



## **Stögra Antriebstechnik GmbH**

Machtlfinger Straße 24  
D-81379 München

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

# **SERS 02, SERS 06 and SERS 12**

## **Version V**

### **PB-DP**

Stepping motor power amplifier board with position  
control and Profibus-DP interface

**Profibus-DP specific additions to the 'SERS installation and  
programming manual' for SERS with RS232 interface**

edition June 2004

All rights reserved.

Without written approval we allow no reprint nor partial copying.

We reserve the right to make engineering changes, refinements and improvements.

Mechanical and electrical ratings and dimensions are, therefore,  
subject to change without notice.

no liability whatsoever is accepted.

## Index

	page
1. General notes	3
1.1 Short overview	3
1.2 Notes on manual	3
1.3 GSD-file and protocol modes	3
1.4 Overview protocol differences in the ASCII-character mode in comparison to the SERS with RS232	4
2. Profibus-DP protocol in the SERS-PB-DP	5
2.1 ASCII - character mode : I/O-section SERS...PB-DP – communication with help of a toggle-byte ( <b>old mode</b> )	5
2.2 Binary mode (8/12 byte I/O) ( <b>old mode</b> )	6
2.2.1 Overview	6
2.2.2 Control word	6
2.2.3 Operation code (Opcode)	6
2.2.4 Operand	9
2.2.5 Status	10
2.2.6 Result	10
2.2.7 Position	10
2.2.8 Inputs	10
2.3 <b>Extended binary mode (22/12 byte I/O) (actual mode)</b>	11
2.3.1 Overview	11
2.3.2 Control word	11
2.3.3 Status	12
2.3.4 Function principle extended binary mode	12
3. Diagnostics	13
4. 7-segment-display	16
5. Examples	
5.1 Example: Positioning and writing parameters in the binary mode (8/12 byte I/O)	17
5.2 Example: Positioning and writing parameters in the extended binary mode (22/12 byte I/O)	19
5.3 Typical parameters to be set	22
5.4 Example: Polynom driving in the extended binary mode (22/12 byte I/O)	23

## 1. General notes

### 1.1 Short overview

- The stepper motor control SERS...PB-DP is a 1-axis position control with Profibus-DP interface.
- The file 'stoegra5.gsd' (delivered together with the SERS...PB-DP control on floppy disc or on the STÖGRA-CD or download at the STÖGRA homepage <http://www.stoegra.de>) contains the corresponding 'node' – information for the Profibus-Master
- Three different protocol modes: ASCII-character mode, binary mode and extended binary mode
- Available functions and commands of the SERS...PB-DP are identically to the functions and commands of the position control SERS with RS232/RS485 interface.

### 1.2 Manual notes

On the following pages the "installation and programming manual" for the SERS with RS232/RS485 interface will be named **SERS-manual**.

Following limitations apply to the SERS...PB-DP, compared to the SERS with RS232/RS485 interface (concerning the SERS-manual):

1. At the 8-pole DIP-switch 1 (see SERS-manual page 11 and page 15) the bits 1 until 3 are without function (selection of the baudrate at the RS232/RS485-Version).
2. The assignment of the 9-pole D-Sub-connector is according to the Profibus-DP-standard

### 1.3 GSD-File and Protocol modes

At the provided file 'stoegra.gsd' there can be selected between different protocol modes::

- "SERP instructions 32 Byte I/O" - ASCII-character mode – Transmit and Receive ASCII-characters via 32 Byte in Input and Output sector
- "SERP instructions 16 Byte I/O" - ASCII-character mode – Transmit and Receive ASCII-characters via 32 Byte in Input and Output sector
- "SERP binary 8/12 Byte I/O" – binary mode – set and read Bits in control-/statuswords and transmit/receive operation codes, operands, results via 8 Byte in Output sector and 12 Byte in Input sector
- "SERP binary 8/12 Byte I/O konsis" – binary mode with consistent data transmission and 8 byte in the Output section and 12 Byte in the Input section
- **"SERP binary 22/12 Byte I/O" → extended binary mode with consistent data transmission and 22 Byte in the Output section and 12 Byte in the Input section**

**For new projects we recommend to use the extended binary mode (22/12 byte I/O) !**

## 1.4 Overview special features in the ASCII-character mode compared to the SERS with RS232

1. A character sequence has to be terminated with a Carriage-Return (Hex-Code '0D') and a "Zero"-Byte (Hex-Code '00') - (see chapter '2. Profibus-DP protocol in the SERS-PB-DP') – at the RS232/RS485-version command lines are terminated only with a Carriage-Return.
2. Commands are accepted only after changing the "Toggle-Bit" in the SERS...PB-DP (see explanations in chapter 2)
3. Commands **may not** start with the '#'-character. (At the RS232/RS485-version every new command line has to be started with the '#'-character.
4. In case of transmitting commands from the master to the SERS correctly (correct syntax and no logic or operational error) then the SERS...PB-DP writes acknowledge messages into the Profibus Data Input sector:

First Byte is the Toggle-Byte (see chapter 2), the second Byte is the linefeed-character (Hex-Code '0A') , afterwards starts a acknowledge-message (e.g: 'ok1') and after that again a Linefeed-character is written, followed by a Carriage-Return (Hex-Code '0D') and a 'Zero'-Byte. The Zero-Byte terminates the actual message.

In case of an error only a "Zero"-Byte is written into the second byte. The values after the "Zero"-Byte are not valid.

e.g.

INPUT-sector (e.g. at 32-Byte definition):

Without error:	In case of an error in a command in the OUTPUT-sector
01 0A 6F 6B 31 0A 0D 00	01 00 6F 6B 33 0A 0D 00    2.Byte is "00"
00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00

## 2. Profibus-DP protocol in the SERS-PB-DP

### 2.1. ASCII-character mode : E/A-sector SERS...PB-DP communication by using a Toggle-Byte (old mode)

The communication is realised according to the Profibus-DP standard via a Data-Input-sector and an Output-sector in the Profibus-Master and the SERS...PB-DP.

The 1. Byte of the Input- and Output-sector of the SERS...PB-DP is used as Toggle-Byte.

The communication between the Profibus-Master and the SERS...PB-DP is as follows:

1. The Profibus master writes a command into the OUT-sector of the SERS...PB-DP and writes a value which is different to the actual value into the Toggle-Byte of the OUT-sector (e.g. increase Toggle-Byte by one or change from '0' to '1')
2. Because of the changed Toggle-Byte the SERS...PB-DP recognises, that the master sent a new command. The SERS...PB-DP executes the command, writes an acknowledge (e.g. ok1, ok0, ... or pgm – see SERS-manual page 19) into the Input-sector, and copies the changed Toggle-Byte from the Output-sector to the Toggle-Byte in the Input-sector.
3. Taking over the Toggle-Byte into the Input-sector signals to the master, that the command was accepted by the SERS-DP. In case the master requested any data, the copied Toggle-byte signals to the Master that the data are ready to be picked up. Also the next command from the master the SERS...PB-DP may be sent now.

