $E^2$Prom
TRON // trace mode on – mode for enabling program execution line by line
TROFF // trace mode off
TR? // requests trace mode
TR // enables trace mode and executes of one a program step
VER // shows program version
ON // equivalent to P134=7 – turn on phase current of motor
OFF // equivalent to P134=0 – turn off phase current of motor

**condition**

>
<=
=
<>

**Character list**

*character*
*character-list character*

**Character**

// any character except quotation marks " and 0x00 (=ASCII-character with code 00)

**Assignment**

*address* = *data*
*X* = operand // accumulator for calculating
*address* ? // shows parameter data (value - content)
*address* ?? // shows parameter identification

**operator**

\+ // adding to the accumulator
– // subtracting from the accumulator
* // multiplying with accumulator, in case '∗' shall be used at begin of line , then the address of the drive has to be set ahead, for the SERS not interpreting the character '∗' as broadcast address
/ // dividing accumulator
& // "AND" accumulator
| // "OR" accumulator
^ // "EXCLUSIV OR" accumulator

**unary_operator**

| | |
|---|---|
| *NOT* | // inverting accumulator bit by bit |
| *NEG* | // inverting sign of accumulator |

**operand**

address
decimal constant

**address**

| | |
|---|---|
| A | // equivalent to P138 - acceleration |
| ADC | // equivalent P1046 – analogue Input |
| C1, C2, C3 | // equivalent P100, P101, P102 - counter |
| D | // equivalent P1100 – delay time in $^{1}/_{10}$ seconds |
| M1, M2, M3 | // equivalent P1101, P1102, P1103 - marker |
| O1 until O16 | // equivalent P1201 until P1216 / output 01 until O16 |
| P *decimal constant* | // parameter - see description of parameters (4.7 page 41) |
| POS | // equivalent P336 – status In-Position |
| V | // equivalent P91 - velocity |
| W | // equivalent P47 – distance to move or position |
| WA | // as W but additionally P1014=2 (absolute positioning) |
| WR | // as W but additionally P1014=0 (relative positioning) |
| WP | // first positioning section in mode polynom positioning |
| WAP | // as WP but additionally P1014=2 (absolute positioning) |
| WRP | // as WP but additionally P1014=0 (relative positioning) |
| WPT | // last positioning section in mode polynom positioning |
| WAPT | // as WPT but additionally P1014=2 (absolute positioning) |
| WRPT | // as WPT but additionally P1014=0 (relative positioning) |
| X | // equivalent P1047 - accumulator for calculating |
| Z | // equivalent P1 - destination address for master mode |

**Command**

| | |
|---|---|
| E | // start actual positioning job – actual value set in W (P47) will be executed |
| H | // start homing |
| S | // stop – the motor will be stopped - see also parameter P1033 – continue after stop |

**Data**

[ - ] *decimal constant*

**End of line**

| | |
|---|---|
| Carriage Return | // RETURN-key of keyboard |
| /r | // Return character – ASCII-Code 13 (decimal) |
| /n | // Linefeed character – ASCII-Code 10 (decimal) |

**Decimal constant**

decimal digit
. decimal digit // e.g. **. 5** (= 0.5)
decimal constant decimal digit // e.g. **10.75**

**Decimal digit**

0,1,2,3,4,5,6,7,8,9

**Separating character**

| | |
|---|---|
| "space character" | |
| , | // comma |
| ; | // semicolon, prevents a carriage return at program input in the actual line |
| "tab" | // tabulator character |

# 4.3 Program lines and numbering

Internally each instruction receives its own line number.
The lines **#1 ON V=1000 W=5000 E**
**#O1=1 D=10 W=2000 E**
are put out in a list (after the instruction : 'list') - with the selected option line numbers (see also parameter P1028) - as follows :

1: **#1 ON V=1000 W=5000 E**
5: **#O1=1 D=10 W=2000 E**

or (depending on the parameter P1028) :

1: **#1 ON**
2: **#V=1000**
3: **#W=5000**
4: **#E**
5: **#O1=1**
6: **#D=10**
7: **#W=2000**
8: **#E**

# 4.4 Limitations for $E^2$Prom-programs

Following limitations for a $E^2$Prom-program apply :

- program memory in standard version : the number of program lines depends on the structure of the program - there are 2Kbyte available, e.g. long lines result in a fewer total number of programmable lines. When programming only one instruction per line, then there can be programmed approximately 300 lines.
- maximum of 60 characters per line
- maximum of 64 labels – L1 to L64 – (plus Label L65, where the program will jump to after a program stop)
- interlaced storage number for subroutines : max 4 (into each other interlaced program loops)

# 4.5 SERS functions and modes

## 4.5.1 Manual mode - manual drive control

For the manual drive control there are 4 different commands :

1. manual drive right slow – RS-function (Right Slow)
2. manual drive right fast – RF-function (Right Fast)
3. manual drive left slow – LS-function (Left Slow)
4. manual drive left fast – LF-function (Left Fast)

Following parameters are available for the manual drive control functions :
Acceleration manual drive control : P1018
Velocity manual drive control fast : P1019
Velocity manual drive control slow : P1020
Enable inputs for manual drive control functions : P1021

The manual drive control functions can be started by

- sending the instructions ‘RS‘ or ‘RF‘ or ‘LS‘ or ‘LF‘ via the serial interface to the SERS

or

- Putting signals at the inputs I1 to I4 at the 25-pole D-Sub-connector (I/O-Port) – see connections page 18.

When using the manual drive control functions via the inputs I1 to I4 the phase current of the motor will be turned on automatically.
When using the manual drive control functions by sending instructions via the serial interface the phase current of the motor must be turned on before (by sending ‘ON‘ to the SERS).

## 4.5.2 Executable program mode for master mode, parallel mode and standalone mode

Following steps are necessary for a automatically executable program mode for the operating modes master mode, parallel mode and standalone mode:

- writing and downloading an executable program into the $E^2$Prom of the SERS
- adjusting all necessary parameters in the SERS
- adjusting the DIP-switches on the SERS

**Executable program**

**Master mode:**

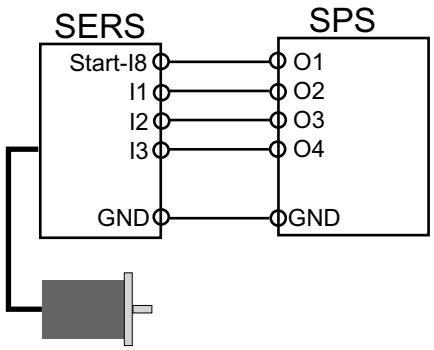
The structure of a master program principally is the same as for programs in other operating modes. Additionally for a master program there are instructions, which are for controlling other SERS-drives (Slaves). This is done by sending the expression ‘Z=*address*‘ to the drive to be controlled, where *address* is the drive address of that slave-drive. All following instructions in the program until the next ‘Z=*address*‘ - expression will be sent to the slave-drive. With the expression ‘Z=0‘ the master assigns the following instructions for himself again (address 0 is the master address).
A typical master-parameter, which can be changed only during operation, is P1110 . With this parameter the master decides, weather after starting a positioning job (instruction ‘E‘) it will proceed with the next instruction only after finishing the positioning job or weather it will execute the next instruction without waiting for the end of the positioning job (in that way the master is able to control other drives while it self is still positioning).
See example program page 40.

**Parallel mode:**

A program for the parallel mode consists of one or more subroutines and is programmed as follows : Each subroutine is started with a label (the label number corresponds to the program address) and is terminated with 'RETURN' (or 'RT').

#L1
:
#RT
#L2
:
#RT
#L3
:
#RT
:

| I1 | I2 | I3 | I4 | I5 | I6 | Jump to Label |
|---|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 | L1 |
| 1 | 0 | 0 | 0 | 0 | 0 | L2 |
| 0 | 1 | 0 | 0 | 0 | 0 | L3 |
| 1 | 1 | 0 | 0 | 0 | 0 | L4 |
| 0 | 0 | 1 | 0 | 0 | 0 | L5 |
| 1 | 0 | 1 | 0 | 0 | 0 | L6 |
| until L64 | | | | | | |




The program waits, until there is a signal edge at the start input (input 8 at I/O-Port). Then, depending on the address, which is created by the inputs I1 to I6 (BCD 0 – 63) the program jumps to the corresponding label (L1 to L64 – see table above) and all instructions until the next RT (RETURN) are executed. After the RETURN the program waits again for the next signal edge at the start input.

**Event controlled mode:**

In the event controlled mode single subroutines (stored in the $E^2$Prom) are called directly via single inputs.
An executable program consists of one or multiple subroutines wich are defines through Label L… and RT.

e.g.: L1 ON RT
L2 V=1000 W=360 E RT

Depending on the activated input the corresponding subroutine will be called: (rising signal edge at input will initiate the jump to the subroutine).

Only the inputs/labels enabled in parameter P1098 are valid for this program mode. The inputs/labels not defined in P1098 may be used for other functions.

P1098 = 1→ jump to L1 at signal edge at input I1
2 → jump to L2 at signal edge at input I2
3 → jump to L1 at signal edge at input I1 and jump to L2 at signal edge at I2
4 → jump to L3 at signal edge at input I3
5 → jump to L1 at signal edge at input I1 and jump to L3 at signal edge at I3
…
255 → jump to L1 until L8 in case of edges at inputs I1 until I8

A positive signal edge at the input will call the subroutine, if at the time of the signal edge no program is executed (P0=0). If there was a positive signal edge at the input and a program was running at the time, and the input is still active after the program is terminated, then the subroutine will be called after termination of the program.

In case of signal edges at multiple inputs (enabled via P1098) at the same time, then "I1" is the input with the highest priority. "I2" is the input with the next lower priority.. If inputs stay active, then all corresponding subroutines will be called after each other. If an input, which did not lead to the call of its corresponding subroutine yet, changes to inactive state, during another program/subroutine is running, then the subroutine will not be called – meaning the signal edges will not be saved internally.

**Standalone mode:**

In the standalone mode the $E^2$Prom-program is started with a start signal (autostart via switched on bit 8 of switch 2 – see 3.1.2 page 14 – or start input I8 at I/O-Port) from the program begin. The program execution can depend on events at the inputs (I/O-Port) , or run independently.

**Necessary SERS-parameter**

**Parallel mode:**

P1022 Digin-label-enable : The inputs, which are used for the addressing in the parallel mode, have to be enabled in P1022 (see parameter description)
and the Start-Input must be set in P1021 (P1021=128)

**Event controlled mode:**

**P1098** enable program jump : the inputs needed for calling the subroutines in the event controlled mode, have to be enabled in P1098.

**DIP-switches**

**General notes:**

Selecting the autostart function at switch 2 (see chapter 3.1.2) – depends on the application

Selecting input signal level at switch 1 (see chapter 3.1.5) – depends on the connected hardware (e.g. control units, to the inputs connected switches are against GND or 24VDC ?)

Switch off service-switch-bit at switch 1 (see page chapter 3.1.4)

## 4.5.3 Programming mode

**When using the SERS-programming software from STOEGRA, you do not need the functions explained as follows !**

For entering the programming mode, the SERS must be stopped (no autostart active or a running program must be stopped via the Stop-input or a sent stop command).
There are two different programming modes :

- programming mode with **erasing** of actual program in $E^2$Prom and
- programming mode with **editing** of actual program in $E^2$Prom

The instruction '**NEW**' will start the programming mode with erasing an $E^2$Prom-program, the instruction '**PGM**' will start the programming mode with editing an $E^2$Prom-program.

A short description of the instructions (LIST, DEL, C, QUIT) for the programming mode can be found in chapter 4.2 syntax definitions in the section instructions.

## 4.5.4 Trace mode

**When using the SERS-programming software from STOEGRA, you do not need the functions explained as follows !**

In the trace mode an executable program can be executed step by step.
The instructions TRON, TROFF, TR, TR? Are explained in chapter 4.2 syntax definitions in the section instructions.

# 4.6 Programming instructions

## 4.6.1 IF : Conditional execution

Syntax : **IF** *event condition digit*
Digit: "0" or "1" respective "0 – 255" in case of programming "IF IN=digit"
The condition depends on the event : "=" , ">" , "<=" oder "<>"

The former syntax (firmware 100101 and earlier) ) using **IF [ ! ]** ***event*** still is accepted.
For the old syntax using the character **'!'** the event will be **inverted**. (**IF !event** means : If the event is not true).
When saving a program with the old syntax into the SERS $E^2$Prom and then reading the program out of the SERS, the program will be listed with the new syntax.

