



## **Stögra Antriebstechnik GmbH**

Machtlfinger Straße 24  
D-81379 München

Tel.: (089)15904000  
Fax.: (089) 15904009  
email [info@stoegra.de](mailto:info@stoegra.de)  
internet <http://www.stoegra.de>

# **SERS 02, SERS 06 und SERS 12**

## **Version V**

### **PB-DP**

Schrittmotorleistungsverstärkerkarte mit  
Positioniersteuerung und Profibus-DP Schnittstelle

**Profibus-DP spezifische Ergänzungen zum Handbuch SERS  
mit RS232/RS485 Schnittstelle**

Ausgabe Juni 2004

Änderungen, die der Verbesserung dienen, bleiben vorbehalten.

Bei der Erstellung von Texten und Bildern wurde mit höchster Sorgfalt  
vorgegangen. Trotzdem können Fehler nicht vollständig ausgeschlossen werden.  
Für fehlerhafte Angaben und deren Folgen können wir keine Haftung übernehmen.

## Inhaltsverzeichnis

	Seite
1. Allgemeine Hinweise	3
1.1 Kurzübersicht	3
1.2 Handbuchhinweise	3
1.3 GSD-File und Protokollmodi	3
1.4 Übersicht der Potokollbesonderheiten im ASCII-Zeichenmode im Vergleich zur SERS mit RS232	4
2. Profibus-DP Protokoll in der SERS-PB-DP	5
2.1 ASCII-Zeichenmode : E/A-Bereich SERS...PB-DP – Kommunikation mit Hilfe eines Toggle-Bytes ( <b>alter Modus</b> )	5
2.2 Binärer Mode (8/12 Byte I/O) ( <b>alter Modus</b> )	6
2.2.1 Übersicht	6
2.2.2 Kontrollwort	6
2.2.3 Operationscode (Opcode)	6
2.2.4 Operand	9
2.2.5 Status	10
2.2.6 Ergebnis	10
2.2.7 Position	10
2.2.8 Eingänge	10
2.3 <b>Erweiterter Binärer Mode (22/12 Byte I/O) (aktueller Modus)</b>	11
2.3.1 Übersicht	11
2.3.2 Kontrollwort	11
2.3.3 Status	12
2.3.4 Funktionsprinzip erweiterter binärer Mode	12
3. Diagnose	13
4. 7-Segmentanzeige	16
5. Beispiele	
5.1 Beispiel: Positionieren und Parameterschreiben im binären Mode (8/12 Byte I/O)	17
5.2 Beispiel: Positionieren und Parameterschreiben im erweiterten binären Mode (22/12 Byte I/O)	19
5.3 Typisch einzustellende Parameter	22
5.4 Beispiel: Polynomfahren im erweiterten binären Mode (22/12 Byte I/O)	23

# 1. Allgemeine Hinweise

## 1.1 Kurzübersicht

- Die Schrittmotor-Ansteuereinheit SERS...PB-DP ist eine 1-Achsen Positioniersteuerung mit Profibus-DP Interface.
- Die entsprechenden 'Knoten' – Informationen für den Profibus-Master sind im auf der Diskette (oder STÖGRA-CD oder Internet download unter <http://www.stoegra.de>) mitgelieferten File 'stoegra5.gsd' enthalten.
- Drei verschiedene Protokoll-Modi : ASCII-Zeichen Mode und Binärer Mode und erweiterter Binärer Mode
- Die implementierten Funktionen der SERS...PB-DP sind identisch mit der Positioniersteuerung SERS mit RS232/RS485 Schnittstelle.

## 1.2 Handbuchhinweise

Im Folgenden wird das "Handbuch zur Inbetriebnahme und Programmierung der SERS mit RS232/RS485 Schnittstelle" als **SERS-Handbuch** bezeichnet.

Folgende Einschränkungen im Vergleich zur SERS mit RS232/RS485 Schnittstelle und das entsprechende SERS-Handbuch gelten:

1. Am 8-poligen DIP-Schalter 1 (siehe SERS-Handbuch Seite 11 und Seite 15) sind die Schalterbits 1 bis 3 ohne Funktion (Einstellung der Baudrate bei der RS232/RS485-Version).
2. Die Belegung der 9-poligen D-Sub-Buchse ist entsprechend der Profibus-DP-Norm

## 1.3 GSD-File und Protokollmodi

In dem mitgeliefertem File 'stoegra5.gsd' können verschiedene Protokollmodi gewählt werden:

- "SERP Anweisungen 32 Byte I/O" - ASCII-Zeichenmode – Senden und Empfangen von ASCII-Zeichen mit 32 Byte im Ein- und Ausgangsbereich
- "SERP Anweisungen 16 Byte I/O" - ASCII-Zeichenmode – Senden und Empfangen von ASCII-Zeichen mit 16 Byte im Ein- und Ausgangsbereich
- "SERP binär 8/12 Byte I/O" – Binärer Mode – Setzen und Lesen von Bits in Kontroll-/Statuswörtern und Senden/Empfangen von Operationscodes, Operanden, Ergebnissen über 8 Byte im Ausgangsbereich und 12 Byte im Eingangsbereich
- "SERP binär 8/12 Byte I/O konsis" – Binärer Mode mit konsistenter Datenübertragung und 8 Byte im Ausgangsbereich und 12 Byte im Eingangsbereich
- "SERP binär 22/12 Byte I/O" → **erweiterter binärer Mode mit konsistenter Datenübertragung und 22 Byte im Ausgangsbereich und 12 Byte im Eingangsbereich**

**Für neue Projekte empfehlen wir die Verwendung des erweiterten binären Modus (22/12 Byte I/O) !**

## 1.4 Übersicht der Potokollbesonderheiten im ASCII-Zeichenmode im Vergleich zur SERS mit RS232

1. Eine Zeichenfolge muß mit einem Carriage-Return (Hex-Code '0D') und einem "Null"-Byte (Hex-Code '00') abgeschlossen werden (siehe unten Kapitel '2. Profibus-DP Protokoll in der SERS-PB-DP') – in der RS232/RS485-Version werden Anweisungszeilen nur mit dem Carriage-Return Zeichen abgeschlossen.
2. Kommandos werden erst mit dem Wechsel des Toggle-Bits in der SERS...PB-DP übernommen (siehe unten Kapitel 2)
3. Die Anweisungen **dürfen nicht** mit dem '#'-Zeichen begonnen werden. (In der RS232/RS485-Version muß jede neue Anweisungszeile mit dem '#'-Zeichen begonnen werden)
4. Wenn die Anweisungen vom Master korrekt an die SERS gesendet wurden (korrekte Syntax und kein logischer oder Ablauffehler) dann schreibt die SERS...PB-DP die Quittungen folgendermaßen in den IN-Bereich:

1. Byte ist das Toggle-Byte (siehe Kapitel 2), das 2. Byte ist ein Linefeed-Zeichen (Hex-Code '0A') , danach beginnt die Quittungsmeldung (z.B: 'ok1') und danach wird wieder ein Linefeed-Zeichen gefolgt von einem Carriage-Return (Hex-Code '0D') und einem 'Null'-Byte gesendet. Das Null-Byte beendet die aktuelle Meldung.

Falls ein Fehler vorliegt wird nur in das 2. Byte ein 'Null'-Byte geschrieben und die Meldung der SERS damit beendet. Die Werte nach dem Null-Byte sind nicht mehr relevant.

z.B.

IN-Bereich (z.B bei 32-Byte):

Ohne Fehler:	Bei Fehler in der Anweisung im OUT-Bereich
01 0A 6F 6B 31 0A 0D 00	01 00 6F 6B 33 0A 0D 00    im 2.Byte steht 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 Protokoll in der SERS-PB-DP

### 2.1. ASCII-Zeichenmode : E/A-Bereich SERS...PB-DP Kommunikation mit Hilfe eines Toggle-Bytes

Die Kommunikation erfolgt nach der Profibus-DP Norm über einen IN-Bereich und einen OUT-Bereich im Profibus-Master und in der SERS...PB-DP.

Das 1. Byte im IN- und OUT-Bereich der SERS...PB-DP wird als Toggle-Byte verwendet.

Folgender Ablauf zwischen Profibus-Master und SERS...PB-DP ergibt sich bei der Kommunikation mit Hilfe des Toggle-Bytes.

