

Originalbetriebsanleitung

PA-CONTROL
Programmierhandbuch
Ab Version 5.25

Ausgabe: Dezember 2011
Art.-Nr.: 1071142

IEF Werner GmbH
Wendelhofstraße 6
78120 Furtwangen
Telefon: 07723-925-0
Telefax: 07723-925-100
www.IEF-Werner.de

Änderungshistorie

Dokumentencode	Datum	Änderung
...Programmierhandbuch_R5a.doc	Mai 2007	Neuerstellung Dokument
...Programmierhandbuch_R5b.doc	Oktober 2007	Korrekturen und Ergänzungen
...Programmierhandbuch_R5c.doc	Oktober 2007	Einfügen Tabelle PA-CONTROL Tastencode
...Programmierhandbuch_R5d.doc	Juli 2008	Titelblatt geändert
...Programmierhandbuch_R5e.doc	September 2008	Basierend auf Version ...D6a.doc: Kap.: 1.7.2.3; 1.7.2.4; 1.7.2.5; 1.7.2.7 um "A0" ergänzt
...Programmierhandbuch_R5f.doc	Februar 2009	Basierend auf Version ...D5f.doc: Korrekturen und Ergänzungen
...Programmierhandbuch_R5g.doc	Februar 2010	Beim Befehl „PARAMETER“ den Achstyp servoTEC S2 ergänzt. Änderungen bei Befehlen: G26; G29; G144; G145; G231; G100.B; G101
...Programmierhandbuch_R5h.doc	Juni 2010	System-Merker, N/R-Register ergänzt G421, G422, G423: Text geändert
...Programmierhandbuch_R5i.doc	März 2011	Änderung: G123-Befehl (nicht Indirket) RUN / SLEEP / CANCEL mit String Neue Befehle: G184, AVERAGE G101-Befehl für servoTEC S2
...Programmierhandbuch_R5j.doc	Dezember 2011	Änderung: G184-Befehl, AVERAGE entfernt

Warenzeichen und Warennamen sind ohne Gewährleistung der freien Verwendbarkeit benutzt. Bei der Erstellung der Texte und Beispiele wurde mit großer Sorgfalt vorgegangen. Trotzdem können Fehler nicht ausgeschlossen werden. Die IEF Werner GmbH kann für fehlende oder fehlerhafte Angaben und deren Folgen weder eine juristische Verantwortung noch irgendeine Haftung übernehmen.

Die IEF Werner GmbH behält sich das Recht vor, ohne Ankündigung die Software oder Hardware oder Teile davon, sowie die mitgelieferten Druckschriften oder Teile davon zu verändern oder zu verbessern.

Alle Rechte der Vervielfältigung, der fotomechanischen Wiedergabe, auch auszugsweise sind ausdrücklich der IEF Werner GmbH vorbehalten.

Für Verbesserungsvorschläge und Hinweise auf Fehler sind wir jederzeit dankbar.

© Dezember 2011; IEF Werner GmbH

Inhalt

1	Befehle der PA-CONTROL-Familie	12
1.1	Informationen zu Softwareständen	12
1.2	Elemente der Programmierung - Übersicht	13
1.2.1	Eingänge	13
1.2.2	Ausgänge	13
1.2.3	Merker	13
1.2.4	Zeiten	13
1.2.5	Realzahlregister	13
1.2.6	Ganzzahlregister	13
1.2.7	Zeichenketten	13
1.2.8	Achsen	13
1.2.9	Marke	13
1.2.10	Kommentar	14
1.2.11	Direkte Adressierung	14
1.2.12	Indirekte Adressierung	14
1.2.13	Display	14
1.2.14	Zähler	14
1.2.15	Serielle Schnittstelle	14
1.2.16	AD	14
1.2.17	DA	14
1.2.18	Temperaturregler	14
1.2.19	Systemmerker	15
1.2.20	System N-Register	24
1.2.21	System R-Register	28
1.3	Übersicht Befehlssatz	30
1.3.1	Programmablauf (Automatik)	30
1.3.1.1	Ablauf	30
1.3.1.2	Sprünge und Verzweigungen	30
1.3.1.3	Unterprogramme aufrufen	30
1.3.1.4	Schleifen	31
1.3.1.5	Parallele Abläufe	31
1.3.1.6	Fehlermanagement	31
1.3.2	Achsbefehle	31
1.3.2.1	Allgemeines	31
1.3.2.2	Modi einer Achse	31
1.3.2.3	Fahren	32
1.3.2.4	Position und Parameter	32
1.3.2.5	Endschalter	32
1.3.2.6	Encoder und Schleppfehler	32
1.3.2.7	Parameter-Befehle	32

1.3.2.8	Interpolation	32
1.3.2.9	Spezielle PA-CONTROL-Befehle	32
1.3.3	Eingänge, Ausgänge, Merker und Register	33
1.3.3.1	Eingänge	33
1.3.3.2	Ausgänge	33
1.3.3.3	Merker	33
1.3.3.4	Register	33
1.3.4	Verweilzeit, Überwachungszeit, Datum, Echtzeituhr	34
1.3.4.1	Warten	34
1.3.4.2	Überwachungszeiten	34
1.3.4.3	Datum	34
1.3.4.4	Uhrzeit	34
1.3.5	Mathematik	34
1.3.5.1	Adressierung	34
1.3.5.2	Mathematische Befehle	34
1.3.5.3	Vergleiche	35
1.3.5.4	Logische Operationen	35
1.3.5.5	Register Operationen	35
1.3.6	Kommunikation	37
1.3.6.1	Serielle Schnittstelle, Display und Tastatur	37
1.3.6.2	CANopen-Bus	37
1.3.6.3	MODBUS	38
1.3.6.4	Stringbefehle	38
1.3.7	Analog / Digital	38
1.3.8	Datei-Befehle	38
1.3.9	Zähler	39
1.3.10	Temperaturregelung	39
1.3.11	Befehlsgruppen der PA-CONTROL	39
1.3.11.1	Befehle der G16x-Gruppe	39
1.3.11.2	Befehle der G17x-Gruppe	39
1.3.11.3	Befehle der G2xx-Gruppe	39
1.3.11.4	Befehle der G4xx-Gruppe	39
1.3.11.5	Befehle der G5xx-Gruppe	40
1.3.11.6	Befehle der G6xx-Gruppe	40
1.3.11.7	Befehle der G7xx-Gruppe	40
1.4	Programmierhinweise	42
1.4.1	Der Ablaufinterpreter	42
1.4.2	Systemfunktionen	43
1.4.3	Steuerung des Automatikbetriebes über externe Eingänge	45
1.4.3.1	Start von Automatik mit Abbruch	45
1.4.3.2	Start von Automatik mit STOP und START NACH STOP	46
1.4.3.3	PA-CONTROL ist im Automatikbetrieb und es tritt eine Störung auf	46

1.4.3.4	PA-CONTROL ist in Grundstellung und es steht eine Störung an	47
1.4.4	Programmstruktur	48
1.5	Erläuterungen zur Syntax	49
1.6	Programmorganisation	50
1.6.1	Ablauf	51
1.6.1.1	BREAK - Abbruch Automatikablauf	51
1.6.1.2	START - Starte Automatikablauf	52
1.6.1.3	STOP - Stoppe Automatikablauf	54
1.6.1.4	Ni:=LINE - Abfrage der aktuellen Zeilennummer eines Programms	55
1.6.1.5	END - Beenden	55
1.6.2	Sprünge und Verzweigungen	56
1.6.2.1	JMP - Unbedingter Sprung	56
1.6.2.2	G21 - Bedingter Sprung	57
1.6.2.3	JMP-LINE.Ni - Start eines Programms bei einer bestimmten Programmzeile	59
1.6.2.4	CASE.JMP - Sprungverteiler	60
1.6.3	Aufruf Unterprogramme	62
1.6.3.1	Prinzipien des Unterprogrammaufrufs	62
1.6.3.2	SUB.Name - Standard Unterprogrammaufruf	63
1.6.3.3	SUB.S0 / SUB.Sn - Unterprogrammaufruf, Programmname steht in einer Zeichenkette	65
1.6.3.4	SUB.Nn - Unterprogrammaufruf, Programmname steht in einem Ganzzahlregister	66
1.6.3.5	G22 - Bedingter Unterprogrammaufruf	67
1.6.3.6	CASE.SUB - Unterprogrammverteiler	68
1.6.4	Schleifen	70
1.6.4.1	INC - Schleife mit bedingtem Sprung	70
1.6.4.2	DEC - Schleife mit bedingtem Sprung	71
1.6.5	Parallele Abläufe	73
1.6.5.1	RUN - Starten eines Parallelablaufes	73
1.6.5.2	SLEEP - Anhalten eines Parallelablaufes	75
1.6.5.3	CANCEL - Beenden eines Parallelablaufes	76
1.6.5.4	CASE.RUN - Starten von Parallelabläufen mit CASE	77
1.6.5.5	CASE.SLEEP - Anhalten von Parallelabläufen mit Case	79
1.6.5.6	CASE.CANCEL - Beenden von Parallelabläufen mit Case	81
1.6.5.7	PROGSTAT - Hole Programmstatus	83
1.6.6	Fehlermanagement	84
1.6.6.1	ERROROFF / ERRORON	84
1.7	Achsbefehle	86
1.7.1	Allgemeines	86
1.7.1.1	Die Zustände einer Achse	86
1.7.2	Modi	88
1.7.2.1	G140 - wechsele in den Zustand DISABLED	88

1.7.2.2	G141 - wechsele in den Zustand OPERATIONAL	89
1.7.2.3	OFF.An - Achse abschalten	90
1.7.2.4	ON.An - Achse einschalten	91
1.7.2.5	STOP.An - Unterbreche die Verfahrbereitschaft einer Achse	92
1.7.2.6	START.An - Herstellen der Verfahrbereitschaft einer Achse	92
1.7.2.7	ABORT.An - Abbruch eines gestoppten Fahrbefehls	93
1.7.3	Fahren	94
1.7.3.1	G25 - Referenzfahrt	94
1.7.3.2	A1 - Positionieren der Achsen	95
1.7.3.3	G90 - Absolutmaßsystem	96
1.7.3.4	G91 - Kettenmaßsystem	97
1.7.3.5	G100 - Beschleunigung	98
1.7.3.6	FAn - Verfahrgeschwindigkeit	99
1.7.3.7	G123	101
1.7.3.8	G123Q - Fahre solange Bedingung erfüllt	103
1.7.3.9	G 150 - Fahre Teilstrecke mit Start-Stop	104
1.7.3.10	Rn:=Fan - Letzte Verfahrgeschwindigkeit einer Achse	105
1.7.4	Position und Parameter	106
1.7.4.1	G26 - Position auf Null setzen / auf Position setzen	106
1.7.4.2	G29 - Position auf Maß setzen	108
1.7.4.3	Rn:=An / Nn:=An - Achsposition holen	110
1.7.5	Endschalter	111
1.7.5.1	G142 Endschalterüberwachung „AUS“	111
1.7.5.2	G143 - Endschalterüberwachung „EIN“	112
1.7.6	Encoder und Schleppfehler	113
1.7.6.1	Ni:=PEAn / Ri:=PEAn - Schleppfehler holen	113
1.7.6.2	ENC - Encoderposition übernehmen	114
1.7.6.3	SSIn.i.Rm / SSIn.i.Nm - Warte auf Position vom SSI-Interface	116
1.7.7	Parameter-Befehle	117
1.7.7.1	PARAMETER.XX.Ai - Parameter schreiben	117
1.7.7.2	Ri:=PARAMETER.XX.Ai - Parameter lesen	119
1.7.7.3	RAn.m:=Ri - Achsparameter laden aus einem Register	120
1.7.7.4	Ri:=RAn.m - Achsparameter in ein Register schreiben	121
1.7.8	Interpolation	122
1.7.8.1	G144 - Schalte die servoTEC Achse in den Interpolationsmode	122
1.7.8.2	G145 - Deaktiviere den Interpolationsmode der servoTEC Achse	123
1.7.8.3	G100.B.n - IPO Bahnbeschleunigung	124
1.7.8.4	FB:=n - IPO Bahngeschwindigkeit	125
1.7.8.5	IPO Bahngeschwindigkeit definieren, eine Achse läuft frei mit	125
1.7.8.6	G01 - Linearinterpolation mit 4 aus 16 Achsen	126
1.7.8.7	Linearinterpolation mit mitlaufender /mitlaufenden Achse(n)	128
1.7.8.8	IPOEND - Linearinterpolation abbrechen	130

1.7.8.9	Rn:=FB - Aktuelle Bahngeschwindigkeit holen	131
1.7.8.10	Rn:=FBAn - Aktuelle Verahrgeschwindigkeit während einer Interpolation	133
1.7.8.11	G16x - Bedienung von Ausgängen bei der Interpolation	134
1.7.8.12	G160 - Bediene Ausgänge vor dem Start der Interpolation	134
1.7.8.13	G161 / G162 - Bediene Ausgänge während der Interpolation	135
1.7.8.14	JMP-LINE-IPO.Ni - Start einer Interpolation bei einer bestimmten Programmzeile	136
1.7.9	Motorstrom Verändern oder Begrenzen G101	137
1.7.9.1	G101 - Motorstrom verändern bei einer PA-CONTROL-MP	138
1.7.9.2	G101 - Spitzenstrom verändern bei einer servoTEC-Achse	139
1.7.9.3	G101 - Strombegrenzung verändern bei einer servoTEC S2	140
1.8	Eingänge, Ausgänge und Merker	141
1.8.1	Eingänge	141
1.8.1.1	In.m - Warten auf log. Zustand des Eingangs	141
1.8.2	Ausgänge	142
1.8.2.1	On.m - Warten auf log. Zustand des Ausgangs	142
1.8.2.2	On:=m - Ausgang setzen / rücksetzen	143
1.8.3	Merker	144
1.8.3.1	Mn:=m - Merker setzen / rücksetzen	144
1.8.3.2	M!2:=0 - Direkte oder Indirekte Adressierung von Merkern	144
1.8.3.3	Mn.m - Warten auf log. Zustand des Merkers	146
1.8.3.4	> = < Vergleichsergebnisse Merker	147
1.8.3.5	Schreibe Merker Nummer nach Vergleich in ein N-Register	148
1.9	Zeiten, Datum und Uhrzeit	149
1.9.1	Warten	149
1.9.1.1	Tn - Verweilzeit	149
1.9.2	Datum und Uhrzeit	150
1.9.2.1	DATE:= - Setzen des Datums aus dem Anwenderprogramm	150
1.9.2.2	TIME:= - Setzen der Zeit aus dem Anwenderprogramm	151
1.9.2.3	S0:=TIME - Auslesen der Echtzeit-Uhr	152
1.10	Mathematik	153
1.10.1	Organisation der Befehle	153
1.10.1.1	Direkte und indirekte Adressierung	153
1.10.1.2	Zuweisungen	153
1.10.2	Mathematische Befehle	154
1.10.2.1	Rn/Nn:= Register laden	154
1.10.2.2	Lade Register mit einem binärem Wert	155
1.10.2.3	Lade Register mit einem hexadezimalen Wert	155
1.10.2.4	Rn:=Rm+K - Addition	156
1.10.2.5	Rn:=Rm-K - Subtraktion	157
1.10.2.6	Rn:=Rm*K - Multiplikation	158
1.10.2.7	Rn:=Rm/K - Division	159

1.10.2.8	Winkelfunktionen	160
1.10.2.9	SIN/ASIN - Sinusfunktionen	160
1.10.2.10	COS/ACOS - Cosinusfunktionen	161
1.10.2.11	TAN/ATAN - Tangensfunktionen	162
1.10.2.12	SQRT - Wurzelfunktion	163
1.10.2.13	INT - Ganzzahliger Anteil	163
1.10.2.14	FRAC - Nachkomma-Anteil einer Realzahl	164
1.10.2.15	ABS - Betrag einer Real- / Ganzzahl	164
1.10.3	Vergleiche	165
1.10.3.1	Vergleiche: Befehlsformen	165
1.10.3.2	Vergleiche: komplexe Beispiele	166
1.10.4	Logische Verknüpfungen mit Eingängen, Ausgängen und Merkern	167
1.10.4.1	Einführung	167
1.10.4.2	LD/AND/OUT/NOT - Logische UND-Verknüpfung	171
1.10.4.3	LD/OR/OUT/NOT - Logische ODER-Verknüpfung	172
1.10.4.4	SET/RES - Ergänzungsbefehle zu logischen Verknüpfungen	173
1.10.4.5	XOR-Verknüpfung (Exklusiv-ODER)	176
1.10.4.6	Mehrstufige logische UND-Verknüpfung	177
1.10.4.7	OR-LD - Mehrstufige logische ODER-Verknüpfung	178
1.10.4.8	Komplexe logische Verknüpfung	179
1.10.4.9	Bitweise Verarbeitung von N-Registern	181
1.10.5	Register-Operationen	182
1.10.5.1	Ni.n - Warten auf den Status eines N-Registers	182
1.10.5.2	Nn:=SNn - Lade Inhalt eines System-N-Registers in ein Ganzzahlregister	183
1.10.5.3	Rn:=SRn - Lade Inhalt eines System-R-Registers in ein Realzahlregister	184
1.10.5.4	Bitweise Verarbeitung von N-Registern	185
1.11	Kommunikation	186
1.11.1	Kommunikation über MODBUS	186
1.11.2	Lesen Werte von einem MODBUS-Teilnehmer	186
1.11.3	Schreiben eines Wertes in einen MODBUS-Teilnehmer	187
1.11.4	Kommunikation über Display PA-CONTROL	188
1.11.4.1	G11 - Anzeige ein- / ausschalten	188
1.11.4.2	Nn:=CHN - Prüfe ob Zeichenübernahme im Hintergrund abgeschlossen	189
1.11.5	String-Befehle	190
1.11.5.1	Sn - Kopieren von Zeichenketten	190
1.11.5.2	S0:=CHN - Kopieren von Zeichenketten	191
1.11.5.3	POS - Suche Position eines Zeichens im lok. Zeichenpuffer S0	192
1.11.5.4	COPY - Wandle Zeichen aus dem lokalen Zeichenpuffer	193
1.11.5.5	Sn:=COPY - Hole Teilstring	195
1.11.5.6	Si:=Sn+Sm - String zusammenfügen	196
1.11.5.7	LENGTH - Hole die Länge eines Strings	196
1.11.5.8	Erzeuge XOR-Quersumme von einem String	197

1.11.5.9 Sn:=Ni / Sn:=Ri - Schreibe den Inhalt eines Registers (Zahl) in einen String	198
1.11.5.10GET - Übertrage den Inhalt des lokalen Zeichenpuffers S0 in ein Ganzzahlregister	199
1.11.5.11GETI - Übertrage Inhalt des lokalen Zeichenpuffer S0 im INTEL-Format in ein Ganzzahlregister	200
1.11.5.12PUT - Übertrage Inhalt eines Registers in den lokalen Zeichenpuffer	201
1.11.5.13PUTI - Übertrage Inhalt eines Registers im INTEL-Format in den lokalen Zeichenpuffer S0	202
1.12 Analog / Digital	203
1.12.1 Analog-Digital-Wandler - Allgemeine Beschreibung	203
1.12.1.1 AD-Werte holen	203
1.12.2 Digital-Analog-Wandler - Allgemeine Beschreibung	205
1.12.2.1 DAi - DA-Werte ausgeben	206
1.13 Datei-Befehle	207
1.13.1 Befehle der G17x-Gruppe	207
1.13.1.1 G170 - Zeichen aus PTX-File lesen	207
1.13.1.2 G171 - Zeichen in PTX-File schreiben	209
1.13.1.3 G172 - Zeile aus PTX-File in String (Sn) schreiben	210
1.13.1.4 G173 - String (Sn) in einer Zeile eines PTX-Files speichern	211
1.13.1.5 STORE - Speichern aktueller Werte in einen Programm-File	213
1.13.1.6 STOREEND / STORESTART - Zuweisung mit Konstanten deaktivieren / aktivieren	215
1.13.1.7 CASE.STORE.N - Speichern wenn Inhalt von N-Register	216
1.14 Zähler	218
1.14.1 Allgemeine Grundlagen	218
1.14.2 Hardware	218
1.14.3 Befehle	219
1.14.3.1 Zähler initialisieren	219
1.14.3.2 Zähler setzen	220
1.14.3.3 Zähler auslesen	220
1.14.3.4 Zähler vergleichen	221
1.15 Temperaturregelung	222
1.15.1 Allgemeine Grundlagen	222
1.15.1.1 Parameter	222
1.15.2 Befehle für die Temperaturregelung	223
1.15.2.1 Parameterwert schreiben	223
1.15.2.2 Parameterwert lesen	224
1.16 Befehlsgruppen der PA-CONTROL	225
1.16.1 Befehle der G18x-Gruppe	225
1.16.1.1 G18x. - Erfassen von mehreren Werten	225
1.16.1.2 G180. - AD-Werte synchron zur Achsbewegung erfassen	226
1.16.1.3 G181 - AD-Werte im definierten Zeitraster erfassen	228