**If the event is true, then the next instruction will be executed, otherwise it will be skipped / ignored.**

**Events** may be **parameters** or **digital inputs** (at I/O-Port).

Following conventions are made for the digital inputs:

- **I1 to I8** : if the inputs is set, then the event is true.
  e.g.: IF I5=1 If input I5 is set, then execute next instruction, otherwise skip / ignore next instruction
  IF I5=0 If input I5 is not set, then execute next instruction, otherwise skip / ignore next instruction
- **IN 0 to IN255** : IF IN=*digit* requests all inputs, where the values of the inputs are binary coded
  (I1=1, I2=2, I3=4, I4=8, I5=16, I6=32, I7=64, I8=128)
  e.g.: IF IN5 - If I1 and I3 are set then execute next instruction (5 = I1 + I3)

Following parameters may be used as events :

- **P0** - request weather a program is executed (for master mode, if the program execution in the slave has terminated) - possible requests: IF P0=0 or IF P0=1
- **C1 to C3** : counter – each IF-request will decrement the counter by 1
  IF C1>1 → (C1=C1-1 and if still C1>1 then execute next instruction)
- **POS (P336)** : **IF POS=1** requests, weather the drive is "In Position", meaning weather it reached the command value position (in master mode for requesting the slave-state).
  IF POS=1 (motor is in position) , IF POS=0 (motor is running)
- **P1015, P1016** – request weather drive is in accelerating phase, or in constant running phase (drive runs with constant velocity)
  possible requests: IF P1015=1, IF P1016=1, IF P1015=0 or IF P1016=0
- **M1 to M3** (P1101 to P1103) : requests marker
  possible requests: IF M1=1 , IF M1=0 , IF M2=1, IF M2=0 , IF M3=1 , IF M3=0
- **P11, P12** : error, warning register : parameter <> 0 → event true
  IF P11<>0 requests, weather there is an error (e.g.: temperature error). In case of an error P11 is not zero and the following instruction will be executed.
  IF P11=0 executes the next instruction, in case there is no error
  IF P12<>0 and IF P12=0 as P11.
- **LP** : IF LP=1 request if P51<P1040 or P51>P1041 → request of status of software-limit switches - possible requests : IF LP=1 and IF LP=0
- **X** : request of accumulator : IF X > 0 or IF X ≤ 0
- **P1123**: request of polynom status: IF P1123=1 → load next polynom section

**Example for an IF request:**

```
IF P12<>0 O1=1
GT 20
```

If P12 is not equal 0 (a warning is occurred) then output O1 will be set (O1=1 is the next instruction following the IF-request). Afterwards the program will jump to Label 20 (GT20). If P12 is 0 (P12<>0 is wrong), then the next instruction "O1=1" will not be executed but skipped. The program executes the instruction afterwards (GT 20) and jumps to Label 20.

Following program is identical to above program:

```
IF P12<>0
O1=1
GT20
```

**Attention:** There is no difference for the SERS weather the instruction following the "IF command" is in the next program line or in the same line as the "IF command".

# 4.6.2 Program labels

Syntax : L *decimal constant* - decimal constant is a value between 1 and 65
e.g. L1 or L64

Labels are used as jump destinations, for creating program loops or subroutines.
Each label number may be defined only once in a program !

In case of P1033=2: The program jumps to Label L65 if the $E^2$PROM-program is terminated suddenly e.g. by a external Stop-command or a drive error (e.g. error temperature). This enables certain actions to be executed in case of an error.

In case the Label L65 does not exist then the program will be terminated after a drive error or a 'stop'.

# 4.6.3 GOTO, GT : jumps

Syntax : GOTO *Label* - Label must be defined anywhere in the program
Alternative syntax : GT *Label*

Program jump to a label (backwards or forwards)

```
#L1
#...
#...
#GOTO 1          // equivalent :  #GT 1
#...
```

or

```
#GOTO 1
#...
#...
#L1
#...
```

# 4.6.4 GOSUB : call of subroutine

Syntax : GOSUB *Label* - Label must be defined anywhere in the program
Alternative Syntax : GS *Label*

Jump to a subroutine
– Each subroutine contains a label at its begin and a ‘RETURN‘ at its end
e.g.:

```
#...
#GOSUB 12        // equivalent :  #GS 1
#...
#...
#L12             // start subroutine
#...
#RETURN          // end of subroutine and continue with instruction after last
                    'GOSUB'
```

# 4.6.5 RETURN : terminating subroutine

Syntax : RETURN
Alternative Syntax : RT

The instruction RETURN terminates a subroutine, which was called with ‘GOSUB‘ and initiates a jump to the instruction, which follows the ‘GOSUB‘.

A RETURN in the parallel mode terminates the called subroutine (called via a signal at the start input and a address at the I/O-Port). Then the SERS waits for the next subroutine call.

# 4.6.6 Programming of positioning jobs

A positioning job is specified by the standard parameter
- **acceleration** – Syntax : **A=*value***
- **velocity** – syntax : **V=*value***
- **way** (relative positioning) / **position** (absolute positioning) – syntax : **W=*value***

and executed with the instruction **'execute positioning job'** – Syntax : **E**

For executing a positioning job with the SERS, only the execute instruction '**E**' has to be sent to the SERS or '**E**' must be programmed in the executable program in the $E^2$Prom.
The parameters A, V and W of a executed positioning job are kept in the memory of the SERS and if not overwritten they will be used for the next positioning job.

**After Power On of the SERS the parameters A and V stored in the $E^2$Prom are valid. The parameter W after Power ON always is 0 (except W was saved with 'POSSAVE') !**

The parameter A and V may be redefined any time. If these parameters are overwritten, during the SERS is executing a positioning job, then they will be valid only for the next positioning job.

Changing W is only possible at standstill of the drive (POS=1), or when the drive runs with constant velocity (constant phase) .
When overwriting W in the constant phase, then in relative positioning mode the new value W will be added to the previous value W and the drive will execute the total new distance W. In absolute positioning mode the drive will position to the new value W. The values W sent during the constant phase may not result in a change of the motor direction. In that case the sent value W will be returned with an error message and only after finishing the actual positioning job, W will be executed with the next instruction 'E' (start positioning).

Additionally to the standard parameters there are some more parameters, which have influence on the positioning jobs :
- **Positioning mode** P1014 (**relative or absolute positioning**)
- **Ramp form of the acceleration** : P1032 (exponential or sinus-ramp form)
  P1005 (acceleration section 1)
  P1006 (acceleration section 2)
  P1007 (velocity section 1)
  P1008 (velocity section 2)
- **Scaling** : P160 (acceleration)
  P44 (velocity)
  P76 (way/distance – position data)

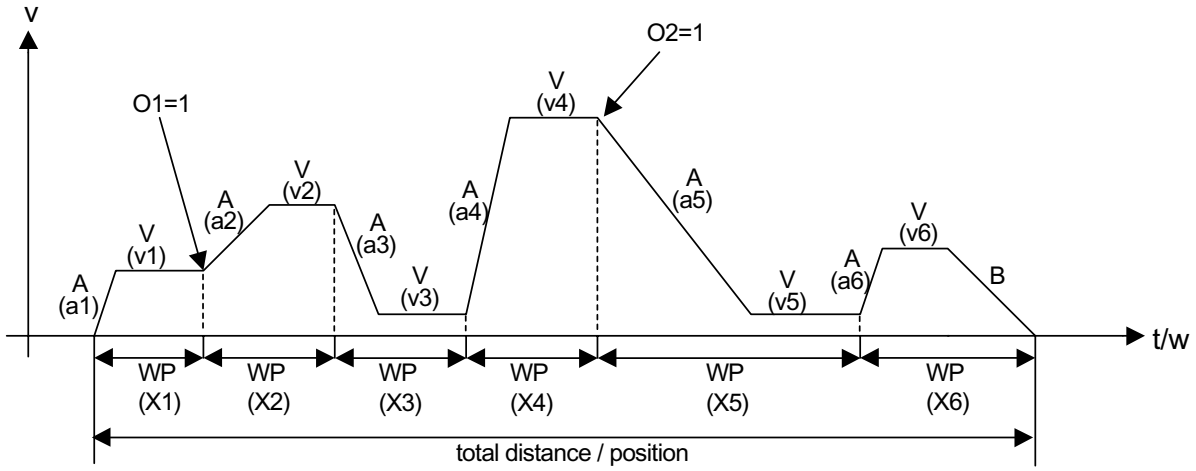
**Alternatively to the assignment W=*value*** there are the commands:
**WR=*value*** : positioning mode is set to relative (P1014=0) **and** W=*value* (e.g. WR=1000)
**WA=*value*** : positioning rmode is set to absolute (P1014=2) **and** W=*value* (e.g. WA=1000)

Following parameters show the actual status of a positioning job:
- **P336** (alternative **'POS'**) – POS = 1 if actual position value = position command value
- **P1015 (acceleration phase)** = 1 during accelerating of the drive
- **P1016 (constant phase)** = 1 when the drive runs with constant speed

In the executable program mode the program only continues with the next instruction, when POS=1 is fulfilled. But with parameter P1110 immediately execution of the next instruction independently of 'POS' can be selected (e.g. for the master mode).

# 4.6.7 Positioning with velocity profiles (polynom positioning)

The drive shall move the distance W = X1 + X2 + X3 + X4 + X5 + X6.
First the drive must accelerate with the value a1 to the velocity v1 and then move with constant velocity v1 until the section W1 is completed. Then the drive accelerates with a2 to velocity v2 in the distance section X2. The sections X3 until X5 are as sections X1 and X2. The last section X6 is defined with the acceleration a6, the velocity v6 and the decelerating ramp B.




**Function principle:**
A velocity profile consists of a single or multiple distance sections. Each distance section consists of an accelerating ramp and a section with constant velocity. The last distance section includes additionally the decelerating ramp B. When positioning with velocity profiles a distance section will be defined with the command “WP“. The acceleration A and velocity V may be redefined for each distance section or the actual values set may be used (no redefinition of acceleration and/or velocity value in the new distance section).
Within the definition of a distance section first the command “WP“ must be defined.
The last distance section must be defined with the command “WPT“.
Each distance section must be terminated with “E“.

In case SERS-outputs, marker or any parameters shall be set/defined after terminating a distance section within a velocity profile, (e.g. in the diagram above the outputs O1 and O2), proceed as follows: at first the next coming distance section has to be defined (during still executing the actual = last defined and started distance section), and after that the parameter, output or/and marker to be set after the still running distance section, may be defined.

Terminating the polynom mode without using “WPT” must be done with by using “POS0“ or “POSR“ - see parameter P1043. After an error (e.g. because of a wrong definition of polynom sections) the polynom mode stays active, until it will be terminated by a command.

The example / diagram above with 6 distance sections must be programmed as follows:

```
WP=100 A=1000 V=200 E
WP=120 A=300 V=400 E
O1=1
WP=100 A=1000 V=50 E
WP=100 A=1500 V=700 E
WP=200 A=500 V=100 E
O2=1
WPT=130 A=1000 V=50 E
```

The output O1 will be set after terminating the first distance section (but note: the command O1=1 comes only after the definition of the second distance section ! ).
The output O2 will be set after terminating the fourth distance section (O2=1 is defined after the definition of the fifth distance section !).
The definition of the acceleration within a distance section is optionally. If the acceleration is not defined within the single distance sections, then the last set acceleration
A is used (with exception of the last decelerating ramp → here parameter B is used).

The command "WP" depends on the positioning mode (P1014 – relative or absolute positioning mode) in the same way as the standard positioning command (positioning/distance definition) "W".

Alternatively to WPT it can be used:
**WPA** : distance section absolut (e.g. WPA=1000)
**WPR** : distance section relativ (e.g. WPR=500)

and alternatively to WPT it can be used:
**WPTA** : terminating distance section absolute (e.g. WPTA=1000)
**WPTR** : terminating distance section relative (e.g. WPTR=1000)

If the single polynom distance sections are sent via a serial interface (serial mode with RS232 or Profibus-DP or CANopen interface), then the SERS indicates via parameter P1123 (P1123=1), when the next polynom section must be sent. The SERS needs to receive the next polynom section definition, before the actual polynom section is terminated by the motor, else there will be created an error message and the motor will be stopped !

# 4.6.8 WAIT instruction

A "WAIT" instruction stops the execution of an operational program until the defined event comes true.

Possible events are: I1..I16, IN, POS, M1...M3, O1..O16

e.g.: WAIT I1=1 waits until input I1 will be set.
WAIT I5=0 waits until input I5 will be erased.

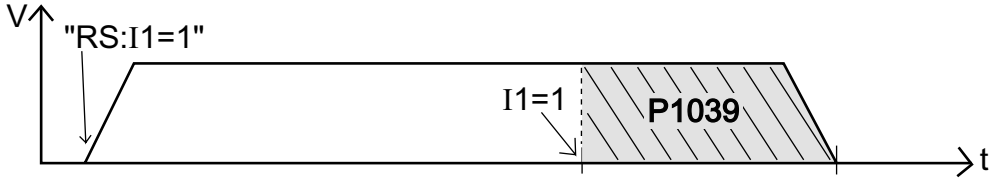
"WAIT" is executed only locally → can not be sent in the master mode to slaves.

# 4.6.9 " : " instruction – jog drive until STOP at an input

Manual driving (jog) with constant velocity until a specified event comes true (then the motor will be decelerated with the jog acceleration value P1018 and then stopped).

e.g. **RS : I1=1** → the motor is driving in jog mode (right slow – with the velocity P1019) until input I1 is set.

Possible commands: RS, LS, RF and LF with the events I1 until I16, IN
If parameter **P1039** <> 0 (a driving distance after stop is defined), the drive will move the relative distance defined in P1039 after the event came true (input is active e.g. at RS:I1=1 or input is deactivated at e.g. RS:I1=0) and then stops. The driving distance after stop (P1039) includes the distance moved during decelerating until motor stop.



# 4.6.10 Arithmetic / Calculating with the SERS

A SERS-E$^2$PROM program may include arithmetical functions. There are following arithmetical functions :

- a 32-bit accumulator **X** - all arithmetical operations have to be assigned to the accumulator
- six 32-bit variables P1080 until P1085, which may be used for any assignments
- Following arithmetical operations exist
  - Adding '**+**' , Subtracting '**–**' , Multiplying '*****' , Dividing '**/**'
  - And '**&**' , Or '|' , Exclusive-Or '^'
  - **NEG** : the accumulator's sign will be inverted
- The operations are executed from left to right (no point before line).
- Arithmetical operations always have to be done via the accumulator X !
- There may be used all parameters and 32-bit constants in arithmetical operations.