**All commands to the SERS are sent as ASCII-characters, as described in the SERS-manual.**

Each command must be terminated by a Carriage Return and a following '0'-Byte. After writing (or time consistent to) the command the Toggle-Byte in the Data-Output-sector has to be changed.

e.g.:

Output-sector (e.g. at 32-Byte definition):

At the beginning:	Writing an 'ON' (switch on motor current)
00 00 00 00 00 00 00 00	01 4F 4E 0D 00 00 00 00
00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00

'01' in the first Byte of the Output-sector is the Toggle-Byte

'4F' = hexadecimal ASCII-Code for the character 'O'

'4E' = hexadecimal ASCII-Code for the character 'N'

'0D' = hexadecimal ASCII-Code for Carriage Return CR

'00' = Zero-Byte

'0A' = hexadecimal ASCII-Code for Linefeed LF

Input-sector of the SERS...PB-DP:

At the beginning	After accepting of the 'ON'-command
00 00 00 00 00 00 00 00	01 0A 4F 4E 31 0A 0D 00 = LF + 'OK1' + LF + CR
00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00

## 2.2 Binary mode (8/12 Byte I/O) (old mode)

### 2.2.1 Overview

Transmitting data to the SERS...PB-DP via 8 Byte in the Profibus data-Output-sector:

Byte 1 and 2 : **Control word**

Byte 3 and 4 : **Operation code**

Byte 5 until 8 : **Operand**

And receiving data from the SERS...PB-DP via 12 Byte in the Profibus data-Input-sector:

Byte 1 and 2 : **Status**

Byte 3 until 6 : **Result** (e.g. after requesting parameter contents of the SERS)

Byte 7 until 10 : **Actual position**

Byte 11 and 12 : **Inputs**

### 2.2.2 Control word

At change (single bits) the Control Word in the Output sector controls following SERS functions ("STOP" is always evaluated - also without change of the corresponding bit, "START\_PROGRAMM" is executed statically, if Parameter P1023 = "1").

Bit 0: SLOW_NEGATIVE jog	1 = active, 0 = Stop
Bit 1: SLOW_POSITIVE jog	1 = active, 0 = Stop
Bit 2: FAST_NEGATIVE jog	1 = active, 0 = Stop
Bit 3: FAST_POSITIVE jog	1 = active, 0 = Stop
Bit 4: HOMING	1 = active, 0 = Stop
Bit 5: SWITCHING CURRENT	1 = ON, 0 = OFF
Bit 6: STOP	1 = Stop active, 0 = enable motor movement
Bit 7: START_PROGRAM	1 = Start
Bit 8: START_POSITIONING	turning from 0 to 1 = Start
Bit 9 : OUTPUT 1	setting output 1 - only if no operational program is active
Bit 10 : OUTPUT 2	setting output 2 - only if no operational program is active
Bit 11 : OUTPUT 3	setting output 3 - only if no operational program is active
Bit 12 : OUTPUT 4	setting output 4 - only if no operational program is active
Bit 13: RESET_ERROR	edge from 0 to 1 → P11=0
Bit 14: RESET_WARNING	edge from 0 to 1 → P12=0

### 2.2.3 Operation Code (Opcode)

- The Opcode is executed when changed from "0" to a value "not 0".
- Both Bytes of the Opcode have to be transmitted time consistent, to prevent a non valid Opcode.
- Depending on the operation an operand is needed. This operand must be valid, before the Opcode is set <> 0.
- After executing the transmitted Opcode the SERS...DP-PB will send an Acknowledge at "Bit 8: HANDSHAKE" in the Status word of the data-Input-sector.
- Before sending a new Opcode the master has to set the Opcode to "0" and after that wait until the Handshake Bit is set to "0" by the SERS...PB-DP.
- The most important Opcodes for a SERS-drive used without operational program are "assignment" and "request".

Following table shows all Opcodes with **binary** code.

The "c" Bit is used for formatting purposes for the internal listing of operational programs.

If 'c' = "1", then the Opcode in the listing is in a new line. If not using (programming) operational programs set always "c" = "0".

"p" bits: parameter numbers (identifiers) - parameter with the address (ppp pppp pppp).

(e.g.: 011 1111 0010 binary = "1010" decimal for parameter P1010)

"n" Bits: number of decimal places (referred to operand)

"e" Bits: event number for conditions used at conditional commands - see next page

"b" Bits: Inputs when using the conditional command "IF IN..."

"l" Bits: Label numbers in an operational program.

"X": accumulator for using the arithmetical functions in an operational program

Command	Opcode, binary	Explanation
Addition	0000 cppp pppp pppp	$X = X + (ppp\ pppp\ pppp)$
	0000 c111 nnnn nnnn	$X = X + \text{operand}$
Subtraction	0001 cppp pppp pppp	$X = X - (ppp\ pppp\ pppp)$
	0001 c111 nnnn nnnn	$X = X - \text{operand}$
Multiplication	0010 cppp pppp pppp	$X = X * (ppp\ pppp\ pppp)$
	0010 c111 nnnn nnnn	$X = X * \text{operand}$
Division	0011 cppp pppp pppp	$X = X / (ppp\ pppp\ pppp)$
	0011 c111 nnnn nnnn	$X = X / \text{operand}$
AND	0100 cppp pppp pppp	$X = X \& (ppp\ pppp\ pppp)$
	0100 c111 nnnn nnnn	$X = X \& \text{operand}$
OR	0101 cppp pppp pppp	$X = X   (ppp\ pppp\ pppp)$
	0101 c111 nnnn nnnn	$X = X   \text{operand}$
EXCL OR	0110 cppp pppp pppp	$X = X \wedge (ppp\ pppp\ pppp)$
	0110 c111 nnnn nnnn	$X = X \wedge \text{operand}$
Accu assignment	0111 cppp pppp pppp	$X = (ppp\ pppp\ pppp)$
	0111 c111 nnnn nnnn	$X = \text{operand}$
<b>Parameter assignment</b>	<b>1000 cppp pppp pppp</b>	<b>(ppp pppp pppp) = operand</b>
	1001 cppp pppp pppp	(ppp pppp pppp) = operand only 16bit
	1010 cppp pppp pppp	(ppp pppp pppp) = X
<b>Parameter request</b>	<b>1011 cppp pppp pppp</b>	<b>operand = (ppp pppp pppp)</b>
Conditional command	1101 ceee eeee eeee	if (eee eeee eeee) (e.g. IF 100)
	1101 c111 bbbb bbbb	if (bbbb bbbb) (e.g. IF IN 12)
	1110 ceee eeee eeee	if !(eee eeee eeee) (e.g. IF !11)
	1110 c111 bbbb bbbb	if !(bbbb bbbb) (e.g. IF !IN12)
Jump	1111 c000 llll llll	GOTO llll llll (e.g. GOTO 5)
Subroutine	1111 c001 llll llll	GOSUB llll llll (e.g. GOSUB 5)
Label	1111 c010 llll llll	Label llll llll (e.g. L5)
Program Start	1111 c011 llll llll	RUN llll llll (e.g. RUN 5)
Text	1111 c100 llll llll	Text with text length llll llll
NOT	1111 c111 0000 0000	$X = !X$
Negation	1111 c111 0000 0001	$X = -X$
	1111 c111 0000 0010	no operation
	<b>1111 c111 0000 0011</b>	<b>E</b>
<b>Stop</b>	<b>1111 c111 0000 0100</b>	<b>S</b>
<b>Home</b>	<b>1111 c111 0000 0101</b>	<b>H</b>
Return	1111 c111 0000 0110	RETURN
Program end	1111 c111 0000 0111	PE
End	1111 c111 1111 1111	END