1.16.1.4	G182 - AD-Werte synchron zum Counterwert (CNT0) holen	229
1.16.1.5	G183 - AD-Werte synchron zum Counterwert (CNT0) holen	232
1.16.2	Befehle der G2xx-Gruppe	235
1.16.2.1	Grundsätzlich Ausführungen zu den Befehlen der G2xx-Gruppe	235
1.16.2.2	G210 - Aktiviere den Startpositioniermodus	236
1.16.2.3	G211 - Positionsbedingter Sprung	237
1.16.2.4	G212 - Positionsbedingter Unterprogrammaufruf	239
1.16.2.5	G213 - Aktiviere den Normalpositioniermodus	241
1.16.2.6	G221 - Positionsbedingter Sprung (aktuelle Position)	242
1.16.2.7	G222 - Positionsbedingter Unterprogramm-Aufruf (akt. Pos.)	244
1.16.2.8	G230 - Warten bis aktuelle Position \neq als Wert	246
1.16.2.9	G231 - Warte bis der Restweg der akt. Positionierung erreicht oder unterschritten ist	248
1.16.3	Befehle der G4xx-Gruppe	250
1.16.3.1	Einleitung, Zeitüberwachungsbefehle der PA-CONTROL-Familie	250
1.16.3.2	G421 - Zeitüberwachung mit bedingtem Sprung	251
1.16.3.3	G422 - Zeitüberwachung mit bedingtem Unterprogrammaufruf	252
1.16.3.4	G423 - mit bedingtem Unterpr.aufruf (zurück in gleicher Zeile)	253
1.16.3.5	G401 - Zurücksetzen der Zeitbedingung	254
1.16.4	Befehle der G5xx-Gruppe	255
1.16.4.1	Einleitung, Datenkanäle der PA-CONTROL	255
1.16.4.2	G500 - Wahl des Datenkanals / Initialisierung der Schnittstellen	256
1.16.4.3	G501 - Lösche die Anzeige	258
1.16.4.4	G502 - Lösche bis zum Zeilenende	259
1.16.4.5	G503/G504 - Positioniere den Cursor	260
1.16.4.6	G510 - Textausgabe	262
1.16.4.7	G511 - Textausgabe mit CR und LF	263
1.16.4.8	G512 - Steuerzeichenausgabe	264
1.16.4.9	G515 - Textausgabe aus einem PTX-File	265
1.16.4.10	G520 - Wertausgabe eines Operanden	266
1.16.4.11	G521 - Wertausgabe eines Operanden mit CR und LF	268
1.16.4.12	G531 - Wertübernahme	270
1.16.4.13	G532 - Zeichenübernahme in den lokalen Zeichenpuffer S0	272
1.16.4.14	G533 - Zeichenübernahme im Hintergrund bis zum Endekriterium	273
1.16.4.15	G534 - Zeichenübernahme einer bestimmten Menge von Zeichen im Hintergrund	275
1.16.4.16	G540 - Zeichenübernahme von der Tastatur, Prüfe ob eine Taste betätigt	277
1.16.4.17	G541 - Hole ein Zeichen von der Tastatur	278
1.16.4.18	G542 - Eingabe eines Wertes über Tastatur	279
1.16.5	Befehle der G6xx-Gruppe	281
1.16.5.1	Möglichkeiten und Befehlsaufbau der G6xx-Befehlsgruppe	281
1.16.5.2	G600 - Binärdarstellung auf Ausgänge	282

1.16.5.3	G601 - BCD-Darstellung auf Ausgänge	283
1.16.5.4	G602 - Binärdarstellung auf Merker	284
1.16.5.5	G603 - Eingänge im Binärformat in Register	286
1.16.5.6	G604 - Merker im Binärformat in Register	287
1.16.6	Befehle der G7xx-Gruppe	288
1.16.6.1	Allgemeines	288
1.16.6.2	G 70x - Netzwerk-Management Befehle	289
1.16.6.3	G701 - Start Remote Node	289
1.16.6.4	G702 - Stop Remote Node	290
1.16.6.5	G703 - Reset Remote Node	290
1.16.6.6	G704 - Enter Pre-Operational-State	291
1.16.6.7	G705 - Reset-Communication	291
1.16.6.8	G711 - Funktionsüberwachung der Teilnehmer	292
1.16.6.9	G72x - Befehle für die Kontrolle von Gerätefehlern, Allgemeines	293
1.16.6.10	G721 / G722 - Prüfe ob eine Gerätefehlernachricht vorhanden ist	293
1.16.6.11	G73x - Befehle für die Bearbeitung von Servicedatenobjekten, Allgemeines	294
1.16.6.12	G730 - Liest den Inhalt eines Servicedatenobjektes (Initiate Domain Upload Rq)	294
1.16.6.13	G731 - Schreibe den Inhalt eines Servicedatenobjektes (Initiate Domain Upload Rq)	295
1.16.6.14	G74x, G75x - Befehle für die Bearbeitung von Prozessdatenobjekten	296
1.16.6.15	G74x - Prozessdaten lesen	298
1.16.6.16	G75x - Prozessdaten senden	299
1.16.6.17	Applikationsbeispiele G7xx- Befehlsgruppe	300
2	PA-CONTROL-Tastencode	302
	INDEX	306

1 Befehle der PA-CONTROL-Familie

1.1 Informationen zu Softwareständen

G25

In der Version 4.xx konnte mit dem Befehl „G25.A0“ die Achse zum Fahren freigeschaltet werden, ohne die Achse zu referenzieren.

Ab der Version 5.00 ist dieser Befehl nicht mehr möglich. Statt dessen sollte der Befehl G26 verwendet werden.

G140 / G141

in der Version 4.xx wurde mit den Befehlen G140 / G141 der MESSMODE aktiviert oder deaktiviert.

ab der Version 5.00 hat eine Achse mehrere Statuszustände der Status MESSMODE wurde durch den Status DISABLED abgelöst. Die Funktion und die Verwendbarkeit der Befehle G140 / G142 hat sich dadurch leicht geändert.

1.2 Elemente der Programmierung - Übersicht

In der PA-CONTROL werden verschiedene Elemente zur Programmierung verwendet.

1.2.1 Eingänge

Ein Eingang kann auf seinen logischen Zustand abgefragt werden (siehe Abschnitt 1.8.1.1, Seite 141). Es sind 2048 Eingänge implementiert.

1.2.2 Ausgänge

Einem Ausgang kann der logische Zustand 0 oder 1 zugewiesen werden (siehe Abschnitt 1.8.2.2, Seite 143). Bei bedingten Sprüngen kann sein logischer Zustand abgefragt werden (Abschnitt 1.8.2.1, Seite 142). Es sind 2048 Ausgänge implementiert.

1.2.3 Merker

Einem Merker kann der logische Zustand 0 oder 1 zugewiesen sein (siehe Abschnitt 1.8.3.1, Seite 144) und sein logischer Zustand kann abgefragt werden. In der PA-CONTROL sind 8192 Merker implementiert.

1.2.4 Zeiten

Zeiten werden zur Erzeugung definierter Wartezeiten benutzt (siehe Abschnitt *Tn - Verweilzeit* Seite 149).

1.2.5 Realzahlregister

Realzahlregister (1-8192) sind spezielle Speicherplätze für reelle Zahlen ($\pm 8.000.000,000$). Die Registerinhalte können für Positionierungen, sowie Rechen- und Vergleichsoperationen benutzt werden. Es sind bis zu 3 Nachkommastellen erlaubt (siehe Abschnitt 1.10, Seite 153).

1.2.6 Ganzzahlregister

Ganzzahlregister (1-8192) sind spezielle Speicherplätze für ganze Zahlen ($\pm 2^{31}$, d.h. $\pm 2.147.483.648$). Die Registerinhalte können für Zeiten, sowie Zähl-, Rechen- und Vergleichsoperationen benutzt werden. Es sind keine Nachkommastellen erlaubt (siehe Abschnitt 1.10.2.1, Seite 154).

1.2.7 Zeichenketten

Zeichenketten (Strings) dienen zur Speicherung von bis 80 ASCII-Zeichen, die z.B. über serielle Schnittstelle empfangen wurden.

Es gibt S1 bis S16 als globale Strings auf die jeder Ablauf zugreifen kann. Jeder Ablauf hat zusätzlich noch seinen eigenen lokalen S0. Die Inhalte der globalen Strings bleiben in der Steuerung bis zum nächsten Überschreiben erhalten, die lokalen Strings werden bei jedem Automatikstart zurückgesetzt.

1.2.8 Achsen

Achsen sind Schrittmotor- oder Servomotorachsen und dienen zur Positionierung. Sie werden mit den Namen A1, A2, A3, A4, A5, A6, A7, A8 angesprochen (nur soweit vorhanden) (siehe Abschnitt 1.7.3.2, Seite 95). Über die PA-CONTROL können maximal 16 Achsen angesteuert werden.

1.2.9 Marke

Marken sind Orientierungspunkte und dienen als Ziele bei bedingten und unbedingten Sprüngen. Beispiel für eine Marke: \$ROTB LAU

1.2.10 Kommentar

Ein Kommentar wird durch ein Semikolon (;) eingeleitet und dient der Programmdokumentation (siehe Beispiele bei den Befehlserklärungen) und der Bedienerführung im Automatikablauf.

1.2.11 Direkte Adressierung

Bei der direkten Adressierung wird Registern direkt ein neuer Wert übergeben bzw. Merkern ein neuer Zustand zugeordnet.

1.2.12 Indirekte Adressierung

Die indirekte Adressierung wird durch ein "!" nach dem Bezeichner gekennzeichnet (R!5, M!6). Bei der indirekten Adressierung von Registern wird auf ein weiteres Register gezeigt, welches durch den Inhalt des aufgerufenen Registers angesprochen wird und der zugeordnete Wert wird an dieses übergeben. Bei der indirekten Merkeradressierung zeigt der Merker auf den Inhalt des Ganzzahlregisters mit gleichem Index und setzt den Inhalt des Ganzzahlregisters als Index des anzusprechenden Merkers.

1.2.13 Display

Grundsätzlich stehen für die PA-CONTROL drei Möglichkeiten der Anzeige zur Verfügung:

- 2-zeilige Anzeige bei den Geräten mit eigener Tastatur
- die 2-zeilige Anzeige innerhalb der IEF-Bedienkonsole sowie
- die Darstellung auf einer SÜTRON-Bedienkonsole über CANopen-Bus.

Für die Anzeige und Steuerung der beiden ersten Möglichkeiten kommen in erster Linie die Befehle der G5xx-Befehlsgruppe (siehe Abschnitt 1.16.4, ab Seite 255) zur Anwendung. Die Anzeige und Steuerung der Konsolen über CANopen-Bus erfolgt über Merker und Register der PA-CONTROL.

1.2.14 Zähler

Zur Anwendung kommen Zähler, die über den CANopen-Bus angeschlossen werden (siehe Abschnitt 1.14 ab Seite 218).

1.2.15 Serielle Schnittstelle

In der PA-CONTROL sind maximal die 4 Schnittstellen COM 1 bis COM 4 realisierbar. Zur Anwendung kommen die Befehle der G5xx-Befehlsgruppe (siehe Abschnitt 1.16.4, ab Seite 255).

1.2.16 AD

Die Verarbeitung analoger Signale, z.B. von Temperaturen oder Drücken, kann über die optional einsetzbaren IEF-Module aber auch über spezielle CANopen-Bus-Module erfolgen. Für die Verarbeitung der analogen Signale kommen die Befehle im Abschnitt 1.12.1 ab Seite 203 zur Anwendung.

1.2.17 DA

Über CANopen-BUS können bis zu 8 DA-Wandler angeschlossen werden (siehe dazu Abschnitt 1.12.2 ab Seite 205).

1.2.18 Temperaturregler

Die Feldbusklemme „**Ksvario**“ kann über den CAN-Bus angeschlossen und zur Temperaturregelung eingesetzt werden.

1.2.19 Systemmerker

Die systeminternen Merker (logischer Zustand 0 oder 1) können vom Anwender nur abgefragt werden (siehe Liste Systemmerker).

Nr.	Funktion
1	Power On: Der Merker wird von der PA-CONTROL nach dem Einschalten gesetzt und nach der Abfrage (LD SM1, G21 SM1.n, ...)
2	An der PA-CONTROL ist eine Tastatur angeschlossen und funktionsfähig in Betrieb
3	Der Schlüsselschalter auf der Tastatur der PA-CONTROL steht in der Position „Program“ (waagrecht, Schlüssel nicht abziehbar)
4	Nicht belegt
5	Der „G542-Befehl“, Eingabe eines Registerwertes, wurde mit ENTER beendet
6	Der „G171-Befehl“, Zeichen in PTX-File schreiben, wurde nicht korrekt ausgeführt werden
7	Die Batterie ist in Ordnung
8	Achsweg ist größer als Interpolations-Bahn-Weg
9	Interpolation läuft (0 = Interpolation nicht aktiv, 1 = Interpolation läuft)
10	Merker Status immer „0“
11	Achse 1 wurde durch die G123-Funktion gestoppt
12	Achse 2 wurde durch die G123-Funktion gestoppt
13	Achse 3 wurde durch die G123-Funktion gestoppt
14	Achse 4 wurde durch die G123-Funktion gestoppt
15	Achse 5 wurde durch die G123-Funktion gestoppt
16	Achse 6 wurde durch die G123-Funktion gestoppt
17	Achse 7 wurde durch die G123-Funktion gestoppt
18	Achse 8 wurde durch die G123-Funktion gestoppt
19	Achse 9 wurde durch die G123-Funktion gestoppt
20	Achse 10 wurde durch die G123-Funktion gestoppt
21	Achse 11 wurde durch die G123-Funktion gestoppt
22	Achse 12 wurde durch die G123-Funktion gestoppt
23	Achse 13 wurde durch die G123-Funktion gestoppt
24	Achse 14 wurde durch die G123-Funktion gestoppt
25	Achse 15 wurde durch die G123-Funktion gestoppt
26	Achse 16 wurde durch die G123-Funktion gestoppt
27	CANopen-Bedienkonsole „Sütron“ ID 63 ist aktiv
28	CANopen-Bedienkonsole „Sütron“ ID 62 ist aktiv
29	CANopen-Bedienkonsole „Sütron“ ID 61 ist aktiv
30	CANopen-Bedienkonsole „Sütron“ ID 60 ist aktiv

Teil 2 Liste der Systemmerker:

Nr.	Funktion
31	Achse 1 ist DRIVE ENABLED (Bereit) / Bereitschaftseingang ist bestromt
32	Achse 2 ist DRIVE ENABLED (Bereit) / Bereitschaftseingang ist bestromt
33	Achse 3 ist DRIVE ENABLED (Bereit) / Bereitschaftseingang ist bestromt
34	Achse 4 ist DRIVE ENABLED (Bereit) / Bereitschaftseingang ist bestromt
35	Achse 5 ist DRIVE ENABLED (Bereit) / Bereitschaftseingang ist bestromt
36	Achse 6 ist DRIVE ENABLED (Bereit) / Bereitschaftseingang ist bestromt
37	Achse 7 ist DRIVE ENABLED (Bereit) / Bereitschaftseingang ist bestromt
38	Achse 8 ist DRIVE ENABLED (Bereit) / Bereitschaftseingang ist bestromt
39	Achse 9 ist DRIVE ENABLED (Bereit) / Bereitschaftseingang ist bestromt
40	Achse 10 ist DRIVE ENABLED (Bereit) / Bereitschaftseingang ist bestromt
41	Achse 11 ist DRIVE ENABLED (Bereit) / Bereitschaftseingang ist bestromt
42	Achse 12 ist DRIVE ENABLED (Bereit) / Bereitschaftseingang ist bestromt
43	Achse 13 ist DRIVE ENABLED (Bereit) / Bereitschaftseingang ist bestromt
44	Achse 14 ist DRIVE ENABLED (Bereit) / Bereitschaftseingang ist bestromt
45	Achse 15 ist DRIVE ENABLED (Bereit) / Bereitschaftseingang ist bestromt
46	Achse 16 ist DRIVE ENABLED (Bereit) / Bereitschaftseingang ist bestromt
47	Blinkmerker-0, 160ms-Takt (Puls plus Pause)
48	Blinkmerker-1, 320ms-Takt (Puls plus Pause)
49	Blinkmerker-2, 640ms-Takt (Puls plus Pause)
50	Blinkmerker-3, 1280ms-Takt (Puls plus Pause)
51	Achse 1 ist referenziert
52	Achse 2 ist referenziert
53	Achse 3 ist referenziert
54	Achse 4 ist referenziert
55	Achse 5 ist referenziert
56	Achse 6 ist referenziert
57	Achse 7 ist referenziert
58	Achse 8 ist referenziert
59	Achse 9 ist referenziert
60	Achse 10 ist referenziert
61	Achse 11 ist referenziert
62	Achse 12 ist referenziert
63	Achse 13 ist referenziert
64	Achse 14 ist referenziert
65	Achse 15 ist referenziert

Teil 3 Liste der Systemmerker:

Nr.	Funktion
66	Achse 16 ist referenziert
67	Grundstellungsprogramm hat einen END-Befehl bearbeitet
68	Es kann mit RUN noch eine weitere TASK gestartet werden
69	Reserve
70	Reserve
71	Achse 1 zum Fahren freigeschaltet (G25.A0)
72	Achse 2 zum Fahren freigeschaltet (G25.A0)
73	Achse 3 zum Fahren freigeschaltet (G25.A0)
74	Achse 4 zum Fahren freigeschaltet (G25.A0)
75	Achse 5 zum Fahren freigeschaltet (G25.A0)
76	Achse 6 zum Fahren freigeschaltet (G25.A0)
77	Achse 7 zum Fahren freigeschaltet (G25.A0)
78	Achse 8 zum Fahren freigeschaltet (G25.A0)
79	Achse 9 zum Fahren freigeschaltet (G25.A0)
80	Achse 10 zum Fahren freigeschaltet (G25.A0)
81	Achse 11 zum Fahren freigeschaltet (G25.A0)
82	Achse 12 zum Fahren freigeschaltet (G25.A0)
83	Achse 13 zum Fahren freigeschaltet (G25.A0)
84	Achse 14 zum Fahren freigeschaltet (G25.A0)
85	Achse 15 zum Fahren freigeschaltet (G25.A0)
86	Achse 16 zum Fahren freigeschaltet (G25.A0)
87	Reserve
88	Reserve
89	Reserve
90	Reserve
91	Achse 1 ist im Status ACTIVE
92	Achse 2 ist im Status ACTIVE
93	Achse 3 ist im Status ACTIVE
94	Achse 4 ist im Status ACTIVE
95	Achse 5 ist im Status ACTIVE
96	Achse 6 ist im Status ACTIVE
97	Achse 7 ist im Status ACTIVE
98	Achse 8 ist im Status ACTIVE
99	Achse 9 ist im Status ACTIVE
100	Achse 10 ist im Status ACTIVE

Teil 4 Liste der Systemmerker:

Nr.	Funktion
101	Achse 11 ist im Status ACTIVE
102	Achse 12 ist im Status ACTIVE
103	Achse 13 ist im Status ACTIVE
104	Achse 14 ist im Status ACTIVE
105	Achse 15 ist im Status ACTIVE
106	Achse 16 ist im Status ACTIVE
107	Reserve
108	Reserve
109	Reserve
110	Reserve
111	Achse 1 ist im Status HALT
112	Achse 2 ist im Status HALT
113	Achse 3 ist im Status HALT
114	Achse 4 ist im Status HALT
115	Achse 5 ist im Status HALT
116	Achse 6 ist im Status HALT
117	Achse 7 ist im Status HALT
118	Achse 8 ist im Status HALT
119	Achse 9 ist im Status HALT
120	Achse 10 ist im Status HALT
121	Achse 11 ist im Status HALT
122	Achse 12 ist im Status HALT
123	Achse 13 ist im Status HALT
124	Achse 14 ist im Status HALT
125	Achse 15 ist im Status HALT
126	Achse 16 ist im Status HALT
127	Reserve
128	Reserve
129	Reserve
130	Reserve
131	Achse 1 ist im Status IDLE
132	Achse 2 ist im Status IDLE
133	Achse 3 ist im Status IDLE
134	Achse 4 ist im Status IDLE
135	Achse 5 ist im Status IDLE

Teil 5 Liste der Systemmerker:

Nr.	Funktion
136	Achse 6 ist im Status IDLE
137	Achse 7 ist im Status IDLE
138	Achse 8 ist im Status IDLE
139	Achse 9 ist im Status IDLE
140	Achse 10 ist im Status IDLE
141	Achse 11 ist im Status IDLE
142	Achse 12 ist im Status IDLE
143	Achse 13 ist im Status IDLE
144	Achse 14 ist im Status IDLE
145	Achse 15 ist im Status IDLE
146	Achse 16 ist im Status IDLE
147	Reserve
148	Reserve
149	Reserve
150	Reserve
151	Achse 1 ist im Status SAFE
152	Achse 2 ist im Status SAFE
153	Achse 3 ist im Status SAFE
154	Achse 4 ist im Status SAFE
155	Achse 5 ist im Status SAFE
156	Achse 6 ist im Status SAFE
157	Achse 7 ist im Status SAFE
158	Achse 8 ist im Status SAFE
159	Achse 9 ist im Status SAFE
160	Achse 10 ist im Status SAFE
161	Achse 11 ist im Status SAFE
162	Achse 12 ist im Status SAFE
163	Achse 13 ist im Status SAFE
164	Achse 14 ist im Status SAFE
165	Achse 15 ist im Status SAFE
166	Achse 16 ist im Status SAFE
167	Reserve
168	Reserve
169	Reserve
170	Reserve

Teil 6 Liste der Systemmerker:

Nr.	Funktion	
171	Achse 1 ist im Status IDLE oder SAFE	Wird gesetzt, wenn die Achse durch ein OFF-Kommando den Status "DISABLED" oder "SAFE" erreicht hat
172	Achse 2 ist im Status IDLE oder SAFE	
173	Achse 3 ist im Status IDLE oder SAFE	
174	Achse 4 ist im Status IDLE oder SAFE	
175	Achse 5 ist im Status IDLE oder SAFE	
176	Achse 6 ist im Status IDLE oder SAFE	
177	Achse 7 ist im Status IDLE oder SAFE	
178	Achse 8 ist im Status IDLE oder SAFE	
179	Achse 9 ist im Status IDLE oder SAFE	
180	Achse 10 ist im Status IDLE oder SAFE	
181	Achse 11 ist im Status IDLE oder SAFE	
182	Achse 12 ist im Status IDLE oder SAFE	
183	Achse 13 ist im Status IDLE oder SAFE	
184	Achse 14 ist im Status IDLE oder SAFE	
185	Achse 15 ist im Status IDLE oder SAFE	
186	Achse 16 ist im Status IDLE oder SAFE	
187	Reserve	
188	Reserve	
189	Reserve	
190	Reserve	
191	Achse 1	<p>Diese Merker sind gesetzt, wenn eine Achse im Status IDLE / SAFE um mehr als den zulässigen Parameterwert verschoben und die Achse in den Status OPERATIONAL oder HALT geschaltet wird, ansonsten wird der Merker bei der Funktion ON-Achse zurückgesetzt.</p> <p>Der Merker wird gesetzt, bevor die Achse den Status OPERATIONAL oder HALT erreicht hat.</p> <p>Ist dieser Merker gesetzt, werden entsprechend dem "Einschaltverfahrenmode" weitere Aktionen beeinflusst.</p> <p>Der Merker wird durch Abfrage oder beim Verlassen der Verfahrbetriebsart zurückgesetzt.</p>
192	Achse 2	
193	Achse 3	
194	Achse 4	
195	Achse 5	
196	Achse 6	
197	Achse 7	
198	Achse 8	
199	Achse 9	
200	Achse 10	
201	Achse 11	
202	Achse 12	
203	Achse 13	
204	Achse 14	
205	Achse 15	
206	Achse 16	

Teil 7 Liste der Systemmerker:

Nr.	Funktion	
207	Mit Hilfe des Programms WINPAC können die 16 möglichen Achsen in Gruppen zusammengefasst werden. Das erfolgt über die Einstellmöglichkeiten der Registerkarte „Achsparemeter“. Es können maximal 4 Gruppen gebildet werden, für die die Systemmerker 207, 208, 209 und 210 zugeordnet sind.	
208	Wird der Systemmerker der Achse gesetzt, dann wird auch der Gruppenmerker gesetzt. Solange einer dieser beiden Merker gesetzt ist, kann die Achse nicht verfahren werden.	
209	Die Merker werden gelöscht durch: - Abfragen im Automatikbetrieb (LD SM191..., G21 SM191..., LD SM207...) - Abbruch AUTOMATIK oder ONLINE	
210	HINWEIS: Ein Gruppenmerker kann nicht direkt gelöscht werden. Der Gruppenmerker wird zurückgesetzt, in dem alle Merker dieser Gruppe gelöscht werden.	
211	Achse 1 ist durch eine Interpolation belegt	0 = Achse wird nicht durch eine Interpolation verfahren 1 = Achse wird durch eine Interpolation verfahren
212	Achse 2 ist durch eine Interpolation belegt	
213	Achse 3 ist durch eine Interpolation belegt	
214	Achse 4 ist durch eine Interpolation belegt	
215	Achse 5 ist durch eine Interpolation belegt	
216	Achse 6 ist durch eine Interpolation belegt	
217	Achse 7 ist durch eine Interpolation belegt	
218	Achse 8 ist durch eine Interpolation belegt	
219	Achse 9 ist durch eine Interpolation belegt	
220	Achse 10 ist durch eine Interpolation belegt	
221	Achse 11 ist durch eine Interpolation belegt	
222	Achse 12 ist durch eine Interpolation belegt	
223	Achse 13 ist durch eine Interpolation belegt	
224	Achse 14 ist durch eine Interpolation belegt	
225	Achse 15 ist durch eine Interpolation belegt	
226	Achse 16 ist durch eine Interpolation belegt	
227	Reserve	
228	Reserve	
229	Reserve	
230	Reserve	
231	Achse 1 ist FAULT	
232	Achse 2 ist FAULT	
233	Achse 3 ist FAULT	
234	Achse 4 ist FAULT	
235	Achse 5 ist FAULT	
236	Achse 6 ist FAULT	

Teil 8 Liste der Systemmerker:

Nr.	Funktion
237	Achse 7 ist FAULT
238	Achse 8 ist FAULT
239	Achse 9 ist FAULT
240	Achse 10 ist FAULT
241	Achse 11 ist FAULT
242	Achse 12 ist FAULT
243	Achse 13 ist FAULT
244	Achse 14 ist FAULT
245	Achse 15 ist FAULT
246	Achse 16 ist FAULT
247	Reserve
248	Reserve
249	Reserve
250	Reserve
251	Achse 1 ist IN POSITION
252	Achse 2 ist IN POSITION
253	Achse 3 ist IN POSITION
254	Achse 4 ist IN POSITION
255	Achse 5 ist IN POSITION
256	Achse 6 ist IN POSITION
257	Achse 7 ist IN POSITION
258	Achse 8 ist IN POSITION
259	Achse 9 ist IN POSITION
260	Achse 10 ist IN POSITION
261	Achse 11 ist IN POSITION
262	Achse 12 ist IN POSITION
263	Achse 13 ist IN POSITION
264	Achse 14 ist IN POSITION
265	Achse 15 ist IN POSITION
266	Achse 16 ist IN POSITION
271	Achse 1 Gantryantrieb ist referenziert
272	Achse 2 Gantryantrieb ist referenziert
273	Achse 3 Gantryantrieb ist referenziert
274	Achse 4 Gantryantrieb ist referenziert
275	Achse 5 Gantryantrieb ist referenziert
276	Achse 6 Gantryantrieb ist referenziert

Nr.	Funktion
277	Achse 7 Gantryantrieb ist referenziert
278	Achse 8 Gantryantrieb ist referenziert
279	Achse 9 Gantryantrieb ist referenziert
281	Achse 10 Gantryantrieb ist referenziert
282	Achse 11 Gantryantrieb ist referenziert
283	Achse 12 Gantryantrieb ist referenziert
284	Achse 13 Gantryantrieb ist referenziert
285	Achse 14 Gantryantrieb ist referenziert
286	Achse 15 Gantryantrieb ist referenziert
285	Achse 16 Gantryantrieb ist referenziert
291	(intern) Achse 1 setze Referenzoffset für Gantry-Slave ist möglich
292	(intern) Achse 1 setze Referenzoffset für Gantry-Slave ist möglich
293	(intern) Achse 1 setze Referenzoffset für Gantry-Slave ist möglich
294	(intern) Achse 1 setze Referenzoffset für Gantry-Slave ist möglich
295	(intern) Achse 1 setze Referenzoffset für Gantry-Slave ist möglich
296	(intern) Achse 1 setze Referenzoffset für Gantry-Slave ist möglich
297	(intern) Achse 1 setze Referenzoffset für Gantry-Slave ist möglich
298	(intern) Achse 1 setze Referenzoffset für Gantry-Slave ist möglich
299	(intern) Achse 1 setze Referenzoffset für Gantry-Slave ist möglich
300	(intern) Achse 1 setze Referenzoffset für Gantry-Slave ist möglich
301	(intern) Achse 1 setze Referenzoffset für Gantry-Slave ist möglich
302	(intern) Achse 1 setze Referenzoffset für Gantry-Slave ist möglich
303	(intern) Achse 1 setze Referenzoffset für Gantry-Slave ist möglich
304	(intern) Achse 1 setze Referenzoffset für Gantry-Slave ist möglich
305	(intern) Achse 1 setze Referenzoffset für Gantry-Slave ist möglich
306	(intern) Achse 1 setze Referenzoffset für Gantry-Slave ist möglich

1.2.20 System N-Register

Die systeminternen Ganzzahlregister können vom Anwender nur abgefragt werden.

Nr.	Funktion
1	Systemlaufzeit der PA-CONTROL in Sekunden (die Laufzeit wird nur durch Laden eines neuen Betriebssystems zurückgesetzt)
2	Systemlaufzeit der PA-CONTROL in Zehntelsekunden (die Laufzeit wird nur durch Laden eines neuen Betriebssystems zurückgesetzt)
3	Laufzeitmessung in Millisekunden (ab V4.74G) Das Register wird bei Abfrage auf Null zurückgesetzt und dann in jeder weiteren Millisekunde bis zur nächsten Abfrage um 1 inkrementiert. Das WINPAC-Diagnosefenster „System-N-Register“ beeinflusst SN3 nicht.
4	Reserve
5	PA-CONTROL-Version
6	PA-CONTROL-Seriennummer
7	Reserve
8	Reserve
9	Reserve
10	Systemfehlernummer
11	Ablauffehlernummer
12	Nummer der Achse oder einer Achse aus einer (bei einem Fehler beteiligten) Achsgruppe
13	Nummer des Parallelablaufes mit dem Fehler
20	PA-CONTROL wurde gestoppt von: <SN20> = 1 → STOP durch Bedienkonsole (IEF oder CANopen-Kons.) <SN20> = 2 → STOP durch Extern-STOP-Eingang <SN20> = 3 → STOP durch Diagnoseschnittstelle (WINPAC) <SN20> = 4 → STOP durch Profibus DP <SN20> = 5 → STOP durch RS232-ONLINE-CMD-Schnittstelle <SN20> = 6 → STOP durch Interbus-S <SN20> = 7 → STOP durch PAB-Programm
21	PA-CONTROL befindet sich im Stop-Mode Wird die PA-CONTROL im Automatikbetrieb gestoppt werden im STOPMODE verschiedene Stufen durchlaufen, die im SN21 abgefragt werden können. <SN21> = 0 → Kein STOP <SN21> = 1 → STOP erkannt <SN21> = 2 → Warte bis alle Achsen stehen <SN21> = 3 → Starte Programm nach STOP <SN21> = 4 → Bearbeite Programm nach STOP <SN21> = 5 → Gestoppt <SN21> = 6 → Gestoppt und warten auf Weiter <SN21> = 7 → Start nach Stop erkannt, starte gestoppte Achsen <SN21> = 8 → Bearbeite Programm START-NACH-STOP <SN21> = 9 → Aktiviere Bearbeite alle Programme

Teil 2 Liste der System-N-Register:

Nr.	Funktion
31	Achse 1 / Zähler Absolut-Positionssystem
32	Achse 2 / Zähler Absolut-Positionssystem
33	Achse 3 / Zähler Absolut-Positionssystem
34	Achse 4 / Zähler Absolut-Positionssystem
35	Achse 5 / Zähler Absolut-Positionssystem
36	Achse 6 / Zähler Absolut-Positionssystem
37	Achse 7 / Zähler Absolut-Positionssystem
38	Achse 8 / Zähler Absolut-Positionssystem
39	Achse 9 / Zähler Absolut-Positionssystem
40	Achse 10 / Zähler Absolut-Positionssystem
41	Achse 11 / Zähler Absolut-Positionssystem
42	Achse 12 / Zähler Absolut-Positionssystem
43	Achse 13 / Zähler Absolut-Positionssystem
44	Achse 14 / Zähler Absolut-Positionssystem
45	Achse 15 / Zähler Absolut-Positionssystem
46	Achse 16 / Zähler Absolut-Positionssystem
47	Reserve
48	Reserve
49	Reserve
50	Reserve
51	Achse 1 / Anzahl der tatsächlich mit dem G180-Befehl geholten AD-Werte
52	Achse 2 / Anzahl der tatsächlich mit dem G180-Befehl geholten AD-Werte
53	Achse 3 / Anzahl der tatsächlich mit dem G180-Befehl geholten AD-Werte
54	Achse 4 / Anzahl der tatsächlich mit dem G180-Befehl geholten AD-Werte
55	Achse 5 / Anzahl der tatsächlich mit dem G180-Befehl geholten AD-Werte
56	Achse 6 / Anzahl der tatsächlich mit dem G180-Befehl geholten AD-Werte
57	Achse 7 / Anzahl der tatsächlich mit dem G180-Befehl geholten AD-Werte
58	Achse 8 / Anzahl der tatsächlich mit dem G180-Befehl geholten AD-Werte
59	Achse 9 / Anzahl der tatsächlich mit dem G180-Befehl geholten AD-Werte
60	Achse 10 / Anzahl der tatsächlich mit dem G180-Befehl geholten AD-Werte
61	Achse 11 / Anzahl der tatsächlich mit dem G180-Befehl geholten AD-Werte
62	Achse 12 / Anzahl der tatsächlich mit dem G180-Befehl geholten AD-Werte
63	Achse 13 / Anzahl der tatsächlich mit dem G180-Befehl geholten AD-Werte

Teil 3 Liste der System-N-Register:

Nr.	Funktion		
64	Achse 14 / Anzahl der tatsächlich mit dem G180-Befehl geholten AD-Werte		
65	Achse 15 / Anzahl der tatsächlich mit dem G180-Befehl geholten AD-Werte		
66	Achse 16 / Anzahl der tatsächlich mit dem G180-Befehl geholten AD-Werte		
67	Reserve		
68	Anzahl freie TASKs (für RUN-Befehl)		
71	Status Achse 1	Mögliche Stati einer Achse: 0 : INIT 1 : DISABLE 2 : SAFE 3 : SAFE_WHILE_DISABLE 4 : SAFE_WHILE_OPERATIONAL 5 : SAFE_WHILE_ACTIVE 6 : IDLE 7 : IDLE_WHILE_DISABLE 8 : IDLE_WHILE_OPERATIONAL 9 : IDLE_WHILE_ACTUVE 10 : OPERATIONAL 11 : ACTIVE 12 : HALT 13 : HALT_WHILE_DISABLE 14 : HALT_WHILE_OPERATIONAL 15 : HALT_WHILE_ACTIVE 16 : FAULT	
72	Status Achse 2		
73	Status Achse 3		
74	Status Achse 4		
75	Status Achse 5		
76	Status Achse 6		
77	Status Achse 7		
78	Status Achse 8		
79	Status Achse 9		
80	Status Achse 10		
81	Status Achse 11		
82	Status Achse 12		
83	Status Achse 13		
84	Status Achse 14		
85	Status Achse 15		
86	Status Achse 16		
91	ab V. 5.01B	CAN-IO-25	IO-Modul-Typen-Nummer
92		CAN-IO-26	IO-Modul-Typen-Nummer
93		CAN-IO-27	IO-Modul-Typen-Nummer
93		CAN-IO-28	IO-Modul-Typen-Nummer
101	ab V. 4.77A	COM 1	Anzahl der empfangenen Zeichen
102		COM 2	
103		COM 3	
104		COM 4	
105		COM 5	
106		COM 6	
107		COM 7	
108		COM 8	

Teil 4 Liste der System-N-Register:

Nr.	Funktion
110	Counter 0 : Istwert
111	Counter 1 : Istwert
112	Counter 2 : Istwert
113	Counter 3 : Istwert
114	Counter 4 : Istwert
115	Counter 5 : Istwert
116	Counter 6 : Istwert
117	Counter 7 : Istwert
118	Counter 8 : Istwert
119	Counter 9 : Istwert
120	Counter 10 : Istwert
121	Counter 11 : Istwert
122	Counter 12 : Istwert
123	Counter 13 : Istwert
124	Counter 14 : Istwert
125	Counter 15 : Istwert
126	Counter 16 : Istwert
130	Counter 0 : Anzahl AD-Werte beim G182-Befehl
131	Counter 1 : Anzahl AD-Werte beim G182-Befehl
132	Counter 2 : Anzahl AD-Werte beim G182-Befehl
133	Counter 3 : Anzahl AD-Werte beim G182-Befehl
134	Counter 4 : Anzahl AD-Werte beim G182-Befehl
135	Counter 5 : Anzahl AD-Werte beim G182-Befehl
136	Counter 6 : Anzahl AD-Werte beim G182-Befehl
137	Counter 7 : Anzahl AD-Werte beim G182-Befehl
138	Counter 8 : Anzahl AD-Werte beim G182-Befehl
139	Counter 9 : Anzahl AD-Werte beim G182-Befehl
140	Counter 10 : Anzahl AD-Werte beim G182-Befehl
141	Counter 11 : Anzahl AD-Werte beim G182-Befehl
142	Counter 12 : Anzahl AD-Werte beim G182-Befehl
143	Counter 13 : Anzahl AD-Werte beim G182-Befehl
144	Counter 14 : Anzahl AD-Werte beim G182-Befehl
145	Counter 15 : Anzahl AD-Werte beim G182-Befehl
146	Counter 16 : Anzahl AD-Werte beim G182-Befehl