Examples:

```
X=V*ADC+100   //The actual selected velocity V is multiplied with the value at the
V=X           //analogue input ADC (0 – 100%) and the value 100 is added

P1081=2
X=V*ADC*3
P1080=X
X=W+10000/P1080*P1081

X=W
NEG    //The accumulator 's sign is inverted (X=–X)
W=X
```

# 4.6.11 Operation with the SERS-Programmer

When controlling a SERS with the SERS-Programmer then the parameter section and the program section can be locked via a **Password**.
Only when the operator of the SERS-Programmer knows the Password, then he is able to see and change all parameters and program lines in the SERS.
Without knowing the password the operator can see and change only **released parameters in the parameter section** and **released program lines with value assignments** (e.g. V=1000 or P1080=5) **in the program section**. These released parameters and value assignments can be changed by the operator via the SERS-Programmer.

The **Password** is defined in the **parameter P1059**.

The parameters to be released are defined via the **parameter masks** (parameters **P1060 until P1065**) and program lines with value assignments to be released have to be defined via the **program masks** (parameters **P1070 until P1074**).
In the program section there can be displayed any text additionally to the released value assignments. To do that the text to be displayed has to be set between quotation marks (e.g. "diameter"). The text has to be placed between the released Label (releasing of program lines with value assignments is done with the help of Labels - see description under P1070 page 68) and the program line with the value assignment to be displayed.

**A detailed description of the operation with the SERS-Programmer can be found in the "manual SERS-Programmer".**

# 4.6.12 Program examples

**Standalone mode:**
Following example demonstrates an endless loop. Because all executed positions/ways are always positive (results in running always into the same direction), the positioning mode must be set to RELATIV_ERASE - (the positioning mode ABSOLUT is possible, doesn't make sense in this case, because the program would position 10 times behind each other to the same position '1000').
The **bold printed characters** are the program, the c*ursive text* behind with the comment characters '// ' are explanations, which are not part of the program.

```
#ON          // switch motor current on
#V=10        // velocity = 10 (e.g. at rotational scaling : 10 U/min)
#W=25.5 E    // way/position = 25.5 and start ('E') P  positioning
#L1          // label 1 - destination for jump with GOTO
#L2          // Label 2
#IF !I2      // if input I2 is not set then execute next instruction
#GOTO2       // program jump to label 2 (loop with waiting for input I2)
#V=1500      // velocity = 1500
#W=2000 E    // way/position = 2000 and start ('E') positioning
#L3          // label 3
#IF IN5      // if inputs I1 and I3 (I1=1 and I3=4, and 1+4=5) are set
#GOTO3       // then program jump to label 3
#P100=10     // set counter P100 to 10 - together with IF P100 (below)→loop with 10
#L4          // label 4
#W=1000 E    // way/position = 1000 and start positioning
#GOSUB10     // call of subroutine starting at label 10
#IF P100     // P100=P100 - 1,  if P100 = 0 then execute next instruction
#GOTO4       // GOTO 4 - loop with 10 runs because of P100=10 and IF P100
#GOTO1       // program jump to label 1(above)
#L10         // label 10
#O2=1        // set output 2
#W=5000 E    // way/position = 5000 and start positioning
#O2=0        // reset (erase) output 2
#RETURN      // end of subroutine - return to instruction following last 'GOSUB'
```

**Parallel mode:**
Each subroutine is preceded by a label and finished with a 'RT' or 'RETURN'.
The instructions may be in one line, or in multiple lines behind other.
A label number corresponds to an address (e.g. L1 = address '1' or L10 = address '10').

```
#L1 ON RT              // program line for address '0' (no address  input is set)
#L2 V=1000 W=2500 E RT // program for address '1' (only I1 is set)
#L3 V=10 W=10.5 E RT   // program for address '2' (only I2 is set)
#L4                    // start of program for address '3' (I1 and I2 set)
#V=20.8
#W=40.6
#E
#RETURN                // end of program for address '3'
```

**Master mode:**
In the following example a master controls two SERS-slaves with the addresses 1 and 2.
The **bold printed characters** are the program, the c*ursive text* behind with the comment characters '// ' are explanations, which are not part of the program.

```
#ON          // switch motor current on
#L1          // label 1
#Z=1         // send following instructions/assignments to drive with address 1
#V=1000      // drive 1 : velocity = 1000
#W=500       // drive 1 : way/position = 500
#L2          // label 2
#IF !I3 GT2  // if input I3 of drive 1 is not active then goto label 2
#E           // drive 1 : start positioning (actual value W of drive 1)
#L3          // label 3
#IF !POS     // if drive 1 is still positioning, then execute next instruction
#GT3         // jump to label 3
#Z=0         // following instruction will be executed from the master
#V=2000      // velocity = 1000
#W=1000 E    // way/position=1000 and start positioning
#Z=2         // send following instructions/assignment to drive with address 2
#RUN 2       // drive 2 : start subroutine at label 2 (in E2Prom program of drive 2!)
#L10         // label 10
#IF P0       // if program of drive 2 is still running, then execute next instruction
#GT10        // jump to label 10
#Z=0         // following instructions will be executed from the master
#V=10        // velocity=10
#W=20 E      // way/position = 20 and start positioning
#GT1         // jump to label 1
```

**Serial mode:**
Bold printed characters between single quotes have to be sent as ASCII-characters via a COM-interface (e.g. COM1 of a PC) to the drive. The Carriage-Return-character is written as ‘#13‘ – for the ASCII-Code 13 of the character.
(Note: The SERS returns a 'ok' after each Carriage-Return).
Explanations and instructions printed in cursive style have to be programmed in the used programming language (e.g. PASCAL or C).

```
‘#1‘ #13          // addressing of the drive – here drive address is '1'
‘#ON‘ #13         // switch on current of motor
‘#V=1000‘ #13     // velocity = 1000 – unit depends on parameter P44
‘#W=2500 E‘ #13   // way to be executed = 2500 – the way really executed depends
                     on scaling (P76), positioning mode (P1014),
                     gear ratio and feed constant (P120 – P122)
                     – 'E' starts the positioning job
‘#P336?‘ #13      // request if P336=1 (POS=1) or P336=0 – drive finished
                  // positioning or not - the SERS will return a ‘1‘ or ‘0‘
‘#P1300?‘ #13     // request of the digital input port – the SERS returns a value
                     between ‘0‘ and ‘255‘. Set and non set inputs can be filtered
                     out of the value, and depending on the result further actions
                     can be started.
```

# 4.7 Parameter overview

**Changing parameters in the SERS**
Parameters are programmed with following syntax : **#P*datum*=value**
- the parameter identifier ***datum*** and the new parameter value ***value***.

**e.g.: #P138=1500**

Please note: The drive, where a parameter is to be changed, must be addressed first - meaning the address of the drive has to be sent once to the drive before sending a parameter assignment to the drive (e.g.: for a drive with the address 1 : #1 P138=1500).
With a following Carriage Return the new value will be accepted by the SERS and in case of no error the SERS will return a 'OK'.
If the parameter value is not valid, the SERS returns an error message.

**Reading actual parameters out of the SERS**
A parameter can be read with the instruction : **#P*datum*?**
- with the parameter identifier *datum* , e.g.: **#P138?**

The SERS returns the parameter value (in case of parameters values with units like e.g. acceleration the unit will be returned together with the value)
In the example above e.g. '1500 rad/s2' will be returned.

After sending '**#list p'** to the SERS, a list of all parameters and their values will be returned.

**Reading parameter identifications out of the SERS**
A parameter identification can be read by sending the instruction : **#P*datum*??**
- with the parameter number *datum* , e.g.: **#P138??**

The SERS returns the parameter identification, e.g.: '**acceleration**'

**List of all parameters:**


| Param.Nr. | page | Param.Nr | page | Param.Nr | page | Param.Nr | page | Param.Nr | page |
|---|---|---|---|---|---|---|---|---|---|
| P0 | 45 | P1 - **Z** | 58 | P2 | 58 | P11 | 46 | P12 | 46 |
| P41 | 51 | P42 | 52 | P44 | 54 | P47 - **W** | 48 | P51 | 49 |
| P76 | 54 | P91 - **V** | 51 | P100 - **C1** | 57 | P101 - **C2** | 57 | P102 - **C3** | 57 |
| P103 | 49 | P108 | 51 | P121 | 55 | P122 | 55 | P123 | 55 |
| P134 | 46 | P138 - **A** | 52 | P147 | 53 | P160 | 54 | P265 | 66 |
| P336 - **POS** | 47 | P403 | 47 | P1001 | 50 | P1002 | 63 | P1003 | 51 |
| P1004 | 66 | P1005 | 52 | P1006 | 52 | P1007 | 52 | P1008 | 52 |
| P1009 | 59 | P1010 | 59 | P1011 | 60 | P1012 | 59 | P1013 | 47 |
| P1014 | 48 | P1015 | 47 | P1016 | 47 | P1017 | 43 | P1018 | 52 |
| P1019 | 51 | P1020 | 51 | P1021 | 43 | P1022 | 44 | P1023 | 44 |
| P1024 | 45 | P1025 | 45 | P1026 | 45 | P1027 | 45 | P1028 | 58 |
| P1029 | 62 | P1030 | 52 | P1031 | 61 | P1032 | 52 | P1033 | 60 |
| P1034 | 60 | P1035 | 61 | P1036 | 44 | P1037 | 55 | P1038 | |
| P1039 | 50 | P1040 | 66 | P1041 | 66 | P1042 - **LP** | 47 | P1043 | 50 |
| P1044 | 65 | P1045 | 65 | P1046 - **ADC** | 44 | P1047 - **X** | 67 | P1050 | 61 |
| P1051 - **WR** | 48 | P1052 - **WA** | 48 | P1053 | 62 | P1054 | 63 | P1055 | 63 |
| P1056 | 44 | P1057 | 63 | P1058 | 51 | P1059 | 67 | P1060-P1068 | 68 |
| P1070-P1073 | 69 | P1080–P1085 | 67 | **R0** - **R1** | 67 | P1092 | 67 | P1093 | 61 |
| P1094 | 65 | P1095 | 61 | P1096 | 52 | P1097 | 71 | P1098 | 44 |
| P1099 | 71 | P1100 - **D** | 57 | P1101 - **M1** | 57 | P1102 - **M2** | 57 | P1103 - **M3** | 57 |
| P1110 | 58 | P1111-P1116 | 48 | **WP** - **WAPT** | 48 | P1117 | 66 | P1118 | 60 |
| P1119 | 65 | P1120 | 65 | P1121 | 47 | P1122 | 66 | P1123 | 47 |
| P1124 | 65 | P1125 | 45 | P1126 | 45 | P1201-P1204 | 44 | P1205-P1216 | 44 |
| P1300 / **I1**-**I8** | 43 | P1301 / **I9**-**I16** | 43 | | | **O1** - **O4** | 44 | **O5** - **O16** | 44 |
| | | | | | | | | | |

page

**4.8 SERS Parameter**
**4.8.1 Interface - I/O**
**4.8.1.1 Handshake mode**..... 43
**4.8.1.2 Inputs**..... 43
**4.8.1.3 Outputs**..... 44
**4.8.2 System parameters**
**4.8.2.1 Mode - programming/operation**..... 45
**4.8.2.2 Power control - drive ON/OFF**..... 46
**4.8.3 Status messages**
**4.8.3.1 Drive errors and warnings**..... 46
**4.8.3.2 Drive status messages**..... 47
**4.8.4 Operating parameters**
**4.8.4.1 Positioning mode**..... 48
**4.8.4.2 Way-distance / position dates**..... 48
**4.8.4.3 Velocity dates**..... 51
**4.8.4.4 Acceleration dates**..... 52
**4.8.4.5 Homing**..... 53
**4.8.4.6 Scaling of operating parameters**..... 54
**4.8.4.7 Mechanics - gear, feeding constant, backlash**..... 55
**4.8.5 Programming - parameter**
**4.8.5.1 Time Delay**..... 57
**4.8.5.2 Counter**..... 57
**4.8.5.3 Marker**..... 57
**4.8.5.4 List option**..... 58
**4.8.5.5 Special master-mode parameters**..... 58
**4.8.6 Diverse parameters**
**4.8.6.1 Phase current adjustment**..... 59
**4.8.6.2 Program continue after stop**..... 60
**4.8.6.3 Manual drive control commands**..... 61
**4.8.6.4 Manual drive control with timeout**..... 61
**4.8.6.5 Drive address**..... 61
**4.8.6.6 Control of load angle**..... 62
**4.8.6.7 Control of motion**..... 65
**4.8.6.8 $E^2$Prom parameters**..... 66
**4.8.6.9 Language selection**..... 66
**4.8.6.10 Software limit switch**..... 66
**4.8.6.11 Arithmetic parameters**..... 67
**4.8.6.12 Service switch external**..... 67
**4.8.6.13 Program-/parameter-mask and password For SERS-Programmer**..... 67
**4.8.6.14 CANopen Parameter**..... 71

# 4.8 SERS Parameter
# 4.8.1 Interface I/O
# 4.8.1.1 Handshake mode

### Handshake mode **P1017**

0 : V24-handshake mode - the pins 2,3,5,7,8 at the 9-pole
serial PC-COM-connection must be connected
- hardware-handshake (using of pins 7 and 8)
- each received character will be returned 1:1

1 : no hardware handshake - the pins 2,3,5 at the 9-pole
serial PC-COM-connection must be connected
- each received character will be returned 1:1

2 : RS485-handshake - no hardware-handshake
- received characters are not returned
- error messages are sent only after a received end of line (Carriage Return)
(see also page 24 - syntax general notes)

# 4.8.1.2 Inputs

### Inputs **P1300**

P1300 contents the status of the digital Inputs I1 to I8

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| Set Input | : | I1 | I2 | I3 | I4 | I5 | I6 | I7 | I8 |
| Parameter value: | : | 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 |

E.g.: in case of P1300 = 11 , then the inputs I1, I2 and I4 are set (1+2+8=11)

I1 - I8 and IN0 - IN255 can be used for IF-requests
see page 32 - chapter 4.6.1

### Inputs I/O-extension **P1301**

P1301 contents the status of the digital inputs I9 until I16
→ valid values only in case of installed option “IO“ – SERS with I/O-extension

### DigIn-enable **P1021**

P1021 specifies, which manual drive function is available at the digital inputs in case of switched off service switch (see page 15 - chapter 3.1.4)
Assignment parameter-value - Inputs - drive-function

| Value (dec) | Input | Function |
|---|---|---|
| 1 | I1 | Left – slow |
| 2 | I2 | Right – slow |
| 4 | I3 | Left – fast |
| 8 | I4 | Right – fast |
| 16 | I5 | Start homing |
| 32 | I6 | Motor ON/OFF , additionally P11=0 and P12=0 at motor → ON |
| 84 | I7 | Stop |
| 128 | I8 | Start program |

e.g.: #P1021=3 → manual drive slow right and left at the inputs
I1 and I2 is always active. (P1021=255 → all functions activated)

### Limit switches and Digin **P1056**

Parameter P1056 is only for internal use.