1. Der Profibus Master schreibt eine Anweisung in den OUT-Bereich der SERS...PB-DP und schreibt einen zum vorherigen Wert veränderten neuen Wert in das Toggle-Byte im OUT-Bereich (z.B. Toggle-Byte wird um eins erhöht oder wechselt von '0' auf '1' bzw. umgekehrt)
2. Durch das geänderte Toggle-Byte erkennt die SERS...PB-DP, daß eine neue Anweisung vom Master gesendet wurde. Die SERS...PB-DP führt diese Anweisung aus, schreibt eine Quittung (z.B. ok1, ok0, ... oder pgm – siehe dazu SERS-Handbuch Seite 19) in den IN-Bereich, und übernimmt den geänderten Wert des Toggle-Bytes aus dem OUT-Bereich in den IN-Bereich.
3. Mit der Übernahme des Toggle-Byte-Werts in den IN-Bereich signalisiert die SERS...PB-DP dem Master, daß sie die Anweisung vom Master übernommen hat, falls vom Master Daten angefordert wurden diese jetzt zum Abholen bereit liegen und die nächste Anweisung vom Master an die SERS...PB-DP gesendet werden kann.

**Alle Anweisung an die SERS werden als ASCII-Zeichen gesendet, wie im SERS-Handbuch beschrieben.**

Jede Anweisung wird mit einem Carriage Return und einem nachfolgendem '0'-Byte beendet und mit dem Toggle-Byte der SERS...PB-DP angezeigt.

z.B.:

OUT-Bereich (z.B bei 32-Byte):

Am Anfang:	Schreiben von 'ON' (zum Bestromen des Motors)
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' im ersten Byte im OUT-Bereich ist das Toggle-Byte

'4F' entspricht dem Hexadezimalen ASCII-Code für das Zeichen 'O'

'4E' entspricht dem Hexadezimalen ASCII-Code für das Zeichen 'N'

'0D' entspricht dem Hexadezimalen ASCII-Code für Carriage Return CR

'00' ist ein Null-Byte

'0A' entspricht dem Hexadezimalen ASCII-Code für Linefeed LF

IN-Bereich der SERS...PB-DP:

Am Anfang:	Nach Übernahme der 'ON'-Anweisung
00 00 00 00 00 00 00 00	01 0A 4F 4E 31 0A 0D 00    entspricht LF + 'OK1' +
00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00    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

## 2.2 Binärer Mode (8/12 Byte I/O)

### 2.2.1 Übersicht

Senden von Daten an die SERS...PB-DP über 8 Byte im Ausgangsbereich:

Byte 1 und 2 : **Kontrollwort**

Byte 3 und 4 : **Operationscode (Opcode)**

Byte 5 bis 8 : **Operand**

Und Lesen von Daten aus der SERS...PB-DP über 12 Byte im Eingangsbereich:

Byte 1 und 2 : **Status**

Byte 3 bis 6 : **Ergebnis** (z.B. einer zuvor erfolgten Parameterabfrage)

Byte 7 bis 10 : **Aktuelle Position**

Byte 11 und 12 : **Eingänge**

### 2.2.2 Kontrollwort

Das Kontrollwort im Ausgangsbereich triggert **bei Veränderung** die folgend definierten Aktionen. "STOP" wird immer (auch ohne Änderung) berücksichtigt.

("START\_PROGRAMM" wird auch statisch ausgewertet, wenn der Parameter "P1023" auf 1 gesetzt ist).

Bit 0: LANGSAM_NEGATIV Handfahren	1 = aktiv, 0 = Stop
Bit 1: LANGSAM_POSITIV Handfahren	1 = aktiv, 0 = Stop
Bit 2: SCHNELL_NEGATIV Handfahren	1 = aktiv, 0 = Stop
Bit 3: SCHNELL_POSITIV Handfahren	1 = aktiv, 0 = Stop
Bit 4: REFERENZFAHREN	1 = aktiv, 0 = Stop
Bit 5: STROM_EINSCHALTEN	1 = ein, 0 = aus
Bit 6: STOP	1 = Stop aktiv, 0 = Fahren möglich
Bit 7: START_PROGRAMM	1 = Start
Bit 8: START_POSITIONIERUNG	Flanke von 0 auf 1 = Start
Bit 9: AUSGANG 1	1 = EIN (nur wenn kein Ablaufprogramm aktiv ist !)
Bit 10: AUSGANG 2	1 = EIN (nur wenn kein Ablaufprogramm aktiv ist !)
Bit 11: AUSGANG 3	1 = EIN (nur wenn kein Ablaufprogramm aktiv ist !)
Bit 12: AUSGANG 4	1 = EIN (nur wenn kein Ablaufprogramm aktiv ist !)
Bit 13: LOESCHEN_FEHLER	Flanke von 0 auf 1 → P11=0
Bit 14: LOESCHEN_WARNUNG	Flanke von 0 auf 1 → P12=0

### 2.2.3 Operationscode (Opcode)

- Der Opcode wird ausgeführt bei einem Wechsel von 0 auf ungleich 0.
- Die beiden Bytes des Opcodes müssen konsistent übertragen werden, damit kein ungültiger Opcode entsteht.
- Je nach Operation wird der Operand benötigt. Dieser muss gültig sein, bevor der Opcode ungleich 0 gesetzt wird.
- Der übertragene Opcode wird nach seiner Ausführung mit "Bit 8: HANDSHAKE" im Statuswort des Eingangsbereich quittiert.
- Bevor ein neuer Opcode übertragen werden kann, muss der Opcode auf 0 gesetzt und gewartet werden, bis das Handshake Bit auch wieder 0 ist.
- Die wichtigsten Opcodes für einen nicht programmierten Antrieb sind "Zuweisung" und "Abfrage".

In der folgenden Darstellung sind alle Opcodes **binär** dargestellt.

Das "c" Bit wird fuer die Formatierung bei ASCII Darstellung verwendet. Wenn das Bit "1" ist, so beginnt dieser Opcode im Listing (z.B. Programmlisting) in einer neuen Zeile. Im "nicht programmierten Antrieb" (kein Ablaufprogramm in SERS) ist "c" normalerweise "0".

Die "p" Bits bilden die Parameternummer. (ppp pppp pppp) bezeichnet den Inhalt des Parameters mit der Adresse "ppp pppp pppp" (z.B.: . 011 1111 0010 für Parameter P1010)

Die "n" Bits bilden die Anzahl der Dezimalnachkommastellen.

Die "e" Bits bilden die Ereignisnummer bei bedingten Verzweigungen.

Die "b" Bits bilden das Eingangsbitmuster bei bedingten Verzweigungen.

Die "l" Bits bilden die Labelnummer. "X" bezeichnet den Rechenakku.

Beschreibung	Opcode, binär Byte 4 / Byte 3	Bedeutung
Addition	0000 cppp pppp pppp 0000 c111 nnnn nnnn	$X = X + (ppp pppp pppp)$ $X = X + \text{operand}$
Subtraktion	0001 cppp pppp pppp 0001 c111 nnnn nnnn	$X = X - (ppp pppp pppp)$ $X = X - \text{operand}$
Multiplikation	0010 cppp pppp pppp 0010 c111 nnnn nnnn	$X = X * (ppp pppp pppp)$ $X = X * \text{operand}$
Division	0011 cppp pppp pppp 0011 c111 nnnn nnnn	$X = X / (ppp pppp pppp)$ $X = X / \text{operand}$
Und	0100 cppp pppp pppp 0100 c111 nnnn nnnn	$X = X \& (ppp pppp pppp)$ $X = X \& \text{operand}$
Oder	0101 cppp pppp pppp 0101 c111 nnnn nnnn	$X = X   (ppp pppp pppp)$ $X = X   \text{operand}$
excl. Oder	0110 cppp pppp pppp 0110 c111 nnnn nnnn	$X = X \wedge (ppp pppp pppp)$ $X = X \wedge \text{operand}$
Akku-Zuweisung	0111 cppp pppp pppp 0111 c111 nnnn nnnn	$X = (ppp pppp pppp)$ $X = \text{operand}$
<b>Parameter Zuweisung</b>	<b>1000 cppp pppp pppp</b> 1001 cppp pppp pppp 1010 cppp pppp pppp	<b>(ppp pppp pppp) = operand</b> (ppp pppp pppp) = operand nur 16 Bit (ppp pppp pppp) = X
<b>Parameter Abfrage</b>	<b>1011 cppp pppp pppp</b>	<b>operand = (ppp pppp pppp)</b>
Verzweigung	1101 ceee eeee eeee 1101 c111 bbbb bbbb 1110 ceee eeee eeee 1110 c111 bbbb bbbb	if (eee eeee eeee) if (bbbb bbbb) if !(eee eeee eeee) if !(bbbb bbbb)
Sprung	1111 c000 llll llll	GOTO llll llll
Unterprogramm	1111 c001 llll llll	GOSUB llll llll
Label	1111 c010 llll llll	Label llll llll
Programm Start	1111 c011 llll llll	RUN llll llll (z.B. RUN 5)
Text	1111 c100 llll llll	Text mit Textlänge llll llll
Invertieren	1111 c111 0000 0000	$X = !X$
Negation	1111 c111 0000 0001 1111 c111 0000 0010	$X = -X$ no operation
<b>Start</b>	<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
Programmende	1111 c111 0000 0111	PE
Ende	1111 c111 1111 1111	ENDE