1.2.21 System R-Register

Die systeminternen Realzahlregister können vom Anwender nur abgefragt werden

Nr.	Funktion
1...9	Reserve
10	Interpolations-Bahnsollgeschwindigkeit
11...30	Reserve
31	Achse A1 / Position im Absolutsystem
32	Achse A2 / Position im Absolutsystem
33	Achse A3 / Position im Absolutsystem
34	Achse A4 / Position im Absolutsystem
35	Achse A5 / Position im Absolutsystem
36	Achse A6 / Position im Absolutsystem
37	Achse A7 / Position im Absolutsystem
38	Achse A8 / Position im Absolutsystem
39	Achse A9 / Position im Absolutsystem
40	Achse A10 / Position im Absolutsystem
41	Achse A11 / Position im Absolutsystem
42	Achse A12 / Position im Absolutsystem
43	Achse A13 / Position im Absolutsystem
44	Achse A14 / Position im Absolutsystem
45	Achse A15 / Position im Absolutsystem
46	Achse A16 / Position im Absolutsystem
47	Reserve
48	Reserve
49	Reserve
50	Reserve
51	Position der Achse 1 wenn sie in den Staus IDLE oder SAFE geschaltet wurde
52	Position der Achse 2 wenn sie in den Staus IDLE oder SAFE geschaltet wurde
53	Position der Achse 3 wenn sie in den Staus IDLE oder SAFE geschaltet wurde
54	Position der Achse 4 wenn sie in den Staus IDLE oder SAFE geschaltet wurde
55	Position der Achse 5 wenn sie in den Staus IDLE oder SAFE geschaltet wurde
56	Position der Achse 6 wenn sie in den Staus IDLE oder SAFE geschaltet wurde
57	Position der Achse 7 wenn sie in den Staus IDLE oder SAFE geschaltet wurde
58	Position der Achse 8 wenn sie in den Staus IDLE oder SAFE geschaltet wurde
59	Position der Achse 9 wenn sie in den Staus IDLE oder SAFE geschaltet wurde
60	Position der Achse 10 wenn sie in den Staus IDLE oder SAFE geschaltet wurde

Teil 2 Liste der System-R-Register:

Nr.	Funktion
61	Position der Achse 11 wenn sie in den Staus IDLE oder SAFE geschaltet wurde
62	Position der Achse 12 wenn sie in den Staus IDLE oder SAFE geschaltet wurde
63	Position der Achse 13 wenn sie in den Staus IDLE oder SAFE geschaltet wurde
64	Position der Achse 14 wenn sie in den Staus IDLE oder SAFE geschaltet wurde
65	Position der Achse 15 wenn sie in den Staus IDLE oder SAFE geschaltet wurde
66	Position der Achse 16 wenn sie in den Staus IDLE oder SAFE geschaltet wurde
67	Reserve
68	Reserve
69	Reserve
70	Reserve
71	Die Zielposition der Achse 1 beim letzten Fahrbefehl.
72	Die Zielposition der Achse 2 beim letzten Fahrbefehl.
73	Die Zielposition der Achse 3 beim letzten Fahrbefehl.
74	Die Zielposition der Achse 4 beim letzten Fahrbefehl.
75	Die Zielposition der Achse 5 beim letzten Fahrbefehl.
76	Die Zielposition der Achse 6 beim letzten Fahrbefehl.
77	Die Zielposition der Achse 7 beim letzten Fahrbefehl.
78	Die Zielposition der Achse 8 beim letzten Fahrbefehl.
79	Die Zielposition der Achse 9 beim letzten Fahrbefehl.
80	Die Zielposition der Achse 10 beim letzten Fahrbefehl.
81	Die Zielposition der Achse 11 beim letzten Fahrbefehl.
82	Die Zielposition der Achse 12 beim letzten Fahrbefehl.
83	Die Zielposition der Achse 13 beim letzten Fahrbefehl.
84	Die Zielposition der Achse 14 beim letzten Fahrbefehl.
85	Die Zielposition der Achse 15 beim letzten Fahrbefehl.
86	Die Zielposition der Achse 16 beim letzten Fahrbefehl.

1.3 Übersicht Befehlssatz

Einige Befehle der PA-CONTROL können nur bei bestimmten PA-CONTROL Typen verwendet werden.

Befehl ¹ → ¹ = PA-CONTROL MP

Befehl ² → ² = PA-CONTROL STEUER mit PLS7 / PLS9

Befehl ³ → ³ = PA-CONTROL COMPACT mit PLS7

Befehl ⁴ → ⁴ = PA-CONTROL servoTEC

In den Beschreibungen der Befehle wird durch die hochgestellte Zahl gekennzeichnet, bei welcher PA-CONTROL Type dieser Befehl nur verwendbar ist.

HINWEIS Alle Seitenangaben der folgenden Befehlsübersicht sind mit einem Link versehen

1.3.1 Programmablauf (Automatik)

1.3.1.1 Ablauf

Abbrechen	BREAK.....	51
Anhalten	STOP	52
Fortführen	START	52
Abfrage der aktuellen Zeilennummer eines Programms	Ni:=LINE Ni:=LINE.Name Ni:=LINE.Si	55
Beenden	END	55

1.3.1.2 Sprünge und Verzweigungen

Springe immer	JMP ZYKLUS	56
Beginne ein Programm an einer bestimmten Programmzeile...	JMP-LINE.Ni.....	59
Springe im Programm wenn Zustand von Eingang	G21 I1.1	57
Springe im Programm wenn Zustand von Ausgang	G21 O1.1	57
Springe im Programm wenn Zustand von Merker	G21 M1.1	57
Springe wenn (logische) Bedingung erfüllt	G21 M5.1 BOHREN	57
Springe wenn Inhalt von N-Register	CASE.JMP.N8	60
Springe wenn Überwachungszeit abgelaufen	G421.1.500 SAVE_1	251
Springe wenn Achse steht / läuft	G211.A1.0 WAIT	237
Springe wenn aktuelle Position der Achse	G221.A2.1 READY	242
Springe wenn neue Gerätefehlernachricht empfangen.....	G721.49 EMERGENCY	293
Beginne Interpolation an einer bestimmten Programmzeile	JMP-LINE-IPO.Ni	136

1.3.1.3 Unterprogramme aufrufen

Rufe UP auf	SUB LAMBLI	63
Rufe UP auf, Programmnamen steht in einem N-Register	SUB.N8.....	66
Rufe UP auf, Programmnamen steht in einem N-Register, Programmnamen hat eine feste Zeichenlänge	SUB.N7.8.....	66
Rufe UP auf, Programm wird auf richtigen Typ überprüft	SUB.N6.PNC	66
Rufe UP wenn Zustand von Eingang	G22 I1.1	67
Rufe UP wenn Zustand von Ausgang	G22 O1.1	67

Rufe UP wenn Zustand von Merker	G22 M1.1	67
Rufe UP wenn (logische) Bedingung erfüllt	G22 M5.1 SAEGEN	67
Rufe UP wenn Inhalt von N-Register	CASE.SUB.N8	68
Rufe UP wenn Zeitüberwachung abgelaufen	G422.1.500 ERROR.....	252
Rufe UP wenn Zeitüberwachung abgel. mit Wiederholen	G423.1.200 ERROR_2.....	253
Rufe UP wenn Achse steht / läuft	G212.A0.0 AUSG	239
Rufe UP wenn aktuelle Position der Achse	G222.1.A2.1500 PASTE ...	244
Rufe UP wenn neue Gerätefehlnachricht empfangen	G722.50 GREIFER.....	293
Unterprogrammaufruf, Programmname steht in einer Zeichenkette	SUB.S0, SUB.Sn	65
1.3.1.4 Schleifen		
Zähler Inkrement	INC.N2.N3 START	70
Zähler Dekrement	DEC .N3 ZYKLUS	71
1.3.1.5 Parallele Abläufe		
Start Parallelablauf	RUN	73
Anhalten Parallelablauf.....	SLEEP	75
Beende Parallelablauf	CANCEL	76
Start Parallelablauf in Abhängigkeit	CASE.RUN	77
Anhalten Parallelablauf in Abhängigkeit.....	CASE.SLEEP	79
Beende Parallelablauf in Abhängigkeit.....	CASE CANCEL	81
Hole Programmstatus	PROGSTAT	83
1.3.1.6 Fehlermanagement		
ERROROFF	ERROROFF	84
ERRORON	ERRORON	84
1.3.2 Achsbefehle		
1.3.2.1 Allgemeines		
1.3.2.2 Modi einer Achse		
Startpositioniermode aktivieren	G210.A3	236
Normalpositioniermode aktivieren	G213.A3	241
Disable, aktiviere die Funktion „OFF-ACHSE“	G140	88
Enable, aktiviere die Funktion „ON-ACHSE“	G141	89
Achse abschalten (IDLE, SAFE)	OFF.An	90
Achse einschalten (OPERATIONAL, ACTIVE)	ON.An.....	91
Unterbreche eine Fahrt (HALT).....	STOP.An.....	92
Stelle die Verfahrbereitschaft einer Achse her (OPERATIONAL, ACTIVE)	START.An	92
Brich einen gestoppten Fahrbefehl (HALT) ab.....	ABORT.An.....	93
Warte bis aktuelle Position </> Wert	G230.1.A1.456	246
Warte bis der Restweg der aktuellen Positionierung über- bzw. unterschritten ist	G231.A4.34	248

1.3.2.3 Fahren

Referenzfahrt	G25.A1.....	94
Verfahren	A1:=20 # A1:=R3	95
Absolut verfahren	G90.A0 # G90.A1	96
Relativ verfahren	G91.A0 # G91.A1	97
Beschleunigung festlegen	G100.A1.15	98
Verfahrgeschwindigkeit festlegen.....	FA1:=200 # FA2:=R4.....	99
Verfahre so lange Bedingung erfüllt	G123.A1 I3.1	101
Fahre Teilstrecke mit Start-Stop Frequenz	G150.....	103
Abfrage der letzten Verfahrgeschwindigkeit einer Achse.....	Rn:=Fan.....	105
Abfrage der Bahngeschwindigkeit.....	Rn:=FB	130
Aktuelle Verfahrgeschwindigkeit während Interpolation.....	Rn:=FBAn	133

1.3.2.4 Position und Parameter

Position setzen	G26.A1 # G26.A1.R1	106
Position auf Maß setzen (auf neuen Wert setzen)	G29.A.1.345 # G29.A2.R3	108
Achsparameter holen (laden)	R1:=A1, N1:=A1	110

1.3.2.5 Endschalter

Endschalterüberwachung ausschalten.....	G142.....	111
Endschalterüberwachung einschalten.....	G143.....	112

1.3.2.6 Encoder und Schleppfehler

Schleppfehler holen	Ni :=PEAn,Ri :=PEAn	113
Encoderposition holen (laden) ¹	R1:=ENC1 # N1:=ENC1....	113
Warte auf Position von SSI-Interface	SSI1.1.R12 # SSI1.0.N4....	116

1.3.2.7 Parameter-Befehle

servoTEC-Parameter schreiben (an servoTEC senden)	PARAMETER.ST.A1.DIS:=N2	117
servoTEC-Parameter lesen (von servoTEC holen).....	N2:=PARAMETER.ST.A1.IN1	119
Achsparameter laden aus einem Register	RA2.8:=R7	120
Achsparameter in ein Register schreiben	Ri:=RAn.m	121

1.3.2.8 Interpolation

IPO Bahnbeschleunigung festlegen	G100.B.1500	122
IPO Bahngeschwindigkeit festlegen.....	FB:=2000, FB:=R4.....	125
Linearinterpolation	G01 A1:=40 A2:=60.....	126
IPO Bahngeschwindigkeit festlegen, eine Achse läuft frei mit ..	FB.2:=1234 # FB.3:=R7	125
Linearinterpolation, eine Achse läuft frei mit	G01 A1:=50 A2:=30	
A3:=FB*R6		
.....		128
Linearinterpolation abbrechen	IPOEND	128
Bediene Ausgänge vor einer Interpolation	G160.20.O123.1	134
Setze Ausgänge während einer Interpolation	G161.1.O333.1	135
Rücksetzen von Ausgängen während einer Interpolation.....	G162.5.O333.0	135
Beginne Interpolation an einer bestimmten Programmzeile	JMP-LINE-IPO.Ni	136

1.3.2.9 Spezielle PA-CONTROL-Befehle

Motorstrom verändern ^{1,4}	G101.A1.50	138
---	------------------	-----

1.3.3 Eingänge, Ausgänge, Merker und Register

1.3.3.1 Eingänge

Warte bis Zustand von Eingang	I1.1.....	141
Logische UND-Verknüpfung mit Eingang	LD / AND	171
Logische ODER-Verknüpfung mit Eingang	LD / OR.....	172
Abbild von Eingängen in ein Register	G603.R3.2.8	286

1.3.3.2 Ausgänge

Warte auf Zustand von Ausgang	O1.1	142
Setze / Rücksetze Ausgang	O1:=1 # O2:=0.....	143
Logische UND-Verknüpfung mit Ausgang	LD / AND / OUT.....	171
Logische ODER-Verknüpfung mit Ausgang	LD / OR / OUT.....	172
Setzen, Rücksetzen, Negieren von Ausgängen	SET / RES / NOT.....	173
Erweiterung der Befehle SET und RES	SET / RES.O1.O16.....	173
Abbild von Register auf Ausgänge (Binär)	G600.R3.5.8	282
Abbild von Register auf Ausgänge (BCD)	G601.N2.2.4	283

1.3.3.3 Merker

Setze / Rücksetze Merker	M1:=1 # M2:=0	144
Direkte oder Indirekte Adressierung	M1.1 # M!10.1.....	144
Warte auf Zustand von Merker	M1.1	146
Logische UND-Verknüpfung mit Merker	LD / AND / OUT	171
Logische ODER-Verknüpfung mit Merker	LD / OR / OUT.....	172
Setzen, Rücksetzen, Negieren von Merker	SET / RES / NOT	173
Erweiterung der Befehle SET und RES	SET / RES.M1.M256	173
Abbild von Register auf Merker	G602.R3.4.8	284
Abbild von Merkern in ein Register	G604.N4.2.8	287
Schreibe Merker Nummer nach Vergleich in eine N-Register....	N3:=GETM.1.128.1	148
Vergleichsergebnis zuweisen	M36:=R1<56	147

1.3.3.4 Register

Direkte und indirekte Adressierung	N1, N!2.....	153
Lade Register mit Wert (Konstante)	N1:=23, R1:=33,5	154
Lade Register mit einem binärem Wert.....	N2:=2#00110000	155
Lade Register mit einem hexadezimalen Wert.....	N3:=16#3E8.....	155
Warte auf Status eines N-Registers	Ni.n, Ni.Nn, N!i.n, N!i.N!n.....	182
Lade Inhalt eines System-N-Registers in ein Ganzzahlregister	Nn:=SNn	183
Lade Inhalt eines System-R-Registers in ein Realzahlregister.....	Rn:=SRn	184
Bitweise Verarbeitung von N-Registern	Ni:=Nn&Nm Ni:=Nn&Konstante Ni:=Nn Nm Ni:=Nn Konstante Ni:=Nn^Nm Ni:=Nn^Konstante	181

1.3.4 Verweilzeit, Überwachungszeit, Datum, Echtzeituhr

1.3.4.1 Warten

Warte mit Programmausführung bis Zeit abgelaufen	T10, TN1.....	149
--	---------------	-----

1.3.4.2 Überwachungszeiten

Starte Überwachungszeit mit Sprung im Programm	G421.1.200 ERROR_1.....	251
Starte Überwachungszeit mit Unterprogrammaufruf	G422.1.500 ERROR_2.....	252
Starte Überwachungszeit mit Wiederholfunktion	G423.1.300 CRASH	253
Lösche (Rücksetzen) Überwachungszeit	G401.1	254

1.3.4.3 Datum

Setze Datum	DATE:=N1.N2.N3.N4	150
-------------------	-------------------------	-----

1.3.4.4 Uhrzeit

Lies Datum / Uhrzeit	S0:=TIME.HOUR.MIN	152
Setze Zeit.....	TIME:=N1.N2.N3	151

1.3.5 Mathematik

1.3.5.1 Adressierung

Direkte und indirekte Adressierung	N1, N2.....	153
--	-------------	-----

1.3.5.2 Mathematische Befehle

Lade – Register	Rn/Nn:=1234, N2:=16#3E8.....	
.....	N3:=2#00110000	154
Addition	R2:=R2+57	155
.....	R1:=R1+R2	155
Subtraktion	R4:=R4-33.....	157
.....	R5:=R5-R6	157
.....	R23:=1000-R7.....	157
Multiplikation	N2:=N1*12	158
.....	R3:=R2*R4.....	158
Division	N5:=N5/12	159
.....	R6:=R6/R7	159
.....	N7:=1000/N8.....	159
Sinusfunktionen	Rn:=SIN.K	160
.....	Rn:=SIN.Ri	160
.....	Rn:=ASIN.K.....	160
.....	Rn:=ASIN.Ri.....	160
Cosinusfunktionen	Rn:=COS.K	161
.....	Rn:=COS.Ri	161
.....	Rn:=ACOS.K	161
.....	Rn:=ACOS.Ri.....	161
Tangensfunktionen	Rn:=TAN.K	162
.....	Rn:=TAN.Ri	162
.....	Rn:=ATAN.K	162
.....	Rn:=ATAN.Ri	162

Wurzelfunktion	R99:=SQRT.R43	163
Ganzzahliger Anteil	R45:=INT.R70	163
Nachkomma Anteil	R43:=FRAC.R55	164
Betrag	N55:=ABS.N22	164
<u>1.3.5.3</u> <u>Vergleiche</u>		
Befehlsformen.....		165
Beispiele		166
Größer	M11:=N22>12.....	147
Gleich	M12:=R21=R22	147
Kleiner	M13:=R23<R24	147
Ungleich	M113:=N100<>123.....	147
<u>1.3.5.4</u> <u>Logische Operationen</u>		
LD	LD I3.1	171
UND	AND I4.0	171
ODER	OR I5.1	172
SET	SET O6.....	173
SET	SET M92.....	173
SET	SET.O1.O6	173
SET	SET.M2.M96.....	173
NOT	NOT O7	171, 172
RESET	RES O32	173
RESET	RES M16	173
RESET	RES.O1.O16.....	173
RESET	RES.M56.M94	173
XOR-Verknüpfung (Exklusiv-ODER).....	XOR In.i	176
Erzeuge XOR-Quersumme von einem String und speichere sie in einem Ganzzahlregister ab	N1:=XOR.S0.....	197
Mehrstufig UND (Klammern)	AND-LD	177
Mehrstufig ODER (Klammern)	OR-LD	178
Bitweise Verarbeitung von N-Registern	Ni:=Nn&Nm Ni:=Nn&Konstante Ni:=Nn Nm Ni:=Nn Konstante Ni:=Nn^Nm Ni:=Nn^Konstante	181
<u>1.3.5.5</u> <u>Register Operationen</u>		
Lade Register mit Wert (Konstante)	N1:=23, R1:=33,5	154
Lade Register mit einem binärem Wert.....	N2:=2#00110000	155
Lade Register mit einem hexadezimalen Wert.....	N3:=16#3E8.....	155
Warte auf Status eines N-Registers	Ni.n, Ni.Nn, N!i.n, N!i.N!n.....	182
Lade Inhalt eines System-N-Registers in ein Ganzzahlregister	Nn:=SNn	183
Lade Inhalt eines System-R-Registers in ein Realzahlregister.....	Rn:=SRn	184

Bitweise Verarbeitung von N-Registern	Ni:=Nn&Nm
	Ni:=Nn&Konstante
	Ni:=Nn Nm
	Ni:=Nn Konstante
	Ni:=Nn^Nm
	Ni:=Nn^Konstante 181

1.3.6 Kommunikation

1.3.6.1 Serielle Schnittstelle, Display und Tastatur

Ablaufanzeigen ausschalten	G11.0	188
Initialisiere	G500.0, G500.1.6.1.0	256
Anzeige löschen	G501	258
Zeile löschen	G502	259
Cursor positionieren	G503.4.1 # G504.N2.N6	260
Text ausgeben	G510.MASCHINENFEHLER	262
Text ausgeben mit CR und LF.....	G511.EINGRIFF	263
Sonderzeichen ausgeben	G512.36	264
Textzeile aus einem PTX-File ausgeben	G515.5.FEHLERTEXT	265
Registerinhalt ausgeben	G520.R2 # G520.R3.3.1	266
Merker-, Ausgangs-, Eingangszustand ausgeben	G520.I7	266
Registerinhalt ausgeben mit CR und LF	G521.N5 # G521.N7.8	268
Merker-, Ausgangs-, Eingangszustand mit CR und LF ausgeben	G521.M25	268
Registerinhalt übernehmen	G531.R7 ERROR_1	270
Merker- oder Ausgangszustand übernehmen	G531.M34.DATA	270
Zeichenkette übernehmen	G531.R2 FEHLER	270
Zeichenübernahme in den lokalen Zeichenpuffer S0.....	G532.13.U_FEHLER	272
Zeichenkette im Hintergrund übernehmen	G533.13	273
Definierte Anzahl Zeichen übernehmen	G534.4	275
Prüfe Übernahme von Schnittstelle	N5:=CHN	189
Prüfe ob Taste gedrückt	G540.N6	277
Hole Zeichen (Tastecode)	G541.N2	278
Eingabe Register	G542.30.1.3.N1	279
Schlüsselschalter abfragen	LD SM3.0	15