**Event numbers ("e")** – for conditional commands if (eee eeee eeee)

Event no.	event true
0	Program running
1	Input 1 is active
2	Input 2 is active
3	Input 3 is active
4	Input 4 is active
5	Input 5 is active
6	Input 6 is active
7	Input 7 is active
8	Input 8 is active
11	Error - P11 $\langle \rangle$ 0
12	Warning - P12 $\langle \rangle$ 0
100	Counter C1 - P100 > 1
101	Counter C2 - P101 > 1
102	Counter C3 - P102 > 1
336	motor is not moving – positioning ready, drive is in position
1015	motor is accelerating
1016	motor is moving with constant speed
1042	software limit position overflow
1047	X > 0
1101	Marker M1 $\langle \rangle$ 0
1102	Marker M2 $\langle \rangle$ 0
1103	Marker M3 $\langle \rangle$ 0
1201	Output O1 active
1202	Output O2 active
1203	Output O3 active
1204	Output O4 active

The event numbers correspond to the Parameter numbers !  
E.g. event "336" = Parameter P336

**Running (operational) programs in SERS ... PB-DP controls:**

Writing / transmitting a new running program into a SERS ... PB-DP, is initiated by the parameter assignment 'P0=2'.

Attention: 'P0=2' is a parameter assignment with a 16-bit operand !

→ OPCODE 1010 0000 0000 0000 = 90 00 (HEX) and operand 00 00 00 02 (HEX)

All other parameter assignments are with 32-bit operands

→ OPCODE 1000 .... = 8... ..

All commands included in above OPCODE-list may be used in a running program.

The definition of a running program initiated by 'P0=2' must be terminated by the parameter assignment 'P0=0' (end of running program definition)

→ OPCODE 1000 0000 0000 0000 = 80 00 (HEX) and operand 00 00 00 00 (HEX)

Via the control word Bit 7 the running program can be started.

Bit 9 in the status word indicates if a running program is operating or stopped.

## 2.2.4 Operand

### Values in binary format

The number of decimal places is defined with the parameter number (in the Opcode) and the scaling of the parameter. Values are assigned and stored in binary format without decimal places. E.g. the position "360.6" degree at rotational scaling has to be assigned and also is indicated as "3606000", because this parameter at this scaling includes 4 decimal places.

When using arithmetical functions with constants, then the decimal places have to be indicated in the Opcode in the field "nnnn nnnn", because constants do not have a standard scaling.

The scalings can be selected via the parameters P44, P76 and P160 (see general “**manual SERS**“ - for SERS with RS232-interface - page 54)

Following table shows all Parameters with decimal places:

Parameter	number of decimal places
41	depending on selected scaling
42	depending on selected scaling
47 (W)	depending on selected scaling
51	depending on selected scaling
91 (V)	depending on selected scaling
103	depending on selected scaling
108	2 100% = 1.00
123	depending on selected scaling
138 (A)	depending on selected scaling
1003	depending on selected scaling
1005	2 100% = 1.00
1006	2 100% = 1.00
1007	2 100% = 1.00
1008	2 100% = 1.00
1012	depending on selected scaling
1018	depending on selected scaling
1019	depending on selected scaling
1020	depending on selected scaling
1024	depending on selected scaling
1026	depending on selected scaling
1030	depending on selected scaling
1037	depending on selected scaling
1039	depending on selected scaling
1040	depending on selected scaling
1041	depending on selected scaling
1044	depending on selected scaling
1046	2 100% = 1.00
1047 (X)	depending calculation result
1052	depending on selected scaling
1053	depending on selected scaling
1054	3
1080... (R0...)	depending assigned parameter
1100 (D)	1

All other parameters are not scaled (without decimal places).

## 2.2.5 Status

The Status word includes following Bit definitions (similar to DRIVECOM profile):

Bit position and signification	set Bit means
Bit 0: READY_TO_SWITCH_ON	is always 1
Bit 1: SWITCHED_ON	P134 $\langle \rangle$ 0 (motor current switched ON)
Bit 2: OPERATION_ENABLED	P134 $\langle \rangle$ 0 (motor current switched ON)
Bit 3: FAULT	P11 (error) $\langle \rangle$ 0
Bit 4: VOLTAGE_DISABLED	is always 0
Bit 5: QUICK_STOP	is always 0
Bit 6: SWITCH_ON_DISABLED	is always 0
Bit 7: WARNING	P12 (warning) $\langle \rangle$ 0
Bit 8: HANDSHAKE	Opcode executed from SERS
Bit 9: REMOTE	P0 is 0 (program is running)
Bit 10: TARGET_REACHED	P336 is 1 (motor is moving)
Bit 11: INTERNAL_LIMIT_ACTIVE	P1042 $\langle \rangle$ 0, software position limit overflow
Bit 12: HOMING_ATTAINED	P403 = 1 - after homing terminated successfully
Bit 13: FOLLOWING_ERROR	Load angle error - only with option Encoder
Bit 14: ACCELERATING_PHASE	P1015 = 1, motor is accelerating
Bit 15: CONSTANT_RUNNING_PHASE	P1016 = 1, motor runs with constant speed

## 2.2.6 Result

"result" of the last interrogation - after sending the Opcode "request" - value of the interrogated parameter value.