**Ereignisnummern ("e") für Verzweigungen** if (eee eeee eeee)

Ereignis Nr.	Ereignis wahr
0	Programm läuft
1	Eingang 1 ist aktiv
2	Eingang 2 ist aktiv
3	Eingang 3 ist aktiv
4	Eingang 4 ist aktiv
5	Eingang 5 ist aktiv
6	Eingang 6 ist aktiv
7	Eingang 7 ist aktiv
8	Eingang 8 ist aktiv
11	Zustandsklasse 1 ungleich 0
12	Zustandsklasse 2 ungleich 0
100	Zähler 1 ist größer als 1
101	Zähler 2 ist größer als 1
102	Zähler 3 ist größer als 1
336	Antrieb steht still
1015	Antrieb beschleunigt
1016	Antrieb fährt konstant
1042	Grenzposition überschritten
1047	X ist größer als 0
1101	Merker 1 ist ungleich 0
1102	Merker 2 ist ungleich 0
1103	Merker 3 ist ungleich 0
1201	Ausgang 1 ist aktiv
1202	Ausgang 2 ist aktiv
1203	Ausgang 3 ist aktiv
1204	Ausgang 4 ist aktiv

Die Ereignisnummern sind identisch mit den SERS Parameter Nummern !  
Z.B. Ereignis "336" = Parameter P336

**Ablaufprogramme in SERS ... PB-DP Steuerungen:**

Das Schreiben / Übertragen eines neuen Ablaufprogramms in eine SERS ... PB-DP, wird mit der Parameterzuweisung 'P0=2' eingeleitet.

Achtung: 'P0=2' ist eine Zuweisung mit einem 16-Bit Operand !

→ OPCODE 1010 0000 0000 0000 = 90 00 (HEX) und Operand 00 00 00 02 (HEX)

Alle anderen Parameterzuweisungen mit 32-Bit Operanden

→ OPCODE 1000 .... = 8... ..

Alle in obiger OPCODE-Liste enthaltenen Kommandos können in einem Ablaufprogramm verwendet werden.

Ein mit 'P0=2' eingeleitetes Definieren eines Ablaufprogramms wird mit der Parameterzuweisung 'P0=0' wieder beendet (Ende des Ablaufprogramms)

→ OPCODE 1000 0000 0000 0000 = 80 00 (HEX) und Operand 00 00 00 00 (HEX)

Über das Kontrollwort Bit 7 kann das Ablaufprogramm gestartet werden.

Bit 9 im Statuswort zeigt an, ob ein Ablaufprogramm gerade läuft.

## 2.2.4 Operand

### Zahlendarstellung im Binärformat

Die Anzahl der Nachkommastellen sind durch die Parameternummer (im Opcode) und der Wichtungsart des Parameters festgelegt. In der Binärarstellung wird die Kommaposition nicht gespeichert, sondern einfach weggelassen. Es müssen jedoch alle Nachkommastellen angehängt werden.

Z.B. wird die Position "360.6" Grad bei rotatorischer Wichtung im Binärformat als "3606000" angegeben, weil dieser Parameter bei dieser Wichtung 4 Nachkommastellen hat.

Bei arithmetischen Berechnungen mit Konstante wird die Anzahl der Nachkommastellen im Opcode im Feld "nnnn nnnn" gespeichert, da es hier keine Standardwichtung gibt.

Die Wichtungen (Skalierungen) werden eingestellt über die Parameter P44, P76 und P160 (siehe allg. Handbuch "Bedienungsanleitung SERS" - für SERS mit RS232-Schnittstelle - Seite 54)

Folgend sind alle Parameter mit Nachkommastellen aufgelistet:

Parameter	Nachkommastellen
41	entsprechend eingestellter Wichtung
42	entsprechend eingestellter Wichtung
47 (W)	entsprechend eingestellter Wichtung
51	entsprechend eingestellter Wichtung
91 (V)	entsprechend eingestellter Wichtung
103	entsprechend eingestellter Wichtung
108	2 → 100% entspricht 1.00
123	entsprechend eingestellter Wichtung
138 (A)	entsprechend eingestellter Wichtung
1003	entsprechend eingestellter Wichtung
1005 - 1008	2 → 100% entspricht 1.00
1012	entsprechend eingestellter Wichtung
1018	entsprechend eingestellter Wichtung
1019	entsprechend eingestellter Wichtung
1020	entsprechend eingestellter Wichtung
1024	entsprechend eingestellter Wichtung
1026	entsprechend eingestellter Wichtung
1030	entsprechend eingestellter Wichtung
1037	entsprechend eingestellter Wichtung
1039	entsprechend eingestellter Wichtung
1040	entsprechend eingestellter Wichtung
1041	entsprechend eingestellter Wichtung
1044	entsprechend eingestellter Wichtung
1046	2 → 100% entspricht 1.00
1047 (X)	entsprechend Berechnungsergebnis
1052	entsprechend eingestellter Wichtung
1053	entsprechend eingestellter Wichtung
1054	5
1080... (R0...)	wie zugewiesener Parameter
1100 (D)	1

Alle anderen Parameter sind ungewichtet (also ohne Nachkommastellen).

## 2.2.5 Status

Der Status hat folgende Bitdefinitionen (angelehnt an DRIVECOM):

Bitposition und Bedeutung	gesetztes Bit bedeutet
Bit 0: READY_TO_SWITCH_ON	ist immer 1
Bit 1: SWITCHED_ON	P134 (Master-Steuerwort) ist ungleich 0
Bit 2: OPERATION_ENABLED	P134 (Master-Steuerwort) ist ungleich 0
Bit 3: FAULT	P11 (Zustandsklasse-1) ist ungleich 0
Bit 4: VOLTAGE_DISABLED	ist immer 0
Bit 5: QUICK_STOP	ist immer 0
Bit 6: SWITCH_ON_DISABLED	ist immer 0
Bit 7: WARNING	P12 (Zustandsklasse-2) ist ungleich 0
Bit 8: HANDSHAKE	Antrieb hat opcode ausgeführt
Bit 9: REMOTE	P0 ist 0
Bit 10: TARGET_REACHED	P336 ist 1
Bit 11: INTERNAL_LIMIT_ACTIVE	P1042 ungleich 0, Grenzposition ueberschritten
Bit 12: HOMING_ATTAINED	P403 ist 1 nach erfolgreichem Referenzfahren
Bit 13: FOLLOWING_ERROR	Lastwinkelfehler - nur bei Version mit Encoder
Bit 14: BESCHLEUNIGUNGSPHASE	P1015 ungleich 0, Beschleunigungsphase
Bit 15: KONSTANTLAUFPHASE	P1016 ungleich 0, Konstantphase

## 2.2.6 Ergebnis

"ergebnis" ist der letzte mit dem Opcode "Abfrage" ausgelesene Parameterwert – als Binärwert.

## 2.2.7 Position

"position" enthält die aktuelle Position (P51, Lage-Istwert 1) – als Binärwert.

## 2.2.8 Eingänge

"Eingänge" enthält den Zustand der digitalen Eingänge I1 bis I8 und der optoentkoppelten Eingänge Stop, Referenzschalter, Endschalter plus und minus.