1.3.6.2 CANopen-Bus

Netzwerk-Management Befehle	Allgemeines	288
Start Remote Node	G701.n	289
Stop Remote Node	G702.n	290
Reset Remote Node	G703.n	290
Enter Pre-Operational-State	G704.n	291
Reset-Kommunikation	G705.n	291
Funktionsüberwachung der Teilnehmer	G711.49.12 FEHLER.....	292
Befehle für die Kontrolle von Gerätefehlern, Allgemeines	G72x	293
Verzweige, wenn eine Fehlernachricht vorhanden ist.....	G721.(ID) (Marke)	293
Rufe ein Unterprogramm auf, wenn eine Fehlernachricht vorhanden ist	G722.(ID) (Marke)	293
Befehle für die Bearbeitung von Servicedatenobjekten	G73x	294
Lies den Inhalt eines Servicedatenobjektes	G730	294
Schreibe den Inhalt eines Servicedatenobjektes.....	G731	295

Bearbeitung von Prozessdatenobjekten, Allgemeines	G74x, G75x	296
Prozessdaten lesen (PDO1)	G741.49.4	298
Prozessdaten lesen (PDO2)	G742.50.2	298
Prozessdaten senden (PDO1)	G751.49.4	299
Prozessdaten senden (PDO2)	G752.50.2	299
1.3.6.3 MODBUS		
Daten von MODBUS-Teilnehmer lesen	Ni:=MODBUS	186
Daten auf MODBUS-Teilnehmer schreiben	MODBUS	187
1.3.6.4 Stringbefehle		
Sn - Kopieren von Zeichenketten	Sn:= Si # Sn:="Hallo Welt"	190
S0 - Kopieren von Zeichenketten	S0:=CHN	191
Eingabe über Tastatur	G542.10.2.3.N1	279
Wertübernahme vom aktuellen Datenkanal	G531.R1 ERROR	270
Wertübernahme vom aktuellen Datenkanal	G531.13	270
Wertübernahme vom aktuellen Datenkanal	G532.13 FINE	272
Suche Position eines Zeichens im lokalen Zeichenpuffer S0 ..	N2:=POS.1.35	192
Wandle Zeichen aus dem lokalen Zeichenpuffer	N3:=COPY.2.5 C_FEHLER	193
Hole Teilstring	S3:=COPY.N5.N7	195
String zusammenfügen	Si:=Sn+SM	196
Hole die Länge eines Strings	N6:=LENGTH.S0	196
Schreibe den Inhalt eines Registers (Zahl) in einen String	S4:=R6.N3.N7	198
Übertrage Inhalt des lokalen Zeichenpuffers S0 in ein Ganzzahlregister	N4:=GET.3.2	199
Übertrage Inhalt des lokalen Zeichenpuffer S0 im INTEL-Format in ein Ganzzahlregister	N1:=GETI.1.2	200
Übertrage Inhalt eines N-Registers in den lokalen Zeichenpuffer	PUT.1.4.N1	201
Übertrage Inhalt eines N-Registers im INTEL-Format in den lokalen Zeichenpuffer S0	PUTI.1.4.N1	202
Erzeuge XOR-Quersumme von einem String	N3:=XOR.S1	197
1.3.7 Analog / Digital		
Analog-Digital-Wandler – Allgemeine Beschreibung		203
AD-Werte holen	N1:=AD1 # R1:=AD1	203
Erfassen von mehreren AD-Werten	G18x	225
AD-Messreihe synchron zur Achsbewegung	G180.A2.R22.R23.1.100.25	226
AD-Messreihe mit Zeitraster	G181.5.1.100.50	228
Digital-Analog-Wandler – Allgemeine Beschreibung		205
DA-Wert (Inkrement) einstellen	DA1:=N2	206
1.3.8 Datei-Befehle		
Hole N-Register aus PTX-File	G170.N1.N2.N10.TEXTE	207
Speichere N-Register in PTX-File	G171.N2.N3.N20.ERR	209
Zeile aus PTX-File in String (Sn) schreiben	G172.4.S2.FEHLERTEXTE	210

String (Sn) in einer Zeile eines PTX-Files speichern	G173.N2.S1.FEHLER_1.....	211
Speichern aktueller Werte in einen Programm-File	STORE TA2:=PE_001.....	213
Zuweisung mit Konstanten deaktivieren.....	STOREEND.....	215
Zuweisung mit Konstanten aktivieren.....	STORESTART.....	215
Speichern wenn Inhalt von Register.....	CASE.STORE.N3	216

1.3.9 Zähler

Zähler initialisieren.....	CNT4.INIT.0	219
Zähler setzen	CNT1:=N1 # CNT3:=25	220
Zähler lesen	N7 :=CNT8.....	220
Zähler vergleichen mit Konstante	CNT3.0.1234	221
Zähler vergleichen mit dem Inhalt eines Ganzzahlregisters	CNT4.1.N7.....	221

1.3.10 Temperaturregelung

Allgemeine Ausführungen	222
Parameterwert schreiben	TC3.SP:=25.5	223
Parameterwert lesen	R27:=TC1.X_EFF	224

1.3.11 Befehlsgruppen der PA-CONTROL

1.3.11.1 Befehle der G16x-Gruppe

Ausgänge vor dem Start der Interpolation.....	G160.20.O123.1	134
Bediene Ausgänge während der Interpolation	G161.1.O333.1#	
.....	G162.5.O333.0	135

1.3.11.2 Befehle der G17x-Gruppe

Hole N-Register aus PTX-File	G170.N1.N2.N10.TEXTE	207
Speichere N-Register in PTX-File	G171.N2.N3.N20.ERR	209
Zeile aus PTX-File in String (Sn) schreiben	G172.4.S2.....	210
String (Sn) in einer Zeile eines PTX-Files speichern	G173.N2.S1.FEHLER_1.....	211

1.3.11.3 Befehle der G2xx-Gruppe

Startpositioniermode aktivieren	G210.A3.....	236
Springe wenn Achse steht / läuft	G211.A1.0 WAIT.....	237
Rufe wenn Achse steht / läuft	G212.A0.0 AUSG	239
Normalpositioniermode aktivieren	G213.A3.....	241
Springe wenn aktuelle Position der Achse	G221.A2.1 READY	242
Rufe wenn aktuelle Position der Achse	G222.1.A2.1500 PASTE	244
Warte bis aktuelle Position < / > als Wert.....	G230.j.An.Konstante	246
Warte bis Restweg der aktuellen Positionierung erreicht ist	G231.An.Konstante	248

1.3.11.4 Befehle der G4xx-Gruppe

Lösche (Rücksetzen) Überwachungszeit	G401.1	254
Starte Überwachungszeit mit Sprung im Programm	G421.1.200 ERROR_1.....	251
Starte Überwachungszeit mit Unterprogrammaufruf	G422.1.500 ERROR_2.....	252
Starte Überwachungszeit mit Wiederholfunktion	G423.1.300 CRASH	253

1.3.11.5 Befehle der G5xx-Gruppe

Initialisiere	G500.0, G500.1.6.1.0	256
Anzeige löschen	G501	258
Zeile löschen	G502	259
Cursor positionieren	G503.4.1 # G504.N2.N6	260
Text ausgeben	G510.MASCHINENFEHLER	262
Text ausgeben mit CR und LF	G511.EINGRIFF	263
Sonderzeichen ausgeben	G512.36	264
Textzeile aus einem PTX-File ausgeben	G515.5.FEHLERTEXT	265
Registerinhalt ausgeben	G520.R2	266
Merker-, Ausgangs-, Eingangszustand ausgeben	G520.I7	266
Registerinhalt ausgeben mit CR und LF	G521.N5	268
Merker-, Ausgangs-, Eingangszustand mit CR und LF ausgeben	G521.M25	268
Registerinhalt übernehmen	G531.R7 ERROR_1	270
Merker- oder Ausgangszustand übernehmen	G531.M34.DATA	270
Zeichenkette übernehmen	G531.R2 FEHLER	270
Zeichenübernahme in den lokalen Zeichenpuffer S0	G532.13.U_FEHLER	272
Zeichenkette im Hintergrund übernehmen	G533.13	273
Definierte Anzahl Zeichen übernehmen	G534.4	275
Prüfe ob Taste gedrückt	G540.N6	277
Hole Zeichen (Tastecode)	G541.N2	278
Eingabe Register	G542.30.1.3.N1	279

1.3.11.6 Befehle der G6xx-Gruppe

Abbild auf Ausgänge (binär)	G600.R2.2.4	282
Abbild auf Ausgänge (BCD)	G601.N3.2.4	283
Abbild auf Merker	G602.R3.17.32	284
Abbild von Eingängen	G603.R4.2.8	286
Abbild von Merkern	G604.R4.2.8	287

1.3.11.7 Befehle der G7xx-Gruppe

Netzwerk-Management Befehle	Allgemeines	288
Start Remote Node	G701.n	289
Stop Remote Node	G702.n	290
Reset Remote Node	G703.n	290
Enter Pre-Operational-State	G704.n	291
Reset-Kommunikation	G705.n	291
Funktionsüberwachung der Teilnehmer	G711.49.12 FEHLER	292
Befehle für die Kontrolle von Gerätefehlern, Allgemeines ...	G72x	293
Verzweige, wenn eine Fehlernachricht vorhanden ist..	G721.(ID) (Marke)	293
Rufe ein Unterprogramm auf, wenn eine Fehlernachricht vorhanden ist	G722.(ID) (Marke)	293
Befehle für die Bearbeitung von Servicedatenobjekten	G73x	294
Lies den Inhalt eines Servicedatenobjektes	G730.	294

Schreibe den Inhalt eines Servicedatenobjektes.....	G731.....	295
Bearbeitung von Prozessdatenobjekten, Allgemeines.....	G74x, G75x.....	296
Prozessdaten lesen (PDO1).....	G741.49.4.....	298
Prozessdaten lesen (PDO2).....	G742.50.2.....	298
Prozessdaten senden (PDO1).....	G751.49.4.....	299
Prozessdaten senden (PDO2).....	G752.50.2.....	299

1.4 Programmierhinweise

1.4.1 Der Ablaufinterpreter

Die PA-CONTROL ist eine Ablaufsteuerung. Sie bearbeitet einen Befehl nach dem anderen (z.B. fahre Achse, setze Ausgang, warte auf Eingang, usw.). Ist ein Befehl abgearbeitet, so wird der nächste Befehl im Programm bearbeitet.

Damit parallele Abläufe in einer Maschine gesteuert werden können, wurde die Ablaufsteuerung als Parallelablaufsteuerung ausgeführt.

Der Ablaufinterpreter erlaubt in der Grundstellung die Abarbeitung eines Programms. In diesem Grundstellungsprogramm sind folgende Befehle bzw. Befehlsgruppen **nicht erlaubt**:

- Verfahrbefehle
- Interpolationsbefehle
- Unterprogrammaufrufe
- RUN

Zulässig sind dagegen folgende Aktionen im Grundstellungsprogramm:

- Aufruf von Bildschirmmasken der CANopen-Bedienerkonsole
- Veränderung von Ausgängen in Abhängigkeit von Eingängen
- Start des Automatik-Betriebes, wenn die vorher definierten Bedingungen erfüllt sind

Der Ablaufinterpreter kann im Automatikbetrieb bis zu 31 parallele Abläufe gleichzeitig bearbeiten. Jeder Parallelablauf kann Unterprogramme bis zu einer Schachteltiefe von 16 Programmen aufrufen. Man spricht von Multitasking.

Als ablauffähige Programme stehen PNC-, PNX- und PAB-Programme zur Verfügung. Alle Befehle können unabhängig vom Programmtyp verwendet werden.

Zum Ablegen von Texten stehen PTX-Programme (PTX-Dateien) zu Verfügung. In diesen Programmen (Dateien) sollten ausschließlich darstellbare ASCII-Zeichen verwendet werden.

HINWEIS Die ASCII-Zeichen 01hex (SOH), 02hex (STX), 03hex (ETX) und 04hex (EOT) dürfen auf keinen Fall zur Verwendung kommen, da sie das Übertragungsprotokoll zwischen der PA-CONTROL und dem Programm WINPAC stören!

Der Unterschied zwischen PNC-, PNX- und PAB-Programmen macht sich nur dann bemerkbar, wenn die PA-CONTROL gestoppt wurde (STOP-MODUS).

Beim Aktivieren eines Parallelablaufes (RUN, ...) wird der Programmtyp gespeichert, damit im STOP-MODUS der Parallelablauf entsprechend behandelt werden kann.

PNC-Abläufe: Achsen werden angehalten, Befehle werden nicht weiter bearbeitet.

PNX-Abläufe: Achsen werden angehalten, Befehle werden nicht weiter bearbeitet.

PAB-Abläufe: Ist gerade ein Verfahrbefehl aktiv, so wird die Achse angehalten und die Befehle nicht weiter bearbeitet, ansonsten werden alle anderen Befehle bearbeitet, als ob kein STOP aktiv ist.

Durch diese Unterscheidung kann im STOP-MODUS zum Beispiel Bedienerführung oder Kommunikation über serielle Schnittstellen weitergeführt werden.

Jeder Parallelablauf hat einen eigenen Zeiger auf einen aktuellen Datenkanal und einen eigenen (lokalen) S0-Zeichenpuffer von max. 80 Zeichen, der beim Automatik-Start gelöscht wird.

Beim Einstieg (START) in den Automatikbetrieb werden vom System keine Ausgänge verändert. Beim Ausstieg (ABBRUCH) aus dem Automatikbetrieb werden vom System alle Ausgänge zurückgesetzt.

1.4.2 Systemfunktionen

Es können folgende Definitionen für die Systemfunktion getroffen werden :

- Programm bei START (Startprogramm)
- Programm bei STOP (Stoppprogramm)
- Programm bei START nach STOP (Start nach Stoppprogramm)
- Programm bei STOERUNG
- Programm „in Grundstellung“ (siehe Abschnitt 1.4.1, Seite 42)

Die obigen Systemfunktionen können auf verschiedene Arten aktiviert werden.

aktiviert durch	Systemfunktion				
	START	STOP	START nach STOP	ENDE- ABBRUCH	STÖRUNG
Tastatur	X	X	X	X	-
digitale Eingänge (extern)	X	X	X	X	-
„Startprogramm“	X	X	X	X	
PAB-Ablauf	-	X	X	X	-
PNC-Ablauf/PNX-Ablauf	-	-	-	X	-
Diagnoseschnittstelle	X	X	X	X	-
RS232 Online	X	X	X	X	-
Interbus-S Kommando	X	X	X	X	-
Profibus-DP Kommando	X	X	X	X	-
RS232-Überwachung	-	-	-	-	X
Interbus-S Überwachung	-	-	-	-	X
Profibus-DP Überwachung	-	-	-	-	X
Achsüberwachung	-	-	-	-	X
Programm in Grundstellung	X				

Tabelle 1: Systemfunktionen

Systemfunktion START

Das als „Programm bei START“ festgelegte Programm (Startprogramm) wird in den ersten Parallelablauf eingetragen und bearbeitet. Mit RUN-Befehlen können dann bei Bedarf weitere parallele Abläufe gestartet werden.

Läuft das Startprogramm auf den END-Befehl auf, so wird der Automatikablauf beendet.

Systemfunktion STOP

Alle laufenden Positioniervorgänge werden angehalten und alle als PNC/PNX-Programme gestarteten Parallelabläufe nicht mehr weiterbearbeitet (wie SLEEP). Nachdem die Positioniervorgänge stehen, wird (falls festgelegt), das Stopp-Programm abgearbeitet.

Die als PAB-Programme gestarteten Parallelabläufe sind vom Stoppvorgang nur insoweit betroffen, dass die Positioniervorgänge angehalten werden. Ansonsten werden sie weiterbearbeitet.

Systemfunktion START nach STOP

Das Programm START nach STOP wird nach Ablauf des Stopp-Programms (falls festgelegt), abgearbeitet. Danach werden alle angehaltenen Positioniervorgänge wieder gestartet und alle Parallelabläufe wieder bearbeitet.

Systemfunktion ENDE-ABBRUCH

Alle Positioniervorgänge werden angehalten. Alle Parallelabläufe werden nicht mehr bearbeitet, und der Automatikbetrieb der Steuerung verlassen (Systemfunktion Störung). Ein STOP-Programm, falls festgelegt, wird nicht bearbeitet.

Systemfunktion STÖRUNG

Alle Positioniervorgänge werden angehalten. Alle Parallelabläufe werden nicht mehr bearbeitet. Das „Programm bei STÖRUNG“ (falls festgelegt) wird abgearbeitet. Der Automatikbetrieb wird erst durch Quittung verlassen.

Mögliche Quittungen können sein:

- BREAK-Befehl
- ESC-Taste auf der Tastatur
- über digitale Eingang „extern START“ ohne „extern STOP“
- ABBRUCH-Kommando von Schnittstelle (Profibus-DP, RS232-Online, usw.)

Beim Verlassen der „AUTOMATIK-Betriebsart“ wird versucht, die Störung zurückzusetzen.

Die Systemstörungen werden in der Steuerung archiviert. Es können bis zu 50 Störungen mit folgenden Informationsinhalt abgelegt werden:

- Zeitpunkt (aus Betriebsstundenzähler)
- Fehlernummer (mit Text)

Sind mehr als 50 Systemfehler aufgetreten, so werden die ältesten Meldungen überschrieben.

