

# **DS - Schrittmotoren**

**Gerätemodell und Softwareentwurf**

**P. Kainberger G.Schwarz**

*Dieses Papier enthält die Beschreibung des Gerätemodells "DS – Schrittmotoren" und den Entwurf der Gerätesoftware für dieses Gerät.*

### Änderungsprotokoll

Datum	Version	Name	Kommentar
6. Apr 92	--	R. Thomitzek	STEP erstellt
1. Jun 92	--	R. Thomitzek	Neue Properties
28. Jul 92	--	R. Thomitzek	Bereinigung
13. Aug 92	--	A. Peters	Korrekturen
02. Sep 93	STEP_02	P. Kainberger	Korrekturen u. Ergänzungen
04. Nov 93	STEP_02	P. Kainberger	Gerätetyp in Infopaket erweitert
24. Mai 94	STEP_02	P. Kainberger	Gerätetyp in Infopaket u. Handhabung von rechts/links korrigiert
13. Feb 98	STEP_06	P. Kainberger	Gerätetyp in Infopaket erweitert
30. Aug 01	--	M. Kühn	Überarbeitete und erweiterte T <sub>E</sub> X-Version, die sowohl in PostScript als auch in HTML konvertiert werden kann.
30. Aug 01	--	VRW Schaa	Eingebettete L <sup>A</sup> T <sub>E</sub> X-Bilder in externe eps-Files verwandelt.
13. Okt 03	STEP_08	G.Schwarz	komplette Ueberarbeitung, Einbindung neues SDI-Interface
31. Mar 04	DS	G.Schwarz	uebernommen nach DS

# Contents

<b>I</b>	<b>Das Gerätemodell</b>	<b>7</b>
<b>1</b>	<b>Die Aufgabe des Gerätes</b>	<b>7</b>
<b>2</b>	<b>Die Hardware des Gerätes</b>	<b>8</b>
2.1	Der SD $\mu$ P . . . . .	9
2.2	Das SD-Interface . . . . .	10
<b>3</b>	<b>Die Schnittstellen zum Gerät</b>	<b>10</b>
3.1	Der SD $\mu$ P . . . . .	10
3.1.1	Funktionscodes der Interfacekarte . . . . .	10
3.1.2	Interlock-Interrupt . . . . .	15
3.1.3	Data Request (DRQ) Interrupts . . . . .	15
3.1.4	Data Ready (DRD) Interrupts . . . . .	15
3.1.5	Umfang eines logischen Gerätes . . . . .	15
3.2	Das SD-Interface . . . . .	15
3.2.1	Funktionscodes der Interfacekarte . . . . .	15
3.3	Definition der Bits des Hardwarestatus . . . . .	18
<b>4</b>	<b>Die Bedienung des Gerätes</b>	<b>19</b>
4.1	Aufgaben im Normalbetrieb . . . . .	19
4.1.1	Schrittmotor fahren . . . . .	19
4.1.2	Positionsüberwachung . . . . .	19
4.1.3	Gerätestatusüberwachung . . . . .	19
4.1.4	Einschalten . . . . .	20
4.1.5	Ausschalten . . . . .	20
4.2	Genauigkeitsanforderungen . . . . .	20
4.3	Zeitkritische Anforderungen . . . . .	20
4.4	Einordnung in das Timing . . . . .	20
4.5	Festlegung von Startwerten . . . . .	20
4.5.1	Kaltstarts . . . . .	20
4.5.2	Warmstarts . . . . .	21
4.6	Handbetrieb . . . . .	21
4.7	Ableitung des Hardwarefehler-Bits aus dem Gerätestatus . . . . .	21
4.8	Verhalten bei Störungen . . . . .	21
4.8.1	Geräteinterlock . . . . .	21
4.8.2	Event-Sequenzfehler . . . . .	21
4.8.3	Event-Overrun . . . . .	21
4.8.4	Emergency-Event . . . . .	21
4.8.5	Ausfall der Kommunikation EC – Gerät . . . . .	21
4.9	Bedienungsfehler vom Operating . . . . .	22
4.10	Sonstige Anforderungen . . . . .	22
<b>5</b>	<b>Die Repräsentation des Gerätes</b>	<b>22</b>

5.1	Kennzeichnung des Gerätemodells . . . . .	22
5.2	Die Master-Properties . . . . .	22
5.2.1	POWER . . . . .	23
5.2.2	STATUS . . . . .	23
5.2.3	INIT . . . . .	23
5.2.4	RESET . . . . .	23
5.2.5	VERSION . . . . .	23
5.2.6	INFOSTAT . . . . .	24
5.2.7	INFO . . . . .	25
5.2.8	CONSTANT . . . . .	25
5.2.9	POSIABSS . . . . .	26
5.2.10	POSIABSI . . . . .	26
5.2.11	REFTEST . . . . .	26
5.2.12	STEPS . . . . .	26
5.2.13	ENDLAGE . . . . .	27
5.2.14	SLITINFO . . . . .	27
5.3	Die Slave-Properties . . . . .	27
5.3.1	ACTIV . . . . .	27
5.3.2	COPYSET . . . . .	28
5.3.3	EQMERROR . . . . .	28
<b>II</b>	<b>Der Entwurf der Software</b>	<b>30</b>
<b>6</b>	<b>Softwareentwurf</b>	<b>30</b>
<b>7</b>	<b>Lokale Datenbasis</b>	<b>30</b>
7.1	Tabelle der Konstanten . . . . .	30
<b>8</b>	<b>Dualport RAM</b>	<b>30</b>
<b>9</b>	<b>USRs - User Service Routinen</b>	<b>30</b>
9.1	Obligatorische USRs . . . . .	30
9.1.1	N_Init . . . . .	30
9.1.2	N_Reset . . . . .	30
9.1.3	R_Status . . . . .	30
9.1.4	R_Power . . . . .	30
9.1.5	W_Power . . . . .	30
9.1.6	R_Active . . . . .	30
9.1.7	W_Active . . . . .	30
9.1.8	W_CopySet . . . . .	30
9.1.9	R_EQMErr . . . . .	30
9.1.10	R_Version . . . . .	30
9.1.11	R_InfoStat . . . . .	30
9.2	Gerätespezifische USRs . . . . .	30
9.2.1	R_RefTest . . . . .	31
9.2.2	R_PosiAbsi . . . . .	31

9.2.3	R_PosAbs	31
9.2.4	W_PosAbs	31
9.2.5	W_PosRel	31
9.2.6	R_Info	31
9.2.7	R_Constant	31
9.2.8	W_StepS	31
9.2.9	W_Endlage	31
9.2.10	R_SlitInfo	31
9.3	Globale Routinen	31
<b>10</b>	<b>EQMs - Equipment Module</b>	<b>31</b>
10.1	Interne Zustände	31
10.1.1	Bedeutung der internen Zustände	31
10.1.2	Übergänge zwischen den Zuständen	32
10.1.3	Emerg_EQM	32
10.2	Periodisch konnektierte EQMs	32
10.2.1	Status_EQM	32
10.2.2	Update_Config_EQM	33
10.3	An externe Interrupts konnektierte EQMs	34
10.3.1	Interlock_EQM	34
10.3.2	DRD_EQM	34
10.3.3	DRQ_EQM	34
10.4	Kommandogetriggerte EQMs	34
10.4.1	Dev_Init_EQM	34
10.4.2	Dev_Reset_EQM	34
10.4.3	Status_EQM	34
10.4.4	Active_EQM	34
10.4.5	Power_EQM	34
10.4.6	Test_EQM	34
10.4.7	Move_EQM	34
10.5	EQMs für die Diagnose vor Ort	35
10.5.1	Display_DPR_EQM	35
10.5.2	Display_DevErr_EQM	35
10.6	Sonstige EQMs	35
10.6.1	Startup_EQM	35
10.7	Globale Routinen	35
10.7.1	Read_and_Update_Status	35
10.7.2	Read_and_Update_GStatus	35
10.7.3	Get_LogDev	36
10.7.4	Set_MStatus	36
10.7.5	Do_Intr_Service_Prep	36
<b>11</b>	<b>Varianten</b>	<b>36</b>
<b>12</b>	<b>Besonderheiten</b>	<b>36</b>
12.1	Supergeräte	36

12.2 Der Response-Algorithmus . . . . .	37
---	----

<b>Index</b>	<b>38</b>
--------------	-----------

### List of Figures

1 Das Koordinatensystem . . . . .	7
2 Die Drehbewegung . . . . .	8
3 Blockschaltbild der Schrittmotorsteuerung . . . . .	8
4 Der Response-Algorithmus . . . . .	37

### List of Tables

15 Zustandsübergangsdiagramm . . . . .	32
16 Legende zu den Zustandsübergangsdiagrammen . . . . .	33

# Part I

## Das Gerätemodell

### 1 Die Aufgabe des Gerätes

In der GSI gibt es an vielen Stellen Objekte, die über kürzere Distanzen möglichst genau bewegt werden müssen. Zur Lösung dieser Aufgabe werden bevorzugt Schrittmotoren eingesetzt.

Eine der häufigsten Anwendungen sind Schlitzbacken oder Blenden, die zur Strahlbegrenzung oder zur Strahlseparierung verwendet werden. Die Schlitzbacken werden über eine Spindel in den Strahl gefahren. Zur Rückmeldung der Position wird in der Regel ein Linearpotentiometer benutzt.

Bei anderen Anwendungen wird zur Ermittlung der aktuellen Position ein Winkelcodierer eingesetzt. Schlitzbacken sind im Normalfall paarweise angeordnet, und sollen auch paarweise bewegt werden können. Motorpaare sollen auch mit unterschiedlichen Positionsaufnehmern bestückbar sein. Aber auch der Anschluss von nur einem Antrieb ist vorgesehen, z. B. bei Targetleitern und Schleusen.

Für alle diese Anwendungen ist geplant, dass zur Benutzerseite hin die gleiche Bedienungsschnittstelle besteht.