Bit 0: Eingang 1	Bit 8: Endschalter Minus
Bit 1: Eingang 2	Bit 9: Endschalter Plus
Bit 2: Eingang 3	Bit 10: Stopschalter
Bit 3: Eingang 4	Bit 11: Referenzschalter
Bit 4: Eingang 5	Bit 12: Intern
Bit 5: Eingang 6	Bit 13: Intern
Bit 6: Eingang 7	Bit 14: Intern
Bit 7: Eingang 8	Bit 15: Intern

## 2.3 Erweiterter binärer Mode (22/12 Byte I/O)

### 2.3.1 Übersicht

Senden von Daten an die SERS...PB-DP über 22 Byte im Ausgangsbereich:

Byte 1 bis 4 : **Kontrollwort**

Byte 5 und 6 : **Operationscode (Opcode)**

Byte 7 bis 10 : **Operand**

Byte 11 bis 14 : **Beschleunigung**

Byte 15 bis 18 : **Geschwindigkeit**

Byte 19 bis 22 : **Lagesollwert / Weg**

Lesen von Daten aus der SERS...PB-DP über 12 Byte im Eingangsbereich:

Byte 1 und 2 : **Status**

Byte 3 bis 6 : **Ergebnis** (z.B. einer zuvor erfolgten Parameterabfrage)

Byte 7 bis 10 : **Aktuelle Position**

Byte 11 und 12 : **Eingänge**

### 2.3.2 Kontrollwort

Das Kontrollwort im Ausgangsbereich triggert **bei Veränderung** die folgend definierten Aktionen. "STOP" wird immer (auch ohne Änderung) berücksichtigt. ("START\_PROGRAMM" wird auch statisch ausgewertet, wenn Parameter P1023 = 1 ist)

Bit 0: LANGSAM_NEGATIV Handfahren	1 = aktiv, 0 = Stop
Bit 1: LANGSAM_POSITIV Handfahren	1 = aktiv, 0 = Stop
Bit 2: SCHNELL_NEGATIV Handfahren	1 = aktiv, 0 = Stop
Bit 3: SCHNELL_POSITIV Handfahren	1 = aktiv, 0 = Stop
Bit 4: REFERENZFAHREN	1 = aktiv, 0 = Stop
Bit 5: STROM_EINSCHALTEN	1 = ein, 0 = aus
Bit 6: STOP	1 = Stop aktiv, 0 = Fahren möglich
Bit 7: START_PROGRAMM	1 = Start
Bit 8: START_POSITIONIERUNG	Flanke von 0 auf 1 = Start
Bit 9: AUSGANG 1	1 = EIN (nur wenn kein Ablaufprogramm aktiv ist !)
Bit 10: AUSGANG 2	1 = EIN (nur wenn kein Ablaufprogramm aktiv ist !)
Bit 11: AUSGANG 3	1 = EIN (nur wenn kein Ablaufprogramm aktiv ist !)
Bit 12: AUSGANG 4	1 = EIN (nur wenn kein Ablaufprogramm aktiv ist !)
Bit 13: LOESCHEN_FEHLER	Flanke von 0 auf 1 → P11=0
Bit 14: LOESCHEN_WARNUNG	Flanke von 0 auf 1 → P12=0
Bit 15: OPCODE_AUSFUEHREN	Flanke von 0 → 1 = "opcode" wird ausgeführt
Bit 16: START_POSITIONIERUNG_TOGGLE	bei Bit-Änderung wird der Antrieb gestartet.
Bit 17: ABSOLUT	1 = setzt bei einem Wegstart den absoluten Positioniermodus (P1014 = 2), 0 = setzt den relativen Positioniermodus (P1014 = 0), wenn vorher absolut aktiv war. Die relativen Positioniermodi (P1014 = 1, P1014 = 3) bleiben unverändert, wenn diese vorher gesetzt waren.
Bit 18: POLYNOM	1 = aktiviert die Polynom Positionierung (wie WP)
Bit 19: POLYNOM_TERM	1 = kennzeichnet das Polynom Ende (wie WPT)
Bit 20: PARAMETER_UEBERNEHMEN	die Parameter "Beschleunigung", "Geschwindigkeit" und "Lagesollwert" werden bei einem Start durch das Kontrollwort verwendet.
Bit 21 bis 31:	reserviert

**'OPCODE' und 'Operand' → Siehe Kapitel 2.2.3 bis 2.2.4**

### 2.3.3 Status

Bitposition und Bedeutung	gesetztes Bit bedeutet
Bit 0: READY_TO_SWITCH_ON	ist immer 1
Bit 1: SWITCHED_ON	P134 $\triangleleft$ 0 (Motor Phasenstrom ist ON)
Bit 2: OPERATION_ENABLED	P134 $\triangleleft$ 0 (Motor Phasenstrom ist ON)
Bit 3: FAULT	P11 $\triangleleft$ 0 (ein SERS Fehler ist aufgetreten)
Bit 4: SETPOINT_ACKNOWLEDGE	P1123 = 1, nächsten Polynomabschnitt laden
Bit 5: QUICK_STOP	ist immer 0
Bit 6: SWITCH_ON_DISABLED	ist immer 0
Bit 7: WARNING	P12 $\triangleleft$ 0 (eine SERS Warnung ist aufgetreten)
Bit 8: HANDSHAKE	Antrieb hat opcode ausgeführt
Bit 9: REMOTE	P0 = 0 (kein Ablaufprogramm aktiv)
Bit 10: TARGET_REACHED	P336 = 1 (Motor steht still)
Bit 11: INTERNAL_LIMIT_ACTIVE	P1042 $\triangleleft$ 0, Grenzposition überschritten
Bit 12: HOMING_ATTAINED	P403 = 1 nach erfolgreichem Referenzfahren
Bit 13: FOLLOWING_ERROR	Lastwinkelfehler - nur bei Version mit Encoder
Bit 14: BESCHLEUNIGUNGSPHASE	P1015 $\triangleleft$ 0, Antrieb in Beschleunigungsphase
Bit 15: KONSTANTLAUFPHASE	P1016 $\triangleleft$ 0, Antrieb in Konstantphase (V=Konstant)

**'Ergebnis', 'Position' und 'Eingänge' → Siehe Kapitel 2.2.6 bis 2.2.8**

### 2.3.4 Funktionsprinzip erweiterter binärer Mode

#### 1. Allgemeines Schreiben und Lesen von Parametern:

In den Bytes 5 und 6 wird der Opcode für das Schreiben oder Lesen von Parametern eingetragen (Zuweisung). Beim Schreiben von Parametern wird zusätzlich der Wert, der zugewiesen werden soll, im Operand eingetragen (Bytes 7 bis 10). Mit einem Wechsel von "0" nach "1" von Bit 15 im Kontrollwort wird dem Slave signalisiert, dass ein neuer OPCODE mit Operand übernommen werden soll.

Nach Abarbeitung des OPCODE's setzt der Slave das Handshakebit auf "1". Bevor ein neuer Opcode übertragen werden kann, muss das Bit 15 im Kontrollwort auf 0 gesetzt und gewartet werden, bis das Handshake Bit auch wieder 0 ist. Beim Lesen von Parametern steht jetzt das Ergebnis (Parameterwert) im "Ergebnis" des Eingangsbereichs (Byte 3 bis 6).

**Eine Beschreibung von Opcode und Operand finden Sie auf den Seiten 6 bis 8.**

#### 2. Positionieren

Die Werte Beschleunigung, Geschwindigkeit und Lagesollwert werden in die entsprechenden Bereiche (Bytes 11 bis 22) geschrieben. Über die Bits 17 bis 19 im Kontrollwort wird festgelegt, um welche Art der Positionierung es sich handelt (einfache Positionierung oder Polynomfahrt, relativ oder absolut).

Mit Bit 20 wird dem Slave mitgeteilt, ob er die neuen Werte Beschleunigung, Geschwindigkeit und Position übernehmen soll.

Eine Flanke von "0" nach "1" von Bit 8 im Kontrollwort führt zum Start der Positionierung. Alternativ startet eine Änderung des Bit 16 (Togglebit) die Positionierung - Bit 8 im Kontrollwort muss dabei permanent gesetzt sein. Das Handshakebit 8 im Statuswort folgt dem Bit 8 (Start\_Positionierung) im Kontrollwort bzw. dem Togglebit Bit 16 im Kontrollwort.