– indicated as binary value.

## 2.2.7 Position

"position" contains the actual position (P51) – binary value.

## 2.2.8 Inputs

"Inputs" contains the status of the digital inputs I1 until I8 and the optoisolated inputs Stop, Homing switch, Limit switch positive and limit switch negative.

Bit 0: input 1	Bit 8: Limit switch negative
Bit 1: input 2	Bit 9: Limit switch positive
Bit 2: input 3	Bit 10: Stop switch
Bit 3: input 4	Bit 11: Homing switch
Bit 4: input 5	Bit 12: Intern
Bit 5: input 6	Bit 13: Intern
Bit 6: input 7	Bit 14: Intern
Bit 7: input 8	Bit 15: Intern

## 2.3 Extended binary mode (22/12 Byte I/O) (actual mode)

### 2.3.1 Overview

Transmitting data to the SERS...PB-DP via 22 Byte in the Output section:

Byte 1 until 4 : **Control Word**  
 Byte 5 until 6 : **Operation Code (Opcode)**  
 Byte 7 until 10 : **Operand**  
 Byte 11 until 14 : **Acceleration**  
 Byte 15 until 18 : **Velocity**  
 Byte 19 until 22 : **Position command value**

Receiving data from the SERS...PB-DP via 12 Byte in the Input section:

Byte 1 until 2 : **Status**  
 Byte 3 until 6 : **Result** (from a parameter value request before)  
 Byte 7 until 10 : **Actual Position**  
 Byte 11 until 12 : **Inputs**

### 2.3.2 Control Word

The Control Word in the Output section is triggering the actions defined below after **being changed**. "STOP" will be evaluated always (also without change of control word).

"START\_PROGRAMM" will be evaluated statically (without change of control word) if  
 Parameter P1023 = 1

Bit 0: SLOW_NEGATIVE Jog	1 = active, 0 = Stop
Bit 1: SLOW_POSITIVE Jog	1 = active, 0 = Stop
Bit 2: FAST_NEGATIVE Jog	1 = active, 0 = Stop
Bit 3: FAST_POSITIVE Jog	1 = active, 0 = Stop
Bit 4: HOMING	1 = active, 0 = Stop
Bit 5: PHASE_CURRENT_ON	1 = ON, 0 = OFF
Bit 6: STOP	1 = Stop aktiv, 0 = Fahren möglich
Bit 7: START_PROGRAMM	1 = Start
Bit 8: START_POSITIONING	edge 0 to 1 = Start
Bit 9: OUTPUT 1	1 = ON (only if no running program is active / P0=0 !)
Bit 10: OUTPUT 2	1 = ON (only if no running program is active / P0=0 !)
Bit 11: OUTPUT 3	1 = ON (only if no running program is active / P0=0 !)
Bit 12: OUTPUT 4	1 = ON (only if no running program is active / P0=0 !)
Bit 13: RESET_ERROR	edge 0 to 1 → P11=0
Bit 14: RESET_WARNING	edge 0 to 1 → P12=0
Bit 15: EXECUTE_OPCODE	edge 0 → 1 = "opcode" will be executed
Bit 16: START_POSITIONING_TOGGLE	at change of this bit the drive will be started
Bit 17: ABSOLUTE	1 = set absolute positioning mode (P1014=2) 0 = set relative positioning mode (P1014=0), if absolute positioning mode was active before. Relative positioning modes (P1014=1, P1014=3) will not be changed, if any relative positioning mode was active before
Bit 18: POLYNOM	1 = activates the Polynom positioning mode (command WP)
Bit 19: POLYNOM_TERM	1 = termination of Polynom / last Polynom section (WPT)
Bit 20: PARAMETER_ACCEPT	the parameter "acceleration", "velocity" and "position command value" in bytes 11 until 22 will be used in case of a start positioning via the Control Word (bit 8 or bit 16).
Bit 21 bis 31: reserviert	

### 2.3.3 Status

Bit position and definition	a set bit means
Bit 0: READY_TO_SWITCH_ON	is always 1
Bit 1: SWITCHED_ON	P134 $\triangleleft$ 0 (phase current is ON)
Bit 2: OPERATION_ENABLED	P134 $\triangleleft$ 0 (phase current is ON)
Bit 3: FAULT	P11 $\triangleleft$ 0 (an error is active)
Bit 4: SETPOINT_ACKNOWLEDGE	P1123=1, next Polynom section is expected
Bit 5: QUICK_STOP	is always 0
Bit 6: SWITCH_ON_DISABLED	is always 0
Bit 7: WARNING	P12 $\triangleleft$ 0 (a warning is active)
Bit 8: HANDSHAKE	SERS finished last opcode execution
Bit 9: REMOTE	P0=0 (no running program active)
Bit 10: TARGET_REACHED	P336 = 1 (motor not running / reached its position)
Bit 11: INTERNAL_LIMIT_ACTIVE	P1042 $\triangleleft$ 0, limit position overflow
Bit 12: HOMING_ATTAINED	P403 = 1 after successful homing procedure
Bit 13: FOLLOWING_ERROR	error load angle – only for SERS with option E50
Bit 14: ACCELERATING_PHASE	P1015 $\triangleleft$ 0, motor is accelerating
Bit 15: CONSTANT_PHASE	P1016 $\triangleleft$ 0, motor runs with constant velocity

**'Result', 'Position' and 'Inputs' → see chapter 2.2.6 until 2.2.8**

### 2.3.4 Function principle extended binary mode

#### 1. Write (assignment) and read (request) parameters:

The Opcode for writing and reading parameters has to be set in the bytes 5 and 6. In case of writing parameters, additionally the value to be assigned to the parameter has to be set in the Operand (bytes 7 until 10). A change from "0" to "1" of Bit 15 in the Control Word will signalise the SERS, that there is a new OPCODE with Operand to be accepted and executed. After executing the OPCODE the SERS will set the handshake bit (Status Word bit 8) to "1". Before writing a new Opcode into the SERS, the Bit 15 in the Control Word must be set to "0" and it must be waited for the Handshake Bit (Status Word bit 8) to be reset to "0". When reading (requesting) parameters, the result of the request (the parameters value) can be read in the "result" in Byte 3 until 6 in the Input section (the result is valid when handshake bit = 1)  
**A definition of the Opcode and Operand can be found at pages 6 until 8.**

#### 2. Positioning

The values acceleration, velocity and position command value have to set in the bytes 11 until 22. Bits 17 until 19 in the Control Word it define the mode of positioning (standard point to point positioning or Polynom driving, relative or absolute mode).