Zum Erreichen der zuvor beschriebenen Ziele wurde eine Hardware entwickelt, die durch ihren modularen Aufbau für jede Schrittmotorenanwendung einsetzbar ist. Für die Linearantriebe gilt:

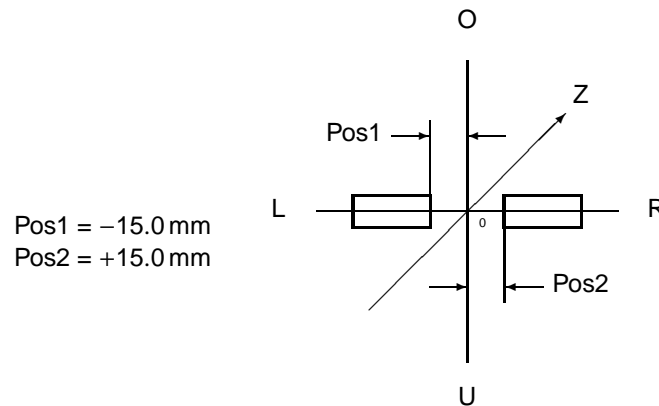


Figure 1: Das Koordinatensystem

- Alle Bewegungen nach rechts, nach oben oder in Strahlrichtung sind positiv.
- Alle Bewegungen nach links, nach unten oder gegen die Strahlrichtung sind negativ.

Bei Drehbewegungen gilt:

- Alle Bewegungen im Uhrzeigersinn sind positiv.
- Alle Bewegungen gegen den Uhrzeigersinn sind negativ.

Bei Geräten, die paarweise angeordnet sind (wie zum Beispiel die Schlitzbacken), gilt:

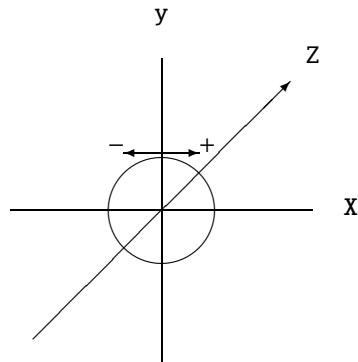


Figure 2: Die Drehbewegung

1. Jeder Antrieb kann nur einzeln gefahren werden.
2. Die Positionen dieser Antriebe können nur gelesen, nicht gesetzt werden.

## 2 Die Hardware des Gerätes

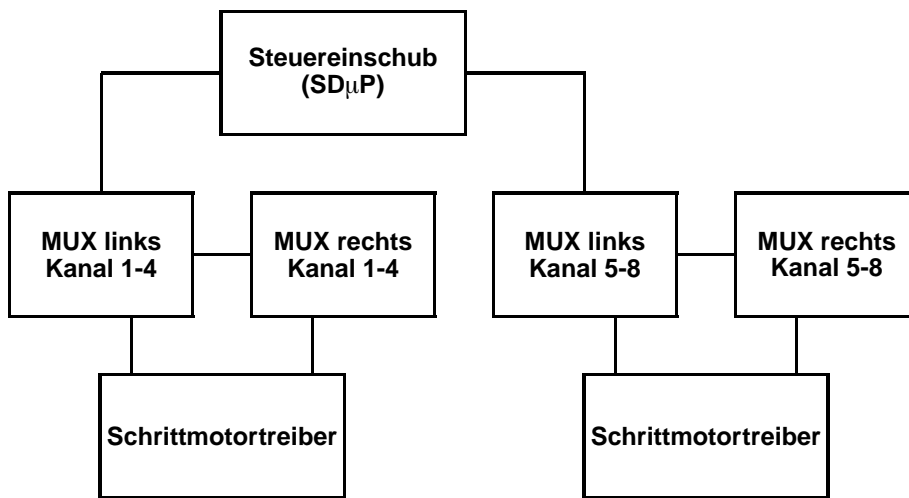


Figure 3: Blockschaltbild der Schrittmotorsteuerung

Um Verstärker zu sparen, werden Multiplexer verwendet, Ueber zwei Multiplexerpaare können an einer Elektronik bis zu 8 Kanäle mit je 2 Einzelantrieben konfiguriert werden. Je nach Anforderung ist jeder Kanal mit einem Potentiometer oder einem Winkelkodierer als Positionsmesseinheit ausgerüstet, ebenso werden für jeden Kanal die Interlock-Eingänge separat verwaltet. Als Treiber kommen in der Regel frei käufliche Schrittmotor-Treiber zum Einsatz. Alle vom VME kommenden Kommandos sind auch lokal ausführbar. Dadurch wird dem Service-Personal eine örtliche Diagnosemöglichkeit



zur Verfügung gestellt. Am Steuereinschub befinden sich zusätzliche Bedienelemente, die eine vereinfachte Handsteuerung ermöglichen.

Um Multiplexer und Verstärker-Elektronik zu schalten, werden folgende Hardware-Varianten unterschieden:

- Zur Zeit wird noch ein Strahldiagnose-Mikro-Prozessor (SD $\mu$ P) verwendet. Er besteht aus einem 8-Bit Prozessor (8085) mit I/O-Peripherie. Dem SD $\mu$ P vorgeschaltet ist eine Interface-Karte, die SE kommuniziert über ein spezielles Protokoll (MIL-SDN) mit Interface-Karte und SD $\mu$ P.
- Die neueren Elektroniken werden nur noch mit dem SD-Interface aufgebaut. Auch dafür gibt es eine Treiberunterstützung (MIL-SDI).

Die Elektronik hält in einem EEPROM (bei SD $\mu$ P EPROM) alle nötigen Informationen über Art und Anzahl der angeschlossenen Antriebe bereit (s. Property INFO in Abs. 3.1.1 auf Seite 12). Aus diesen Informationen werden auf der SE-Ebene einzelne Antriebe sowie zusammengehörige Antriebe als Mehrfachantriebe (Paare und Kollimatoren) als *logische* Geräte gebildet.

## 2.1 Der SD $\mu$ P

Die Verwaltung und Konfiguration der einzelnen Kanäle, *welcher* Kanal ist *womit* belegt, findet im SD $\mu$ P statt. Der SD $\mu$ P seinerseits stellt jede einzelne Komponente (z. B. Kanal-1 rechter Antrieb, Kanal-1 linker Antrieb und Kanal-1 Antriebspaar rechts/links) als *logische* Geräteeinheit dar, damit können aus physikalisch 8 Kanälen bis zu 24 *logische* Geräteeinheiten werden.

Über ein Bit im Gerätestatus kann SE-seitig festgestellt werden, ob eine *logische* Geräteeinheit existiert (*online* ist) oder nicht. Im Infopaket jedes einzelnen Gerätes ist als Gerätetyp beschrieben, um welches Gerät es sich handelt (z. B. rechter Antrieb horizontal, der Teil eines Antriebspaares ist; oder linker Einzelantrieb vertikal (*links* bedeutet dann *unten*)).

Dabei ist zu beachten, dass die Information *rechter* oder *linker* Antrieb, sich nur auf die Einbauweise des Antriebs an der Diagnosekammer bezieht. Für die Gerätesoftware ist jedoch entscheidend, ob der Antrieb am *rechten* oder *linken* Multiplexer angeschlossen ist. Dabei kann es durchaus vorkommen, dass ein *rechter* Antrieb an einem *linken* Multiplexer angeschlossen ist. Dann muss die Gerätesoftware den Status und die Position vom *linken* Multiplexer auf den Status des *rechten* Antriebs übertragen, damit für den Benutzer der Anschluss der Geräte an den verschiedenen Multiplexern transparent bleibt.

Zusätzlich zu dieser Aufteilung auf *logische* Geräteeinheiten, können auf der VME-Ebene 3 oder 4 Antriebe zu einem zu einem Kollimator zusammengefasst werden.

Durch die Verwendung von Multiplexern (mit Relais) ergeben sich einige Konsequenzen für die Software im SD $\mu$ P und auf der SE:

- Wenn bei einem Gerätezugriff erst die Multiplexer auf einen anderen Kanal geschaltet werden müssen, muss eine längere Wartezeit in Kauf genommen werden (ca. 100 ms).
- Solange ein Motor "fährt" können die Multiplexer nicht auf einen anderen Kanal geschaltet werden.
- Solange ein Motor "fährt" kann der Status und die aktuelle Position der anderen Kanäle nicht abgefragt werden. Nur die Motoren des selektierten Kanals können überprüft werden.

Um all diesen Problemen einigermaßen aus dem Weg gehen zu können, bietet der SD $\mu$ P einen sog. *Blockstatus* (siehe Kapitel 3.1.1 auf Seite 14) von allen 8 Kanälen in einem Zugriff an. Der SD $\mu$ P sorgt auch dafür, dass die Daten zu jedem einzelnen Kanal (Geräte-Status, Ist-Position rechts und links) so aktuell wie möglich sind. D.h. der Kanal, der gerade über den Multiplexer angewählt ist, wird periodisch aktualisiert, bei den anderen Kanälen kann sich "eigentlich" (siehe Kapitel 4.10 auf Seite 22) nichts ändern. SE-seitig muss dieser *Blockstatus*, der ja kanalbezogene Informationen beinhaltet, auf die einzelnen *logischen* Geräteeinheiten je nach Konfiguration separiert werden.

## 2.2 Das SD-Interface

Seit 2003 gibt es zur Schrittmotorsteuerung das Strahl Diagnoseinterface, kurz SDI. Es bildet zur Mil-Schnittstelle hin keine logischen Geräteeinheiten ab, sondern legt die statischen Informationen über die Konfiguration der 8 Kanäle bzw. 16 Antriebe in einem EEPROM ab. Die Informationen über Status und Positionen der Antriebe werden in einem DPRAM gehalten und ständig refresht. Antriebspaare können hier nicht konfiguriert werden. Alle Mehrfachantriebe (Paare und Kollimatoren) werden auf der SE gebildet.