Beim Polynomfahren signalisiert der Slave über das Bit 4, dass der nächste Polynomabschnitt geschrieben werden kann. Die letzte Bremsrampe Parameter "B" (Parameter P1096) - Definition und Wichtung wie Parameter A - P138) kann nur über OPCODE definiert werden.

### 3. Diagnose

Die Diagnose im Profibus-Protokoll der SERS...PB-DP ist folgendermaßen aufgebaut:

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

Die Bytes **X1** bis **X6** sind in der Profibus-DP Norm festgelegt.

Die Bytes **Z1** bis **W2** werden durch die SERS...PB-DP beschrieben.

**Z1** gibt die Anzahl der folgenden Bytes an (inklusive Z1) - immer '06', wenn eine Diagnosemeldung vorliegt.

**D1** gibt eine Diagnose-Fehlernummer an, die unten beschrieben ist.

**E1** entspricht dem High-Byte des 16-Bit Parameters P11 (SERS-Antriebsfehler).

**E2** entspricht dem Low-Byte des 16-Bit Parameters P11 (SERS- Antriebsfehler).

**W1** entspricht dem High-Byte des 16-Bit Parameters P12 (SERS-Warnung).

**W2** entspricht dem Low-Byte des 16-Bit Parameter P12 (SERS-Warnung ).

Diagnose Fehlernummern (Byte **D1**)

Hex-Wert (Dezimal)

1	(1)	: zu groß
2	(2)	: zu klein
3	(3)	: ungültig
4	(4)	: ungültiger Ausgang
5	(5)	: EEPROM Speicher voll
6	(6)	: EEPROM acknowledge timeout
7	(7)	: EEPROM no acknowledge
8	(8)	: EEPROM no page begin
9	(9)	: run Dezimalkonstante zu klein
A	(10)	: Dezimalkonstante zu groß
B	(11)	: unbekanntes if Ereignis
C	(12)	: Zugriff verweigert
D	(13)	: Parameter nicht vorhanden
E	(14)	: adc erwartet
F	(15)	: Textende erwartet
10	(16)	: Texteingabe nur im pgm Modus
11	(17)	: Text zu lang
12	(18)	: [Dezimalkonstante pgm psave] erwartet
13	(19)	: * nur bei P1 oder z erlaubt
14	(20)	: Datum oder z erwartet
15	(21)	: Anweisung erwartet
16	(22)	: Programmiermodus nicht aktiv
17	(23)	: if erwartet
18	(24)	: if Ereignis erwartet
19	(25)	: goto oder gosub oder GT oder GS erwartet
1A	(26)	: goto oder gosub erwartet
1B	(27)	: goto erwartet
1C	(28)	: goto Dezimalkonstante erwartet
1D	(29)	: gosub erwartet
1E	(30)	: gosub Dezimalkonstante erwartet
1F	(31)	: [return RT run rs rf] erwartet
20	(32)	: return erwartet
21	(33)	: [Dezimalkonstante list ls lf] erwartet
22	(34)	: = oder ? erwartet

23	(35)	: [Dezimalkonstante on off] erwartet
24	(36)	: Dezimalkonstante oder n erwartet
25	(37)	: Dezimalkonstante erwartet
26	(38)	: run erwartet
27	(39)	: [new, neg, not] erwartet
28	(40)	: new oder neg erwartet
29	(41)	: list erwartet
2A	(42)	: quit erwartet
2B	(43)	: off erwartet
2C	(44)	: Programm läuft noch
2D	(45)	: pgm erwartet
2E	(46)	: Programmiermodus nicht aktiv
2F	(47)	: del erwartet
30	(48)	: Datum erwartet
31	(49)	: Change: nur Konstante erlaubt
32	(50)	: Dezimalkonstante oder pos erwartet
33	(51)	: pos erwartet
34	(52)	: psave erwartet
35	(53)	: [tr tron troff] erwartet
36	(54)	: Programm läuft nicht
37	(55)	: troff erwartet
38	(56)	: ver erwartet
39	(57)	: 1, 2, 3 oder 4 erwartet
3A	(58)	: pos oder possave erwartet
3B	(59)	: lp erwartet
3C	(60)	: possave erwartet
3D	(61)	: Datum oder Parameter erwartet
3E	(62)	: Komma nicht erlaubt
3F	(63)	: not erwartet
40	(64)	: unbekannter Zustand
41	(65)	: Programmstart nicht möglich wenn Serviceschalter an ist
42	(66)	: Programmstart nicht möglich, Zustandsklasse-1 Fehler
43	(67)	: Stop Schalter ist aktiv
44	(68)	: Stop Schalter ist offen
45	(69)	: kein gültiges Programm vorhanden
46	(70)	: Antrieb muss stehen
47	(71)	: Sprungziel unbekannt
48	(72)	: Sprungziel ungültig
49	(73)	: Stack Überlauf
4A	(74)	: unbekannter Opcode, return vergessen ?
4B	(75)	: unbekannter fxxx Opcode
4C	(76)	: ungültiger Opcode für Zieladresse
4D	(77)	: unbekannter f7xx Opcode
4E	(78)	: Endschalter offen
4F	(79)	: Antrieb ist nicht freigegeben
50	(80)	: unbekannter Positioniermodus
51	(81)	: Antrieb muss konstant fahren oder stehen
52	(82)	: ungültiger EEPROM Modus Wert
53	(83)	: label bereits definiert: L
54	(84)	: Die Wegdifferenz ist zu groß
55	(85)	: Die neue Position ist zu groß

56	(86)	: Die neue Position ist zu klein
57	(87)	: neuer Restweg zu kurz
58	(88)	: Vergleich Position 1 ist zu groß
59	(89)	: Vergleich Position 1 ist zu klein
5A	(90)	: Vergleich Position 2 ist zu groß
5B	(91)	: Vergleich Position 2 ist zu klein
5C	(92)	: Der neue Modulwert ist zu groß
5D	(93)	: nicht schreibbar, während der Antrieb positioniert
5E	(94)	: Die Lötbrücke ist fuer diesen Bereich falsch eingestellt.
5F	(95)	: Die negative Grenzposition ist größer als die positive
60	(96)	: Exponent zu groß
61	(97)	: Exponent zu klein
62	(98)	: Das Rechenergebnis ist zu groß
63	(99)	: Das Rechenergebnis ist zu klein
64	(100)	: Das Rechenergebnis ist zu groß zum anzeigen
65	(101)	: Das Rechenergebnis ist zu klein zum anzeigen
66	(102)	: Division durch 0
67	(103)	: Bus Stopbit ist aktiv
68	(104)	: Subindex nicht vorhanden
69	(105)	: Wert kann nicht geschrieben werden
6A	(106)	: Wert kann nicht gelesen werden
6B	(107)	: Polynom mit Getriebeispiel nicht erlaubt
6C	(108)	: Keine neuen Polynomdaten verfügbar
6D	(109)	: Wait erwartet
6E	(110)	: <=0 erwartet
6F	(111)	: =0 erwartet
70	(112)	: 0 erwartet
71	(113)	: 0 oder 1 erwartet
72	(114)	: > erwartet
73	(115)	: 1 erwartet
74	(116)	: = erwartet
75	(117)	: 3 erwartet
76	(118)	: Polynom Ende zu kurz zum Anhalten (fehlerhafte Definition Polynom Abschnitt)
77	(119)	: A zu klein
78	(120)	: A zu groß
79	(121)	: V zu klein
7A	(122)	: V zu groß

Bei Auftreten eines Fehlers (SERS Parameter P11<>0) oder einer Warnung (P12 <>0) wird eine entsprechende Diagnosemeldung erzeugt (Bytes Z1 bis W2 werden belegt).

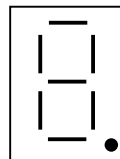
Die Diagnosemeldung kann durch Löschen des Fehlers bzw. der Warnung (über das Kontrollwort) zurückgesetzt werden.

Falls der Fehler bzw. die Warnung nach wie vor aktiv ist, bleibt die Meldung bestehen.

## 4. 7-Segmentanzeige

Die 7-Segment Anzeige zeigt den Status der SERS an

Anzeigeelemente :  
7 Segmente



Fehler und Statusmeldungen werden durch konstant leuchtende Zeichen angezeigt.