### DigIn-label-enable **P1022**

P1022 specifies, which inputs may be used as address inputs for the parallel-mode.

E.g.: #P1022=15 → inputs I1 - I4 will be used as address-inputs in the parallel mode.

Attention : If a input is enabled also in P021 then it is not available in the parallel mode ! (Attention P1021 must content value 128)

### Enable program jump **P1098**

Enable program jump in case of a signal at a digital input
→ see chapter 4.5.2 “event controlled mode“

### Start-enable **P1023**

0 : Program is started with a positive signal edge at the start input I8
1 : Program is started , when the start input I8 is set (static signal)

### Analogue-Input **P1046**

Alternative Syntax : **ADC**
Standard 8-bit resolution with 1bit hysteresis (the value at the input has to change at least 2 LSB, for changing the actual value in P1046 (ADC)
The analogue input is scanned and updated every time when being assigned or used in a formula.
E.g.: X=ADC or X=V*ADC result in reading the analogue input.

## 4.8.1.3 Outputs

### Outputs **P1201**

P1201 - output 1 - alternative syntax O1 until
P1202 - output 2 - alternative syntax O2 **P1204**
P1203 - output 3 - alternative syntax O3
P1204 - output 4 - alternative syntax O4

#P1201=1 or #O1=1 sets the output O1
#P1201=0 or #O1=0 resets the output O1

### Outputs I/O-extension **P1205 (O5)**

Only in case of SERS option “IO“ - I/O-extension: until
P1205 - output 5 (O5) until P1216 - output 16 (O16) **P1216 (O16)**

### Break output **P1036**

The Ready-signal (usable as signal for activating a break in case of a driver error or motor error or in case of power off e.g. for holding of a z-axis) can be mapped to a digital output by writing

1..4 : switching on output O1..O4 in case of an error
-1..-4 : switching off output O1..O4 in case of an error
0 : function is disabled

### Compare position 1 P1024

If the position, defined in parameter P1024, is reached , then
the output or marker, defined in P1025, will be set or reset
see P1025

### Compare output 1 P1025

When actual-position (P51) = P1024 and P1025=
0 → no action
1 to 4 → corresponding output will be set (e.g.: P1025=3 → O3 will be set)
-1 to -4 → corresponding output will be reset
5 to 7 → marker 1 to 3 will be set (5 = marker 1 , 6 = marker 2 , 7 = marker 3)
-5 to -7 → marker 1 to 3 will be reset (e.g.: P1025=-6 → marker 2 will be reset)

### Compare position 2 P1026

see P1024

### Compare output 2 P1027

See P1025

### Output "Motor IN-Position" (P336) P1125

P1125 may be configured as follows:
1..4 : output O1..O4 =1 if P336=1 (SERS IN-Position) , O1..O4 =0 if P336=0
-1..-4 : O1..O4 =0 if P336=1 (SERS IN-Position) , O1..O4 =1 if P336=0
0 : function disabled
→ e.g.: P1125=2 → when motor is running (P336=0), then O2=0, at standstill → O2=1

### Output "program active" (P0) P1126

P1126 may be configured as follows:
1..4 : output O1..O4 =1 if P0=RUN (program is running) , O1..O4=0 if P0=0
-1..-4 : O1..O4 =0 if P0=RUN (program is running) , O1..O4=1 if P0=0
0 : the function is disabled
→ e.g: P1126=1 → if a program is running (P0=RUN), then O1=1, else → O1=0

# 4.8.2 System parameters

# 4.8.2.1 Mode programming/operation

### $E^2$PROM Modus P0

0 : The $E^2$Prom - program is waiting for an action (e.g. a command via the serial interface or a start command)
If the programming mode is active, then it will be terminated with P0=0
(alternative syntax : **quit**)

1 : the assignment P0=1 starts the $E^2$Prom-program during operation ($E^2$Prom-program is running), parameter P0 = 1
(alternative syntax : **run**)

2 : P0=2 starts the programming mode - the actual $E^2$Prom-program will be erased
all following instructions (except 'list', 'quit', 'pgm', 'trace') will be stored into the $E^2$Prom
After each Carriage Return the SERS returns a 'pgm'
(alternative syntax : **new**)
The instruction **pgm** also enters the programming mode, but the actual $E^2$Prom-program will not be erased, but can be edited.

# 4.8.2.2 Power control - drive ON/OFF

## Master-control word P134

0 : Switch motor phase current off (alternative syntax : **OFF**)
7 : Switch motor phase current on (alternative Syntax : **ON**)

# 4.8.3 Status messages

# 4.8.3.1 Drive error and warnings

## Drive error P11

In case of an error in the SERS, parameter P11 contents the corresponding errors (each error will set a certain bit in P11).
A drive error always results in decelerating of the drive (with the value defined in parameter P1030). When the motor stopped, the motor phase current will be switched off and afterwards the ready signal will be reset (see page 9 - relay will open). Via parameter P1036 there may be defined that an output (O1-O4) will be set in case of an error.

**Reset of drive error messages**
After an error occurred, the motor phase current may be switched on and the drive can be moved only if there is no error anymore and the error message was reset with the instruction **P11=0 .**
In case of an error ´short circuit power amplifier or motor´ (P11=128) before writing ´P11=0´ the reset button at the front panel (chapter 2.1.5) of the SERS must be pressed.

**Assignment error number - drive error:**
2 : error over temperature - the heat sink temperature is over 85 °C (+/- 10°C)
32 : error control of load angle – see P1029 (only with installed option control of load angle)
128 : short circuit at amplifier board or motor
512 : error undervoltage - the power supply is to low (see page 10)
2048 : error motion control (see P1044)
8192 : error position limit control - drive is on limit switch (input limit switch is open)

A value 514 has the signification : error over temperature (2) and undervoltage (512) occurred.

## Drive warnings P12

If a warnings occurs, it will be indicated in P12. After the reason for the warning disappeared, P12 still contents the warning (reset P12 with the assignment P12=0). (except values 2 and 4 – these warnings are reset automatically, when the reason for the warning disappeared)

**Assignment parameter values - drive warnings:**
1 : Warning limit position – P51< P1040 or P51>P1041 - software limit switch reached.
2 : Warning over temperature - the heat sink temperature exceeds 75 °C (+/- 10°C)
4 : Warning under voltage - the voltage supply is to low (see page 10)
8 : Internal position limit - the internally calculated position value can not be imaged to the scaled value.
16 : An error message was generated - e.g. "parameter value to big" (after trying to assign a parameter with a not valid - to big - value), or "destination does not exist", when trying to jump to a not existing label in an executable program in the $E^2$Prom
32 : Error control of load angle – see P1029 (only with installed option control of load angle)
64 : Position limit command value - the set position command value is too big
128 : Program error – the operational program was ended due to an error in the program

# 4.8.3.2 Drive Status messages

### Position status - In_Position P336 (POS)

Alternative syntax : POS
While the drive is running → P336 = 0 else → P336 = 1

### Independently messaging In_Position P1121

0 : disabled
1 : after terminating a positioning job the string "@nPOS=1" will be sent, where n is the drive address (e.g. SERS with address 2 → message "@2POS=1"

### Position feedback value status P403

If the position feedback value is valid (after Power-On of the SERS there was at least once a successful homing procedure), then → P403 = 0
After a Power-On (and still no successful homing procedure) → P403 = 3

### Accelerating phase P1015

While the drive is accelerating → P1015=1

### Constant phase P1016

While the drive is running with constant velocity → P1016=1

### Drive status and limit switch P1013

word (2 byte) – only read parameter
Consists of the driver status and the status of the limit switch inputs
Set (active) bit :

- 0 (decimal value 1) : limit switch right open (drive error)
- 1 (decimal value 2) : limit switch left open (drive error)
- 2 (decimal value 4) : STOP-switch open (drive stops)
- 3 (decimal value 8) : home switch open (drive is not on home switch position)
- 7 (decimal value 128) : service-switch open (inactive)

All other bits are used internally.
For checking e.g. the limit switch left:
AND-function of P1013 and the decimal value of the bit to be checked
→ P1013 & 2 = "2", if the bit is active, else the result of the AND-function is "0"

### Limit position exceeded P1042 (LP)

Alternative syntax: **LP** - see also description P1040/P1041 page 65
If P51 < P1040 or P51 > P1041 then P1042=1 else P1042=0

### Load next polynom section P1123

The SERS expects the definition of the next polynom section, when P1123=1.
Necessary for polynom driving in the serial mode → see chapter 4.6.7 (polynom driving)

# 4.8.4 Operating parameter

## 4.8.4.1 Positioning mode

### Positioning mode P1014

0 : RELATIVE - the new position command value is the actual position command value plus the value stored in W (P47). The drive is running the distance stored in W (P47) after receiving the instruction 'E'.

1 : RELATIVE ERASE - After receiving the instruction ('E') the drive is running the distance stored in W (P47) and P51 (actual position) is set to zero before starting. This mode is used for endless positioning. **Also pay attention to P103.**

2 : ABSOLUTE - An instruction 'E' starts positioning to the absolute position stored in W (P47).
After finishing positioning the position value (P51) = W (P47) !

3 : RELATIVE_WITH_OVERFLOW_COUNTER
The 16 Bit counter P100 (C1) is used as overflow counter.
The position with overflow is: X=C1 *P103 +P51
So the max. relative position is +/- 5.8982 $*10^9$ Grad (rotational)
respectively +/- 4.7186 $*10^9$ mm (linear).
→ but the max. value for a single positioning job is the max. value of P103 (via the polynom driving mode there may be positioned larger distances).

## 4.8.4.2 Way/distance/position dates

### Position command value P47 (W)

Alternative syntax : **W**
The really executed motor movement depends on the positioning mode (P1014), on the scaling of the position dates (P76) and on the parameters P121 - P123.
The value ranges are in case of 1:1 gear ratio and feeding constant = 1 (P121=P122=P123=1)
Value range : Rotational : -214748.3647 to 214748.3647 (°)
Linear : -167772.1599 to 167772.1599 (mm)
Incremental : --2147483639 to 2147483639 (increments/steps)
Starting a positioning procedure is done by the command ´**E**´ - e.g. ´#W=360 E´

### New relative position (position command value - relative) P1051 (WR)

Alternative syntax : **WR**
Description as **W** (P47) but additionally the positioning mode is set to RELATIVE (P1014=0)

### New absolute position (position command value - absolute) P1052 (WA)

Alternative syntax : **WA**
Description as **W** (P47) but additionally the positioning mode is set to ABSOLUTE (P1014=2)

### Position command value Polynom P1111 (WP)

Alternative syntax : **WP**
Description as "**W**" (P47), but definition of distance section at polynom driving (see chapter 4.6.7)

## Position command value relative Polynom P1112 (WRP)

Alternative syntax : **WRP**
Description as "**WR**", but definition of distance section at polynom driving
(see chapter 4.6.7)

## Position command value absolute Polynom P1113 (WAP)

Alternative syntax : **WAP**
Description as "**WA**", but definition of distance section at polynom driving
(see chapter 4.6.7)

## Position command value Polynom termination P1114 (WPT)

Alternative syntax : **WPT**
Description as "**W**", but definition of last distance section at polynom driving
(see chapter 4.6.7)

## Position command value relative Polynom termination P1115 (WRPT)

Alternative syntax : **WRPT**
Description as "**WR**", but definition of last distance section at polynom driving
(see chapter 4.6.7)

## Position command value absolute Polynom termination P1116 (WAPT)

Alternative syntax : **WAPT**
Description as "**WA**", but definition of last distance section at polynom driving
(see chapter 4.6.7)

## Position value P51

Stores the actual position of the drive.
P51 may be assigned with any value (e.g. for setting a zero-position).
After Power-On of the SERS P51 is set to the value '0', because P51 is not saved automatically into the $E^2$Prom of the SERS.
With the command **POSSAVE** the actual value in P51 cab be saved into the $E^2$Prom.
After the next Power-On of the SERS, P51 is set to this saved value.