## 3 Die Schnittstellen zum Gerät

### 3.1 Der SD $\mu$ P

#### 3.1.1 Funktionscodes der Interfacekarte

Die für die Geräteansteuerung definierten Funktionscodes sind in der folgenden Tabelle aufgelistet. Als Modus ist angegeben, ob Daten von der Interfacekarte gelesen werden, ob Daten zu der Interfacekarte geschrieben werden, oder ob nur eine Funktion ausgeführt wird.

Funktionscode		Modus	Bedeutung
Name	Hex		
IFB_Rdstat	C0	Lesen	Elektronikstatus lesen
IFB_Rdstat_1	C1	Lesen	Gerätestatus lesen
IFB_Info	82	Lesen	Infopaket lesen
IFB_Posit	83	Lesen	Positionen lesen
IFB_Test	84	Lesen	Testspannung lesen
IFB_Block	85	Lesen	Blockstatus lesen
IFB_Ein	06	Schreiben	Einzelantrieb fahren
IFB_AufZu	08	Schreiben	Auf-/Zufahren eines Antriebpaares
IFB_Par	09	Schreiben	Parallelfahren eines Antriebpaares
IFB_Rein	10	Funktion	Motor an Endanschlag innen fahren
IFB_Raus	11	Funktion	Motor an Endanschlag außen fahren
IFB_Move	12	Schreiben	Motor n Schritte fahren

Die Fahrfunktionen fuer Paare werden nicht mehr benutzt.

## IFB\_Rdstat

Status der Elektronik lesen. Dieser kann zu jeder Zeit gelesen werden, da dafür kein Schalten des Multiplexers erforderlich ist.

Das vom Gerät gelieferte Statuswort hat folgenden Aufbau:

Bit	Name	Bedeutung	
		High (1)	Low (0)
0	Power (Electronic)	OK	Fail
1	reserved		
:	:		
7	reserved		
8	Power (Interface)	on	off
9	Remote/Local	Remote	Local
10	reserved		
:	:		
14	reserved		
15	Online/Offline	online	offline

Die beiden Bits für Power (Electronic und Interface) sind im Gerätestatus zu einem Power-Bit zusammengefasst.

## IFB\_Rdstat\_1

Status des Gerätes explizit lesen. D. h. falls das Gerät noch nicht über den Multiplexer erreichbar ist, wird dieser auf den entsprechenden Kanal umgeschaltet.

Das vom Gerät gelieferte Statuswort hat folgenden Aufbau:

Bit	Name	Bedeutung	
		High (1)	Low (0)
0	reserved		
1	Interlock A links/unten	ok	Interlock
2	Interlock B links/unten	ok	Interlock
3	Interlock A rechts/oben	ok	Interlock
4	Interlock B rechts/oben	ok	Interlock
5	Minimalabstand	eingehalten	nicht eingehalten
6	Motor rechts/oben steht	steht	fährt
7	Motor links/unten steht	steht	fährt
8	reserved		
:	:		
10	reserved		
11	Fahrfehler	ok	Fahrfehler

Bit	Name	Bedeutung	
		High (1)	Low (0)
12	Endlage links/unten außen	erreicht	nicht erreicht
13	Endlage rechts/oben außen	erreicht	nicht erreicht
14	Endlage links/unten innen	erreicht	nicht erreicht
15	Endlage rechts/oben innen	erreicht	nicht erreicht

## IFB\_Info

Infopaket für ein Gerät (*logische* Geräteeinheit) lesen.

Die 19 Daten (Integer16) bedeuten im einzelnen:

- 1: Versionsnummer
- 2: Nomenklatur 4 Byte (bei SDuP generiert)
- 3: Nomenklatur 4 Byte (bei SDuP generiert)
- 4: Gerätetyp (beschreibt die Art des Antriebes und der Positionsaufnehmer)

Bit	Name	Bedeutung	
		High (1)	Low (0)
0	rechter Antrieb	ja	nein
1	linker Antrieb	ja	nein
2	Teil eines Antriebspaares	ja	nein
3	Typ der Positionsmesseinheit	Winkelkodierer	Linearpotentiometer
4	Fahrgeschwindigkeit	schnell	langsam
5	Anschluss an Multiplexer	rechts	links
6	Lage des Antriebs	vertikal	horizontal
7	Verfügbarkeit über Kontrollsystem	nicht verfügbar	verfügbar
8	Mindestabstandsschalter	nicht vorhanden	vorhanden
9	ADC-Type	16 Bit	12 Bit
10	unused		
:	⋮		
15	unused		

Ist ein Antrieb sowohl rechter als auch linker Antrieb, so handelt es sich um ein Antriebspaar.

- 5: Minimalabstand (gibt in 1/10 mm an, wieviel Abstand die Antriebe mindestens einhalten müssen)
- 6: Pulse pro cm
- 7: mm/U bei Winkelkodierer
- 8: Maximale Position
- 9: Minimale Position

**10:** Maximale Position gegenueber

**11:** Minimale Position gegenueber

**12:** Positionsfaktor

**13:** Startfrequenz

**14:** Maximalfrequenz

**15:** Pulsbreite

**16:** Polspolaritaet

**17:** Rampensteilheit

**18:** Positionsoffset

**19:** Potilaenge

**20:** reserve

### **IFB\_Posit**

Liest die aktuellen Positionen der Antriebe:

1. Position links/unten (in 1/10 mm),
2. Position rechts/oben (in 1/10 mm).

Die Daten sind vom Typ Integer16.

### **IFB\_Test**

Liest die Referenzspannung der Antriebe:

1. Referenzspannung links/unten (mV),
2. Referenzspannung rechts/oben (mV).

Die Daten sind vom Typ Integer16.

### **IFB\_Ein**

Einzelnen Motor auf absolute Position fahren. Als Datum ist die absolute Position in 1/10 mm zu uebergeben (Typ Integer16).

### **IFB\_AufZu**

Antriebspaar auf-/zu-fahren. Diese Fahrfunktion wird nicht mehr benutzt.

### IFB\_Par

Antriebspaar parallel fahren. Diese Fahrfunktion wird nicht mehr benutzt.

### IFB\_Rein

Motor an den inneren Endanschlag fahren.

### IFB\_Raus

Motor an den äußeren Endanschlag fahren.

### IFB\_Move

Einzelnen Motor schrittweise fahren. Als Datum ist die Anzahl der Schritte zu übergeben (Typ Integer16).

### IFB\_Block

Liest in einem Block alle Statusinformationen und Positionen der 8 Kanäle. Dabei ist zu beachten, dass sich *rechts/links* immer auf den Multiplexer bezieht. Es werden immer 24 Worte übertragen:

**1...8** = Status der 8 Kanäle,

Bit	Name	Bedeutung	
		High (1)	Low (0)
0		reserved	
1	Interlock A links	ok	Interlock
2	Interlock B links	ok	Interlock
3	Interlock A rechts	ok	Interlock
4	Interlock B rechts	ok	Interlock
5	Minimalabstand	eingehalten	nicht eingehalten
6	Motor rechts steht	steht	fährt
7	Motor links steht	steht	fährt
8		reserved	
⋮		⋮	
10		reserved	
11	Fahrfehler	ok	Fahrfehler
12	Endlage links außen	erreicht	nicht erreicht
13	Endlage rechts außen	erreicht	nicht erreicht
14	Endlage links innen	erreicht	nicht erreicht
15	Endlage rechts innen	erreicht	nicht erreicht

**9...16** = Ist-Positionen der Antriebe am linken Multiplexer,

17...24 = Ist-Positionen der Antriebe rechten Multiplexer.

### 3.1.2 Interlock-Interrupt

Diese Interruptleitung wird nur zur Kommunikation zwischen der SE und dem SD $\mu$ P benutzt (siehe Beschreibung der SD $\mu$ P -Treiberrouinen).

### 3.1.3 Data Request (DRQ) Interrupts

Diese Interruptleitung wird nur zur Kommunikation zwischen der SE und dem SD $\mu$ P benutzt (siehe Beschreibung der SD $\mu$ P -Treiberrouinen).

### 3.1.4 Data Ready (DRD) Interrupts

Diese Interruptleitung wird nur zur Kommunikation zwischen der SE und dem SD $\mu$ P benutzt (siehe Beschreibung der SD $\mu$ P -Treiberrouinen).

### 3.1.5 Umfang eines logischen Gerätes

An einen SD $\mu$ P können bis zu 8 Schrittmotorsteuerungen angeschlossen werden. Jede einzelne Steuerung kann bis zu 3 Geräte repräsentieren (Antrieb rechts, Antrieb links und Antriebspaar rechts/links). Also werden pro SD $\mu$ P bis zu 8 physikalische Geräteeinheiten als maximal 24 logische Geräteeinheiten repräsentiert (24 nur dann, wenn alle 8 Steuerungen Antriebspaare sind, deren Motoren aber nur einzeln gefahren werden können).

Aus dem Gerätetyp im Infopaket jedes einzelnen Antriebs ist erkennbar, ob es sich um einen Einzelantrieb (rechts, links, oben oder unten) oder ein Antriebspaar handelt. Ebenso wird markiert, ob die Fahrtrichtung der Antriebe vertikal oder horizontal ist (horizontal: rechts/links, vertikal: rechts heißt oben/ links heißt unten).

Für die Software der EC-Ebene bedeutet dies, dass der Status einer physikalischen Geräteeinheit bis zu 3 verschiedenen logischen Geräteeinheiten zugeordnet werden muss. Siehe hierzu auch Abschnitt 2 auf Seite 8.

## 3.2 Das SD-Interface

### 3.2.1 Funktionscodes der Interfacekarte

Die für die Geräteansteuerung definierten Funktionscodes sind in der folgenden Tabelle aufgelistet. Als Modus ist angegeben, ob Daten von der Interfacekarte gelesen werden, ob Daten zu der Interfacekarte geschrieben werden, oder ob nur eine Funktion ausgeführt wird.