Vorwarnungen werden durch blinkende Zeichen gemeldet.

Anzeige	Beschreibung	Kommentar
-	SERS Initialisierung	Wird 1 Sekunde lang nach einem Power-On-Reset angezeigt - Initialisierungsphase
0	Profibus Aufsatzplatine erkannt	extern kein Profibus angeschlossen
1	Baudrate eingestellt	physikalische Profibusverbindung vorhanden
2	Warten auf Konfigurationstelegramm	noch kein korrektes Konfigurationstelegramm gesendet oder falsche Slaveadresse
3	Datenaustausch findet statt	Motor kann jetzt bestromt werden
4	kein Profibusinterface auf der SERS gefunden	interner SERS-Hardwarefehler
5	Motor ist bestromt	Antrieb kann verfahren werden
7	E <sup>2</sup> PROM Checksummenfehler	Stellen Sie die Standardparameter (Auslieferungszustand) durch Schreiben von P1004=3 ein
8	Temperaturfehler - ab 85°C (± 10%) am Kühlblech	Überprüfen Sie die Kühlung der SERS - Belüftung (Lüfter und 24VDC in ELK) o.k.?
9	Fehler Unterspannung	Prüfen Sie die SERS-Versorgungsspannung
8 (blinkend)	Temperaturvorwarnung - ab 75°C (± 10%) am Kühlblech	Siehe Kommentar für Anzeige '8'
9 (blinkend)	Vorwarnung Unterspannung	Prüfen Sie die SERS-Versorgungsspannung
A (blinkend)	Warnung Positionsüberlauf - der aktuelle Positioniervorgang erreicht die Positionsgrenze	der Parameter W (P47) ist zu groß Überprüfen Sie den Positioniermodus - für Endlosbetrieb muß P1014=1 sein - Seite 48
C	Ein Endschalter ist angefahren	Überprüfen sie die Endschalter - siehe 2.1.2
E	Fehler Kurzschluß im Motor oder auf der Verstärkerkarte	Bei der Installation : Überprüfen Sie den korrekten Anschluß der Motorkabel
F	Fehler Lastwinkelüberwachung der Motor konnte der Positionssollwertvorgabe nicht folgen siehe Beschreibung P1029	Nur bei Option Lastwinkelüberwachung - überprüfen Sie den Encoderanschluß - die Beschleunigung ist zu hoch - das externe Drehmoment ist zu hoch
F (blinkend)	Warnung Lastwinkelüberwachung	Siehe Kommentar unter F
H (blinkend)	Warnung Programmfehler - das Ablaufprogramm wurde angehalten wegen eines Fehlers	Überprüfen Sie die Parameter und Label im Ablaufprogramm
L (blinkend)	Warnung Softwareendschalter	Die aktuelle Position hat den in P1040 oder P1041 eingestellten Grenzwert erreicht

**Das Löschen einer Fehlermeldung erfolgt durch 'P11=0'**

**Das Löschen einer Warnmeldung erfolgt durch 'P12=0'**

## 5.1 Beispiel: Positionieren und Parameterschreiben im binären Mode (8/12 Byte I/O)

1. Setzen der Verfahrgeschwindigkeit beim Positionieren : P91=1000 (V=1000)
2. Setzen der Beschleunigung beim Positionieren : P138=2000 (A=2000)
3. Dauerhaftes Abspeichern der Parameter im SERS E<sup>2</sup>Prom : P1004=2
4. Setzen eines Lagesollwerts : P47=3600 (W=3600)
5. Starten der Positionierung über das Kontrollwort
6. Nochmaliges Starten der Positionierung

### 1. P91=1000

#### a) Ermitteln und Schreiben des Operanden:

P91, bei rotatorischer Wichtung (P44=2 -StandardEinstellung), hat 4 Nachkommastellen → P91=1000.0000 → Schreiben von P91=10000000

(die Nachkommastellen werden ohne das "Komma" eingetragen !)

→ Umwandeln in HEX-Wert → 00 98 96 80

→ Operand Bytes 5 - 8 : Byte 8 7 6 5

→ Eintragen des Operanden im OUT-Bereich in den Bytes 5 bis 8

Bemerkung: Byte 5 ist das niederwertigste Byte und Byte 8 ist das höchstwertigste Byte in dem 32-Bit Operanden

#### b) Ermitteln und Schreiben des OPCODE

Der OPCODE muß jetzt "0" sein. Ein neuer OPCODE wird durch eine Änderung vom Wert "0" auf einen Wert ungleich Null von der SERS verarbeitet.

Die beiden Bytes 3 und 4 des OPCODE müssen konsistent übertragen, d.h. gleichzeitig als Wort in den OUT-Bereich eingetragen werden.

→ OPCODE für Zuweisung von P91 :

→ binär 1000 000 0101 1011 (binär 0101 1011 = dezimal 91)

Byte 4 3

→ Hex-Wert für OPCODE für Zuweisung von P91: 80 5B

Byte 4 3

c) Nach Eintrag des OPCODE in den Profibus OUT-Bereich setzt die SERS nach Verarbeitung des OPCODE das Handshakebit im Statuswort.

d) Danach muß der OPCODE wieder mit "00 00" beschrieben werden

e) Nach dem Erhalt des Wertes "00 00" im OPCODE setzt die SERS das Handshakebit im Statuswort wieder auf "0"

Jetzt kann der nächste Parameter geschrieben werden :

### 2. P138=2000 (Beschleunigung standardmäßig rotatorisch gewichtet - P160=2 - und hat dabei 3 Nachkommastellen → aus 2000 wird 2000000)

a) bis e) wie bei 1) aber mit

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

OPCODE 4 = "80" und Byte 3 = "8A" (HEX-Werte)

(alles HEX-Werte !)

### 3. P1004=2

a) bis e) wie bei 1) aber mit

OPCODE 4 = "83" und Byte 3 = "EC" (HEX-Werte)

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

(alles HEX-Werte !)

4. **P47=3600** (Weg standardmäßig rotatorisch gewichtet - P76=2 - und hat dabei 4 Nachkommastellen → aus 3600 wird 36000000)  
a) bis e) wie bei 1) aber mit  
OPCODE Byte 4 = "80" und Byte 3 = "2F" und  
OPERAND Byte 8 = "02" und Byte 7 = "25" und Byte 6 = "51" und Byte 5 = "00"  
(alles HEX-Werte !)
5. **Starten der Positionierung über das Kontrollwort**  
Schreiben von "20" (HEX-Wert) in Byte 1 im OUT-Bereich → dadurch wird der Motor bestromt  
Danach:  
Schreiben von "01" (HEX-Wert) in Byte 2 im OUT-Bereich → dadurch wird die Positionierung gestartet und Bit 10 im Statuswort wird gelöscht  
Wenn der Antrieb seine Sollposition erreicht hat, setzt die SERS das Bit 10 im Statuswort.  
Danach Schreiben von "00" in Byte 2 im Out-Bereich (das Bit 8 im Kontrollwort kann auch schon während dem Positionieren zurückgenommen werden - nachdem die Positionierung gestartet wurde !)
6. **Nochmaliges Starten der Positionierung über das Kontrollwort**  
Schreiben von "01" (HEX-Wert) in Byte 2 im OUT-Bereich → dadurch wird die Positionierung nochmals gestartet und Bit 10 im Statuswort wieder gelöscht usw.

**Allgemeine Bemerkung:**

Die Punkte 1. bis 3. sind für die meisten Anwendungen nur einmal notwendig.

Die Geschwindigkeit und die Beschleunigung werden definiert und dauerhaft im EEPROM abgespeichert. Natürlich können diese Werte aber auch vor jeder neuen Positionierung wieder neu definiert werden.