Bit 20 in the Control Word tells the SERS to use the values in bytes 11 until 22 (acceleration, velocity, position command value) as new positioning parameters (Bit 20 = 1) or to use the values from the last positioning job (Bit 20 = 0).

An edge from "0" to "1" of Bit 8 in the Control Word results in starting the positioning job. Alternatively a change of Bit 16 (Toggle bit) starts the positioning job – additionally the Bit 8 in the Control Word must be set permanently. The handshake bit 8 in the Status Word follows the Bit 8 (Start\_Positioning) respectively the Togglebit Bit 16 in the Control Word.

During Polynom driving the SERS signalises via Bit 4 in the Status Word that it expects the definition of the next polynom section. The final decelerating ramp (Parameter B - P1096 - definition and scaling as parameter A – P138) can be defined via OPCODE only.

### 3. Diagnostics

The diagnostics of the Profibus-protocol in the SERS...PB-DP is realised as follows:

#### **X1 X2 X3 X4 X5 X6 Z1 D1 E1 E2 W1 W2**

Bytes **X1** until **X6** are specified in the Profibus-DP standard.

Bytes **Z1** until **W2** are written by the SERS...PB-DP.

**Z1** indicates the quantity of the following Bytes (inclusive Z1) - always '06' if there is a diagnostics message.

**D1** indicates a diagnostics error number - see list below.

**E1** corresponds to the High-Byte of the 16-Bit parameter P11 (SERS-drive error).

**E2** corresponds to the Low-Byte of 16-Bit Parameters P11 (SERS-error).

**W1** corresponds to the High-Byte of the 16-Bit Parameter P12 (SERS-warning).

**W2** corresponds to the Low-Byte of the 16-Bit Parameter P12 (SERS-warning ).

#### Diagnostics error numbers (Byte **D1**)

Hex-Wert (Dezimal)

1	(1)	: to big
2	(2)	: to small
3	(3)	: not valid
4	(4)	: invalid output
5	(5)	: EEPROM storage full
6	(6)	: EEPROM acknowledge timeout
7	(7)	: EEPROM no acknowledge
8	(8)	: EEPROM no page begin
9	(9)	: run decimal constant to small
A	(10)	: decimal constant to big
B	(11)	: unknown if event
C	(12)	: admission refused
D	(13)	: Parameter not existing
E	(14)	: adc expected
F	(15)	: end of text expected
10	(16)	: input text only in pgm mode
11	(17)	: text to long
12	(18)	: [decimal constant pgm psave] expected
13	(19)	: * permitted only at P1 or z
14	(20)	: data or z expected
15	(21)	: command expected
16	(22)	: programming mode not active
17	(23)	: if expected
18	(24)	: if event expected
19	(25)	: goto or gosub or GT or GS expected
1A	(26)	: goto or gosub expected
1B	(27)	: goto expected
1C	(28)	: goto decimal constant expected
1D	(29)	: gosub expected
1E	(30)	: gosub decimal constant expected
1F	(31)	: [return RT run rs rf] expected
20	(32)	: return expected
21	(33)	: [decimal constant list ls lf] expected

22	(34)	: = or ? expected
23	(35)	: [decimal constant on off] expected
24	(36)	: decimal constant or n expected
25	(37)	: decimal constant expected
26	(38)	: run expected
27	(39)	: [new, neg, not] expected
28	(40)	: new or neg expected
29	(41)	: list expected
2A	(42)	: quit expected
2B	(43)	: off expected
2C	(44)	: program still running
2D	(45)	: pgm expected
2E	(46)	: Programming mode not active
2F	(47)	: del expected
30	(48)	: data expected
31	(49)	: change: only constant allowed
32	(50)	: decimal constant or pos expected
33	(51)	: pos expected
34	(52)	: psave expected
35	(53)	: [tr tron troff] expected
36	(54)	: program not running
37	(55)	: troff expected
38	(56)	: ver expected
39	(57)	: 1, 2, 3 or 4 expected
3A	(58)	: pos or possave expected
3B	(59)	: lp expected
3C	(60)	: possave expected
3D	(61)	: data or parameter expected
3E	(62)	: semicolon not allowed
3F	(63)	: not expected
40	(64)	: not known status
41	(65)	: program start not possible when service switch is ON
42	(66)	: programmstart not possible, error P11
43	(67)	: stop switch is active
44	(68)	: stop switch is open
45	(69)	: not valid program existing
46	(70)	: drive must stand still
47	(71)	: unknown destination
48	(72)	: destination not valid
49	(73)	: Stack overflow
4A	(74)	: unknown Opcode, forgot return ?
4B	(75)	: unknown fxxx Opcode
4C	(76)	: invalid Opcode for destination address
4D	(77)	: unknown f7xx Opcode
4E	(78)	: limit switch open
4F	(79)	: drive not enabled (OFF)
50	(80)	: unknown positioning mode
51	(81)	: drive must run constant or stand still
52	(82)	: invalid EEPROM mode value
53	(83)	: label already defined: L
54	(84)	: position difference to big

55	(85)	: new position to big
56	(86)	: new position to small
57	(87)	: new residual position to short
58	(88)	: compare position 1 is to big
59	(89)	: compare Position 1 is to small
5A	(90)	: compare Position 2 is to big
5B	(91)	: compare Position 2 is to small
5C	(92)	: new modulo value is to big
5D	(93)	: not writable, during drive is positioning
5E	(94)	: solder bridge is set wrong for this range
5F	(95)	: negative software limit position is bigger than positive position
60	(96)	: exponent to big
61	(97)	: exponent to small
62	(98)	: calculation result is to big
63	(99)	: calculation result is to small
64	(100)	: calculation result is to big to be displayed
65	(101)	: calculation result is to small to be displayed
66	(102)	: division through 0
67	(103)	: bus stopbit is active
68	(104)	: subindex not existing
69	(105)	: value can not be written (read only parameter)
6A	(106)	: value can not be read
6B	(107)	: Polynom with backlash not allowed
6C	(108)	: Missing Polynom data for next section
6D	(109)	: Wait expected
6E	(110)	: <=0 expected
6F	(111)	: =0 expected
70	(112)	: 0 expected
71	(113)	: 0 or 1 expected
72	(114)	: > expected
73	(115)	: 1 expected
74	(116)	: = expected
75	(117)	: 3 expected
76	(118)	: Polynom end to short for deceleration (false definition of polynom section)
77	(119)	: A to small
78	(120)	: A to big
79	(121)	: V to small
7A	(122)	: V to big

In case of an error (SERS parameter P11 <> 0) or a warning (P12 <> 0) a diagnostics message will be created (bytes Z1 until W2 <> 0).