## Modulo value for position command value P103

Important parameter for an "endless"-operation at positioning mode RELATIV ERASE.
**P103 must be bigger than the biggest single distance to be executed !**
Also **P103 must be an endless to be scaled value**, that means P103 must be able to be executed without a rest distance (corresponding to the physically resolution of the drive of 12800 Increments/revolution).
Suitable values at a gear ratio of 1:1 and a feeding constant =1 are e.g. 360 and multiples of 360 at rotational scaling, 12800 and multiples of 12800 at incremental scaling, or 1 and multiples of 1 at linear scaling.
Different gear ratios and feeding constants must be taken into account,
e.g.: linear scaling, gear ratio 2:1 and feeding constant = 5

→ 2 motor revolutions are 5 mm → 1 rev = 2,5 mm
→ because 1 motor revolution always can be executed without rest and 2.5 mm = 1 rev.
→ suitable values for P103 : Multiples of 2.5mm e.g. 2500 mm (selected value must be greater than the greatest to be executed single distance !)

## Erase phase position / position correction P1043

0 : no function

1 : alternative syntax: **POS0**

When the actual position was stored with the command **POSSAVE** and the real physical position of the drive was changed, e.g. after switching the drive off (what results in shutting the phase current off and because off that losing torque), then the phase-zero-position can be reset to its origin only with the command POS0 (equal to P1043=1).

At the phase-zero-position the current in phase 1 of the stepping motor is set to positive 100% and the current in phase 2 is set to 0%. If the POSSAVE-command is never used, then after Power-On of the SERS the motor is always set to a phase-zero-position and P51=0°.

With 'POS0' the drive first runs from the actual set phase-position (e.g. after a POSSAVE-command P51=361,8°) to the next phase-zero-position - to 360° in the example - and afterwards P51 is set to 0° (P51=0°).

POS0 reset also the polynom mode (started by a WP command).

2 : alternative Syntax : **POSR**

Motor is moved into position command value P47 with homing velocity P41 – command can be used only after an error P11=32 or a warning P12=32 (presupposition is a motor with encoder, SERS with option E50, P1029=1 or P1029=2, and P1053=8).

After the motor stopped because of e.g. a mechanical overload (and the max. load angle was exceeded) the real motor position is detected by the encoder signals. The difference between motor position and position command value will be corrected by the command **POSR**.

POSR reset also the polynom mode (started by a WP command).

3 : Command P1043=3 will execute POSR also without an existing error P11=32 or a warning P12=32.

## Leave Zero Phase P1001

P1001=0 : normal operation

P1001=1 : Phase positions, where one of the phases carries a very low current, are left.

When reaching a position, which is multiple of 1,8° or is within 0,1° after or before (e.g 0°, 1.74° , 1.8°, 1.85° , 3.57° , 3.6°, 3.69° ...) , the drive moves on or returns until reaching a position which is 0,1° after or before a multiple of 1,8° (e.g. position command value is 3.6° → motor moves until 3.7° or position command value is 5.35° → motor moves until 5.3°)

This is for evitating a possible noise of the chopper at the motor at motor stand still.

P1001=2 : move 0.2 degree away from zero phase

P1001=3 : move 0.3 degree away from zero phase

## Distance after stop P1039

The distance after Stop is activated in case it is not Zero (P1039<>0).

Scaled value (according to P76)

Relative distance into the same direction with identical velocity and jog acceleration. The value P1039 is the relative distance executed after the Stop event of a " : " - command.

e.g.: RS:I1=1 (see chapter 4.6.9).

A set backlash (P1037<>0) will be ignored in case of a set P1039 (P1039<>0).

During executing the distance P1039 the parameter P108 (Feedrate Override) is ignored.

**Save position in case of under voltage** **P1058**

P1058=0 : no function

P1058=1 : If the supply voltage is lower then a minimum value - $U_B$ (see page 10) - (→ error under voltage in P11) then the command **POSSAVE** (see P51) is executed. Presupposition is, that the supply voltage falls down slowly (sufficient big electrolytical capacitors in the AC/DC power supply) >10ms between $U_B$ and $U_L$ (see page 10)

# 4.8.4.3 Velocity dates

All velocity values depend on the scaling P44

**Velocity - Positioning** **P91 (V)**

Alternative syntax : **V**

Maximum selectable value at rotational scaling (U/min) : 10000

**Velocity manual slow** **P1019**

**Velocity manual fast** **P1020**

**Velocity homing** **P41**

Velocity after start homing, if the homing switch input is not set.

**Velocity homing slow** **P1003**

Velocity during homing, when the homing switch input is set.
Also used for velocity for backlash-function (see P1037)

**Feedrate override** **P108**

Possible values : 0 - 100 (unit is %)
The homing velocity P41 and the jog velocities P1019 and P1020 will be multiplied with this value.

# 4.8.4.4 Acceleration dates

All accelerating values depend on the scaling (P160)

### Acceleration - positioning P138

Alternative syntax : **A**
Maximum selectable value at rotational scaling : 100000 rad/s$^2$

### Acceleration - manual drive P1018

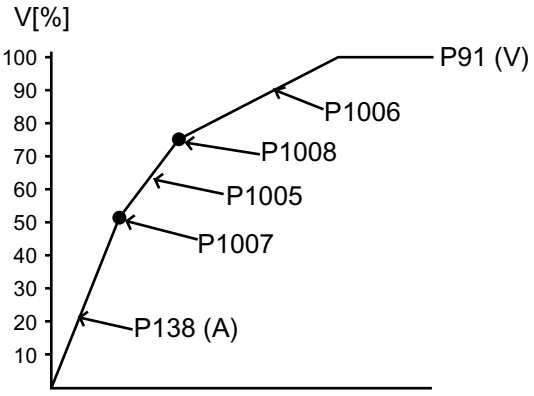
### Acceleration - homing P42

### Acceleration at drive error P1030

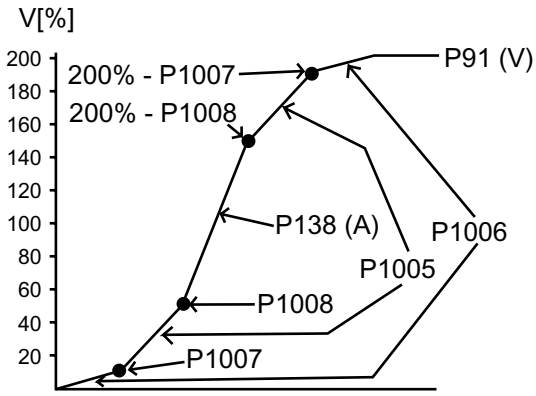
When an error occurs (see P11) or the Stop input is set, then the drive will decelerate with the value defined in P1030.
The value must be as great as possible, but only as great as the stepping motor will not loose it synchronous running at worst conditions.

### Ramp form of acceleration P1032

0 : exponential
1 : sinus
(see drawing below)




**Exponential acceleration**




**Sinus Acceleration**

### Acceleration section 1 P1005

Indication in % of P138 (A) , value range : 0 to 100

### Velocity section 1 P1007

Indication in % of P91 (V) , value range : 0 to 100

### Acceleration section 2 P1006

Indication in % of P138 (A) , value range : 0 to 100

### Velocity section 2 P1008

Indication in % of P91 (V) , value range : 0 to 100

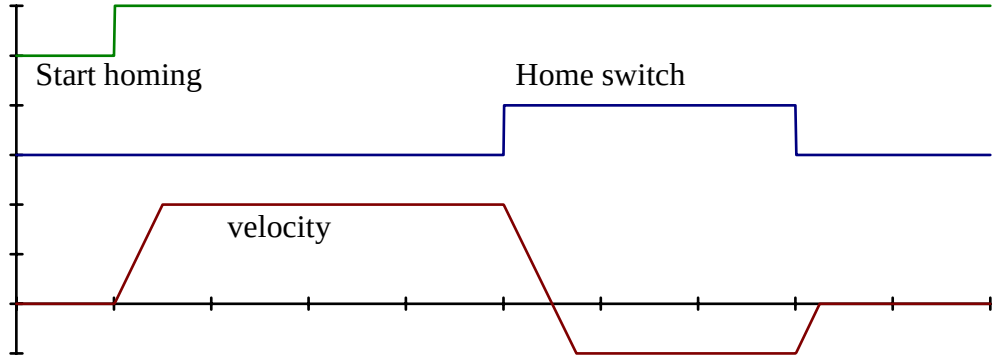
### Ramp down / Deceleration at end of polynom P1096 (B)

Alternative syntax : **B**
Max. adjustable value in case of rotational scaling : 100000 rad/s$^2$

# 4.8.4.5 Homing parameter

## Start Homing

When receiving the command **H** or in case of an active signal at the digital input I5 (depends on P1021 and the service switch) the drive starts running to the home switch. The drive accelerates with the "acceleration-homing“ (P42) to the velocity "velocity-homing“ (P41). At a positive edge of the home switch the drive decelerates until motor stop. Then the drive runs with the "velocity-homing slow“ (P3) into the opposite direction, until reaching the negative edge of the home switch.
In case of start homing when the drive is on the reference switch, then the drive runs with "velocity homing slow" until reaching a negative edge of the home switch.
The homing direction is defined in P147.




## Homing parameter - motor direction P147

Bit 0 (1 decimal) : Homing - direction :
- 1 = negative
- 0 = positive

Bit 1 (2 decimal) : Homing mode :
- 1 = homing to limit switch
- 0 = homing to homing switch

Bit 2 (4 decimal) : Reset position :
- 1 = homing to switch and afterwards moving on to the next electr. reset-position (every 7,2° at motor - in case of 1,8° stepping motor)
- 0 = homing to switch only

Bit 3 (8 decimal) : Controlling software limit switch :
- 0 = ignore P1040 und P1041 when homing
- 1 = do not ignore P1040 und P1041 when homing

E.g.: P147=4 specifies homing in positive direction to home switch and afterwards stop at next electrical reset position (every 7,2°).
P147=3 means homing in negative direction to limit switch and afterwards stop immediately.

### Homing to limit switch - P147 Bit 1 set :

During homing the function of the limit switches is identical to a home switch.
If a limit switch openes during homing (no 24VDC anymore at limit switch input), then the motor decelerates and returns into the opposite direction until there are 24 VDC at the input again (in case of running over a limit switch the motor returns and moves until reaching a positive signal edge at the limit switch input).

# 4.8.4.6 Scaling of operating parameters

Following scaling modes are possible :
**Incremental** - unit [increments] - 12800 increments = 1 motor revolution
**Linear** - unit [mm] - 1mm = 1 motor revolution
**Rotational** - unit [ ° ] - 360° = 1 motor revolution

In case of the scaling modes rotational and incremental the reference may be the load or the motor shaft. In case of a reference load the parameters P121 and P122 (gear ratios) will be taken into account when calculating the operating dates.

In case of the scaling mode linear the parameters P121, P122 and P123 (feeding-constant) will be taken into account always when calculating the operating dates !

Most of the scaled values include decimal places. The number of decimal places varies - e.g. in case of linear position datas $\rightarrow$ 4 decimal places.
The value 120mm is stored as 120.**0000** mm. When writing values in ASCII-format (SERS with RS232-interface) only the value ´120´ must be written.
In case of option Profibus-interface (SERS.. PB-DP and binary mode) or CANopen (SERS ... CAN) the value ´1200000´ must be written for '120' mm (includes the 4 decimal places)

## Scaling position dates **P76**

| Bit | 76543210 | decimal | unit | scaling mode | reference | decimal places |
|---|---|---|---|---|---|---|
| | 00000000 | 0 | incr | Incremental | Motor | 0 |
| | 01000000 | 64 | incr | Incremental | Load | 0 |
| | 00000010 | 2 | ° | Rotational | Motor | 4 |
| | 01000010 | 66 | ° | Rotational | Load | 4 |
| | 00000001 | 1 | mm | Linear | Load | 4 |
| | 00010001 | 17 | in | Linear | Load | 6 |

Pre-selected is 2 (° – rotational motor)

## Scaling Velocity dates **P44**

| Bit | 76543210 | decimal | unit | scaling mode | reference | decimal places |
|---|---|---|---|---|---|---|
| | 00000000 | 0 | U/min | Incremental | Motor | 4 |
| | 01000000 | 64 | U/min | Incremental | Load | 4 |
| | 00000010 | 2 | U/min | Rotational | Motor | 4 |
| | 01000010 | 66 | U/min | Rotational | Load | 4 |
| | 00000001 | 1 | mm/min | Linear | Load | 3 |
| | 00010001 | 17 | in/min | Linear | Load | 5 |

Pre-selected is 2 (U/min – rotational motor)

## Scaling Acceleration dates **P160**

| Bit | 76543210 | decimal | unit | scaling mode | reference | decimal places |
|---|---|---|---|---|---|---|
| | 00000000 | 0 | $rad/s^2$ | Incremental | Motor | 3 |
| | 01000000 | 64 | $rad/s^2$ | Incremental | Load | 3 |
| | 00000010 | 2 | $rad/s^2$ | Rotational | Motor | 3 |
| | 01000010 | 66 | $rad/s^2$ | Rotational | Load | 3 |
| | 00000001 | 1 | $mm/ s^2$ | Linear | Load | 3 |
| | 00010001 | 17 | $in/ s^2$ | Linear | Load | 5 |

Pre-selected is 2 ($rad/s^2$ – rotational motor)

# 4.8.4.7 Mechanics - gear ratio, feeding constant

## Feeding constant P123

The unit depends on P44 (scaling position dates)
The feeding constant transforms a rotational movement into a linear movement (e.g. spindle)
The feeding constant is equivalent to the made linear distance at one revolution (of the spindle).
Pre-selected value: 1

**Example.: Spindle with lead of 5 (=5mm per revolution) → P123 = 5**

## Gear ratio:

## Gear - input revolutions P121

Input revolutions at the first stage of all gears
between the motor shaft and the load
- seen at the motor shaft
Pre-selected value: 1

## Gear - output revolutions P122

Output revolutions at the last stage of all gears
between the motor shaft and the load
- seen at the load
Pre-selected value: 1

$$\text{Gear ratio} = \frac{\text{Gear input revolutions}}{\text{Gear output revolutions}}$$