## 5.2 Beispiel: Positionieren und Parameterschreiben im erweiterten binären Mode (22/12 Byte I/O)

1. Einstellen der Skalierung (Wichtung) von Positionswerten auf translatorisch : P76=1
2. Einstellen der Vorschubkonstante (Zuordnung 1 Motorumdrehung zu [mm]) : P123=5 (mm)
3. Dauerhaftes Abspeichern der Parameter im SERS E<sup>2</sup>Prom mit Kommando : P1004=2
4. Setzen der Verfahrgeschwindigkeit beim Positionieren : V=1000 (U/min)
5. Setzen der Beschleunigung beim Positionieren : A=2000 (rad/s<sup>2</sup>)
6. Setzen des Lagesollwerts : W=15 (mm)
7. Bestromen des Motors (Einschalten des Motorphasenstroms)
8. Starten der Positionierung über das Kontrollwort
9. Erneuter Start der Positionierung

### 1. P76=1

Einstellen von translatorischer Wichtung (Skalierung) für alle Positionswerte (Lagesoll, Lageist, Backlash, Vorschubkonstante, usw.) → alle Werte in mm unter Berücksichtigung der Parameter P121 bis P123 – Getriebekonstanten und Vorschubkonstante)

#### a) Ermitteln und Schreiben des **OPCODE**

→ OPCODE für Zuweisung von P76 :

→ binär 1000 000 0100 1010 (binär 0100 1010 = dezimal 76)

Byte 6 5

→ Hex-Wert für OPCODE für Zuweisung von P76: 80 4C

Byte 6 5

→ Eintragen des OPCODE 80 4C im OUT-Bereich in den Bytes 6 und 5

#### b) Ermitteln und Schreiben des **Operanden**:

Wert (Operand) = 1

→ Umwandeln in HEX-Wert → 00 00 00 01

→ Operand Bytes 7 - 10 : Byte 10 9 8 7

→ Eintragen des Operanden im OUT-Bereich in den Bytes 7 bis 10

Bemerkung: Byte 7 ist das niederwertigste Byte und Byte 10 ist das höchstwertigste Byte in dem 32-Bit Operanden (Doppelwort)

#### c) Ausführen des OPCODES (Parameterschreiben) durch Setzen von Bit 15 im Kontrollwort

→ Kontrollwort Bit 15 = 1 (Flanke von "0" auf "1") startet Ausführung vom OPCODE)

→ nach 2 bis 4ms setzt die SERS das Handshakebit Bit 8 im Statuswort auf "1"

→ danach im Kontrollwort Bit 15 = 0 setzen

→ 2 bis 4ms danach setzt die SERS das Handshakebit Bit 8 im Statuswort auf "0" zurück

#### Bemerkung:

Die meisten Profibus-Master (z.B. SPS) haben relativ große Zykluszeiten, durch die auf die Auswertung des Handshakebits verzichtet werden kann → Setzen und Rücksetzen des Bit 15 im Kontrollwort mit einer Pause von 2ms dazwischen ist ausreichend. 2ms danach kann der nächste Parameter (OPCODE) geschrieben werden (Bit 15 im Kontrollwort gesetzt werden).

	Kontrollwort				OPCODE		Operand				
Byte	1	2	3	4	5	6	7	8	9	10	
a) + b)	00	00	00	00	<b>4C</b>	<b>80</b>	01	00	00	00	(OPCODE und Operand eintragen - P76=1)
c)	00	<b>80</b>	00	00	4C	80	01	00	00	00	(OPCODE Ausführen)
c)	00	<b>00</b>	00	00	4C	80	01	00	00	00	(OPCODE Bit 15 zurück auf Null setzen)

**2. P123=5**

z.B. bei einer Spindel mit 5mm Steigung (1 Motorumdrehung → lineare Bewegung von 5mm)

- Ermitteln des OPCODES wie unter 1. → OPCODE = 80 7B (HEX-Wert)
- Ermitteln des Operanden:
  - Wert=5 , P123 ist ein gewichteter Wert (durch P76=1 → in [mm] mit 4 Nachkommastellen – siehe SERS-Handbuch P76 auf Seite 54)
  - Wert=5.0000 → **Wert=50000** → Operand = C3 50 (HEX-Wert)
- Kontrollwort Bit 15 setzen und rücksetzen (evtl. Kontrolle über Handshakebit Bit 8 im Statuswort) um OPCODE auszuführen

Byte Nr.	Kontrollwort				OPCODE		Operand				
	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 und Operand – P123=5.0000)
c)	00	<b>80</b>	00	00	7B	80	50	C3	00	00	(OPCODE Ausführen)
c)	00	<b>00</b>	00	00	7B	80	50	C3	00	00	(OPCODE Bit 15 zurück auf Null setzen)

**3. P1004=2**

Dauerhaftes Abspeichern der unter 1. und 2. eingestellten Parameter im E<sup>2</sup>PROM der SERS  
Ermitteln von OPCODE und Operand und Ausführen des OPCODE wie unter 1.

Byte Nr.	Kontrollwort				OPCODE		Operand				
	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 und Operand – P1004=2)
	00	<b>80</b>	00	00	EC	83	02	00	00	00	(OPCODE Ausführen)
	00	<b>00</b>	00	00	EC	83	02	00	00	00	(OPCODE Bit 15 zurück auf Null setzen)

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

Eintragen der Positioniergeschwindigkeit V im OUT-Bereich Bytes 15 bis 18

- Ermitteln des einzutragenden Wertes:
    - V (P91), bei rotatorischer Wichtung (P44=2 -Standardeinstellung), hat 4 Nachkommastellen (siehe SERS-Handbuch P44) → V=1000.0000 → Schreiben von V=10000000 (die Nachkommastellen werden ohne das "Komma" eingetragen !)
    - Umwandeln in HEX-Wert → 80 96 98 00
    - Bytes 15 - 18 :                      Byte 15 16 17 18
    - Eintragen des HEX-Wertes im OUT-Bereich in den Bytes 15 bis 18
- Bemerkung: Byte 15 ist das niederwertigste Byte und Byte 18 ist das höchstwertigste Byte in dem 32-Bit Wertes

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

Eintragen der Positionierbeschleunigung A im OUT-Bereich Bytes 11 bis 14

- Ermitteln des einzutragenden Wertes:
    - A (P138), bei rotatorischer Wichtung (P138=2 -Standardeinstellung), hat 3 Nachkommastellen (siehe SERS-Handbuch P160) → A=2000.000 → Schreiben von A=2000000 (die Nachkommastellen werden ohne das "Komma" eingetragen !)
    - Umwandeln in HEX-Wert → 80 84 1E 00
    - Bytes 11 - 14 :                      Byte 11 12 13 14
    - Eintragen des HEX-Wertes im OUT-Bereich in den Bytes 11 bis 14
- Bemerkung: Byte 11 ist das niederwertigste Byte und Byte 14 ist das höchstwertigste Byte in dem 32-Bit Wertes

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

Eintragen der Position bzw. des Positionierwegs im OUT-Bereich Bytes 19 bis 22

a) Ermitteln des einzutragenden Wertes:

W (P47), bei translatorischer Wichtung (P76=1 unter 1. eingestellt), hat 4 Nachkommastellen (siehe SERS-Handbuch P76) → W=15.0000 → Schreiben von W=150000 (die Nachkommastellen werden ohne das "Komma" eingetragen !)

→ Umwandeln in HEX-Wert → 00 02 49 F0

→ Bytes 19 - 22 :           Byte 22 21 20 19

→ Eintragen des HEX-Wertes im OUT-Bereich in die Bytes 19 bis 22

Bemerkung: Byte 19 ist das niederwertigste Byte und Byte 22 ist das höchstwertigste Byte in dem 32-Bit Wertes

Bemerkung: Negative Positionierwerte werden durch das 2er-Komplement des 32-Bitwerte angegeben (Beispiel : -0,0001 → -1 mm → entspricht FF FF FF FF)

**7. Bestromen des Motors**

Einschalten des Motorphasenstroms durch → Kontrollwort Bit 5 = 1

**8. Start der Positionierung**

Festlegen Positionierart und Start der Positionierung

a) Positionierart: Im Beispiel wird "Relativ Positionierung" (Kettenmaß) verwendet – von der aktuellen Position aus soll der Wert in "W" relativ positioniert werden.

→ Kontrollwort Bit 17 = 0 (bei Absolut Positionierung müsste Bit 17 = 1 gesetzt werden)

b) Start Positionierung

→ Kontrollwort Bit 20 = 1 (Werte A, V, W in Bytes 11 bis 22 für Positionierung übernehmen) und Bit 8 = 1 (Flanke von "0" auf "1" in Bit 8 startet die Positionierung)

c) Warten auf Bit 10 im Statuswort (TARGET\_REACHED) – Bit 10 wird von der SERS auf "0" gesetzt nach Start der Positionierung (2 bis 4ms nach Setzen von Bit 8 im Kontrollwort) Danach kann Bit 8 im Kontrollwort wieder zurückgesetzt werden auf "0".