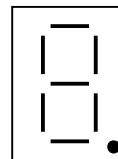
The diagnostics message may be reset by resetting the error / warning via the control word.

In case the error or the warning is still active, the diagnostics message also stays active.

## 4. 7-Segment display

the 7 Segment display shows the actual status of the SERS

Elements of indication :  
7 segments



Error and status indications are indicated by constantly illuminated characters.

Warnings are indicated by blinking characters.

Indication	description	comment
-	initialisation phase	indicated 1 second after power-on-reset
0	Profibus interface on SERS recognized	no external Profibus connection
1	Baudrate found	Profibus connection exists
2	Waiting for configuration telegram	yet no correct configuration telegram received, or wrong slave address
3	Profibus datas are exchanged	motor current can be switched on
4	no Profibus interface at SERS found	internal SERS hardware error
5	phase current of motor is ON positioning jobs can be executed	
7	checksum error of dates in E <sup>2</sup> Prom	adjust standard parameters (factory default) by writing parameter P1004=3
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	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	check power supply
A (blinking)	warning position overflow – positioning job is reaching position limit	- Parameter W (P47) to big - check positioning mode (for endless positioning → P1014=1)
C	Limit switch is open	check limit switches at machine and limit switch inputs of SERS
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 - acceleration to high - external load to big (not enough motor torque)
F (blinking)	warning step angle control – description as F – see P1029	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 description on P11 in the standard SERS manual )

**Reset of a warning with 'P12=0'** (see description on P12 in the standard SERS manual )

## 5.1 Example: Positioning and writing parameters in the binary mode (8/12 Byte I/O)

1. Set velocity for positioning job : P91=1000 (V=1000)
2. Set acceleration for positioning job : P138=2000 (A=2000)
3. Save parameters permanently into SERS E<sup>2</sup>Prom : P1004=2
4. Set position command value for positioning job: P47=3600 (W=3600)
5. Start positioning job via the Control Word
6. Start positioning job again

### 1. P91=1000

#### a) Calculating and writing of the **Operand**:

P91, at rotational scaling (P44=2 – standard adjustment), includes 4 decimal places

→ P91=1000.0000 → write P91=10000000

(the decimal places will be written without the decimal point !)

→ Conversion into HEX-value → 00 98 96 80

→ Operand Bytes 5 - 8 :       Byte 8 7 6 5

→ Writing of the Operand into the OUT-section to the Bytes 5 until 8

Note: Byte 5 is the least significant Byte and Byte 8 is the highest significant Byte in the 32-Bit Operand

#### b) Calculating and writing of the **OPCODE**

The OPCODE must be "0" now. A new OPCODE will be accepted and executed by the SERS if the OPCODE is changed from the value zero to the new value not zero.

The Bytes 3 and 4 of the OPCODE must be transmitted consistent, that means they must be sent to the SERS simultaneously as word.

→ OPCODE for the assignment of P91 :

→ binary 1000 000 0101 1011       (binary 0101 1011 = decimal 91)

Byte       4       3

→ Hex-value for OPCODE for assignment of P91: 80 5B

Byte 4 3

c) After transmitting the OPCODE into the Profibus OUTPUT section of the SERS, the SERS will set the handshake bit in the Status Word after the execution of the OPCODE.

d) Then an OPCODE "00 00" must be transmitted to the SERS

e) After receiving the OPCODE "00 00" the SERS will reset the Handshake bit in the Status Word to "0"

Now the next parameter can be transmitted :

2. **P138=2000** (acceleration is scaled rotational in standard setting - P160=2 - and in this case includes 3 decimal places → value 2000.000 will be converted to 2000000)

a) until e) as under 1) but with

OPERAND Byte 8 = "00" and Byte 7 = "1E" and Byte 6 = "84" and Byte 5 = "80"

OPCODE 4 = "80" and Byte 3 = "8A" (HEX-values)

(all values HEX-values !)

### 3. P1004=2

a) until e) as under 1) but with

OPCODE 4 = "83" and Byte 3 = "EC" (HEX-value)

OPERAND Byte 8 = "00" and Byte 7 = "00" and Byte 6 = "00" and Byte 5 = "02"

(all values HEX-values !)

4. **P47=3600** (position scaled rotational in standard setting - P76=2 – and in this case includes 4 decimal places → 3600.0000 is converted to 36000000)  
a) until e) as under 1) but with  
OPCODE Byte 4 = "80" and Byte 3 = "2F" and  
OPERAND Byte 8 = "02" and Byte 7 = "25" and Byte 6 = "51" and Byte 5 = "00"  
(all values HEX-values !)
5. **Start positioning job via the Control Word**  
Writing "20" (HEX-value) to Byte 1 in OUT-section → switches Motor phase current ON  
Then:  
Writing "01" (HEX-value) to Byte 2 in OUT-section → starts positioning job and therefore Bit 10 in the Status Word will be reset to "0" (motor is running now)  
After reaching the position command value, the SERS will set Bit 10 in the Status Word.  
Then writing "00" to Byte 2 in Out-section (Bit 8 in the Control Word can also be reset already during positioning – after the positioning job was started → check via Bit 10 in the Status Word !)
6. **Start positioning job once again**  
Writing "01" (HEX-value) to Byte 2 in OUT-section → the positioning job will be started again and Bit 10 in the Status Word will be rest to "0" again, ...

**General notes:**

Points 1. until 3. are necessary only once for most applications.

The velocity and the acceleration are defined and then saved permanently into the SERS E<sup>2</sup>Prom.  
But of course these parameters can also be redefined for any positioning job again.