1.4.3 Steuerung des Automatikbetriebes über externe Eingänge

Definition

I [START]	Starteingang aus den Systemparametern (Taster Schliesser, Ausgang von einer Mastersteuerung)
I [STOP]	Stoppeingang aus den Systemparametern (Taster Öffner, Ausgang von einer Mastersteuerung)
O [BEREIT]	Bereitgang aus den Systemparametern (Signale-Lampe, Eingang einer Mastersteuerung)
O [AUTOMATK AKTIV]	wird durch einen Befehl (On:=) zu Beginn des Programmes gesetzt (Signale-Lampe, Eingang einer Mastersteuerung)
O [AUTOMATIK GESTOPPT]	wird in den Programmen „STOP“, „START NACH STOP“ und „STÖRUNG“ durch Befehle beeinflusst (Signale-Lampe, Eingang einer Mastersteuerung)

Tabelle 2: Definition externer Eingänge

1.4.3.1 Start von Automatik mit Abbruch

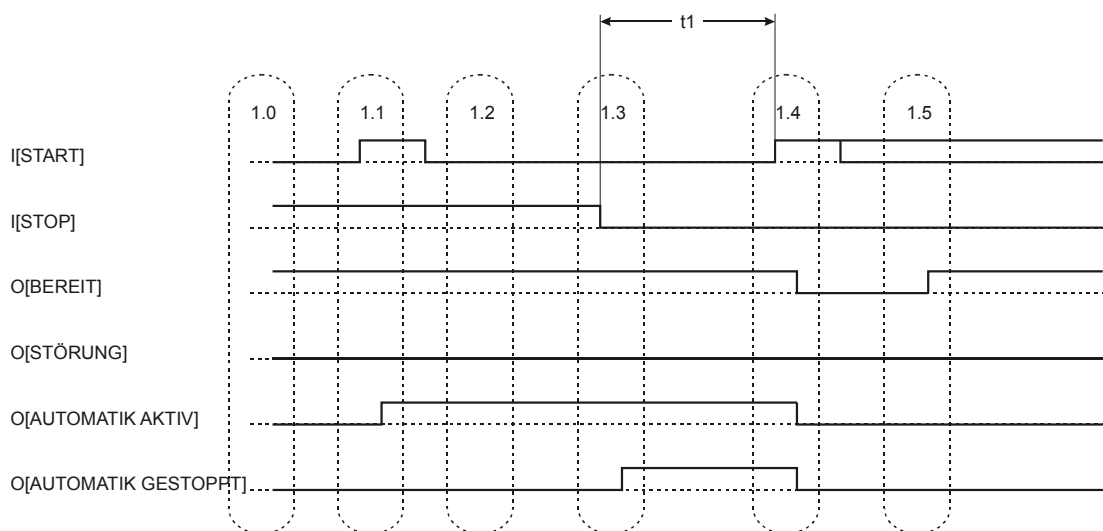


Bild311D

Ablaufdiagramm 1: Start Automatikbetrieb

Beschreibung der Zustände:

- 1.0 → PA-CONTROL ist in Grundstellung, es ist kein Fehler vorhanden
- 1.1 → Automatik wird gestartet
- 1.2 → PA-CONTROL ist im Automatikbetrieb, es ist kein Fehler vorhanden
- 1.3 → PA-CONTROL wird gestoppt
- Wartezeit von „START“ $t_1 \geq 10$ ms
- 1.4 → PA-CONTROL bricht den Automatikbetrieb ab und wechselt in die Grundstellung
- 1.5 → PA-CONTROL ist in Grundstellung, es ist kein Fehler vorhanden

1.4.3.2 Start von Automatik mit STOP und START NACH STOP

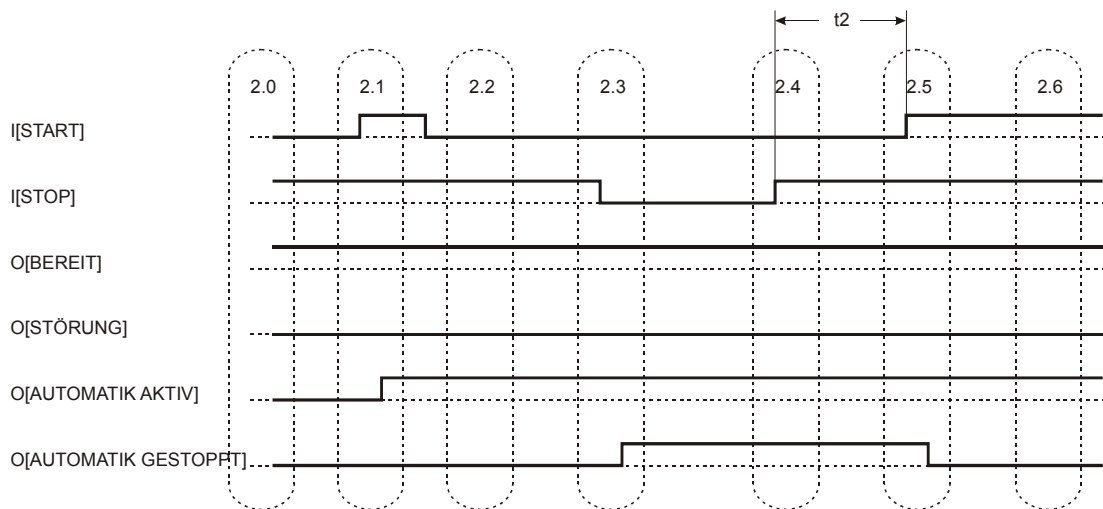


Bild312D

Ablaufdiagramm 2: Start Automatikbetrieb mit STOP und START nach STOP

Beschreibung der Zustände:

- 2.0 → PA-CONTROL ist in Grundstellung, es ist kein Fehler vorhanden
- 2.1 → Automatik wird gestartet
- 2.2 → PA-CONTROL ist im Automatik-Modus, es ist kein Fehler vorhanden
- 2.3 → PA-CONTROL wird gestoppt
- 2.4 → STOP-Eingang wird inaktiviert
- Wartezeit $t \geq 2ms$
- 2.5 → START-Eingang wird aktiviert, PA-CONTROL macht im Automatik weiter
- 2.6 → PA-CONTROL ist im Automatik-Modus, es ist kein Fehler vorhanden

1.4.3.3 PA-CONTROL ist im Automatikbetrieb und es tritt eine Störung auf

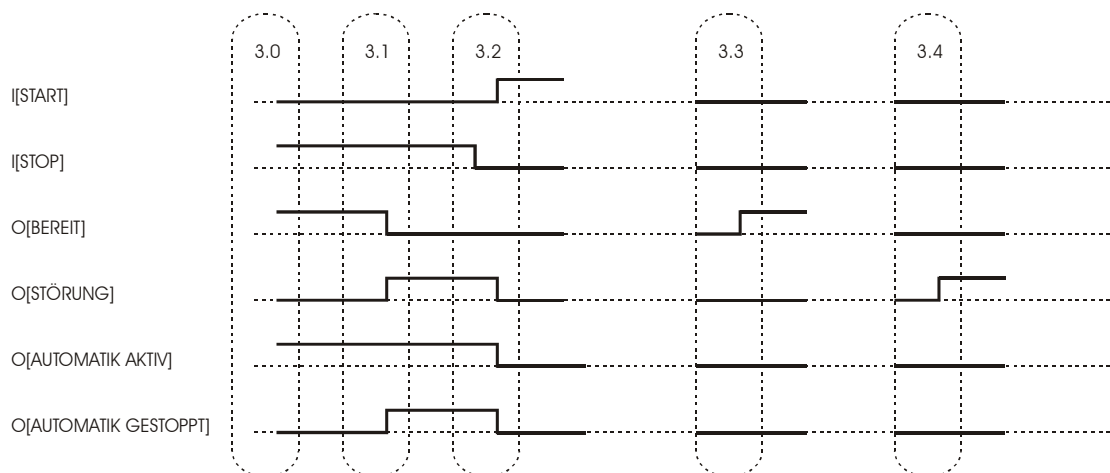


Bild313D

Ablaufdiagramm 3: Ablauf Störung im Automatikbetrieb

Beschreibung der Zustände

- 3.0 → PA-CONTROL ist im AUTOMATIK, es ist kein Fehler vorhanden
- 3.1 → Störung kommt
- 3.2 → Störung wird quittiert, AUTOMATIK wird abgebrochen, die PA-CONTROL wechselt in die Grundstellung
- 3.3 → die PA-CONTROL ist in der GRUNDSTELLUNG und die Störung ist behoben, mögliche Fehler:
- Endschalter angefahren
 - Wert zu groß
 - Drehüberwachung
 - Leistungsteil nicht bereit
 - Temperaturfehler
- 3.4 → die PA-CONTROL ist in Grundstellung und die Störung steht weiter an mögliche Ursache:
- Fehler in der Versorgungsspannung
 - etc.

HINWEIS Die PA-CONTROL muss ausgeschaltet und der Hardwarefehler behoben werden!

1.4.3.4 PA-CONTROL ist in Grundstellung und es steht eine Störung an

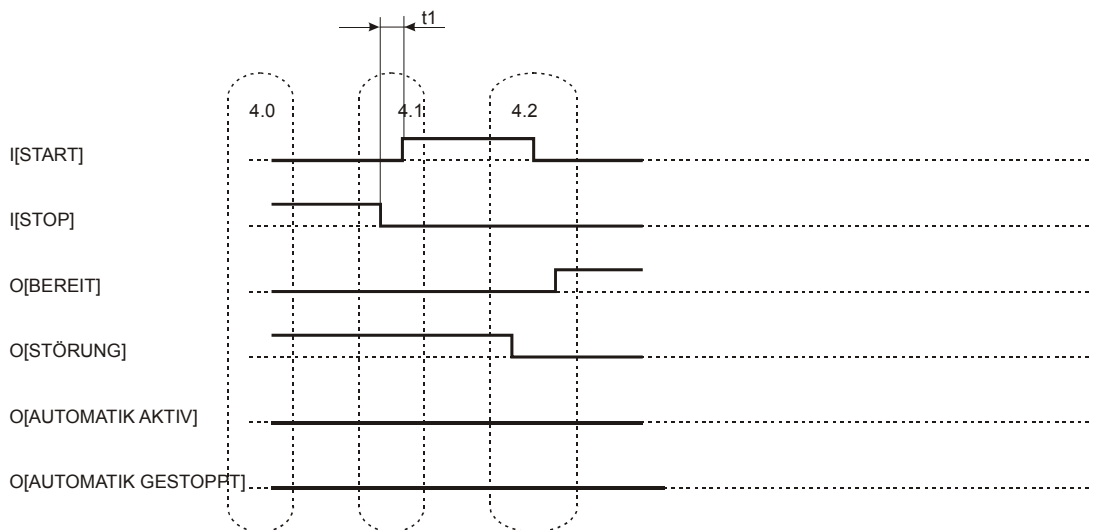


Bild314D

Ablaufdiagramm 4: Ablauf Störung, PA-CONTROL in Grundstellung

Beschreibung der Zustände

- 4.0 → PA-CONTROL ist in GRUNDSTELLUNG, es ist ein Fehler vorhanden
Wartezeit $t_1 \geq 80\text{ms}$
- 4.1 → der Fehler wird zurückgesetzt
- 4.2 → der Fehler ist zurückgesetzt, die PA-CONTROL meldet BEREIT

1.4.4 Programmstruktur

Ein Programm ist durch einen Programmnamen gekennzeichnet und mit einer Erweiterung (Extension) versehen. Diese Erweiterung dient der Typerkennung der Programme.

Der Programmnamen besteht aus maximal 20 alphanumerischen Zeichen (0-9, A-Z). Als Sonderzeichen innerhalb des Namens ist nur der Bindestrich „-“ und der Unterstrich „_“ zulässig. Der Programmtyp besteht aus drei Buchstaben.

Ein Programm besteht aus einzelnen Befehlen die nacheinander abgearbeitet werden und ist mit dem END-Befehl abzuschließen. Das Programm darf maximal 9990 Programmzeilen haben.

Ein Programm kann ein weiteres Programm mit dem SUB-Befehl als Unterprogramm aufrufen (bis zu 16 Verschachtelungen sind möglich), oder mit dem RUN-Befehl einen weiteren parallelen Ablauf aktivieren (bis 31 Parallelablaufprogramme).

Innerhalb eines Programms können bedingte Sprünge und unbedingte Sprünge auf Marken (Label) ausgeführt werden. Die Markennamen (Labelnamen) sind nur innerhalb eines Programms bekannt. Es ist deshalb möglich/sinnvoll, den gleichen Markennamen in verschiedenen Programmen zu benutzen.

Sprungbefehle aus einem Programm in ein anderes Programm sind nicht möglich.

Ein Markenname besteht aus maximal 20 alphanumerischen Zeichen (0-9, A-Z). Als Sonderzeichen innerhalb des Namens ist nur der Bindestrich „-“ und der Unterstrich „_“ zulässig. Der Markenname wird durch ein Leerzeichen, Semikolon oder das Zeilenende abgeschlossen. Bei der Definition der Marke muss die Marke, angeführt durch das Zeichen „\$“, in der ersten Spalte einer Zeile stehen. Innerhalb eines Programms kann die Marke nur einmal definiert werden, ein Sprung von verschiedenen Stellen auf diese eine Marke ist selbstverständlich erlaubt.

1.5 Erläuterungen zur Syntax

Befehlskürzel

Beschreibung:

In diesem Abschnitt wird der Befehl mit allgemeinen Worten beschrieben.

HINWEIS An diesem Punkt wollen wir Sie auf Besonderheiten hinweisen und Ihnen mittels Querverweisen Hilfen geben.

Befehlsform:

mmmm nnnn

Die Befehle sind aufgeteilt in Operand und Operator, oder mit anderen Worten in Befehlskürzel und in eine Zahl.

HINWEIS Die Leerstellen zwischen Operand und Operator wurden im Abschnitt Befehlsform zur Übersichtlichkeit eingefügt, bei der Eingabe in die PA-CONTROL sind sie nicht erlaubt!

In einer Programmzeile können mehrere Befehle stehen. Die Befehle einer Zeile müssen durch mindestens ein Leerzeichen voneinander getrennt werden.

Anwendung:

In diesem Abschnitt werden ein paar Worte zum praktischen Bezug des Befehls gemacht.



VORSICHT

Die Beispiele sind ohne Kenntnisse der aktuellen Mechanik erstellt und sind, auf ihre sinnvolle Anwendung hin, unbedingt zu prüfen.

Beispiel

1	18.1	Steuerung wartet bis Eingang 8 logisch 1 ist
2	14.0	Steuerung wartet bis Eingang 4 logisch 0 ist
3	END	

Die Zahlen am Zeilenanfang sind die Zeilennummern, die in der PA-CONTROL vom Editor selbst generiert werden und nur der Orientierung im Programmeditor dienen.

Die Beispiele können direkt übernommen werden. Kommentare sind, wie in der PA-CONTROL, mit einem Semikolon abgetrennt und können, soweit sie die maximale Zeilenlänge von 80 Zeichen nicht überschreiten, mit in das Programm übernommen werden. Alle Programme werden generell mit dem END-Befehl abgeschlossen.

HINWEIS Bei der Übertragung eines Programms vom PC zur PA-CONTROL kann im Auswahlfenster für die zu übertragenden Programme festgelegt werden, ob das Programm mit oder ohne Kommentar in die Steuerung übernommen werden soll.

Werden bei der Programmierung vor dem Kommentar zwei Semikola geschrieben, wird der Kommentar in jedem Fall in die Steuerung übertragen!

1.6 Programmorganisation

Die Befehle der Programmgruppe „Programmorganisation“ realisieren Sprünge und Verzweigungen, starten Unterprogramme, starten, beenden oder halten „Parallele Abläufe“ an und sind für das Fehlermanagement verantwortlich.

1.6.1 Ablauf

1.6.1.1 BREAK - Abbruch Automatikablauf

BREAK

Beschreibung:

Der Befehl BREAK bricht den Automatikablauf der PA-CONTROL ab. Die PA-CONTROL kehrt zum Hauptmenü zurück.

Befehlsform:

BREAK

Anwendung:

Der Befehl BREAK dient zum Abbrechen des Automatikablaufes der PAC, wenn zum Beispiel in einem Unterprogramm ein Maschinenfehler aufgetreten ist. Bei der Anwendung des Befehles BREAK erspart man sich den Rücksprung in die Programmstruktur, um aus mehreren Unterprogrammebenen in das Hauptprogramm zurückzugehen und dann dort an das Programmende zu verzweigen.

HINWEIS Wird der BREAK-Befehl im „Programm bei Störung“ benutzt, erhalten die Bediener keine Meldung über die Fehlernummer „EXXX“. Die PA-CONTROL geht in die Grundstellung über, es besteht aber keine Möglichkeit der Analyse des Fehlers. Ist ein „Programm bei STOP“ festgelegt, so wird dieses nicht bearbeitet!

Beispiel

```

1  $SCHLEIFE
2  I10.1                ;Start
3  SUB HOLEN           ;Unterprogramm zum Teil holen
4  SUB BEARBEIT       ;Unterprogramm zum Teil bearbeiten
5  SUB ABLEGEN        ;Unterprogramm zum Teil ablegen
6  G21 I11.1 SCHLEIFE ;weiter produzieren
7  END

```

Programm: HOLEN.PNC

```

1  G421.1.100 FEHLER  ;Fehler, wenn Teilholvorgang nicht in 1 s erledigt
2  O1:=1              ;Greifer oeffnen
3  I1.0 I2.1          ;Greifer offen
4  O2:=1              ;Greifer senken
5  I3.0 I4.1          ;Greifer unten
6  O1:=0              ;Greifer schließen
7  I2.0 I1.1          ;Greifer geschlossen
8  O2:=0              ;Greifer heben
9  I4.0 I3.1          ;Greifer oben
10 G401.1
11 END
12 $FEHLER
13 I12.1              ;QUITTUNG FEHLER BEI TEIL HOLEN
14 BREAK

```

1.6.1.2 START - Starte Automatikablauf

START

Beschreibung:

Der Befehl **START** startet die angehaltenen Programme bzw. Positionierbefehle und hebt somit den Befehl Stop wieder auf. Das bedeutet, dass angehaltene Fahrbefehle wieder aufgenommen und an ihrer vorgesehen Position beendet werden.

Befehlsform:

START

Anwendung:

Der Befehl **START** startet angehaltene Achsen, sowie die weitere Abarbeitung gestoppter PNC/PNX-Abläufe (mit RUN gestartet).

HINWEIS Der START-Befehl kann nur in einem PAB-Programm verwendet werden.

Beispiel

```
1  $SCHLEIFE
2  I10.1                ;Start
3  RUN SCHAUEN         ;starten eines PAP-Programms
4  RUN WEITER          ;starten eines PAP-Programms
5  SUB BEARBEIT        ;Unterprogramm zum Teil bearbeiten
6  SUB ABLEGEN         ;Unterprogramm zum Teil ablegen
7  G21 I11.1 SCHLEIFE ;weiter produzieren
8  END
```

Programm: SCHAUEN.PAB

```
1  G421.1.100 Fehler   ;Fehler, wenn Teilholvorgang nicht in 1 s erledigt
2  I1.0 I2.1           ;Greifer offen
3  I3.0 I4.1           ;Greifer unten
4  I2.0 I1.1           ;Greifer geschlossen
5  I4.0 I3.1           ;Greifer oben
6  G401.1
7  END
8  $FEHLER
9  STOP                ;aktiviere Systemfunktion STOP
10 END
```

Programm: WEITER.PAB

```
1  $LOOP               ;Schleifenanfang
2  G21 I3.1 Start      ;zu Start wenn Eingang 3 bestromt
3  JMP LOOP            ;gehe an Anfang
4  $Start              ;Sprungmarke Start
5  Start               ;aktiviere Systemfunktion START nach STOP
6  END
```

1.6.1.3 STOP - Stoppe Automatikablauf

STOP

Beschreibung:

Der Befehl STOP hält alle Achsen an, unabhängig davon, ob sie durch ein PNC- oder PAB-Programm gestartet wurden. Alle Abläufe verhalten sich entsprechend der Festlegung bei RUN. Das bedeutet, dass PAB-Programme weiterhin laufen.

Befehlsform:

STOP

Anwendung:

Der Befehl STOP dient zum Anhalten aller Positionierbefehle bzw. zum Stoppen aller Achsen und PNC-Abläufe welche mit RUN gestartet wurden. Tritt zum Beispiel ein Anwendungsfehler auf, so kann das Programm auf elegante Weise angehalten werden, ohne abzubrechen.

HINWEIS Der STOP-Befehl kann nur in einem PAB-Programm verwendet werden.

Beispiel

```

1    $SCHLEIFE
2    I10.1
3
4    RUN GREIFER           ;Start
5    M11.1                 ;Starten eines PAP- Programms
6    SUB BEARBEIT          ;Unterprogramm zum Teil bearbeiten
7    SUB ABLEGEN           ;Unterprogramm zum Teil ablegen
8    G21 I11.1 SCHLEIFE   ;weiter produzieren
9    END

```

PAB-Programm: GREIFER.PAB

```

1    M10.1                 ;Warte bis Fertig-Merker
2    G421.1.200 Fehler     ;Fehler, wenn Teilholvorgang nicht in 2 s erledigt
3    O1:=0 O2:=1          ;Greifer öffnen
4    I1.0 I2.1             ;Greifer offen
5    O3:=0 O4:=1          ;Greifer senken
6    I3.0 I4.1             ;Greifer unten
7    O1:=1 O2:=0          ;Greifer schliessen
8    I1.1 I2.0             ;Greifer geschlossen
9    O3:=1 O4:=0          ;Greifer heben
10   I4.0 I3.1             ;Greifer oben
11   G401.1
12   M11:=1
13   END
14   $FEHLER
15   STOP                  ;aktiviere Systemfunktion STOP
16   END

```

1.6.1.4 Ni:=LINE - Abfrage der aktuellen Zeilennummer eines Programms

Ni:=LINE

Beschreibung:

Mit dem LINE-Befehl kann die Zeilennummer eines Programms ermittelt werden, um dann später mit dem Befehl „JMP-LINE...“ wieder an diese Programmstelle zu verzweigen .