Wenn die Position erreicht ist, dann setzt die SERS im Statuswort Bit 10 auf "1"

Bemerkung: Das Rücksetzen von Bit 8 im Kontrollwort kann auch später erfolgen.

	Kontrollwort				OPCODE		Operand				Beschl. A				Geschw. V				Position W			
Byte Nr.	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. Erneuter Start der Positionierung**

Erneutes Positionieren mit den selben Werten für A, V, W

→ Setzen von Bit 8 im Kontrollwort (Flanke von "0" auf "1" startet die Positionierung erneut – Bit 20 im Kontrollwort und die Werte A, V, W sind nach wie vor gesetzt !)

Start Pos. 20 01 10 00    00 00    00 00 00 00    80 84 1E 00    80 96 98 00    F0 49 02 00

**Allgemeine Bemerkung:**

Die Punkte 1. bis 3. sind für die meisten Anwendungen nur einmal notwendig.

Diese Parameter werden definiert und dauerhaft im E<sup>2</sup>PROM der SERS abgespeichert. Natürlich können diese Werte aber jederzeit neu definiert werden.

Das Schreiben aller SERS-Parameter funktioniert nach dem Prinzip, wie im Beispiel unter 1. bis 3. dargestellt.

## 5.3 Typisch einzustellende Parameter

Für die meisten Anwendungen müssen nur ein paar wenige typische Parameter einmal eingestellt werden. Eine detaillierte Beschreibung der im Folgenden gelisteter Parameter finden Sie im allgemeinen Handbuch **“Bedienungsanleitung SERS“** (für SERS mit RS232-Schnittstelle) ab Kapitel 4.7 auf Seite 41. Alle anderen zusätzlichen Parameter in der SERS (beschrieben im allgemeinen Handbuch) können für viele verschiedene Sonderfunktionen verwendet werden.

1. Phasenstrom Parameter **P1010** (muss immer abhängig vom angeschlossenen Schrittmotor eingestellt werden) - z.B. bei 4A Phasenstrom → P1010=4000 (in [mA])
2. Skalierung (Wichtungsart)
  - a) **P76** – Lagewichtung → alle Positionsdaten (sowohl Sollwerte als auch Lageistwert) → z.B. für lineare Systeme (Spindelantriebe) muss P76 = 1 gesetzt werden
  - b) **P44** – Geschwindigkeitswichtung → alle Geschwindigkeitswerte (Positionieren, Referenzfahren und Handfahren) → z.B. um bei linearen Systemen (Spindelantriebe) die Werte in [mm/min] angeben zu können, muss P44 = 1 gesetzt werden
  - b) **P160** – Beschleunigungswichtung → alle Beschleunigungswerte (Positionieren, Referenzfahren und Handfahren)

Die Wichtungen für Lagewerte, Geschwindigkeitswerte und Beschleunigungswerte dürfen auch unterschiedlich eingestellt werden (z.B. Lagedaten in [mm] – P76=1 – und Geschwindigkeitswerte in [U/min] – P44=2 oder P44=66)
3. Parameter der angeschlossenen Mechanik
  - a) Vorschubkonstante **P123** (gewichteter Wert – abhängig von P76)
  - b) Getriebekonstanten (Übersetzungsverhältnis) **P121** und **P122** (ungewichtete Werte)
4. Referenzfahrparameter (P41, P42 und P1003 sind gewichtete Parameter)
  - a) allgemeine Referenzfahreinstellung Parameter **P147** (z.B. Referenzfahrrichtung)
  - b) Geschwindigkeit Referenzfahren **P41** und **P1003** und Beschleunigung Referenzfahren **P42**
5. Handfahrparameter (alle Handfahrparameter sind gewichte Parameter)
  - a) Geschwindigkeit Handfahren langsam und schnell **P019** und **P1020**
  - b) Beschleunigung Handfahren **P1018**
6. Bremsrampe im Fehlerfall oder bei aktivem STOP → **P1030** (gewichteter Wert)  
Wenn ein Fehler auftritt (z.B. Endschalter angefahren) oder der Stop-Eingang aktiv wird (extern oder Stop-Bit über Kontrollwort), dann bremst der Motor immer mit der in P1030 definierten Rampe ab (egal ob der Antrieb gerade Positioniert, Referenz fährt oder im Handfahrmodus ist)
7. Dauerhaftes Abspeichern aller geänderten Parameter im SERS E<sup>2</sup>PROM durch Beschreiben von Parameter P1004 → **P1004=2** (ungewichteter Wert)

Einige der oben gelisteten Parameter sind bereits mit für viele Anwendungen passenden Werten voreingestellt:

**P1010=6000 [mA]** bei SERS 06.. Versionen (bei SERS 02... → 2800, bei SERS 12... → 8000)  
**P76=2, P44=2, P160=2** (alle rotatorisch gewichtet, P121 bis P123 werden dann nicht benötigt)  
**P147=4** (Referenzfahren auf Referenzschalter in positive Drehrichtung)  
**P42=500 [rad/s<sup>2</sup>], P41=1000 [U/min], P1003=100 [U/min]**  
**P1018=500 [rad/s<sup>2</sup>], P1019=30 [U/min], P1020=150 [U/min]**  
**P1030=4000 [rad/s<sup>2</sup>]**

## 5.4 Beispiel: Polynomfahren im erweiterten binären Mode (22/12 Byte I/O)

Ein Geschwindigkeitsprofil bestehend aus 3 Teilen wird folgendermaßen programmiert  
(im Beispiel angegebene Werte setzen rotatorische Wichtung voraus):

### Ausgangszustand:

Kontrollwort (Byte 1 - 4) = 0;

Opcode (Byte 5 - 6) = 0;

Operand (Byte 7 - 10) = 0;

Beschleunigung (Byte 11 - 14) = 1000000 (Dez); Beschleunigung 1000.000 rad/s<sup>2</sup>

Geschwindigkeit (Byte 15 - 18) = 30000000 (Dez); 3000.0000 U/min

Lagesoll (Byte 19 - 22) = 3600000 (Dez); 360.0000 Grad

### 1. Abschnitt Starten:

Beschleunigung = 1500000 (Dez)

Geschwindigkeit = 20000000 (Dez)

Lagesoll = 7200000 (Dez)

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

relativen Polynomabschnitt starten

Bit 8 (START\_POSITIONING) and Bit 15 (START\_POSITIONING\_TOGGLE) werden beim ersten Start beide gesetzt.

Warten, bis der Profibus-Puffer ausgewertet ist:

→ Warten, bis Handshake = 1 (Statuswort, Bit 8: HANDSHAKE)

Warten, bis die nächsten Daten geladen werden können:

→ Warten, bis SETPOINT\_ACKNOWLEDGE = 1 (Statuswort, Bit 4)

### 2. Abschnitt Schreiben und Starten (wird am Ende des ersten gestartet):

Beschleunigung = 1800000 (Dez)

Geschwindigkeit = 25000000 (Dez)

Lagesoll = 3600000 (Dez)

Kontrollwort = 0x00140120 (Hex) relativen Polynomabschnitt starten, Togglebit (Bit 16) verändert

Warten, bis Handshake = 1 (Statuswort, Bit 8: HANDSHAKE)

Warten, bis die Daten übernommen wurden (geschieht erst am Ende des ersten Wegabschnittes):

Warten, bis SETPOINT\_ACKNOWLEDGE = 1 (Statuswort, Bit 4)

### Letzten Abschnitt Schreiben und Starten:

Beschleunigung = 2000000 (Dez)

Geschwindigkeit = 12000000 (Dez)

Lagesoll = 7200000 (Dez)

Kontrollwort = 0x001D0120 letzten Polynomabschnitt starten (Togglebit Bit 16 verändert und Bit 19 – Polynom-Ende – gesetzt)

Warten bis Handshake = 1 (Statuswort, Bit 8: HANDSHAKE) → danach Bit 8 im Kontrollwort (START\_POSITIONING) wieder zurücksetzen, um die Befehlsabarbeitung neuer "OPCODE"-Kommandos wieder freizugeben:

Setzen Bit 8 im Kontrollwort = 0