## 5.2 Example: Positioning and writing parameters in the extended binary mode (22/12 Byte I/O)

1. Adjusting scaling of position data to linear scaling : P76=1
2. Adjusting of the feeding constant (assignment 1 motor revolution to [mm]) : P123=5 (mm)
3. Permanent saving of the parameter into the SERS E<sup>2</sup>Prom with command : P1004=2
4. Adjusting of the positioning velocity : V=1000 (rpm)
5. Adjusting of the positioning acceleration : A=2000 (rad/s<sup>2</sup>)
6. Selecting position command value: W=15 (mm)
7. Switching on motor phase current via the control word
8. Starting the positioning job via the control word
9. Starting again the positioning job

### 1. P76=1

Adjusting of linear scaling for all position data (position command value, actual position value, backlash, feeding constant, ..) → all values in [mm] by taking into account the parameter P121 until P123 – gear head ratio and feeding constant)

#### a) calculating and writing of the **OPCODE**

→ OPCODE for assignment of P76 :

→ binary 1000 000 0100 1010 (binary 0100 1010 = decimal 76)  
           byte      6      5

→ Hex-value for OPCODE for the assignment of P76: 80 4C  
   byte 6 5

→ writing the OPCODE 80 4C into the Profibus OUT-section into bytes 6 and 5

#### b) calculating and writing of the **Operand**:

Value (Operand) = 1

→ Changing to HEX-value → 00 00 00 01

→ Operand bytes 7 - 10 : byte 10 9 8 7

→ Writing the operand into the Profibus OUT-section into bytes 7 until 10

Note: Byte 7 is the lowest value byte and byte 10 is the highest value byte in the 32-Bit operand (double word)

#### c) executing the OPCODE (writing the parameter) by setting bit 15 in the control word

→ control word bit 15 = 1 (edge from “0“ to “1“) starts execution of the OPCODE)

→ after 2 to 4ms the SERS will set the handshakebit bit 8 in the status word to “1“

→ afterwards set bit 15 = 0 in the control word

→ 2 to 4ms afterwards the SERS will reset the handshakebit bit 8 in the status word to “0“

### Remarks:

The cycle times of most of the Profibus-Master controls (e.g. PLC) relatively long, what allows to ignore the evaluation of the handshakebit in the status word → set and reset of bit 15 in the control word with a delay of 2ms will be sufficient. Another 2ms afterwards the next parameter (OPCODE) may be written (bit 15 in the control word may be set).

	control word	OPCODE	operand	
byte no.	1 2 3 4	5 6	7 8 9 10	
a) + b)	00 00 00 00	<b>4C 80</b>	01 00 00 00	(write OPCODE and Operand - P76=1)
c)	00 <b>80</b> 00 00	4C 80	01 00 00 00	(execute OPCODE)
c)	00 <b>00</b> 00 00	4C 80	01 00 00 00	(reset OPCODE - bit 15 to zero)

**2. P123=5**

e.g. in case of a spindle with 5mm pitch (1 motor revolution → linear movement of 5mm)

- calculating the OPCODE as under 1. → OPCODE = 80 7B (HEX-value)
- calculating the operand:
  - value=5 , P123 is a scaled value (because of P76=1 → in [mm] with 4 decimal places – see SERS-manual P76 page 54)
  - value=5.0000 → **value=50000** → operand = C3 50 (HEX-value)
- set control word bit 15 and reset it (possibly control via handshakebit bit 8 in the status word) for executing the OPCODE

	control word				OPCODE		operand				
byte no.	1	2	3	4	5	6	7	8	9	10	
a) + b)	00	00	00	00	<b>7B</b>	<b>80</b>	50	C3	00	00	(OPCODE and operand – P123=5.0000)
c)	00	<b>80</b>	00	00	7B	80	50	C3	00	00	(OPCODE execute)
c)	00	<b>00</b>	00	00	7B	80	50	C3	00	00	(OPCODE Bit 15 reset to zero)

**3. P1004=2**

Permanent saving of the parameters adjusted at 1. and 2. into the E<sup>2</sup>PROM of the SERS.  
Calculating the OPCODE and the operand and executing the OPCODE as at 1.

	control word				OPCODE		operand				
byte no.	1	2	3	4	5	6	7	8	9	10	
	00	00	00	00	<b>EC</b>	<b>83</b>	02	00	00	00	(OPCODE and Operand – P1004=2)
	00	<b>80</b>	00	00	EC	83	02	00	00	00	(OPCODE execute)
	00	<b>00</b>	00	00	EC	83	02	00	00	00	(OPCODE Bit 15 reset to zero)

**4. V=1000 (U/min)**

Setting the positioning speed V in the Profibus OUT-section bytes 15 until 18

- calculating the value to be set:
  - V (P91), at rotational scaling for speed values (P44=2 – standard setting), includes 4 decimal places (see SERS-manual parameter P44) → V=1000.0000 → writing V=10000000 (the decimal places are written without decimal point !)
  - changing into HEX-value → 80 96 98 00
  - bytes 15 - 18 :                      byte 15 16 17 18
  - writing the HEX-value into the Profibus OUT-section into the bytes 15 until 18
  - remarks: byte 15 is the lowest value byte and byte 18 is the highest value byte within the 32-bit value

**5. A=2000 (rad/s<sup>2</sup>)**

Setting the positioning acceleration A in the Profibus OUT-section bytes 11 until 14

- calculating the values to be set:
  - A (P138), at rotational scaling (P138=2 – standard setting), includes 3 decimal places (see SERS-manual P160) → A=2000.000 → write A=2000000 (the decimal places are written without the decimal point !)
  - change into HEX-value → 80 84 1E 00
  - bytes 11 - 14 :                      byte 11 12 13 14
  - writing the HEX-value into the Profibus OUT-section into the bytes 11 until 14
  - remarks: byte 11 is the lowest value byte and byte 14 is the highest value byte within the 32-bit value

**6. W=15 (mm)**

Setting the position command value in the Profibus OUT-section bytes 19 until 22

a) Calculating the value to be set:

W (P47), at linear scaling (P76=1, adjusted at 1.), includes 4 decimal points

(see SERS-manual parameter P76) → W=15.0000 → write W=150000

(the decimal places are written without the decimal point !)

→ change into HEX-value → 00 02 49 F0

→ bytes 19 - 22 :       Byte 22 21 20 19

→ write the HEX-value into the Profibus OUT-section into bytes 19 until 22

remarks: byte 11 is the lowest value byte and byte 14 is the highest value byte within the 32-bit value

Negative position command values are indicated by writing the complement by 2 value of the 32-bit value (example: - 0,0001 mm → - 1 → is written as HEX-value FF FF FF FF)

**7. Switching on motor phase current**

Switching on motor phase current via → control word bit 5 = 1

**8. Start positioning job**

Setting the positioning mode and start the positioning job

a) positioning mode: in the example “relative positioning” is used – the motor shall move the indicated value “W” (= distance) relative to the actual position.

→ control word bit 17 = 0 (in case of absolute positioning bit 17 = 1 must be set)

b) start positioning job

→ control word bit 20 = 1 (use values A, V, W in bytes 11 until 22 for the positioning job) and bit 8 = 1 (edge from “0“ to “1“ in bit 8 starts the positioning)

c) wait for bit 10 in the status word (TARGET\_REACHED) – the SERS will set bit 10 = 0 after starting the positioning job (2 to 4ms after setting bit 8 in the control word)

Then bit 8 in the control word may be reset to “0“.

After reaching the position the SERS will set bit 10 = 1 in the status word.