**Example.: gear 8:1 → P121=8 and P122=1**

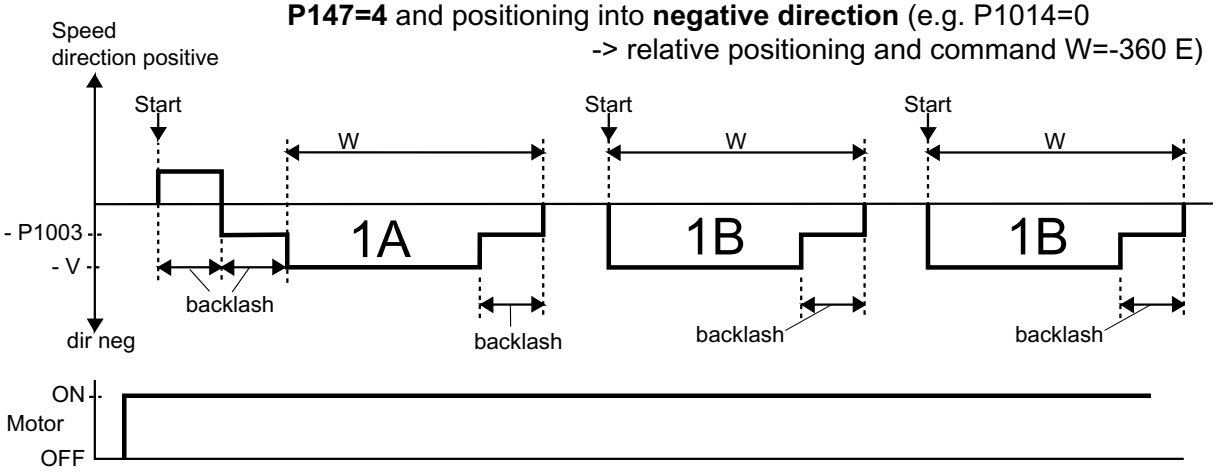
## Backlash P1037

Backlash correction - a spindle or a chain includes a backlash. By writing the backlash value (value depends on scaling for position dates P76 - e.g. 0,1° in case of rotational scaling) into parameter P1037, the backlash when reversing direction or rebounding after positioning with high acceleration will be corrected (when executing positioning jobs). A value ‘0’ in P1037 disables the backlash function.
The backlash correction-function depends on the running direction. Parameter P147 (homing parameter - homing direction) defines the 'backlash direction'.

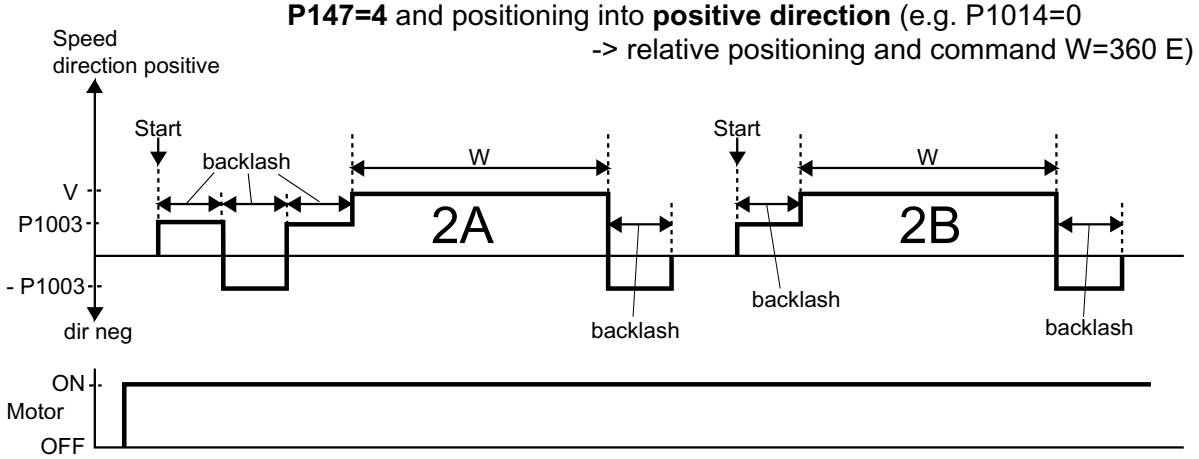
- when positioning into positive direction (and P147=4 homing into negative) the motor first runs the distance 'W minus backlash' with velocity 'V' into positive direction and afterwards the distance 'backlash' with velocity P1003 into the same (positive) direction.
- when changing direction (positioning job into negative direction), the drive first runs the backlash distance stored in P1037 with velocity P1003 into the positioning direction (negative direction). Then it executes the distance 'W' with the velocity 'V' into the same (negative) direction. At last the drives returns to the opposite direction (positive direction) and runs the backlash distance with the P1003-velocity.

## SERS Backlash-function
## Timing Diagram when executing a positioning job (command "E")

**P147=4** and positioning into **negative direction** (e.g. P1014=0
-> relative positioning and command W=-360 E)




**P147=4** and positioning into **positive direction** (e.g. P1014=0
-> relative positioning and command W=360 E)




**P147=5** and positioning into **positive direction** : running characteristics **1A** and **1B**

**P147=5** and positioning into **negative direction** : running characteristics **2A** and **2B**

## Backlash function when positioning and before executed manual drive command:

Depending on the executed manual drive function and the covered distance at manual driving (in relation to the backlash distance), the running characteristics 1A or 2A (depending on the direction and P147) or a different running characteristics is executed (e.g. positioning to the position command value only with backlash-velocity, if the covered distance with manual driving is smaller than the backlash value).

# 4.8.5 Parameters for programming

## 4.8.5.1 Time delay

**Time delay** **P1100 (D)**
Alternative syntax : D
Indication in $^{1}/_{10}$ seconds = 100ms (D=1)
Minimum programmable delay: 10ms (D=0.1)
Example.: **D=20** → 2 seconds delay or **D=0.5** → 50ms delay

## 4.8.5.2 Counter

**Counter 1 –** unsigned 16 bit counter (0 until 65536) **P100 (C1)**
Alternative syntax : **C1**
In case P1014=3 (relative positioning with overflow counter), then C1 is signed (-32768 until 32768) and it will be used as overflow counter.

**Counter 2 –** unsigned 16 bit counter (0 until 65536) **P101 (C2)**
Alternative syntax : **C2**

**Counter 3 -** unsigned 32 bit counter (0 until 4,295 x $10^9$) **P102 (C3)**
Alternative syntax : **C3**

Counters together with IF-requests can be used for programming loops.

E.g.: **#C1=20**
#L1
*Instruction block*
#IF C1>1
#GOTO 1

In the example the *instruction block* will be executed 20 times.

## 4.8.5.3 Marker

**Marker 1** **P1101 (M1)**
Alternative Syntax : **M1**

**Marker 2** **P1102 (M2)**
Alternative Syntax : **M1**

**Marker 3** **P1103 (M3)**
Alternative Syntax : **M1**

Markers may be set and reset any time in a executable program.
Markers may be used as events in IF-requests.

E.g.: **#M1=1**
#L1
*Instruction block*
#IF M1=1
#GOTO 1

In the example the *instruction block* will be executed as long as the marker P1101 is set.

# 4.8.5.4 List options

## List options P1028

For listing the E$^2$Prom-program with the command 'LIST', different options may be selected. Following Bits are assigned in the 8-Bit value P1028:

Bit 0 (decimal 1) : The number of the instruction (line number) will be listed
Bit 1 (decimal 2) : units will be listed
Bit 2 (decimal 4) : listing with offset of lines - labels will be listed in the 1st row and instructions will be listed in the 4th row
Bit 3 (decimal 8) : Carriage Returns within the program will be listed in the same way the user programmed them (else each instruction will be printed in an separate line)
Bit 4 (decimal 16): When sending 'LIST P' the parameter texts are listed too

E.g.: 'P1028=5' results in listing all instructions of the program in the E$^2$Prom with line numbers when sending the command 'LIST' writing labels in the 1st row and instructions in the 4th row (P1028=5=1+4 , → Bit 0 and Bit 2 will be set).

# 4.8.5.5 Special Master-mode parameter

## Wait until end of positioning P1110

When the master is executing a positioning job, it will wait (depending on parameter P1110) until the end of the positioning job, before it continues with the next instruction of the program.

0 : No waiting for the end of a positioning job - in this way the master is able to control other axis while executing its own positioning job
1 : Wait for the end of the actual positioning job

**Please note** :
After Power On of the SERS always P1110 = **1** !
The parameter P1110 can not be stored into the parameter section of the E$^2$Prom.

## Destination address P1

(alternative syntax : z)
destination drive address - slave address - used only from the master drive destination for assignments, commands and 'IF'-requests
(e.g. Z=3 -> following commands are sent to drive address 3)

## IF send/receive P2

IF' send/receive will be initiated by the master drive. The master drive sends an assignment to a slave drive. The operand is the requested event. The slave returns the value of the event as assignment to the master.
P2 is used internally. (e.g. P2=336 is equivalent to IF 336 after addressing a slave drive).

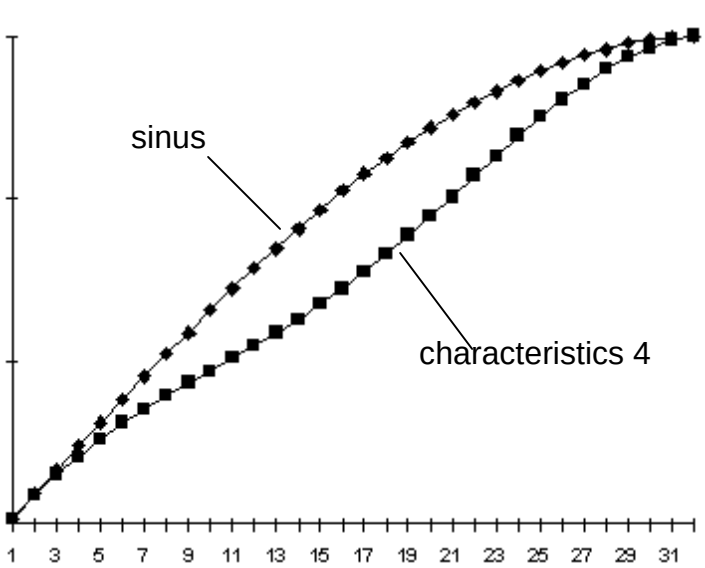
# 4.8.6 Miscellaneous parameter
# 4.8.6.1 Phase current adjustment

## Commuting table P1009

Selection of the current characteristics for the stepping motor current.
The characteristics to be chosen depends on the used stepping motor.
The better the current characteristics matches the motor-characteristics,
the softer and less noisier the stepping motor runs at low speed ranges.

Following selections are available:

0 : sinus-characteristics (pure sinus current)
1 : 87 style – characteristics (adjusted to the STÖGRA stepping motor-series SM 87)
2 : characteristics 3 (sinus$^2$ – characteristics)
3 : characteristics 4 (damped sinus$^2$ – characteristics)




32 entspricht einem Winkel von 1,8 ° am Schrittmotor.

The '87' – characteristics and the Characteristics 3 are between the sinus characteristics and the characteristics 4.

Recommended current characteristics to be selected for STÖGRA-stepping motors :

series SM 56 : '87'-characteristics
series SM 87 : '87'-characteristics
series SM 107/108 : characteristics 4
series SM 168 : characteristics 4

## Phase current P1010

The unit of the assigned values is [mA]
The unit may not be assigned.
The valid range of values depends on the SERS version.

| | SERS 01 | SERS 02 | SERS06 | SERS12 |
|---|---|---|---|---|
| Value range [mA] : | 0 - 1400 | 0 – 2800 | 0 - 8400 | 0 - 14500 |

e.g.: '#P1010=6000' sets the phase current to 6A (=6000mA)

## Acceleration value for current boost P1012

The unit depends on the scaling mode for acceleration (P160).
In case of a set bit 2 in P1011 the phase current will be risen during accelerating, if the value stored in P1012 is exceeded - P1012 may not be chosen smaller than 10 rad/s$^2$.

### current parameters P1011

Following assignment bit by bit apply (the explanations are valid for a set bits) :

Bit 0 (decimal 1) : automatic **current reduction** at motor stop is active
When the motor stops, the phase current will be reduced to 50% of the value stored in P1010.

Bit 1 (decimal 2) : automatic **current boost** is active -for getting more torque for acceleration
When the acceleration value stored in P1012 is exceeded during accelerating, then the phase current will be risen by 20%.

Bit 2 (decimal 4) : Current boost is active at motor stop
When the motor stops the phase current will be risen by 20%

E.g..: '#P1011=3' results in reducing automatically the phase current by 50% when the motor stops and rising the phase current by 20% during accelerating (depending on P1012).

### current range P1034

may not be changed - set by factory !!!

1 : 1A-Versions - e.g. SERS 01.85 V02
2 : 2A-Versions - e.g. SERS 02.24 V02
4 : 4A-Versions - e.g. SERS 04.85 V02
6 : 6A-Versions - e.g. SERS 06.85 V02
12 : 12A-Versions - e.g. SERS 12.120 V02

## 4.8.6.2 Program continue after Stop

### Continue after Stop P1033

The parameter defines, how to continue in the program respective with a positioning job after an one time Stop-command (sent via serial interface) or Stop-Signal (at input I9) and then again a Start-command or signal (sent /input I8).
Following possibilities can be selected :

0 : **New start** of the program in the $E^2$Prom (Start at line 1) respective new complete execution of the value in W (P47) in case of relative positioning.
1 : After one time Stop : Continuing of the program at that point, where it was interrupted, respective continuing the interrupted positioning job.
After two times Stop: proceed as described for P1033=0
2 : Jump to Label L65 in the program (after a stop-command or if there is an error, which causes a program termination – for possible errors see description parameter P11
3: as 2 but after a jump to L65 all further errors will not result in a program stop or a further jump to L65, until the internal flag "L65 error handling" is reset by the command "P1118=0" or "P11=0".