Funktionscode		Modus	Bedeutung
Name	Hex		
IFB_Rdstat	C0	Lesen	Elektronikstatus lesen
IFB_soll_1	06	Schreiben	Daten ins Vorbereitungsregister
IFB_soll_2	07	Schreiben	Daten ins Vorbereitungsregister
IFB_soll_3	08	Schreiben	Starten neue Fahrposition
IFB_DEV_FCT_12	1F	Funktion	DRD-Interrupt zueuecksetzen

Funktionscode		Modus	Bedeutung
Name	Hex		
IFB_ist_1	80	Lesen	Handshakeregister lesen
IFB_RESET	01	Funktion	Reset
IFB_BLOCK	8F	Lesen	DPRAM oder EEPROM Daten lesen

### IFB\_Rdstat

Status der Elektronik lesen. Dieser kann zu jeder Zeit gelesen werden.

Das vom Gerät gelieferte Statuswort soll folgenden Aufbau haben: Zur Zeit (April 2004) wird allerdings noch das invertierte Statuswort geliefert, d.h. im Moment ist der Status ok, wenn alle Bits 0 sind. Die Geraetesoftware invertiert alle Bits vor der weiteren Verarbeitung.

Bit	Name	Bedeutung	
		High (1)	Low (0)
8	Interlock L	off	on
9	Interlock R	off	on
10	Motor laeuft	keiner laeuft	laeuft
11	Remote/Local	Remote	Local
12	Power L 1..4;	ok	fail
13	Power R 1..4;	ok	fail
14	Power L 5..8;	ok	fail
15	Power R 5..8;	ok	fail

### IFB\_Block

Liest in einem Block Daten aus dem DPRAM oder dem EEPROM. Im DPRAM liegen alle Statusinformationen und Positionen der 8 Kanäle. (16 Antriebe) Dabei ist zu beachten, dass sich *rechts/links* immer auf den Multiplexer bezieht. Es werden immer 64 Worte übertragen:

Die SDI-Schnittstelle liefert fuer die einzelnen Antriebe Status- Informationen im DPRAM:

Bit	Name	Bedeutung	
		High (= 1)	Low (= 0)
0	Power L 1..4;	ok	fail
1	Power R 1..4;	ok	fail
2	Power L 5..8;	ok	fail
3	Power R 5..8;	ok	fail
4	InterlockA L	off	on
5	InterlockB L	off	on
6	InterlockA R	off	on
7	InterlockB R	off	on



8	Endlagenschalter links außen	erreicht	nicht erreicht
9	Endlagenschalter links innen	erreicht	nicht erreicht
10	Minimalabstandsschalter links	erreicht	nicht erreicht
11	Motorbremse links	ein, Motor steht	auf, Motor laeuft
12	Endlagenschalter rechts außen	erreicht	nicht erreicht
13	Endlagenschalter rechts innen	erreicht	nicht erreicht
14	Minimalabstandsschalter rechts	erreicht	nicht erreicht
15	Motorbremse rechts	ein, Motor steht	auf, Motor laeuft
16...19	Nummer des Kanals		
20	Remote/Local	Remote	Local
21...31	dies und das		

### IFB\_Block

Ebenfalls im Blockmode, diesmal vom EEPROM wird das Infopaket für die Antriebe gelesen. Die 32 Daten (Integer16) bedeuten im einzelnen:

**1...3:** Nomenklatur

**4:** Gerätetyp (beschreibt die Art des Antriebes und der Positionsaufnehmer)

Bit	Name	Bedeutung	
		High (1)	Low (0)
0	rechter Antrieb	ja	nein
1	linker Antrieb	ja	nein
2	Teil eines Antriebspaares	ja	nein
3	Typ der Positionsmesseinheit	Winkelkodierer	Linearpotentiometer
4	frei	-	-
5	Anschluss an Multiplexer	rechts	links
6	Lage des Antriebs	vertikal	horizontal
7	Verfügbarkeit über Kontrollsystem	nicht verfügbar	verfügbar
8	Mindestabstandsschalter	nicht vorhanden	vorhanden
9	ADC-Type	16 Bit	12 Bit
10	Endschalter	nicht vorhanden	vorhanden
11	Drehrichtung	invertiert	normal
12		unused	
:		:	
15		unused	

Ist ein Antrieb sowohl rechter als auch linker Antrieb, so handelt es sich um ein Antriebspaar.

**5:** Minimalabstand (gibt in 1/10 mm an, wieviel Abstand die Antriebe mindestens einhalten müssen)

**6:** Pulse pro cm

- 7: 1/10mm/U bei Winkelkodierer
- 8...9: Maximale Position in 1/10 mm
- 10...11: Minimale Position in 1/10 mm
- 12...13: Maximale Position Kanal gegenueber in 1/10 mm
- 14...15: Minimale Position Kanal gegenueber in 1/10 mm
- 16: Positionsfaktor \*100
- 17: Startfrequenz
- 18: Maximalfrequenz
- 19: Pulsbreite
- 20: Polspolaritaet
- 21: Rampensteilheit
- 22..23: Positions-Offset
- 24: Potilaenge in mm
- 25...31: Reserve

### 3.3 Definition der Bits des Hardwarestatus

Die von SD $\mu$ P und SD-Interface gelieferten unterschiedlichen Statusinformationen ( GSTATUS) und ( ESTATUS) werden auf der SE zu einem gemeinsamen Statuswort zusammengefasst:

Die Bits 0. . . 7 sind die systemweiten generierten Softwarestatusbits (in engl. derived status bits).

Die Statusbits im Einzelnen sind in der folgenden Tabelle zusammengefasst.

Bit	Name	Bedeutung	
		High (1)	Low (0)
0	Power	on	off
1	Remote/Local	Remote	Local
2		reserved	
3		reserved	
4	Emergency	no	yes
5	Interlock	no	yes
6	HW Error	no	yes
7	SW Error	no	yes
8	fahrbereit(Bit 9 ... 13 true	bereit	nicht bereit
9	dieser Antrieb	steht	fahrt
10	Fahrfehler	kein Fehler	Fehler
11	Interlock B	kein Interlock	Interlock
12	Interlock A	kein Interlock	Interlock

Bit	Name	Bedeutung	
		High (1)	Low (0)
13	alle Antriebe	stehen	mind. einer fährt
14		reserved	
15		reserved	
16	Minimalabstand	eingehalten	nicht eingehalten
17	Endlage innen	erreicht	nicht erreicht
18	Endlage außen	erreicht	nicht erreicht
19	ges. Elektronik Endlage außen	erreicht	nicht erreicht
20 ... 23	reserve		
24 ... 30	reserve		
31	medPosok	ok	nicht ok

Fuer Mehrfachantriebe ist die Statusinformation identisch, die Informationen gelten jeweils fuer alle zugehoerigen Antriebe. Bei den Endlagen gilt: Endlage innen erreicht: mindestens einer der Antriebe hat die Endlage erreicht. Endlage außen erreicht: alle zugehoerigen Antriebe haben die Endlage erreicht.

## 4 Die Bedienung des Gerätes

### 4.1 Aufgaben im Normalbetrieb

#### 4.1.1 Schrittmotor fahren

Die Schrittmotoren können einzeln (auf Position oder Schrittweise) und auf Endlage (außen oder innen) gefahren werden. Vor jeder Fahrfunktion muss geprüft werden, ob bereits ein Antrieb fährt. Bei allen Fahrfunktionen werden immer automatisch die Multiplexer auf die entsprechenden Kanäle umgeschaltet. Die Überwachung eines fahrenden Motors (lesen der aktuellen Positionsdaten bis die gewünschte Funktion vollständig ausgeführt ist), wird vom SD $\mu$ P übernommen, d. h. die SE wartet nicht bis ein Fahrkommando abgeschlossen ist.

Dies entspricht jedoch nicht der "normalen" Vorgehensweise bei Gerätesteuern über das Kontrollsystem und sollte bei der nächsten Überarbeitung der Software dahingehend geändert werden, dass von der SE aus mit Hilfe eines periodischen Auftrags (mit einem timeout, der der Länge des Fahrweges angepasst ist) überwacht wird, wann die veranlasste Fahrfunktion beendet ist.

#### 4.1.2 Positionsüberwachung

Für alle Kanäle einer Schrittmotorsteuerung müssen die aktuellen Positionen periodisch (Periode 1 s) überwacht werden.

#### 4.1.3 Gerätestatusüberwachung

Alle Kanäle einer Schrittmotorsteuerung müssen periodisch (Periode ca 10 Minuten) explizit selektiert werden (siehe Abschnitt 2 auf Seite 8).

#### **4.1.4 Einschalten**

Die Schrittmotorsteuerung kann nicht über das Kontrollsystem eingeschaltet werden.

#### **4.1.5 Ausschalten**

Die Schrittmotorsteuerung kann nicht über das Kontrollsystem ausgeschaltet werden.

### **4.2 Genauigkeitsanforderungen**

Alle Genauigkeitsanforderungen werden durch die verwendete Hardware erfüllt.

### **4.3 Zeitkritische Anforderungen**

Keine

### **4.4 Einordnung in das Timing**

Das Gerät nimmt nicht an der PPM teil.

### **4.5 Festlegung von Startwerten**

#### **4.5.1 Kaltstarts**

Bei einem Kaltstart werden folgende Aktionen durchgeführt:

- Für jeden SD $\mu$ P und jedes SD-Interface wird ein Reset durchgeführt.
- Das Status\_EQM wird für jeden SD $\mu$ P als periodischer Auftrag (Periode 1 s) konnektiert (dadurch werden die Istpositionen aller Komponenten eines SD $\mu$ P aktualisiert).
- Das GStatus\_EQM wird für jeden SD $\mu$ P als periodischer Auftrag (Periode 60 s) konnektiert (dadurch wird der Gerätestatus aller Komponenten eines SD $\mu$ P aktualisiert).
- Alle Sollwerte werden auf 0 gesetzt.
- Die Infopakete der Geräte werden gelesen.
- Die Statusinformationen werden gelesen.
- Die Ist-Positionen werden gelesen.
- Alle Geräte werden für alle Beschleuniger auf inaktiv gesetzt.
- Die SE wird in den Kommandmode-Betrieb geschaltet (nur bei Kaltstart der SE).