Der Befehl „Ni:=LINE“ ermittelt die Zeilennummer des eigenen Programms, also die Zeile in der dieser Befehl steht.

Die Befehle „Ni:=LINE.Name“ oder Ni:=LINE.Si suchen in allen TASKs und deren Unterprogrammebenen nach dem Programmnamen.

Ist das Programm nicht aktiv, so ist das Ergebnis 0. Ist das Programm aber gar nicht vorhanden, erfolgt die Ausgabe der Fehlermeldung „E513=Programm nicht vorhanden“.

HINWEIS Als String können nur S1 bis S16 verwendet werden!

Befehlsform:

Ni:=LINE

Ni:=LINE.Name

Ni:=LINE.Si

1.6.1.5 END - Beenden

1.6.2 Sprünge und Verzweigungen

1.6.2.1 JMP - Unbedingter Sprung

JMP

Beschreibung:

Mit der Funktion JMP wird ein „unbedingter Sprung“ auf die angegebene Sprungmarke innerhalb des Programms ausgeführt.

Eine Sprungmarke besteht aus maximal 20 alphanumerischen Zeichen (0-9, A-Z). Als Sonderzeichen innerhalb des Namens ist nur der Bindestrich „-“ und der Unterstrich „_“ zulässig.

Befehlsform:

JMP (Marke)

Anwendung:

Bei Programmschleifen

Beispiel

```
1   $ENDLOS
2   I1.1           ;warte bis der Eingang 1 bestromt ist
3   O1:=1
4   T100
5   O1:=0
6   JMP ENDLOS   ;springe zur Marke „ENDLOS“
7   END
```

HINWEIS Ein Sprungziel (Label) oder ein Programmnamen kann sowohl in Groß- als in Kleinschreibung bezeichnet werden. Es können Ziffern und Buchstaben verwendet werden. Die PA-CONTROL arbeitet intern nur mit Großbuchstaben. Die Sprungziele „ANFANG“ und „Anfang“ sind gleich!

1.6.2.2 G21 - Bedingter Sprung

G21

G21 (Bedingung) (Marke)

Beschreibung:

Mit der Funktion **G21** wird ein „bedingter Sprung“ ausgeführt. Die im Operand angegebene Bedingung kann der logische Zustand eines Ein-, Ausgangs, Merkers oder Systemmerkers sein. Ist die Bedingung erfüllt, wird ein Sprung auf die im Operator angegebene Marke ausgeführt und das Programm dort fortgeführt. Ist die Bedingung nicht erfüllt, wird das Programm in der nächsten Zeile fortgeführt.

Befehlsform:

G21 I1.0 Marke

G21 O2.1 Marke

G21 M4.1 Marke

G21 MI3.1 Marke

G21 SM51.1 Marke

Anwendung:

Programmverzweigung in Abhängigkeit von dem logischen Zustand eines Eingangs, Ausgangs, Merkers oder Systemmerkers.

Beispiel

1	G21 I1.1 EIN_5	<i>;falls Eingang 1 bestromt ist, wird die Programmabarbeitung bei der Marke „EIN_5“ fortgesetzt, ansonsten mit dem nächsten Befehl</i>
2	O1:=1 T100 O1:=0	<i>;Ausgang 1 für 1 Sekunde gesetzt</i>
3	\$EIN_5	
4	T10	
5	G21 O5.1 MERK_31	<i>;falls Ausgang 5 auf 1 gesetzt ist, wird das Programm bei der Marke „MERK_31“ fortgeführt, ansonsten mit dem nächsten Befehl</i>
6	O1:=1 T100 O1:=0	<i>;Ausgang 1 für 1 Sekunde gesetzt</i>
7	\$MERK_31	
8	T10	
9	G21 M31.0 ENDE	<i>;falls Merker 31 auf logisch 0 ist, wird das Programm bei der Marke „ENDE“ fortgeführt, ansonsten mit dem nächsten Befehl</i>
10	O1:=1 T100 O1:=0	<i>;Ausgang 1 für 1 Sekunde gesetzt</i>
11	\$ENDE	
12	T10	
13	END	

1.6.2.3 JMP-LINE.Ni - Start eines Programms bei einer bestimmten Programmzeile

JMP-LINE.Ni

Beschreibung:

Mit diesem Befehl kann zu einer bestimmten Zeile im Programm gesprungen werden.

HINWEIS Liegt das Sprungziel in einer Interpolation (zweiter, dritter G01-Befehl) oder ist der angesprungene Befehl ein G161-, G162- oder FB-Befehl, so wird der Fehler „E755=falsches Sprungziel“ erzeugt.

HINWEIS Die Zeilennummer eines FB-Befehls kann über die Funktionen „Ni:=LINE.Name“ oder „Ni:=LINE-Sn“ nicht ermittelt werden. Dadurch ist es eigentlich nicht möglich mit dem JMP-LINE- (oder JMP-LINE-IPO-) Befehl auf diesen Befehl zu springen.

Beispiel

Anmerkung: <N2> wird z.B. durch ein PAP-Programm modifiziert

```
1      N2:=2
2      $A
3      JMP-LINE.N2
4      JMP A
5      ;
6      O1:=1 T10 O1:=0 T10
7      JMP A
8      ;
9      O1:=2 T10 O2:=0 T10
10     JMP A
11     ;
12     O3:=1 T10 O3:=0 T10
13     JMP A
14     ;
15     END
```

CASE.JMP.N

CASE.JMP.(Variable)**Beschreibung:**

Der CASE.JMP-Befehl ist eine Programmverzweigung in Abhängigkeit von dem Inhalt eines Ganzzahlregisters. Mit diesem Befehl lassen sich auf einfache Weise Sprungverteiler aufbauen.

Die PA-CONTROL prüft den Inhalt des Ganzzahlregisters und springt entsprechend dem Wert zu den Marken. Ist der Inhalt des Ganzzahlregisters kleiner 1 oder größer der Anzahl der Elemente der Markentabelle (im Beispiel: (ROT), (BLAU), (GELB), (ROSA)), so wird ein Sprung zu der Marke hinter ELSE durchgeführt, ansonsten wird ein Sprung auf die entsprechende Marke ausgeführt.

Befehlsform:

CASE.JMP.Nn

(Marke1)

(Marke2)

...

(Marke_i)

ELSE Marke_e

Oder mit indirekter Adressierung des Registers

CASE.JMP.N!n

(Marke1)

(Marke2)

...

(Marke_i)

ELSE Marke_e

Anwendung:

Programmverzweigungen bei mehr als zwei Möglichkeiten, z.B. verschiedenen Fertigungstypen, Verzweigung auf Grund eines Zählerinhaltes usw.

Beispiel

```

1   $GRAU
2   N8:=N13           ;Wert über Register einlesen
3   CASE.JMP.N8      ;prüfe den Inhalt von N8 und verzweige bei:
4   (ROT)            ;N8=1 : Sprung zur Marke „ROT“
5   (BLAU)           ;N8=2 : Sprung zur Marke „BLAU“
6   (GELB)           ;N8=3 : Sprung zur Marke „GELB“
7   (ROSA)           ;N8=4 : Sprung zur Marke „ROSA“
8   ELSE GRAU        ;Sprung zur Marke „GRAU“ wenn N8<1 oder N8>4
                        ;(Wert von N8 nicht innerhalb des vorgesehenen Be-
                        ;reiches, hier 1 bis 4) ist

9   $ROT
10  O1:=1 T100 O1:=0 ;setze Ausgang 1 für 1 s
11  JMP ENDE
12  $BLAU
13  O2:=1 T100 O2:=0 ;setze Ausgang 2 für 1 s
14  JMP ENDE
15  $GELB
16  O3:=1 T100 O3:=0 ;setze Ausgang 3 für 1 s
17  JMP ENDE
18  $ROSA
19  O4:=1 T100 O4:=0 ;setze Ausgang 4 für 1 s
20  $ENDE
21  END

```

1.6.3 Aufruf Unterprogramme

1.6.3.1 Prinzipien des Unterprogrammaufrufs

Beschreibung:

Der Unterprogrammaufruf **SUB** und der Bedingte Unterprogrammaufruf **G22** ermöglichen dem Anwender ein gut strukturiertes Programm zu erstellen. So können Funktionen, die mehrmals benötigt werden, in einem Unterprogramm hinterlegt werden. Damit bleiben die Programme überschaubar und die Unterprogramme können einzeln getestet werden.

Durch den unbedingten oder bedingten Unterprogrammaufruf erfolgt ein Sprung auf ein in der PA-CONTROL abgespeichertes Programm. Der Name kann explizit angegeben werden (siehe Abschnitt 1.6.3.2, Seite 63) oder aus einer Zeichenkette (siehe Abschnitt 1.6.3.3, Seite 65), oder einem Ganzzahlregister (siehe Abschnitt 1.6.3.4, Seite 66) entnommen werden. Für den bedingten Unterprogrammaufruf → siehe Abschnitt 1.6.3.5, Seite 67.

HINWEIS Ein Unterprogramm darf sich nicht selbst aufrufen. Unterprogramme müssen mit dem END-Befehl abgeschlossen werden. Es können bis zu 15 Unterprogrammaufrufe ineinander verschachtelt werden.

HINWEIS Ein Sprungziel (Label) oder ein Programmnamen kann sowohl in Groß- als in Kleinschreibung bezeichnet werden. Es können Ziffern und Buchstaben verwendet werden. Die PA-CONTROL arbeitet intern nur mit Großbuchstaben. Die Sprungziele „ANFANG“ und „Anfang“ sind gleich!

Befehlsformen:

SUB.name (siehe Abschnitt 1.6.3.2, Seite 63)

SUB.S0 (siehe Abschnitt 1.6.3.3, Seite 65)

SUB.S1 (siehe Abschnitt 1.6.3.3, Seite 65)

SUB.N1 (siehe Abschnitt 1.6.3.4, Seite 66)

SUB.N2.PNX (siehe Abschnitt 1.6.3.4, Seite 66)

G22 I1.1 Name

G22 O2.1 Name

G22 M2.0 Name

G22 M!3.1 Name

G22 SM6.0 Name (siehe Abschnitt 1.6.3.5, Seite 67)

CASE.SUB.(Variable) (siehe Abschnitt 1.6.3.6, Seite 68)

1.6.3.2 SUB.Name - Standard Unterprogrammaufruf

SUB.name

Beschreibung:

Durch den Unterprogrammaufruf **SUB.name** erfolgt ein Sprung auf ein abgespeichertes Programm mit dem angegebenen Namen.

Befehlsform:

SUB.Name

Beispiel nächste Seite

Beispiel

```

1      I2.1
2      SUB LAMBLI

3      I3.1
4      I4.1
5      SUB LAMBLI

```

*;Unterprogramm „LAMBLI“ wird aufgerufen und abgearbeitet
;nach erkennen des END-Befehls im Unterprogramm erfolgt eine Weiterbearbeitung des nächsten Befehles im aufrufenden Programm (hier I3.1)*

```

6      I5.1
7      END

```

*;Unterprogramm „LAMBLI“ wird aufgerufen und abgearbeitet,
;nach erkennen des END-Befehls im Unterprogramm erfolgt eine Weiterbearbeitung des nächsten Befehles im aufrufenden Programm (hier I5.1)*

Programm: LAMBLI

```

I1     O1:=1
2      T100
3      O1:=0
4      END

```

;Ausgang 1 wird für 1s gesetzt, dann zurückgesetzt

Beispiele für Verschachtelungen:

Programm : Beispiel

```

1      I2.1
2      SUB LAMBLI

3      I5.1
4      END

```

Programm : LAMBLI

```

1      O1:=1
2      T100
3      O1:=0
4      END

```

→→→→
←←←←

Programm: Beispiel

```

1      I2.1
2      SUB TEST

3      I5.1
4      END

```

Programm: TEST

```

1      T10
2      SUB LAMBLI

3      I6.1
4      END

```

→→→→
←←←←

Programm: LAMBLI

```

1      O1:=1
2      T100
3      O1:=0
4      END

```

1.6.3.3 SUB.S0 / SUB.Sn - Unterprogrammaufruf, Programmname steht in einer Zeichenkette

SUB.Sx

Beschreibung:

Befehlsform:

SUB.S0[.Programmtype]

SUB.Sn[.Programmtype]

SUB.S0 / SUB.Sn / SUB.S0.<Programmtype> / SUB.Sn.<Programmtype>

Der Programmname des aufzurufenden Programms steht in einer Zeichenkette. Ist das Programm nicht vorhanden, so wird die Fehlermeldung „E513 – Programm nicht vorhanden“ erzeugt.

Die Angabe zum Programmtype (PNC, PAB oder PNx) ist eine Option des Befehls. Wird kein Programmtyp angegeben, so wird das Unterprogramm ohne Überprüfung auf den Programmtyp aufgerufen.

Bei angegebenem Programmtyp, z.B. SUB.S0.PNX, wird geprüft ob das Programm vorhanden ist und dem geforderten Typ entspricht. Bei fehlender Übereinstimmung wird die Fehlermeldung „E513 – Programm nicht vorhanden“ erzeugt.

Beispiel

Inhalt S0-Zeichenkette im ASCII-Format: **BLINK**

1	\$A	
2	N2:=PROGSTAT.S0	Statusabfrage, Programmname aus Zeichenkette S0
3	T1	
4	M1:=N2>0	Programm vorhanden
5	G21 M1.0 A	
6	SUB.S0	Aufruf des Programms, dessen Namen in der S0-Zeichenkette steht
7	JMP A	
8	END	

Programm: **BLINK**

11	\$ANFANG	
2	O10:=1	Ausgang 10 setzen
3	T10	Warte 100ms
4	O10:=0	Ausgang 10 zurücksetzen
5	T10	Warte 100ms
6	END	

HINWEIS Bei der Ausführung der Befehle **SUB.S0** und **SUB.Sn** wird der Inhalt der Zeichenkette bis zu einer Länge von maximal 20 Zeichen oder bis zum ersten Leerzeichen übernommen!

1.6.3.4 SUB.Nn - Unterprogrammaufruf, Programmname steht in einem Ganzzahlregister

SUB.Nn

Befehlsform:

SUB.Nn[.Programmtype]

SUB.Nn.i[.Programmtype]

SUB.Nn / SUB.Nn.i / SUB.Nn.<Programmtype> / SUB.Nn.i<Programmtype>

Der Inhalt des angegebenen Ganzzahlregisters wird in eine Zeichenkette, die aus Zahlen besteht, gewandelt. Diese Zeichenkette ergibt den Programmnamen des gewünschten Unterprogramms. Beim Befehl SUB.Nn.i wird das N-Register in eine Zeichenkette mit „i“ Zeichen gewandelt. Ist die Zahl im N-Register kleiner als die vorgegebene Stellenanzahl, so werden vorlaufende Nullen eingeführt (siehe nachfolgendes Beispiel).

Die Angabe zum Programmtype (PNC, PAB oder PNX) ist eine Option des Befehls. Wird kein Programmtyp angegeben, so wird das Unterprogramm ohne Überprüfung auf den Programmtyp aufgerufen.

Bei angegebenem Programmtyp, z.B. SUB.N24.PNC, wird geprüft ob das Programm vorhanden ist und dem geforderten Typ entspricht. Bei fehlender Übereinstimmung wird die Fehlermeldung „E513 – Programm nicht vorhanden“ erzeugt.

Beispiel

1	\$A	
2	N5:=4567	Lade die Zahl 4567 in das Ganzzahlregister N5
...		
10	SUB.N5	Aufruf des Programms 4567 ohne Kontrolle auf den Programmtyp
...		
20	SUB.N5.PNC	Aufruf des Programms 4567 und Kontrolle auf den Programmtyp PNC
...		
30	SUB.N5.7	Aufruf des Programms 0004567 ohne Kontrolle auf den Programmtyp. Der Inhalt des Registers wird mit drei vorlaufenden Nullen gewandelt, also 7 Zeichen.
...		
40	SUB.N5.10.PNX	Aufruf des Programms 0000004567 und Kontrolle auf den Programmtyp. Der Inhalt des Registers wird mit sechs vorlaufenden Nullen gewandelt, also 10 Zeichen
41	END	

Programm: 4567

11	\$ANFANG	
2	O10:=1	Ausgang 10 setzen
3	T10	Warte 100ms
4	O10:=0	Ausgang 10 zurücksetzen
5	T10	Warte 100ms
6	END	

1.6.3.5 G22 - Bedingter Unterprogrammaufruf

G22

G22 (Bedingung) (Name)

Beschreibung:

Mit der Funktion **G22** wird ein „bedingter Unterprogrammaufruf“ ausgeführt. Die im Operand angegebene Bedingung kann der logische Zustand eines Ein-, Ausgangs Merkers oder Systemmerkers sein. Ist die Bedingung erfüllt, wird der Unterprogrammaufruf entsprechend dem Operator ausgeführt. Sobald das Unterprogramm abgearbeitet ist, oder wenn die Bedingung nicht erfüllt ist, wird mit dem nächsten Befehl des aufrufenden Programmes fortgefahren.

Befehlsform:

G22 I1.1 Name

G22 O2.1 Name

G22 M2.0 Name

G22 M!3.1 Name

G22 SM6.0 Name

Anwendung:

In Abhängigkeit von dem logischen Zustand eines Eingangs, Ausgangs, Merkers oder Systemmerkers wird ein Unterprogramm ausgeführt.

Beispiel

1	G25.A1 G90.A1	<i>;Referenzfahrt A1-Achse, Absolutmaßsystem</i>
2	G22 I1.1 AUSG_2	<i>;falls Eingang 1 bestromt ist, wird das Unterprogramm „AUSG_2“ ausgeführt und danach mit dem nächsten Befehl des aufrufenden Programmes die Abarbeitung fortgeführt</i>
3	G22 M21.1 AUSG_2	<i>;falls Merker 21 gesetzt ist, wird das Unterprogramm „AUSG_2“ ausgeführt und danach mit dem nächsten Befehl des aufrufenden Programmes die Abarbeitung fortgeführt</i>
4	G22 O7.1 AUSG_2	<i>;falls Ausgang 7 gesetzt ist, wird das Unterprogramm „AUSG_2“ ausgeführt und danach mit dem nächsten Befehl des aufrufenden Programmes die Abarbeitung fortgeführt</i>
5	END	

Programm: AUSG_2

1	O2:=1 T100 O2:=0 END	<i>;setze Ausgang 2 für 1s</i>
---	----------------------	--------------------------------

1.6.3.6 CASE.SUB - Unterprogrammverteiler

CASE.SUB.N

CASE.SUB.(Variable)

Beschreibung:

Der CASE.SUB-Befehl ist eine Programmverzweigung in Abhängigkeit von einem Ganzzahlregister. Mit diesem Befehl lassen sich auf einfache Weise Programmverteiler aufbauen. Die PA-CONTROL prüft den Inhalt des Ganzzahlregisters und führt entsprechend dem Wert ein Unterprogrammaufruf durch. Ist der Inhalt des Ganzzahlregisters kleiner 1 oder größer der Anzahl der Elemente der Namentabelle (hier im Beispiel: KLEIN, MITTEL, GROSS), so wird ein Sprung auf die Marke, die hinter ELSE definiert ist, durchgeführt. Ansonsten wird das entsprechende Unterprogramm aufgerufen und abgearbeitet und anschließend mit der Programmzeile nach dem Else-Zweig (im Beispiel: Zeile 8 mit O12:= 0) weitergemacht.

Es gilt folgende Zuordnung:

Wert in Register	Name in Namentabelle
1	1. Name
2	2. Name
....
N	n. Name

HINWEIS Die Unterprogramme müssen vorhanden sein, ansonsten erfolgt die Fehlermeldung „Unterprogramm nicht gefunden“.

Unterprogramme dürfen sich nicht selbst aufrufen.

Befehlsform:

CASE.SUB.Nn

(Name1)

(Name2)

...

ELSE Marke_e

oder mit indirekter Adressierung des Registers

CASE.SUB.N!n

(Name1)

(Name2)

...