### Flag - L65 error handling P1118

Possible values:
0 : End of L65 error handling , it may be jumped again to L65

1 : L65 is executed

# 4.8.6.3 Manual drive control commands

### Manual drive control - command P1031

Different manual drive functions can be executed by assigning P1031 as follows :

1 : Left - slow (alternative syntax : **LS**)
2 : Right - slow (alternative syntax : **RS**)
4 : Left - fast (alternative syntax : **LF**)
8 : Right - fast (alternative syntax : **RF**)
16 : Homing (alternative Syntax : **H**)

### Stop when manual driving (jog) P1093

In case a manual drive (jog) function is activated at a digital Input (I 1 until I 6) , and a $E^2$Prom-operational program is active, then the reaction of the SER drive depends on P1093 as follows:

0 : the operational program will be interrupted and the jog function will be executed. After deactivating the jog input the operational program will be continued.
1 : the operational program will be terminated and the jog function will be executed. After deactivating the jog input the operational program will **not** be continued.

### Manual drive (jog) functions at Power-ON P1095

If manual drive functions are enabled (via P1021 or service-switch) with P1095 it is specified, how to react after a Power-ON-Reset (voltage supply ON).
0 : Start jog function if there is a **static signal** at the (jog) input
1 : Start jog function only if there is **signal edge** to ´1´ at the (jog) input

# 4.8.6.4 Manual drive control with Timeout

### Manual drive with Timeout P1035

If a manual drive function was started by sending one of the manual drive commands LS,RS,LF,RF or by assigning the parameter P1031 as described above, then depending on the parameter P1035 the manual drive function will be stopped again, if the manual drive command is not repeated latest 500ms after the last command.
This parameter is necessary only for using the manual drive functions with a terminal.

0 : no timeout when using manual drive functions
1 : timeout manual drive is active - max. after 500ms a manual drive function will be stopped

# 4.8.6.5 Drive address

### Drive address P1050

The SERS stores the drive address selected at DIP switch 2 (see 3.1.1 page 14) in parameter P1050.

# 4.8.6.6 Control of load angle

## Control of load angle assignment error/warning P1029

Definition of the reaction of the SERS in case of a load angle error

0 : no reaction - without control of load angle
1 : Message drive error - control of load angle (see P11)
2 : Message warning - control of load angle (see P12)
3 : Message drive error - control of load angle (P11) - but motor current stays ON

Parameter values ’1’ to ’3’ for P1029 work only if option E50 – SERS... E50 ... (e.g. SERS 06.85 V02 E50) is installed and the stepper motor includes a 2-channel incremental encoder.

P1029<>0 means controlling the load angle of the stepper motor. The position command value is compared with the real motor position. If the allowed max. load angle is exceeded then a warning or an error is created.

If P1053 = 4 or P1053 = 8, then not only the exceeding of the load angle is detected, but additionally the real motor position – see P1053.

## Option Input pulse / electrical gearing / hand wheel / input encoder / control of load angle with correction of position / frequency measurement P1053

0 : Function disabled.

1 : Pulses at digital Input I 3 move the motor

2 : as 1, but additionally with direction signal at digital Input I 4 (active = negativ)

3 : Type SERS ... (without option E50):
Pulses at digital Input I 3 -> motor moves into positive direction,
Pulses at digital Input I 4 -> motor moves into negative direction,
Type SERS ...E50... (including option E50):
Encoder signals – signal A at Input I 3 and Signal B at Input I 4 → motor is moved by encoder signals. Evaluation of direction automatically from encodersignals A and B

4 : Control of load angle – connection of a 2-channel-encoder :
channel A at digital Input I 3 and channel B at digital Input I 4

5 : as 1, but encoder input (9-pole D-Sub-connector "ENC") instead of digital input

6 : as 2, but encoder input (9-pole D-Sub-connector "ENC") instead of digital inputs

7 : as 3, but encoder input (9-pole D-Sub-connector "ENC") instead of digital inputs

8 : Control of load angle – connection of a 2-channel-encoder with signals A and B and inverted signals /A and /B at 9-pole D-Sub-connector "ENC"

9 : for SERS ... (without option E50) – input for frequency measurement
the frequency of pulses at digital input I 3 are measured (result can be read in P1124)
P1055 defines the timer interval for the measurement

**Handwheel / electrical gear (P1053=3 or P1053=7):**
The motor follows the encoder pulses of a handwheel or of a ´leading´ axis (motor).

While following the encoder pulses the acceleration and the velocity of the drive are limited by the parameters **A (P138)** and **V (P91)**. If an “error limit switch“ occurs (e.g. limit switch left opens) the motor stops and the phase current will be switched off. Afterwards moving the handwheel or the ´leading´ axis into the opposite direction (e.g. right direction), will switch on the motor current again automatically and the motor follows the encoder signals again - see also parameter P1094.

**Control of load angle (P1053=8 or P1053=4) with detection of real motor position:**
**P1053=4** normally is not good for the control of load angle, becuase for a secure control of load angle the inverted signals /A and /B should be evaluated also. Only in exceptionally cases P1053=4 should be used.

For the control of load angle always the option **P1053=8** should be used !
Additionally to P1053=8 the parameter P1029 should be set to a value <> 0 (see P1029)

If an "error load angle" happens (P11=32 or P12=32 – depending on P1029) – motor looses position due to mechanical overload – following commands can be used

- ´**POSR**´ : the motor will be moved with homing velocity (P41) to the actual position command value (parameter ´W´ - P47) – see also parameter **P1043**
- ´**P11=0**´ or ´**P12=0**´ : the real motor position (detected by the encoder signals) will be written into parameter **P51** (position of the stator field).

If an "error load angle" happens, then the SERS can detect (count) a max. load angle error (difference between position command value and real motor position) of +/- 32768 increments. In case of an encoder type E50 with 2 x 50 pulses per revolution (and 4-times evaluation → 200 impulses per rev.) there can be detected (and corrected with command "POSR") max. 163.8 revolutions of load angle error.

## Scaling factor input pulses P1054

This parameter specifies the resolution of the connected encoder or the required step width per pulse (e.g. in case of P1053=1) - unit: 1/200 rev.
- unit: 1/200 Umdr.
- 16 bit value with 5 decimal points
- value range -32.76800 until 32.76700
  internally there are stored only 16 bit. At values out of the range [-0.32768 , 0.32767] the resolution will be reduced by setting the last two decimal points to 0 (e.g. 21.54700)

Examples: P1054=2.5 for an encoder with 2 x 20 pulses per rev. e.g. encoder "E20"
P1054=1 for an encoder with 2 x 50 pulses per rev. e.g. encoder "E50"
P1054=0.25 for an encoder with 2 x 200 pulses per rev. e.g. encoder "H200"
P1054=0.05 for an encoder with 2 x 1000 pulses per rev.

## Time slot for measuring the frequency P1055

Indication in multiples of 2ms - For the function handwheel / electrical gear : defines the time interval for counting the pulses for calculating the velocity command value of the drive. P1055=1 means that every 2ms the counted pulses result in a new velocity command value. A larger value - e.g. P1055=50 (interval 100ms) - results in a smother running of the motor at low speed when following the encoder pulses.

At P1053=9 the parameter P1055 defines the time interval for the frequency measurement.
→ e.g: At P1055=50 every 100ms the pulses at input I3 are counted. The counting result after one time interval can be read in parameter P1124.

## Max. load angle P1057

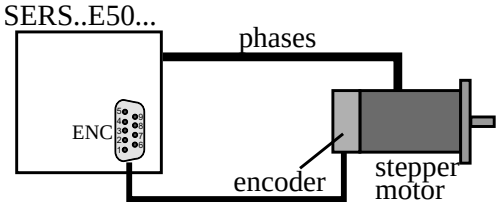
For internal use only

## Controlled value load angle P1002

Limit value of load angle – when the load angle exceeds this value, then an error load angle will be created.
Non scaled value – Indication in increments (unit 1/12800, 12800 increments = 360°)
Standard value : 7,2° degree (in case of 1,8° stepper motor)→ P1002=255

## Overview functions P1053

### Control of load angle

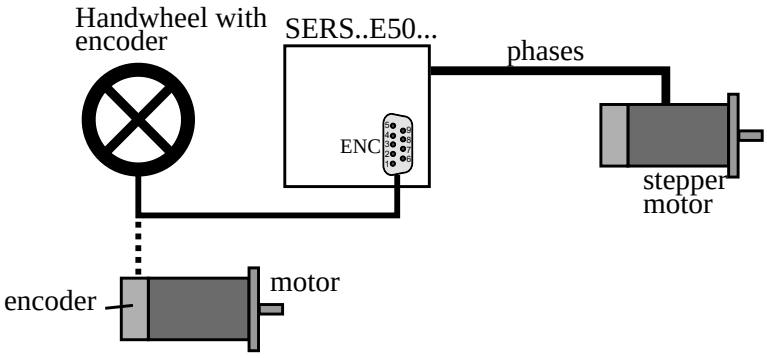



Necessary parameters:

P1053 : 0 or 8
P1029 : 1 or 2 or 3

Possible command **POSR**

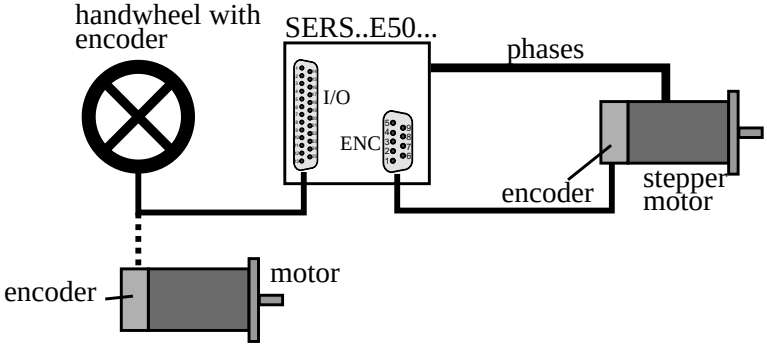
### Function handwheel / electrical gear




Necessary parameters:

P1053 : 3 or 7
P1029 : 0

### Control of load angle and function handwheel / electrical gear

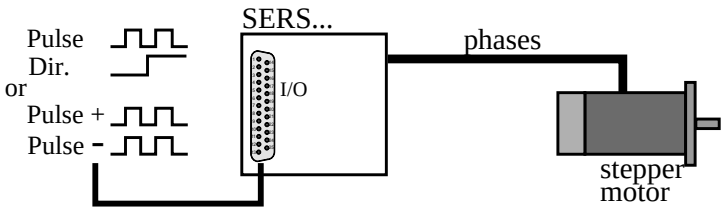



Necessary parameters:

P1053 : 3 or 7
P1029 : 1 or 2 or 3

Command **POSR** not possible

### Pulses input




Necessary parameters:

P1053 : 1 oder 2
P1029 : 0

### Keep absolute value in case of error limit switch and function electrical shaft / handwheel P1094

In cese of P1053=3 or P1053=7 if the motor moves on a limit switch (when controled by the external encoder signals), then the drive will be stopped and the motor phase current will be switched off. If afterwards the external encoder signals are into the opposite direction, then depending on P1094 the motor reacts as follows:

0 : encoder signals into the direction of the open limit switch will be ignored. Encoder signals into the opposite direction result in switching on the phase current and following of the motor to the encoder signals.

1 : encoder signals, into the direction of the open limit switch are not executed, , but will be counted internally (increment the counter). Encoder signals into the opposite direction result in dercrementing the internal counter . If the counter is '0', then the motor phase current will be switched on and the motor follows the encoder signals. So the motor position is allways identically (proportionally) to the position of the handwheel (or leading axis in case of encoder signals from an other 'leading' motor).

### Frequency at P1053=9 P1124

Frequency measurement result at P1053=9
Unit: counted pulses / P1055 (time interval)

### Pulse input counter 1 P1119

Is used internally for the P1053 modes and contents the actual count.
→ do not overwrite, because else in case of activated functions electrical gearing / pulse input the motor could receive new position command values !

### Pulse input counter 2 P1120

Definition as P1119

# 4.8.6.7 Control of motion

### Motion control P1044

Running distance value (scaled value depending on P76).
When running, after covering this distance there must be a signal (edge 0 → 1 with high-active input signals or edge 1 → 0 with low-active input signals) at the input specified in P1045.

e.g.: P1044 = 10mm and P1045 = 2
When during running there is not a signal edge at the input I2 at least all 10mm then the drive will stop and the error bit 'motion control' in P11 will be set.

### Motion control – assignment input for control P1045

Definition of the input used for the motion control - see P1044
P1045 = 1 → input I1
P1045 = 2 → input I2
:
:

P1045 = 8 → input I8

# 4.8.6.8 E²PROM-Parameter

### E²PROM Parameter P1004

Following values can be selected :

2 : Save the actual parameter in the SERS-RAM into the E²PROM
alternative syntax : **PSAVE**
Please note : Changed parameters in the SERS are stored into the SERS E²PROM only by sending P1004=2 or **PSAVE** !
Before sending **PSAVE**, the actual parameters are only in the SERS-RAM, that means that after switching off the SERS (disconnecting from power supply) without **PSAVE** all parameter changes are lost !!!