## 4.5.2 Warmstarts

Bei einem Warmstart werden folgende Aktionen durchgeführt:

- Das Status\_EQM wird für jeden SD $\mu$ P als periodischer Auftrag (Periode 1 s) konnektiert (dadurch werden die Istpositionen aller Komponenten eines SD $\mu$ P aktualisiert).
- Das GStatus\_EQM wird für jeden SD $\mu$ P als periodischer Auftrag (Periode 60 s) konnektiert (dadurch wird der Gerätestatus aller Komponenten eines SD $\mu$ P aktualisiert).
- Falls noch nicht erfolgreich geschehen, werden die Infopakete gelesen.
- Die Statusinformationen werden gelesen.
- Die Ist-Positionen werden gelesen.

## 4.6 Handbetrieb

Im Handbetrieb können zwar die Statusinformationen, nicht aber die Positionen gelesen werden. Aus diesem Grund wird dann eine Fehlermeldung generiert.

## 4.7 Ableitung des Hardwarefehler-Bits aus dem Gerätestatus

Eine Abbildung von Hardwarefehlern im Status des Gerätes ist nicht vorgesehen.

## 4.8 Verhalten bei Störungen

### 4.8.1 Geräteinterlock

Die Schrittmotorsteuerung liefert keinen Geräteinterlock.

### 4.8.2 Event-Sequenzfehler

Event-Sequenzfehler sind nicht möglich.

### 4.8.3 Event-Overrun

Event-Overrun-Fehler sind nicht möglich.

### 4.8.4 Emergency-Event

Emergency-Events müssen nicht berücksichtigt werden.

### 4.8.5 Ausfall der Kommunikation EC – Gerät

Der Ausfall der Kommunikation zwischen EC und Gerät führt zu Timeouts, Checksum-Fehlern oder ähnlichen Fehlermeldungen. Über die Protokollstatistik des EC kann ermittelt werden *wie oft welche* Fehler aufgetreten sind (Menüpunkt 30 im NODAL-Programm COMP.TEST). Lässt sich das Kommunikationsproblem nicht durch einen Reset am SD $\mu$ P beseitigen, so sind in den meisten Fällen ausführlichere Untersuchungen des SD $\mu$ P durch M. Hartung oder M. Fradj notwendig.

## 4.9 Bedienungsfehler vom Operating

Bei jedem Fahrkommando wird auf der EC-Ebene überprüft, ob schon ein Antrieb aus dieser Gerätegruppe (siehe hierzu Kapitel 3.1.5 auf Seite 15) fährt. Ist dies der Fall, so wird das Kommando mit einer entsprechenden Fehlermeldung abgewiesen.

## 4.10 Sonstige Anforderungen

Es ist möglich (aber unwahrscheinlich), dass sich der Status eines Antriebs ändert, ohne dass vom Kontrollsystem aus eine Aktion ausgelöst wurde (z. B. Manipulation *von Hand*, Kabeldefekt am Endschalter, ...). Durch die *normale* Statusüberwachung werden solche Änderungen nur dann erfasst, wenn der betroffene Antrieb explizit gefahren wird (d. h. der Multiplexer auf den entsprechenden Kanal geschaltet ist). Deshalb muss periodisch (ca alle 10 Minuten) der Gerätestatus jedes einzelnen Antriebs überprüft werden (siehe hierzu auch Kapitel 3.1.1 auf Seite 11), indem der entsprechende Kanal explizit über den Multiplexer angewählt wird.

# 5 Die Repräsentation des Gerätes

## 5.1 Kennzeichnung des Gerätemodells

Das Gerätemodell hat die Bezeichnung **DS**.

Die Gerätemodellnummer ist  $11_{dez}$ .

## 5.2 Die Master-Properties

Master Properties							
Property	Klasse	Parameter		Daten		Größe	
		Anz.	Typ	Anz.	Typ	Einh.	Exp.
POWER	R/W	0	--	1	BitSet16	1	0
STATUS	R	0	--	1	BitSet32	1	0
INIT	N	0	--	0	--	--	--
RESET	N	0	--	0	--	--	--
VERSION	RA	0	--	48	BitSet8	1	0
INFOSTAT	RA	0	--	25	BitSet32	1	0
INFO	RA	0	--	6	Integer32	1	0
CONSTANT	RA	0	--	5	Integer32	1	0
POSIABSS	R/W	0	--	1	Integer32	m	-4
POSIABSI	R	1	Integer32	1	Integer32	m	-4
POSIREL	W	0	--	1	Integer16	1	0
STEPS	W	0	--	1	Integer16	1	0
ENDLAGE	W	0	--	1	BitSet16	1	0
REFTEST	R	0	--	1	Integer32	m	-4
SLITINFO	RA	3	Integer32	3	Integer32	1	0

### 5.2.1 POWER

**Bedeutung:** Gibt an, ob der Leistungsteil des Gerätes ein- oder ausgeschaltet ist.

**Parameter:** keine

**Daten:** Das Gerät ist immer eingeschaltet. Der Wert der Property sollte immer Eins sein.

### 5.2.2 STATUS

**Bedeutung:** Auslesen des 32-Bit-Gerätstatus

**Parameter:** keine

**Daten:** Das 32-Bit-Statuswort. Die Bits entsprechen den Statusbits, wie sie in Abschnitt 3.3 auf Seite 18 und in der Tabelle 3.3 auf Seite 18 erklärt sind.

### 5.2.3 INIT

**Bedeutung:** Initialisierung des Gerätes (Kaltstart): Für die dabei durchzuführenden Aktionen siehe Abschnitt 4.5.1 auf Seite 20.

**Parameter:** keine

**Daten:** keine

### 5.2.4 RESET

**Bedeutung:** Reset des Gerätes (Warmstart). Für die dabei durchzuführenden Aktionen siehe Abschnitt 4.5.2 auf Seite 21.

**Parameter:** keine

**Daten:** keine

### 5.2.5 VERSION

**Bedeutung:** Lesen der Versionskennung der Gerätesoftware

**Parameter:** keine

**Daten:** Versionskennung als ASCII-String, pro Datum ein ASCII-Zeichen

Bytes	Inhalt
1...12	Version der USRs
13...24	Version der EQMs
25...36	Version des Standard-MIL-Treibers
37...48	Variante der EQMs

## 5.2.6 INFOSTAT

**Bedeutung:** Diese Property liefert einige wichtige Geräteinformationen in einem Zugriff. Die Informationen werden direkt aus dem Dualport-RAM gelesen, also ohne den expliziten Aufruf eines EQMs, und sind daher in der Abarbeitung nicht abhängig von Kommandoevents.

**Parameter:** Keine

**Daten:** Die 25 Langworte enthalten im Einzelnen:

- 1:** Gerätestatus (wie in der Property STATUS)
- 2:** Gibt in den oberen 16 Bits an, welcher virtuelle Beschleuniger aktiv gesetzt ist (ein Bit pro Beschleuniger). Das niederwertigste Bit (Bit 16) gibt den Beschleuniger 15 an, das Bit 31 den Beschleuniger 0. Die unteren 16 Bit sind nicht verwendet. Dabei bedeutet Null, dass der Beschleuniger inaktiv ist und Eins, dass er aktiv ist.
- 3:** Master-Fehler. Hier ist derjenige Master"-Gerätefehlercode mit dem schwersten Fehler"-grad eingetragen. Bei mehreren Fehlern mit dem gleichen Fehlergrad wird der erste eingetragen, der gefunden wurde.
- 4:** Slave Fehler für virtuellen Beschleuniger 0. Entsprechend dem Master-Fehler wird hier der nach dem Fehlergrad schwerste Slave-Gerätefehlercode für den Beschleuniger 0 eingetragen.
- 5:** Entsprechend Punkt 4, aber für virtuellen Beschleuniger 1.
- ⋮
- 19:** Entsprechend Punkt 4, aber für virtuellen Beschleuniger 15.
- 20:** EC-Mode. In den oberen 16 Bit des Langwortes steht der von der Gerätesoftware eingestellte Default-EC-Mode, in den unteren 16 Bit der aktuelle EC-Mode. Folgende Modi sind definiert:
  - 0:** *not set*
  - 1:** *Preset\_Command* Der ECM hat das Umschalten in Command-Mode vorbereitet aber noch nicht beendet.
  - 2:** *Command* Der ECM läuft im Command-Mode.
  - 3:** *Preset\_Event* Der ECM hat das Umschalten in Event-Mode vorbereitet aber noch nicht beendet.
  - 4:** *Event* Der ECM läuft im Event-Mode.
- 21:** EC-Performance-Mode. In den oberen 16 Bit des Langwortes steht der von der Gerätesoftware eingestellte Default-Performance-Mode, in den unteren 16 Bit der aktuelle Performance-Mode. Folgende Modi sind definiert:
  - 0:** *not set*
  - 1:** *Display* Der ECM läuft im Display-Mode.
  - 2:** *Preset\_Turbo* Der ECM hat das Umschalten in den Turbo-Mode vorbereitet aber noch nicht beendet.
  - 3:** *Turbo* Der ECM läuft im Turbo-Mode.
- 22:** HW\_Warning\_Maske. Die 32 Bits geben an aus welchen Bits im Gerätestatus das HW-Warning-Bit im Status abgeleitet wird.