Remarks: Reset bit 8 in the control word may be done also later.

	control word				OPCODE		operand				accelerat. A				velocity V				position W			
byte no.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
A,V,W	20	00	00	00	00	00	00	00	00	00	80	84	1E	00	80	96	98	00	F0	49	02	00
bit 20=1	20	00	10	00	00	00	00	00	00	00	80	84	1E	00	80	96	98	00	F0	49	02	00
start pos.	20	01	10	00	00	00	00	00	00	00	80	84	1E	00	80	96	98	00	F0	49	02	00
	20	00	10	00	00	00	00	00	00	00	80	84	1E	00	80	96	98	00	F0	49	02	00

**9. Start positioning job again**

Starting the positioning job again with the same values for A, V, W

→ Set bit 8 in the control word (edge from “0“ to “1“ starts the positioning job again

– bit 20 in the control word and values A, V, W are still set !)

start pos. 20 01 10 00   00 00   00 00 00 00   80 84 1E 00   80 96 98 00   F0 49 02 00

**General notes:**

Points 1. to 3. are necessary only once for most applications.

These parameters are defined and saved permanently into the E<sup>2</sup>PROM of the SERS.

Of course these values can be redefined any time.

For writing any other SERS-parameter please follow the principle as shown in the above example under 1. to 3.

## 5.3 Typical parameters to be set

For most applications there are only few basic parameters to be set. A detailed description of the below listed basic parameters can be found in the general manual “**SERS manual**“ (for SERS drives with RS232-interface) chapter 4.7 page 41 to 71. All other additional parameter implemented in the SERS (described in the SERS manual) can be used for many different special functions.

1. Phase current parameter **P1010** (to be adjusted always depending on the connected motor)
  - e.g. for 4A phase current → P1010=4000 (in [mA])
2. Scaling mode
  - a) **P76** – scaling position data → all position data (position command value, actual position, feeding constant, backlash, ...)
    - e.g. for linear systems (e.g. spindle applications) P76 = 1 must be set
  - b) **P44** – scaling velocity data → all velocity values (positioning, homing, jog mode)
    - e.g. at linear systems (e.g. spindle applications) for indicating the values in [mm/min] P44 = 1 must be set
  - b) **P160** – scaling acceleration data → all acceleration data (positioning, homing, jog mode)

The scaling for position data, velocity data and acceleration data may be set to different values (e.g. position data linear in [mm] – P76=1 – and velocity data in [rpm] – P44=2 or P44=66)
3. Parameter of the mechanics
  - a) feeding constant **P123** (scaled value – depending on P76)
  - b) gear ratio constants **P121** and **P122** (not scaled values)
4. Homing parameter (P41, P42 and P1003 are scaled values)
  - a) general adjustments for homing parameter **P147** (e.g. homing direction)
  - b) velocity homing **P41** and **P1003** and acceleration homing **P42**
5. Jog / manual drive parameter (all jog parameter are scaled parameter)
  - a) jog velocity slow and fast **P019** and **P1020**
  - b) jog acceleration **P1018**
6. Stop (deceleration) ramp in case of error or at active STOP → **P1030** (scaled value)
 

When an error occurs (e.g. limit switch open) or the Stop-input gets active (external or (Stop-Bit in the control word), then the motor decelerates always with a ramp defined P1030 (independent whether the drive is positioning, homing or in jog mode at the moment)
7. Permanent saving of all changed parameter into the SERS E<sup>2</sup>PROM by writing the parameter P1004 → **P1004=2** (not scaled value)

All parameters are pre-adjusted. So some of the parameter listed above, are adjusted with values suitable for many applications already:

**P1010=6000 [mA]** for SERS 06.. versions (for SERS 02... → 2800, for SERS 12... → 8000)  
**P76=2, P44=2, P160=2** (all scaled rotational, P121 until P123 are not used then)  
**P147=4** (homing to the input “homing switch” in positive direction)  
**P42=500 [rad/s<sup>2</sup>], P41=1000 [rpm], P1003=100 [rpm]**  
**P1018=500 [rad/s<sup>2</sup>], P1019=30 [rpm], P1020=150 [rpm]**  
**P1030=4000 [rad/s<sup>2</sup>]**

## 5.4 Example: Polynom driving with extended binary mode (22/12 Byte I/O)

A velocity profile consisting of 3 sections can be programmed as follows  
(the values in the example are based on rotational scaling):

### Initial status:

Control Word (Byte 1 - 4) = 0;

Opcode (Byte 5 - 6) = 0;

Operand (Byte 7 - 10) = 0;

Acceleration (Byte 11 - 14) e.g. = 1000000 (decimal); acceleration = 1000.000 rad/s<sup>2</sup>

Velocity (Byte 15 - 18) e.g. = 30000000 (decimal); 3000.0000 rpm

Position command value (Byte 19 - 22) e.g. = 3600000 (decimal); 360.0000 degree

### 1. section Start:

Acceleration = 1500000 (decimal)

Velocity = 20000000 (decimal)

Position command value = 7200000 (decimal)

Control word = 0x00150120 (Hex) (Byte4 = 00, Byte3 = 15, Byte2 = 01 , Byte1 = 20)

Start relative Polynom section (relative positioning in polynom driving mode)

Bit 8 (START\_POSITIONING) and Bit 15 (START\_POSITIONING\_TOGGLE ) are set both for the first start.

Waiting until, Profibus-buffer is accepted and evaluated by the SERS:

→ Wait, until Handshake =1 (Status Word, Bit 8: HANDSHAKE)

Waiting, until the next data (definition of next polynom section) can be loaded:

→ Wait, until SETPOINT\_ACKNOWLEDGE = 1 (Status Word, Bit 4)

### 2. section write and start (will be started right after the end of the first section):

Acceleration = 1800000 (decimal)

Velocity = 25000000 (decimal)

Position command value = 3600000 (decimal)

Control Word = 0x00140120 (Hex) start relative Polynom section, Togglebit (Bit 15) is changed

Wait, until Handshake =1 (Status Word, Bit 8: HANDSHAKE)

Wait until data are accepted (will be only at end of first polynom section):

Wait, until SETPOINT\_ACKNOWLEDGE = 1 (Status Word, Bit 4)

### Last section write and start:

Acceleration = 2000000 (decimal)

Velocity = 12000000 (decimal)

Position command value = 7200000 (decimal)

Control Word = 0x001D0120 start last Polynom section

After Handshake = 1 (Status Word, Bit 8: HANDSHAKE) → reset handshake bits, for not causing a new start and for releasing the execution of new commands via "Opcode":

Set Control Word = 0