3 : Write standard parameter values in E²PROM
**ATTENTION !** : By sending **P1004=3** to the SERS all actual parameters stored in the E²PROM will be erased and replaced by standard values (pre-selected values) !!!

4 : Save actual position (P51) into E²PROM
alternative syntax : **POSSAVE**

5 : Save register R0 until R5 and W into E²PROM

### Save R0 until R5 and W P1117

Possible values to be written:
0 : disabled (=standard adjustment)
1 : Save also parameters "R0" until "R5" and "W" into E²PROM with the command P1004=2 (“PSAVE”)

### Free E²PROM memory P1122

Indicates the actual fee program memory in the E²Prom
Indication in words (1 word = 2 bytes)

# 4.8.6.9 Language selection

### Language selection P265

0 : German
1 : English

# 4.8.6.10 Software limit switch / position limit

### Position limit negative P1040

scaled value (depends on P76)
When running into negative direction and the actual position (P51) is smaller than this value P1040, then the drive stops, the bit "limit position" in P12 (warnings) is set, P1042 is "1" and the 7-Segment-displays shows a blinking "L".

### Position limit positive P1041

Definition as P1040 but positive drive direction

# 4.8.6.11 Arithmetic Parameters

### Accumulator P1047

Alternative Syntax : **X**
Variable for all arithmetic-functions - all calculations have to be done via the accumulator X - see also chapter arithmetic page 38
Example:
X=V-ADC*P1080

### Register 1 P1080 (R0)

Alternative syntax : **R0**
Free usable 32 bit signed variable
e.g. for saving results of arithmetical operations

### Register 2 P1081 (R1)

Alternative syntax : **R1** - see P1080

### Register 3 P1082 (R2)

Alternative syntax : **R2** - see P1080

### Register 4 P1083 (R3)

Alternative syntax : **R3** - see P1080

### Register 5 P1084 (R4)

Alternative syntax : **R4** - see P1080

### Register 6 P1085 (R5)

Alternative syntax : **R5** - see P1080

# 4.8.6.12 Service switch external

### Service switch external P1092

0 : disabled
1 : if there is an active signal at the opoisolated input ´service switch external´, then the digital inputs I 1 to I 8 are enabled for the manual drive functions – see description ´Enable manual drive functions´ at page 15 (chapter 3.1.4). The ´service switch external´ can be used alternatively to the ´service switch´ at DIL-switch 1 Bit 6 (chapter 3.1.4).

# 4.8.6.13 Program-/Parameter-Masken and Password for SERS-Programmer

### Password definition P1059

In P1059 there can be defined any password - 4-digit number - as password to enable showing and changing parameters and the $E^2$Prom-program in the SERS when operating with the SERS-Programmer !
P1059=0 means, that there is no password defined and all parameters and program lines may be seen and changed without password.
In case of the definition of a password :
If the password is not entered correctly when operating the SERS with the SERS-Programmer (e.g. just enter the RET-key when being asked for the password) then only the parameters enabled in P1060 until P1065 and the program lines enabled in P1070 until P1073 are displayed .
When entering the password correctly in the SERS-Programmer , then the operator can see and edit all parameter and the complete program.

## Parameter Mask [0] P1060

Enable the parameters P0 until P103 - set bit means enabled and the parameter can be seen and edited by the SERS-Programmer also without knowing the password in P1059.

| P0 | P1 | P2 | P11 | P12 | P41 | P42 | P44 |
|---|---|---|---|---|---|---|---|
| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 |
| P47 | P51 | P76 | P91 | P100 | P101 | P102 | P103 |
| 256 | 512 | 1024 | 2048 | 4096 | 8192 | 16384 | 32768 |

e.g: P1060 = 2144 (=32+64+2048) → parameters P41, P42 and P91(V) will be displayed in the parameter section of the SERS-Programmer and can be edited even without knowing the password, defined in P1059.

## Parameter Mask [1] P1061

Explanation as P1060 and valid for following parameters:

| P108 | P121 | P122 | P123 | P134 | P138 | P147 | P160 |
|---|---|---|---|---|---|---|---|
| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 |
| P265 | P336 | P403 | P1001 | P1002 | P1003 | P1004 | P1005 |
| 256 | 512 | 1024 | 2048 | 4096 | 8192 | 16384 | 32768 |

## Parameter Mask [2] P1062

Explanation as P1060 and valid for following parameters:

| P1006 | P1007 | P1008 | P1009 | P1010 | P1011 | P1012 | P1013 |
|---|---|---|---|---|---|---|---|
| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 |
| P1014 | P1015 | P1016 | P1017 | P1018 | P1019 | P1020 | P1021 |
| 256 | 512 | 1024 | 2048 | 4096 | 8192 | 16384 | 32768 |

## Parameter Maske [3] P1063

Explanation as P1060 and valid for following parameters:

| P1022 | P1023 | P1024 | P1025 | P1026 | P1027 | P1028 | P1029 |
|---|---|---|---|---|---|---|---|
| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 |
| P1030 | P1031 | P1032 | P1033 | P1034 | P1035 | P1036 | P1037 |
| 256 | 512 | 1024 | 2048 | 4096 | 8192 | 16384 | 32768 |

## Parameter Maske [4] P1064

Explanation as P1060 and valid for following parameters:

| P1038 | P1039 | P1040 | P1041 | P1042 | P1043 | P1044 | P1045 |
|---|---|---|---|---|---|---|---|
| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 |
| P1046 | P1047 | P1050 | P1051 | P1052 | P1053 | P1054 | P1055 |
| 256 | 512 | 1024 | 2048 | 4096 | 8192 | 16384 | 32768 |

## Parameter Maske [5] P1065

Explanation as P1060 and valid for following parameters:

| P1056 | P1057 | P1058 | P1059 | P1060 | P1061 | P1062 | P1063 |
|---|---|---|---|---|---|---|---|
| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 |
| P1064 | P1065 | P1066 | P1070 | P1071 | P1072 | P1073 | P1080 |
| 256 | 512 | 1024 | 2048 | 4096 | 8192 | 16384 | 32768 |

## Parameter Mask [6] P1066

Explanation as P1060 and valid for following parameters:

| P1081 | P1082 | P1083 | P1084 | P1085 | P1092 | P1093 | P1094 |
|---|---|---|---|---|---|---|---|
| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 |
| P1095 | P1096 | P1097 | P1098 | P1099 | P1100 | P1101 | P1102 |
| 256 | 512 | 1024 | 2048 | 4096 | 8192 | 16384 | 32768 |

## Parameter Mask [7] P1067

Explanation as P1060 and valid for following parameters:

| P1103 | P1110 | P1111 | P1112 | P1113 | P1114 | P1115 | P1116 |
|---|---|---|---|---|---|---|---|
| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 |
| P1117 | P1118 | P1119 | P1120 | P1121 | P1122 | P1123 | P1124 |
| 256 | 512 | 1024 | 2048 | 4096 | 8192 | 16384 | 32768 |

## Parameter Mask [8] P1068

Explanation as P1060 and valid for following parameters:

| P1125 | P1126 | P1201 | P1202 | P1203 | P1204 | P1205 | P1206 |
|---|---|---|---|---|---|---|---|
| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 |
| P1207 | P1208 | P1209 | P1210 | P1211 | P1212 | P1213 | P1214 |
| 256 | 512 | 1024 | 2048 | 4096 | 8192 | 16384 | 32768 |

## Program Mask [1] P1070

By using the program mask there can be enabled specific program lines (only value assignments) in the $E^2$PROM-Program when using a SERS-Programmer
The enabled program lines (value assignments) will be displayed in the program-menu of the SERS-Programmer and can be edited, even if the password, defined in P1059, was not entered correctly in the SERS-Programmer.

Following rules apply :

- Enabling a program line refers to a Label. (e.g. L1 or L23)
- The Label L1 until L64 may be enabled
- The by the Label followed assignment is enabled (e.g. L1 V=1000)
- Only value assignments may be enabled together with Labels
- If there is an additional TEXT-string (defined through quotations marks) between the enabled Label and the value assignment (e.g. L1 "velocity" V=1000), then this TEXT-string will be displayed, too

Example: Following program is stored in the $E^2$PROM of the SERS:

```
P1014=0
L1
A=2000.000
L2
"Velocity:      "
V=500.000
W=1300 E
```

The Label L1 and L2 are enabled with P1070=3 (=1+2)
When using the SERS-Programmer without entering the correct password (or just entering the RET-Taste when asked for the password) then following program lines will be displayed:

A=2000.000
Velocity:
V=500.000

**NOTES to TEXT-strings:**
When using a TEXT-string - defined through the quotation marks " at the beginning and the end of the text, then the length of the text should be exactly 16 characters, because the SERS-Programmer does not insert automatically a carriage return at the end of the text - the display of the SERS-Programmer is 4 x 16 characters. In that way the text and the following value assignment will be shown at the display of the SERS-Programmer in separate lines.
The text can be filled with space characters.
e.g.: "length " - to the text '*length'* are added 10 space characters.

| L1 | L2 | L3 | L4 | L5 | L6 | L7 | L8 |
|---|---|---|---|---|---|---|---|
| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 |
| L9 | L10 | L11 | L12 | L13 | L14 | L15 | L16 |
| 256 | 512 | 1024 | 2048 | 4096 | 8192 | 16384 | 32768 |

e.g.: P1070=42 (=2+8+32)→ Labels L2, L4 and L6 are enabled

## Program Mask [2] P1071

Explanation as P1070 for following program Label:

| L17 | L18 | L19 | L20 | L21 | L22 | L23 | L24 |
|---|---|---|---|---|---|---|---|
| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 |
| L25 | L26 | L27 | L28 | L29 | L30 | L31 | L32 |
| 256 | 512 | 1024 | 2048 | 4096 | 8192 | 16384 | 32768 |

## Program Mask [3] P1072

Explanation as P1070 for following program Label:

| L33 | L34 | L35 | L36 | L37 | L38 | L39 | L40 |
|---|---|---|---|---|---|---|---|
| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 |
| L41 | L42 | L43 | L44 | L45 | L46 | L47 | L48 |
| 256 | 512 | 1024 | 2048 | 4096 | 8192 | 16384 | 32768 |

## Program Mask [4] P1073

Explanation as P1070 for following program Label:

| L49 | L50 | L51 | L52 | L53 | L54 | L55 | L56 |
|---|---|---|---|---|---|---|---|
| 1 | 2 | 4 | 8 | 16 | 32 | 64 | 128 |
| L57 | L58 | L59 | L60 | L61 | L62 | L63 | L64 |
| 256 | 512 | 1024 | 2048 | 4096 | 8192 | 16384 | 32768 |

# 4.8.6.14 CANopen Parameter

## Operand P1097

For use with SERS-versions with CANopen interface → SERS...CAN
Operand for the Opcode defined in P1099.
An assignment to this parameter will not be saved into the $E^2$PROM, but executed immediately (only 32 bit assignments).

## Opcode P1099

For use with SERS-versions with CANopen interface → SERS...CAN
An assignment to this parameter will not be saved into the $E^2$PROM, but executed immediately (only 32 bit assignments).
The value of these parameter will be assigned to the Opcode.
In the programming mode (started through P0=2 → new) the Opcode will be saved.
In the normal (non programming) mode the Opcode will be executed.
In case of Opcodes, which require a Operand, the Operand must be defined before in P1097.
E.g. a "RUN 5" Opcode will be executed as "P1099=62213".
A "E" Opcode will be executed as "P1099=63235".

**For writing an operational program with a SERS...CAN with CANopen-interface for saving it into the SERS $E^2$PRom the parameters P1097 und P1099 must be used.**

# 5. Technical specifications

## Protection of the device

SERS 01, SERS 02, SERS 06 and SERS 12 : IP 00
ELK with SERS : IP 20

Protection against over temperature, under voltage and
protection against short circuit (phase against phase and phase against GND)

## Weight

SERS 01/02 : 0,4 kg , SERS 06 : 0,77 kg , SERS 12 : 1,1 kg
ELK2 : 6,1 kg , ELK3 : 7,0 kg , ELK4 : 7,9 kg

## Ambient conditions

Ambient temperature : 0°C to 50°C
From 6A adjusted phase current a forced draft is necessary.

## Noise immunity

In case of correct installation : according to EN50082-2

## Noise radiation

In case of correct installation and shielding of the leads : according to EN55011 class B

## Voltage supply

SERS 02.24 , SERS 06.24 : 18 - 40 VDC (max Ripple 5%)
SERS 01.85 , SERS 06.85 , SERS 12.85 : 50 - 85 VDC (max Ripple 5%)
SERS 06.120 SERS 12.120 : 60 - 120 VDC (max Ripple 5%)
SERS 12.240 : 120 - 240 VDC (max Ripple 5%)

## Phase currents

SERS 01 : 0 - 1,4 A/phase
SERS 02 : 0 - 2,8 A/phase
SERS 06 : 0 – 8,4 A/phase
SERS 12 : 0 – 14,5 A /phase

## Inputs

**2 limit switch inputs, 1 reference switch input, 1 Stop input:**
Optoisolated inputs with common Opto-GND
Signal level : 13 VDC – 30 VDC

**8 free programmable digital inputs**
configurable Low or High active, TTL-level or SPS-level
optionally **additional 8 free programmable digital inputs** (only option I/O) with PLC-level

**1 analogue input**
Differential analogue input 0 - 5 VDC with 8 bit resolution

## Outputs

1 potential-free ready signal - (2 relay contacts) - max. load 100mA / 50VDC
4 free programmable galvanic isolated PNP outputs – max. load 500mA / 5-24 VDC
additional 12 not galvanic isolated PNP outputs (only option I/O) - max. 100mA / 5-24 VDC