- 23: Reserviert für Erweiterungen
- ⋮
- 25: Reserviert für Erweiterungen

### 5.2.7 INFO

**Bedeutung:** Liefert das Info-Packet eines Geraets

**Parameter:** keine

**Daten:** Die 20 Langworte enthalten im Einzelnen:

- 1: VersionsNr
- 2: Nomenklatur 4Byte( bei SD $\mu$ P Dummy)
- 3: Nomenklatur 4Byte( bei SD $\mu$ P Dummy)
- 4: DeviceType
- 5: Mindestabstand in 1/10 mm
- 6: Pulse pro cm
- 7: mm/U bei Winkelkodierer
- 8: max. Position (bei Paaren links)
- 9: min. Position
- 10: max. Position Kanal gegenueber (bei Paaren rechts)
- 11: min. Position Kanal gegenueber
- 12: PositionsFaktor (nur SDI)
- 13: Startfrequenz (nur SDI)
- 14: Maximalfrequenz (nur SDI)
- 15: Pulsbreite (nur SDI)
- 16: Polspolaritaet (nur SDI)
- 17: Rampensteilheit (nur SDI)
- 18: Positionsoffset (nur SDI)
- 19: Potilaenge in mm
- 20: reserve

Das Lesen der Info sollte nur noetig sein um die am Geraet eingestellten Werte auch mal zu pruefen.  
Das Operating sollte mit der Property CONSTANT auskommen.

### 5.2.8 CONSTANT

**Bedeutung:** Liefert die fuer das Operating relevanten konstanten Geraeteinfos

**Parameter:** keine

**Daten:** Die Langworte enthalten im Einzelnen:

- 1: devtyp
- 2: Mindestabstand

- 3:** min. Position
- 4:** max. Position
- 5:** Pulse pro cm

### 5.2.9 POSIABSS

**Bedeutung:** Bringt einen Antrieb auf eine absolute Soll-Position.

**Parameter:** keine

**Daten:** Positionsangabe in 1/10 mm

### 5.2.10 POSIABSI

**Bedeutung:** Liefert von einem Antrieb die absolute Ist-Position.

**Parameter:** 1 Dummy siehe (siehe Seite 37)

**Daten:** Positionsangabe in 1/10 mm

### subsubsectionPOSIREL

**Bedeutung:** Ein Antrieb wird relativ zur aktuellen Soll-Position bewegt. Um eine genauere Positionierung zu erreichen, wird die angegebene Distanz auf die Anzahl der zu fahrenden Schritte umgerechnet und der Antrieb um diese Anzahl von Schritten bewegt.

**Parameter:** keine

**Daten:** Angabe der Distanz in 1/10 mm

### 5.2.11 REFTEST

**Bedeutung:** Lesen der Referenzspannungswerte eines Antriebspaares.

**Parameter:** keine

**Daten:** In 2 Integer16-Werten werden die Referenzspannungswerte für den linken (1. Wert) und den rechten (2. Wert) geliefert.

### 5.2.12 STEPS

**Bedeutung:** Ein Antrieb wird um die angegebene Anzahl von Schritten bewegt.

**Parameter:** keine

**Daten:** Anzahl der Schritte

### 5.2.13 ENDLAGE

**Bedeutung:** Der Antrieb wird an die angegebene Endlage gefahren.

**Parameter:** keine

**Daten:** Folgende Bedeutungen sind vereinbart:

**0:** Endlage außen

**1:** Endlage innen

### 5.2.14 SLITINFO

**Bedeutung:** Es werden Soll-, Ist-Position und Status aus dem DPRAM geliefert.

**Parameter:** 3 Dummies (siehe Seite 37)

**Daten:** Die drei Daten haben folgende Bedeutung:

**1:** Status

**2:** Soll-Position in 1/10 mm

**3:** Ist-Position in 1/10 mm

## 5.3 Die Slave-Properties

Slave-Properties							
Property	Klasse	Parameter		Daten		Größe	
		Anz.	Typ	Anz.	Typ	Einh.	Exp.
ACTIV	R/W	0	--	1	BitSet16	1	0
COPYSET	W	0	--	1	BitSet16	1	0
EQMERROR	RA	217	BitSet32	348	BitSet32	1	0

### 5.3.1 ACTIV

**Bedeutung:** Gibt an, ob das Gerät für den zugehörigen virtuellen Beschleuniger an der Puls-zu-Puls-Modulation (kurz PPM) teilnehmen soll bzw. teilnimmt.

Auch nichtgepulste Geräte müssen diese *obligatorische* Property haben. Ungepulste Geräte liefern beim Versuch ACTIV zu schreiben einen entsprechenden Fehler und beim Versuch ACTIV zu lesen eine Information oder eine Warnung zurück.

**Parameter:** keine

**Daten:** Das Datum kann nur zwei Werte annehmen. Null heißt, das Gerät nimmt für den zugeordneten Beschleuniger *nicht* an der PPM teil bzw. soll *nicht* an der PPM teilnehmen. Eins heißt, das Gerät nimmt für den zugeordneten Beschleuniger an der PPM teil bzw. soll an der PPM teilnehmen.

### 5.3.2 COPYSET

**Bedeutung:** Kopiert alle Geräteeinstellungen (Sollwerte) eines virtuellen ("fremden") Beschleunigers in den zugehörigen ("eigenen") Beschleuniger. Die Property ist bei den Schrittmotoren wirkungslos (siehe oben)!

**Parameter:** keine

**Daten:** Nummer des virtuellen ("fremden") Beschleunigers, von dem die Einstellungen (Sollwerte) kopiert werden sollen.

### 5.3.3 EQMERROR

**Bedeutung:** Fehlermeldungen der auf der SE installierten Gerätesoftware. Es werden die aktuellen Fehlermeldungen sowohl für die Masterfehler als auch für die Slavefehler der Geräteebene geliefert. Dazu wird auch der Inhalt des Fehlerpuffers zurückgegeben, in dem die letzten aufgetretenen Fehler abgespeichert wurden.

**Parameter:** Hier hat nur der erste der 217 Parameter eine Bedeutung.

**1:** Wird bei konnektierten Aufträgen ausgewertet.

0: Es wird bei jeder Ausführung des Auftrages eine Antwort verschickt.

1: Es wird bei jeder Ausführung des Auftrages nur dann eine Antwort verschickt, wenn sich seit dem letzten Aufruf der Inhalt der Daten geändert hat.

**2...217:** Dummy, sie werden vom MOPS intern verwendet und können vom Benutzer beliebig gesetzt werden.

**Daten:** Die Anzahl der Fehlermeldungen sei bezeichnet durch:

$m$  Zahl der Master-Fehlermeldungen

$s$  Zahl der Slave-Fehlermeldungen

$b$  Größe des Fehlerpuffers

Weiterhin soll gelten:

$$l = m + s$$

$$t = m + s + b$$

Die Daten im Einzelnen:

1 : In den unteren beiden Bytes sind die Anzahl der Master-Fehlermeldungen  $m$  und die Anzahl der Slave-Fehlermeldungen  $s$  angegeben:

0	0	$s$	$m$
---	---	-----	-----

2 : erste Master-Fehlermeldung

⋮

$m + 1$  : letzte Master-Fehlermeldung

$m + 2$  : erste Slave-Fehlermeldung

⋮

$l + 1$  : letzte Slave-Fehlermeldung

$l + 2$  : Länge  $b$  des Fehlerpuffers

- $l + 3$  : Zahl der Einträge im Fehlerpuffer
- $l + 4$  : Index des ersten freien Platzes im Fehlerpuffer  
(der Fehlerpuffer ist ein Ringpuffer)
- $l + 5$  : Erster Speicherplatz im Fehlerpuffer
- $\vdots$
- $t + 4$  : Letzter Speicherplatz im Fehlerpuffer

# Part II

## Der Entwurf der Software

### 6 Softwareentwurf

Keine erwähnenswerten Besonderheiten.

### 7 Lokale Datenbasis

#### 7.1 Tabelle der Konstanten

Einträge in der Konstantentabelle der lokalen Datenbasis gibt es nicht.

### 8 Dualport RAM

In den Datenstrukturen des Dualport RAM sind keine erwähnenswerten Besonderheiten enthalten.

### 9 USRs - User Service Routinen

#### 9.1 Obligatorische USRs

##### 9.1.1 N\_Init

##### 9.1.2 N\_Reset

##### 9.1.3 R\_Status

##### 9.1.4 R\_Power

##### 9.1.5 W\_Power

##### 9.1.6 R\_Active

##### 9.1.7 W\_Active

##### 9.1.8 W\_CopySet

##### 9.1.9 R\_EQMErr

##### 9.1.10 R\_Version

##### 9.1.11 R\_InfoStat

#### 9.2 Gerätespezifische USRs

Zuzüglich der obligatorischen USRs werden für die Steuerung des Schrittgerätes folgende gerätespezifischen USRs benötigt:

### **9.2.1 R\_RefTest**

Liefert die Referenzspannungswerte eines Antriebs.

### **9.2.2 R\_PosiAbsi**

Liefert von einem Antrieb die absolute Ist-Position.

### **9.2.3 R\_PosiAbss**

Liefert von einem Antrieb die absolute Soll-Position.

### **9.2.4 W\_PosAbs**

Setzt für einen Antrieb die absolute Soll-Position.

### **9.2.5 W\_PosiRel**

Bewegt einen Antrieb um die angegebene Distanz.

### **9.2.6 R\_Info**

Liefert das Infopaket zu einem Antrieb.

### **9.2.7 R\_Constant**

Liefert Gerätekonstanten eines Antriebs.

### **9.2.8 W\_StepS**

Bewegt einen Antrieb schrittweise.

### **9.2.9 W\_Endlage**

Führt einen Antrieb zur Endlage (außen oder innen).

### **9.2.10 R\_SlitInfo**

Liefert Status, Sollposition und Istposition in einem Zugriff.

## **9.3 Globale Routinen**

# **10 EQMs - Equipment Module**

## **10.1 Interne Zustände**

### **10.1.1 Bedeutung der internen Zustände**

Für die Gerätesoftware sind folgende interne Zustände definiert:

not_set	Initzustand. Dieser Zustand sollte nie auftreten.
emergency	Ein Emergency-Event wurde empfangen. Dieser Zustand darf nur durch Rücksetzen vom Operating verlassen werden.
error	Während der Abarbeitung eines EQMs wurde ein Fehler erkannt.
ready	Das Gerät ist bereit für Aktionen. Ausgangszustand am Beginn eines virtuellen Beschleunigers.
busy	Das Gerät bearbeitet den letzten Fahrbefehl (wird bisher noch nicht benutzt).

### 10.1.2 Übergänge zwischen den Zuständen

Erläuterung, welche Übergänge zwischen den internen Zuständen vorgesehen sind und wodurch sie ausgelöst werden sollen.

Die Zustände und die Übergänge zwischen denselben sind in Tabelle 15 zusammengefasst. Die Legende zu diesen Tabellen ist in Tabelle 16 zu finden.

Tabelle der Zustandsübergänge				
von↓	nach→	emergency	error	ready
emergency	U:	--	--	RESET
	B:	--	--	--
	A:	--	--	Reset_EQM
error	U:	Evt_Emerg	--	RESET
	B:	--	--	--
	A:	Emerg_EQM	--	Reset_EQM
ready	U:	Evt_Emerg	--	--
	B:	--	--	--
	A:	Emerg_EQM	--	--

Table 15: Zustandsübergangsdigramm

### 10.1.3 Emerg\_EQM

**Event:** Evt\_Emergency.

**Aktion:** Internen Zustand auf "Emergency" setzen. Keine gerätespezifische Aktion ausführen.

## 10.2 Periodisch konnektierte EQMs

### 10.2.1 Status\_EQM

**Zeit:** 1 s

**Anzahl:** unendlich

**Aktion:** Liest im sogenanntem Blockmode den Status und die Position aller an einer Elektronik SD $\mu$ P und SDI angeschlossenen Antriebe. Dieses EQM wird pro angeschlossener Elektronik nur einmal konnektiert. Der Status der Mehrfachantriebe wird dabei aus den



## Legende

- Die Priorität der Zustände (höchste Priorität zuerst): emergency, interlock, local, power\_""off und power\_seq, error, ready.

Liegen mehrere Bedingungen für verschiedene Zustände gleichzeitig vor (z. B. Netz aus und Gerät auf Handbetrieb), muss der jeweils wichtigste Zustand eingenommen werden.

- U: Auslösende Ursache.
 

Evt_Emerg	Pulszentrale verschickte Emergency-Event.
RESET	Reset wird per Kommando oder Knöpfchendrücken ausgelöst.
- B: Abzuprüfende Bedingung.
 

R	Remotebit des Status steht auf Remote.
r	Remotebit des Status steht auf Local.
P	Powerbit des Status steht auf Power on.
p	Powerbit des Status steht auf Power off.
- A: Ausführende Stelle des Zustandübergangs.
 

Status lesen (period.)	Beim periodischen (oder regelmäßigen) Lesen des Status.
..._EQM	Innerhalb des EQMs ..._EQM.

Table 16: Legende zu den Zustandsübergangsdiagrammen

Stati der zugehoerigen Einzelantriebe generiert. (siehe hierzu auch Abschnitt 3.1.1 auf Seite 14)

### 10.2.2 Update\_Config\_EQM

**Zeit:** 60 s

**Anzahl:** unendlich

**Aktion:** Aktualisieren der Geräteverfügbarkeit: SDI und SD $\mu$ P werden hier unterschiedlich behandelt. Beim SD $\mu$ P wird versucht, von allen möglichen Geräteadressen den Status zu lesen. Erfolgt eine Reaktion, wird das Gerät (Einzelantrieb und Paar) als "¿known"¿ geführt. Alle Geraetenummern die Kollimatoren zugeordnet sind (ifbAdr ¿= BaseAdr+24), werden immer als "¿known"¿ geführt. Das SDI verwaltet keine einzelnen Geraeteadressen. Wenn hier der Status der Grundadresse der Elektronik lesbar ist, wird anschliessend das Infopaket geprueft: Einzelantriebe werden "¿known"¿, wenn im Infopaket eine Nomenklatur fuer diesen Antrieb eingetragen ist, Paare werden "¿known"¿, wenn im Infopaket fuer beide zugehoerigen Einzelantriebe eine Nomenklatur eingetragen ist. Alle Geraetenummern die Kollimatoren zugeordnet sind (ifbAdr ¿= BaseAdr+24), werden immer als "¿known"¿ geführt.

## 10.3 An externe Interrupts konnektierte EQMs

### 10.3.1 Interlock\_EQM

**Interrupt:** Summen-Interlock

**Aktion:** keine Aktion

### 10.3.2 DRD\_EQM

**Interrupt:** Data Ready Interrupt

**Aktion:** keine Aktion

### 10.3.3 DRQ\_EQM

**Interrupt:** Data Request Interrupt

**Aktion:** keine Aktion

## 10.4 Kommandogetriggerte EQMs

### 10.4.1 Dev\_Init\_EQM

### 10.4.2 Dev\_Reset\_EQM

### 10.4.3 Status\_EQM

### 10.4.4 Active\_EQM

### 10.4.5 Power\_EQM

**Parameter:** keine

**Daten:** keine

**Aktion:** Explizites Lesen des Gerätestatus (inklusive Multiplexer schalten)

### 10.4.6 Test\_EQM

**Parameter:** keine

**Daten:** keine

**Aktion:** Lesen der Referenzspannung für einen Motoreinschub

### 10.4.7 Move\_EQM

**Parameter:** Das EQM benötigt 1 Parameter.

1. Fahrmodus:

**move\_single (1):** Einzelnen Antrieb auf Position fahren

**move\_step (2):** Einzelnen Antrieb um eine Anzahl von Schritten fahren.

**move\_endl (3):** Einzelnen Antrieb auf Endlage (innen oder außen) fahren.

**Daten:** keine

**Aktion:** Veranlasst ein Fahren des Antriebs im angegebenen Modus.

## 10.5 EQMs für die Diagnose vor Ort

### 10.5.1 Display\_DPR.EQM

**Parameter:** Das EQM benötigt 2 Parameter.

1. virtueller Beschleuniger (in Hex angeben)
2. logische Gerätenummer (in Hex angeben)

**Daten:** keine

**Aktion:** Zeigt am Bildschirm vor Ort die wichtigsten Daten aus dem DPRAM für das gewählte Gerät und den gewählten virtuellen Beschleuniger an.

### 10.5.2 Display\_DevErr.EQM

**Parameter:** Das EQM benötigt 2 Parameter.

1. virtueller Beschleuniger (in Hex angeben)
2. logische Gerätenummer (in Hex angeben)

**Daten:** keine

**Aktion:** Zeigt am Bildschirm vor Ort die Error-Codes aus der aus der Datenstruktur im Dualport-RAM für das gewählte Gerät und den gewählten virt. Beschleuniger an.

## 10.6 Sonstige EQMs

### 10.6.1 Startup.EQM

Installiert die Event-EQM-Konnektierung für alle virtuellen Beschleuniger (siehe hierzu auch Abschnitt 4.4 auf Seite 20).

## 10.7 Globale Routinen

### 10.7.1 Read\_and\_Update\_Status

Liest vom SD $\mu$ P den *Blockstatus* (also Status und Istposition der einzelnen Kanäle) und verteilt die Informationen mit Hilfe von *Get\_LogDev* und *Set\_MStatus* auf die einzelnen *logischen* Geräteeinheiten. Siehe Abschnitt 2 auf Seite 8.

### 10.7.2 Read\_and\_Update\_GStatus

Aktualisiert den Elektronikstatus und liest explizit den Gerätestatus (GSTATUS) eines einzelnen Gerätes. Dadurch werden ggf. die Multiplexer auf den entsprechenden Kanal umgeschaltet.

### 10.7.3 Get\_LogDev

Liefert zu einer gegebenen physikalischen Geräteadresse die logische device-number oder 255 (= -1) falls das Gerät nicht *online* ist.

### 10.7.4 Set\_MStatus

Generiert aus Elektronikstatus (ESTATUS) und Gerätestatus (GSTATUS) den eigentlichen Gerätestatus (m\_sts) im DPRAM.

### 10.7.5 Do\_Intr\_Service\_Prep

Macht im Wesentlichen eigentlich nichts, weil der Interlockinterrupt bei Schrittmotoren ohnehin nicht verwendet wird.

## 11 Varianten

Dieses Gerätemodell besitzt keine Varianten.

## 12 Besonderheiten

### 12.1 Supergeräte

In der lokalen Datenbank der VME-Ebene können Supergeräte vereinbart werden, die aus mehreren Einzelantrieben (rechts, links, oben und unten) bestehen. Die von den Elektroniken nicht abgebildeten Antriebe werden von der SE als 'known' geführt und werden 'online', sobald es einen entsprechenden Eintrag in der Datenbasis gibt. Für diese Geräte das gilt das gleiche Gerätemodell *STEP* es werden aber ( mit createMap) nur die Standardproperties bekannt gemacht.

## 12.2 Der Response-Algorithmus

Um Rechen- und Transferzeiten auf den Computern der Operating-Ebene zu sparen, wird ein Algorithmus angewandt, der bei konnektierten Aufträgen nur bei Änderungen Daten liefert. Dazu ist prinzipiell das Problem zu lösen, dass eine USR kein *eigenes* RAM besitzt, in dem sie sich von Aufruf zu Aufruf etwas (z. B. die letzten Istwerte) *merken* könnte. Allerdings liegt während des gesamten Zeitraums einer bestehenden Konnektierung das ursprüngliche Auftragspaket (Parameter und Daten) im DPRAM des Ethernet-Controllers vor.

Der *Trick*, dass die USR sich von Aufruf zu Aufruf etwas merken kann, besteht nun darin, dass bei der Konnektierung der USR genau die Anzahl der zu erwartenden Daten als Parameter (mit beliebigen Werten, also Dummies) mitgeschickt werden. Dadurch ergibt sich für die USR die Möglichkeit, sich im Empfangsparameter Teil des Auftragspaketes die letzten aktuellen Daten zu *merken* und diese beim nächsten Aufruf mit den aktuellen zu vergleichen. Um diesen Algorithmus nutzen zu können, muss

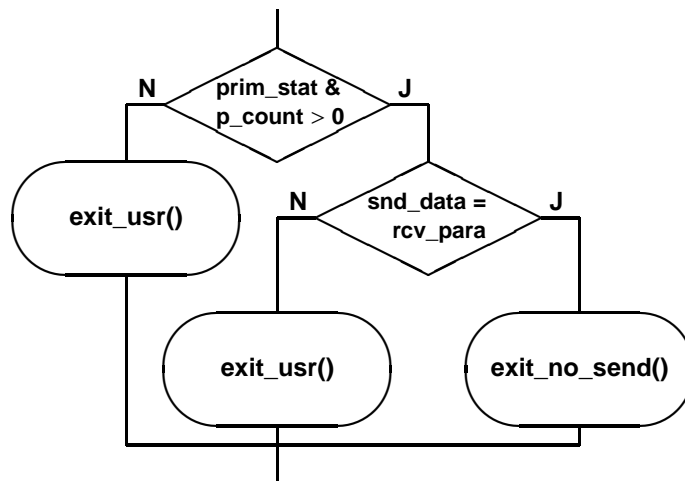


Figure 4: Der Response-Algorithmus

das Operatingprogramm eine, der USR-Definition entsprechende, Anzahl von Dummy-Parametern bei der Konnektierung mitschicken. Wenn der Vergleich zwischen alten und aktuellen Werten eine Differenz zeigt oder ein Fehler aufgetreten ist, werden die aktuellen Werte gemeldet. Werden keine Parameter mitgeschickt, werden immer die aktuellen Werte zurückgemeldet.

# Index

## —Symbole —

Änderungsprotokoll ..... 2

## —A—

Abriss ..... 2  
Active\_EQM ..... 34  
An externe Interrupts konnektierte EQMs .. 34  
Aufgabe des Gerätes ..... 7  
Ausschalten ..... 20

## —B—

Bedienung des Gerätes ..... 19  
Bedienungsfehler ..... 22  
Besonderheiten ..... 36  
Blockschaltbild ..... 8

## —D—

Das SD-Interface ..... 10, 15  
Datenbasis ..... 30  
Der SD $\mu$ P ..... 9, 10  
Dev\_Init\_EQM ..... 34  
Dev\_Reset\_EQM ..... 34  
Display\_DevErr\_EQM ..... 35  
Display\_DPR\_EQM ..... 35  
DRD Interrupt ..... 15  
DRD\_EQM ..... 34  
DRQ Interrupt ..... 15  
DRQ\_EQM ..... 34  
Dualport RAM ..... 30

## —E—

Einschalten ..... 20  
Emerg\_EQM ..... 32  
Emergency-Event ..... 21  
EQMs ..... 31

- An externe Interrupts konnektierte .. 34
  - DRD\_EQM ..... 34
  - DRQ\_EQM ..... 34
  - Interlock\_EQM ..... 34

- Eventkonnektierte
  - Emerg\_EQM ..... 32
- für die Diagnose vor Ort ..... 35
  - Display\_DevErr\_EQM ..... 35
  - Display\_DPR\_EQM ..... 35
- Globale Routinen ..... 35
  - Do\_Intr\_Service\_Prep ..... 36
  - Get\_LogDev ..... 36
  - Read\_and\_Update\_GStatus ..... 35
  - Read\_and\_Update\_Status ..... 35
  - Set\_MStatus ..... 36
- Kommandogetriggerte ..... 34
  - Active\_EQM ..... 34
  - Dev\_Init\_EQM ..... 34
  - Dev\_Reset\_EQM ..... 34
  - Move\_EQM ..... 34
  - Power\_EQM ..... 34
  - Status\_EQM ..... 34
  - Test\_EQM ..... 34
- Periodisch konnektierte ..... 32
  - Status\_EQM ..... 32
  - Update\_Config\_EQM ..... 33
- Sonstige ..... 35
  - Startup\_EQM ..... 35

Event-Overrun ..... 21  
Event-Sequenzfehler ..... 21  
Eventkonnektierungen ..... 20

## —F—

Funktionscodes

- IFB\_AufZu ..... 13
- IFB\_Block ..... 14, 16, 17
- IFB\_Ein ..... 13
- IFB\_Info ..... 12
- IFB\_Move ..... 14
- IFB\_Par ..... 14
- IFB\_Posit ..... 13
- IFB\_Raus ..... 14
- IFB\_Rdstat ..... 11, 16
- IFB\_Rdstat\_1 ..... 11
- IFB\_Rein ..... 14
- IFB\_Test ..... 13

Funktionscodes SD-Interface .....	15
Funktionscodes SDuP .....	10

—G—

Genauigkeitsanforderungen .....	20
Gerät	
• Aufgabe .....	7
• Bedienung .....	19
• Hardware .....	8
• logisches .....	15
• Repräsentation .....	22
• Schnittstellen SD $\mu$ P .....	10
Gerätemodell .....	7
• Kennzeichnung .....	22
• Master-Properties .....	22
• Slave-Properties .....	27
Gerätstatusüberwachung .....	19
Globale Routinen .....	31, 35

—H—

Handbetrieb .....	21
Hardware des Gerätes .....	8
Hardwarefehler-Bit .....	21
Hardwarestatus .....	18

—I—

IFB_AufZu .....	13
IFB_Block .....	14, 16, 17
IFB_Ein .....	13
IFB_Info .....	12
IFB_Move .....	14
IFB_Par .....	14
IFB_Posit .....	13
IFB_Raus .....	14
IFB_Rdstat .....	11, 16
IFB_Rdstat_1 .....	11
IFB_Rein .....	14
IFB_Test .....	13
Init .....	20
Interfacekarte .....	10, 15
Interlock .....	15, 21
Interlock-Eingänge .....	8
Interlock_EQM .....	34

Interne Zustände .....	31
Interrupt	
• DRD Interrupt .....	15
• DRQ Interrupt .....	15
• Interlock .....	15

—K—

Kaltstarts .....	20
Kommandogetriggerte EQMs .....	34

—L—

logisches Gerät .....	15
Lokale Datenbasis .....	30
Lokalen Datenbasis	
• Tabelle der Konstanten .....	30

—M—

Master-Properties .....	22
Move_EQM .....	34
Multiplexer .....	8

—N—

N_Init .....	30
N_Reset .....	30
Normalbetrieb .....	19

—O—

Overrun .....	21
---------------	----

—P—

Periodisch konnektierte EQMs .....	32
Positionsüberwachung .....	19
Positionsmesseinheit .....	8
Potentiometer .....	8
Power_EQM .....	34
PPM .....	27
Properties	
• ACTIV .....	27
• CONSTANT .....	25
• COPYSET .....	28

- ENDLAGE ..... 27
- EQMERROR ..... 28
- INFO ..... 25
- INFOSTAT ..... 24
- INIT ..... 23
- Master- ..... 22
- POSIABSI ..... 26
- POSIABSS ..... 26
- POSIREL ..... 26
- POWER ..... 23
- REFTTEST ..... 26
- RESET ..... 23
- Slave- ..... 27
- SLITINFO ..... 27
- STATUS ..... 23
- STEPS ..... 26
- VERSION ..... 23

Puls-zu-Puls-Modulation ..... 27

—R—

- R\_Active ..... 30
- R\_Constant ..... 31
- R\_EQMErr ..... 30
- R\_Info ..... 31
- R\_InfoStat ..... 30
- R\_PosAbsi ..... 31
- R\_PosAbs ..... 31
- R\_Power ..... 30
- R\_RefTest ..... 31
- R\_SlitInfo ..... 31
- R\_Status ..... 30
- R\_Version ..... 30
- Repräsentation des Gerätes ..... 22
- Reset ..... 21
- Respons-Algorithmus ..... 37

—S—

- Schnittstellen zum SD $\mu$ P ..... 10
- Schrittmotor fahren ..... 19
- Sequenzfehler ..... 21
- Service ..... 8
- Slave-Properties ..... 27
- Softwareentwurf ..... 30
- Softwarestatus ..... 18

- Sonstige EQMs ..... 35
- Störungen ..... 21
  - Emergency-Event ..... 21
  - Event-Overrun ..... 21
  - Event-Sequenzfehler ..... 21
  - Interlock ..... 21
  - Kommunikation EC – Gerät ..... 21
- Startup\_EQM ..... 35
- Startwerte ..... 20
- Status\_EQM ..... 32, 34
- Statusbits ..... 18
- Supergeräte ..... 36

—T—

- Test\_EQM ..... 34
- Timing ..... 20

—U—

- Update\_Config\_EQM ..... 33
- USRs ..... 30
  - gerätespezifische ..... 30
    - R\_Constant ..... 31
    - R\_Info ..... 31
    - R\_PosAbsi ..... 31
    - R\_PosAbs ..... 31
    - R\_RefTest ..... 31
    - R\_SlitInfo ..... 31
    - W\_Endlage ..... 31
    - W\_PosAbs ..... 31
    - W\_PosRel ..... 31
    - W\_StepS ..... 31
  - Globale Routinen ..... 31
  - obligatorische ..... 30
    - N\_Init ..... 30
    - N\_Reset ..... 30
    - R\_Active ..... 30
    - R\_EQMErr ..... 30
    - R\_InfoStat ..... 30
    - R\_Power ..... 30
    - R\_Status ..... 30
    - R\_Version ..... 30
    - W\_Active ..... 30
    - W\_CopySet ..... 30
    - W\_Power ..... 30



—V—

Varianten . . . . .	36
• Betriebs- . . . . .	20
• Software- . . . . .	36

—W—

W_Active . . . . .	30
W_CopySet . . . . .	30
W_Endlage . . . . .	31
W_PosAbs . . . . .	31
W_PosiRel . . . . .	31
W_Power . . . . .	30
W_StepS . . . . .	31
Warmstarts . . . . .	21
Winkelkodierer . . . . .	8

—Z—

Zeitkritische Anforderungen . . . . .	20
Zustände	
• Interne . . . . .	31
– Übergänge . . . . .	32
– Bedeutung . . . . .	31