ELSE Marke_e

Anwendung:

Programmverzweigung bei mehr als zwei Möglichkeiten, wie verschiedene Fertigungstypen, Varianten von Verfahrenfrequenzen.

Beispiel

```
1   $FEHLER
2   N8:=N13           ;Wert über Register einlesen
3   CASE.SUB.N8      ;prüfe den Inhalt von N8 und Aufruf von Unterprogramm
                        bei:
4   (KLEIN)          ;N8=1: Unterprogramm „KLEIN“ abarbeiten
5   (GROSS)          ;N8=2: Unterprogramm „GROSS“ abarbeiten
6   (MITTEL)         ;N8=3: Unterprogramm „MITTEL“ abarbeiten
7   ELSE FEHLER      ;Sprung zur Marke „FEHLER“, wenn der Wert von N8
                        außerhalb der vorgesehenen Grenzen (hier wenn N8<1
                        oder N8>3) ist

8   O12:=0
9   I1.1
10  END
```

Programm: KLEIN

```
1   O1:=1 T100 O1:=0 ;setze Ausgang 1 für 1 s
2   END
```

Programm: GROSS

```
1   O2:=1 T100 O2:=0 ;setze Ausgang 2 für 1 s
2   END
```

Programm: MITTEL

```
1   O3:=1 T100 O3:=0 ;setze Ausgang 3 für 1 s
2   END
```

1.6.4 Schleifen

1.6.4.1 INC - Schleife mit bedingtem Sprung

INC

INC.(Zähler).(Zielwert) (Marke)

Beschreibung:

Der INC-Befehl dient zum Aufbau von Programmschleifen. Er ist mit dem Pascal-Befehl Repeat...Until vergleichbar.

Der INC-Befehl prüft den Inhalt des Ganzzahlregisters Nn (Beispiel N2). Wenn der Inhalt des Ganzzahlregisters kleiner ist, als der des Vorgabewertes des Ganzzahlregister oder der Zahl, wird der Inhalt (N2) um 1 inkrementiert und ein Sprung zur Marke ausgeführt, ansonsten wird die Bearbeitung mit dem nächstfolgenden Befehl (nächste Zeile) fortgesetzt.

HINWEIS Der INC-Befehl ist nur für Ganzzahlregister anwendbar.

Befehlsform:

INC.Nn.Nn Marke

INC.Nn.n Marke

Anwendung:

Zum Aufbau von Programmschleifen, deren Anzahl von Durchläufen als Zahl oder in einem Ganzzahlregister hinterlegt ist, z.B. n-Teile bearbeiten, n-mal verfahren etc.

Beispiel

1	<i>N3:=5</i>	<i>;Schleifenanzahl ist gleich 5</i>
2	<i>N2:=1</i>	<i>;Schleifenzähler</i>
3	<i>\$ANFANG</i>	<i>;Schleifenanfang</i>
4	<i>O1:=1</i>	
5	<i>T100</i>	
6	<i>O1:=0</i>	
7	<i>T100</i>	
8	<i>INC.N2.N3 ANFANG</i>	<i>;wiederhole bzw. springe an „ANFANG“, solange N2<N3 ist.</i>
Oder:		
8	<i>INC.N2.5 ANFANG</i>	<i>;wiederhole bzw. springe an „ANFANG“, solange N2<5 ist.</i>
9	<i>END</i>	

1.6.4.2 DEC - Schleife mit bedingtem Sprung

DEC

DEC.(Zähler) (Marke)

Beschreibung:

Der DEC-Befehl dient zum Aufbau von Programmschleifen. Er ist mit dem Pascal-Befehl „Repeat...until“ oder dem C-Befehl „Do... While“ vergleichbar.

Der DEC-Befehl prüft den Inhalt des Ganzzahlregisters. Wenn der Inhalt des Ganzzahlregisters größer 0 ist, wird der Inhalt um 1 dekrementiert und ein Sprung zur Marke ausgeführt, ansonsten wird die Bearbeitung mit dem nächstfolgenden Befehl (nächste Zeile) fortgesetzt.

HINWEIS Der DEC-Befehl ist nur für Ganzzahlregister anwendbar.

Befehlsform:

DEC.Nn Marke

Anwendung:

Zum Aufbau von Programmschleifen, deren Anzahl von Durchläufen in einem Ganzzahlregister hinterlegt ist, z.B. n-Teile bearbeiten, n-mal verfahren etc.

Beispiele für Schleifen mit dem DEC-Befehl:

Beispiel 1

```
1      N2:=8                      ;Schleifenanzahl
2      $ANFANG
3      O1:=1
4      T100
5      O1:=0
6      T100
7      DEC.N2 ANFANG              ;alle Wiederholungen getätigt? Nein, dann sprin-
                                   ge nach"ANFANG"
8      END
```

Beispiel 2

Der Schleifenprogrammteil wird mindestens einmal durchlaufen (Repeat...Until, Do...While), d.h. der Programmteil wird auf jeden Fall bearbeitet und anschließend wird geprüft (DEC.), ob der Programmteil noch einmal bearbeitet werden muss.

```
1      N3:=10
2      $MACHWAS
3      O1:=1
4      T100
5      O1:=0
6      T100
7      DEC.N3 MACHWAS
8      END
```

Beispiel 3

Der Schleifenprogrammteil wird je nach Wert nicht durchlaufen (While...), d.h. es wird zuerst das Register überprüft (DEC.N3) und dann je nach Inhalt des Registers der Programmteil bearbeitet.

```
1      N3:=15
2      $SCHLEIFE
3      DEC.N3 MACHWAS
4      JMP FERTIG
5      $MACHWAS
6      O1:=1
7      T100
8      O1:=0
9      T100
10     JMP SCHLEIFE
11     $FERTIG
12     END
```

1.6.5 Parallele Abläufe

1.6.5.1 RUN - Starten eines Parallelablaufes

RUN

Beschreibung:

Mit dem Befehl RUN können weitere parallele Tasks (=Abläufe) gestartet werden.

Bei der Ausführung des Befehls RUN wird der Programmtyp (*.PNC/ *.PNX / *.PAB) geprüft und der parallele Ablauf entsprechend gestartet.

Der Unterschied zwischen dem Ablauf eines PNC/PNX-Programms und dem eines PAB-Programms zeigt sich nur, wenn im AUTOMATIK-Betrieb gestoppt wird.

Für die Situation STOP-Automatik gilt:

Für PNC/PNX-Ablauf (Task):

- Achsen werden angehalten,
- Programme werden angehalten (wie SLEEP) und nicht weiter bearbeitet.

Für PAB-Ablauf (Task):

- Programme werden weiterbearbeitet,
- wird gerade ein Verfahrbefehl bearbeitet, oder soll ein Verfahrbefehl aktiviert werden, so bleibt der Ablauf an diesem Befehl stehen.

Diese unterschiedliche Arbeitsweise der beiden Programmabläufe stellt einen Vorteil dar. Ist die PA-CONTROL gestoppt, kann über die PAB-Abläufe eine Ausgabe auf das Display, eine Kommunikation über die serielle Schnittstelle, oder /und ein Blinken von Warnlampen realisiert werden.

HINWEIS Das Programm wird so lange bearbeitet, bis es durch einen END-Befehl oder CANCEL beendet wird..

HINWEIS Ein Programm kann nur einmal in den Ablaufinterpreter eingetragen werden. Es können bis zu 47 Programme (Tasks) gleichzeitig im Ablaufinterpreter bearbeitet werden.

Befehlsform:

RUN *NAME*
RUN.Sn (ab V5.24)
RUN.Sn.*Programmtyp* (ab V5.24)

Anwendung:

Starten von parallelen Ablaufprogrammen.

Beispiel 1

PNC-Programm

1	M1:=0	;Merker 1 zurücksetzen
2	M2:=0	;Merker 2 zurücksetzen
3	RUN BLINKEN	;starte das PAB-Programm „BLINKEN“
4	RUN TUER_AUF	;starte das Programm „TUER_AUF“
5	M1.1	;warte bis der Merker 1 auf logisch „1“ gesetzt wurde
6	M2.1	;warte bis der Merker 2 auf logisch „1“ gesetzt wurde
7	O3:=1	;setze den Ausgang 1
8	END	

PAB-Programm „TUER_AUF.PAB“

1	O30:=1	;Setze den Ausgang 30, Tür öffnen
2	I30.1	;Tür ist offen
3	O30:=0	;Rücksetzen von Ausgang 30
4	M2:=1	;Setze den Merker 2, Tür geöffnet
5	END	

PAB-Programm „BLINKEN.PAB“

1	\$B_ZYKLUS	
2	O2:=1	;setze den Ausgang 2
3	T20	;Wartezeit von 200ms
4	O2:=0	;Rücksetzen von Ausgang 2
5	M1:=1	;setze den Merker 1
6	JMP B_ZYKLUS	; Sprung zur Marke B_ZYKLUS
7	END	

Während das PAB-Programm „TUER_AUF“ nach dem Öffnen beendet wird und damit aus dem Parallelablaufinterpreter gelöscht wird, arbeitet das PAB-Programm „BLINKEN“ so lange, bis es in einem anderen Ablauf mit dem Befehl „CANCEL“ beendet wird.

1.6.5.2 SLEEP - Anhalten eines Parallelablaufes

SLEEP

Beschreibung:

Mit dem Befehl SLEEP können Ablaufprogramme angehalten werden. Das Programm wird durch den SLEEP-Befehl der PA-CONTROL in einen Schlafzustand (SLEEP-Modus) versetzt, d.h. es wird nicht weiter bearbeitet, es bleibt auf dem aktuellen Befehl stehen. Mit einem RUN-Befehl kann dieses Programm später ab dem aktuellen Befehl fortgeführt werden.

HINWEIS Der SLEEP-Befehl kann nur auf Programme angewendet werden, die sich im RUN-Modus befinden.

Befehlsform:

SLEEP *Name*
 SLEEP.Sn (ab V5.24)
 SLEEP.Sn.*Programmtyp* (ab V5.24)

Anwendung:

Anhalten von parallelen Ablaufprogrammen, z.B. wenn eine Sicherheitstür geöffnet wurde.

Beispiel

PNC-Programm:

```

1      M1:=0           ;Merker 1 zurücksetzen
2      M2:=0           ;Merker 2 zurücksetzen
3      RUN AUS1        ;starte das Programm „AUS1“
4      M1.1            ;warte bis der Merker 1 auf logisch „1“ gesetzt
                       ;wurde
5      SLEEP AUS1      ;halte das Programm „AUS1“ an
6      M1:=0           ;Merker 1 zurücksetzen
7      I1.1            ;wartet bis der Eingang 1 auf logisch „1“ gesetzt
                       ;ist
8      RUN AUS1        ;fortführen des Programms „AUS1“
9      M2.1            ;warte bis der Merker 2 auf logisch „1“ gesetzt
                       ;wurde
10     O3:=1           ;setze den Ausgang 1
11     END

```

PAB-Programm: „AUS1.PAB“

```

1      O1:=1           ;setze den Ausgang 1
2      M1:=1           ;setze den Merker 1
3      M1.0            ;warte bis der Merker 1 auf logisch „0“ gesetzt
                       ;wurde
4      T10             ;Wartezeit von 100ms
5      O1:=0           ;Ausgang 1 zurücksetzen
6      M2:=1           ;setze den Merker 2
7      END             ;Programm „AUS1“ beenden

```

1.6.5.3 CANCEL - Beenden eines Parallelablaufes

CANCEL

Beschreibung:

Mit dem Befehl CANCEL können Ablaufprogramme beendet werden.

HINWEIS Der CANCEL-Befehl kann auch auf Programme angewendet werden, die sich nicht im RUN-Modus oder SLEEP-Modus befinden.

Das Programm wird bei dem momentan bearbeiteten Befehl abgebrochen, d.h. es sind unter Umständen noch von ihm beeinflusste Elemente zu versorgen (Merker, Ausgänge, Register).

HINWEIS Ist das im CANCEL-Befehl angesprochene Programm nicht in der Steuerung vorhanden, bleibt die Steuerung mit dem Fehler E513, Programm nicht vorhanden, stehen.

Befehlsform:

CANCEL *Name*
CANCEL.Sn (ab V5.24)
CANCEL.Sn.*Programmtype* (ab V5.24)

Anwendung:

Sofortiges Beenden von Ablaufprogrammen. Dadurch ist im Ablaufinterpreter Platz für ein neu zu startendes Programm, z.B. bei Betriebsartwechsel einer Fertigungsanlage.

Beispiel

```
1 M1:=0 ;Merker 1 zurücksetzen
2 M2:=0 ;Merker 2 zurücksetzen
3 RUN AUS1 ;starte das Programm „AUS1“
4 I7.1 ;warte bis der Eingang 7 bestromt ist
5 CANCEL AUS1 ;beende das Programm „AUS1“
6 END
```

Programm: AUS1.PAB

```
1 $ANFANG
2 O1:=1 ;setze den Ausgang 1
3 T10 ;Wartezeit von 100ms
4 O1:=0 ;Ausgang 1 zurücksetzen
5 T10 ;Wartezeit von 100ms
6 JMP ANFANG ;Sprung zur Marke „ANFANG“
7 END
```

1.6.5.4 CASE.RUN - Starten von Parallelläufen mit CASE

CASE.RUN

Beschreibung:

Der CASE.RUN-Befehl startet Parallelläufe in Abhängigkeit von einem Ganzzahlregister. Mit diesem Befehl lassen sich auf einfache Weise Programmverteiler aufbauen.

Die PA-CONTROL prüft den Inhalt des Ganzzahlregisters und startet entsprechend dem Wert ein Parallellauf. Ist der Inhalt des Ganzzahlregisters kleiner 1 oder größer der Anzahl der Elemente der Namentabelle (hier im Beispiel: PROG-1, PROG-2, PROG-3), so wird ein Sprung auf die Marke, die hinter ELSE definiert ist, durchgeführt. Ansonsten wird das entsprechende Programm aufgerufen und anschließend mit der Programmzeile nach dem ELSE-Zweig (im Beispiel: Zeile 8 mit O12:= 0) weitergemacht.

Es gilt folgende Zuordnung:

<u>Wert im Register</u>	<u>Name in der Namentabelle</u>
1	1. Name
2	2. Name
...	...
n	n. Name

HINWEIS Die Programme müssen vorhanden sein, ansonsten erfolgt die Ausgabe der Fehlermeldung „Programm nicht vorhanden“.
Ist ein Parallellauf bereits gestartet, darf er nicht noch einmal gestartet werden, ansonsten erfolgt die Fehlermeldung „Programm läuft schon“.

Befehlsform:

CASE.RUN.Ni

(Name1)

(Name2)

...

ELSE Marke_e

Anwendung:

Abhängig von z. B. verschiedenen Fertigungstypen können verschiedene Bearbeitungsabläufe auftreten, welche parallel zu anderen Vorgängen stattfinden. Diese können mit CASE.RUN variabel ausgelöst werden.

Beispiel

1	G11.0	;Ablaufanzeige ausschalten
2	G500.0	;LC-Anzeige ist aktueller Datenkanal
3	G501	;Anzeige löschen
4	\$FEHLER	;
5	G503.1.1	;Cursor positionieren
6	G510.EINGABE PROG.-NUMMER	;Ausgabe der Bedieneranforderung
7	G542.24.1.2.N8	;Wert über Tastatur eingeben
8	CASE.RUN.N8	;prüfe den Inhalt von N8 und starte das PAB-Programm bei:
9	(PROG-1)	;N8=1: starte das Programm PROG-1
10	(PROG-2)	;N8=2: starte das Programm PROG-2
11	(PROG-3)	;N8=3: starte das Programm PROG-3
12	ELSE FEHLER	;Sprung zur Marke „FEHLER“, wenn der Wert von N8 außerhalb ;der vorgesehenen Grenzen (hier wenn N8<1 oder N8>3) ist
13	O12:=0	
14	I1.1	
15	END	

Programm: PROG-1.PAB

1	O1:=1 T100 O1:=0	;setze Ausgang 1 für 1 s
2	END	

Programm: PROG-2.PAB

1	O2:=1	;setze Ausgang 2
2	I1.1	;Warte auf Eingang 1
3	O2:=0	;setze Ausgang 2 zurück
4	END	

Programm: PROG-3.PAB

1	O3:=1 T100 O3:=0	;setze Ausgang 3 für 1 s
2	END	

1.6.5.5 CASE.SLEEP - Anhalten von Parallelabläufen mit Case

CASE.SLEEP

Beschreibung:

Mit dem CASE.SLEEP-Befehl können Parallelabläufe in Abhängigkeit von einem Ganzzahlregister angehalten werden. Mit diesem Befehl lassen sich auf einfache Weise Programmverteiler aufbauen.

Die PA-CONTROL prüft den Inhalt des Ganzzahlregisters und startet entsprechend dem Wert ein Parallelablauf. Ist der Inhalt des Ganzzahlregisters kleiner 1 oder größer der Anzahl der Elemente der Namentabelle (hier im Beispiel: PROG-1, PROG-2), so wird ein Sprung auf die Marke, die hinter ELSE definiert ist, durchgeführt. Ansonsten wird das entsprechende Programm aufgerufen und anschließend mit der Programmzeile nach dem ELSE-Zweig fortgesetzt.

Es gilt folgende Zuordnung:

<u>Wert im Register</u>	<u>Name in der Namentabelle</u>
1	1. Name
2	2. Name
...	...
N	n. Name

HINWEIS Die Programme müssen vorhanden sein, ansonsten erfolgt die Fehlermeldung „Programm nicht vorhanden“.

Ist ein Parallelablauf bereits angehalten, darf er nicht noch einmal angehalten werden, ansonsten erfolgt die Fehlermeldung „Programm schon in Sleep“.

Befehlsform:

CASE.SLEEP.Ni

(Name1)

(Name2)

...

ELSE Marke_e

Anwendung:

Abhängig von z. B. verschiedenen Fertigungstypen können verschiedene Bearbeitungsabläufe auftreten, welche parallel zu anderen Vorgängen stattfinden. Diese können mit CASE.SLEEP variabel angehalten werden.

Beispiel

1	RUN PROG-1	;starten des Programms PROG-1
2	RUN PROG-2	;starten des Programms PROG-2
3	\$ANFANG	
..	...	
19	G11.0	;Ablaufanzeige ausschalten
20	G500.0	;LC-Anzeige ist aktueller Datenkanal
21	G501	;Anzeige löschen
22	\$FEHLER	;
23	G503.1.1	;Cursor positionieren
24	G510.EINGABE PROG.-NUMMER	;Ausgabe der Bedieneranforderung
25	G542.24.1.2.N10	;Wert über Tastatur eingeben
26	CASE.SLEEP.N10	;prüft den Inhalt von N10 und hält das entsprechende PAB-Programm bei N10 an:
27	(PROG-1)	;N10=1: hält das Programm PROG-1.PAB an
28	(PROG-2)	;N10=2: hält das Programm PROG-2.PAB an
29	ELSE FEHLER	;Sprung zur Marke „FEHLER“, wenn der Wert von N10 außerhalb der vorgesehenen Grenzen (hier wenn N10<1 und N10>2 ist.
..	...	
100	END	

Programm: PROG-1.PNC

1	\$ANFANG	;Sprungmarke
2	O1:=1	;setze Ausgang 1
3	I1.1	;warte bis Eingang 1 bestromt
4	O1:=0	;setze Ausgang 1 zurück
5	T100	;warte 1s
6	JMP ANFANG	;gehe zur Marke ANFANG
7	END	

Programm: PROG-2.PAB

1	\$ANFANG	;Sprungmarke
2	O2:=1	;setze Ausgang 2
3	I2.1	;warte bis Eingang 2 bestromt
4	O2:=0	;setze Ausgang 2 zurück
5	T50	;warte 0,5s
6	JMP ANFANG	;gehe zur Marke ANFANG
7	END	

1.6.5.6 CASE.CANCEL - Beenden von Parallelabläufen mit Case

CASE.CANCEL

Beschreibung:

Mit dem CASE.CANCEL-Befehl können Parallelabläufe in Abhängigkeit von einem Ganzzahlregister beendet werden.

Die PA-CONTROL prüft den Inhalt des Ganzzahlregisters und beendet auf Grund des im Ganzzahlregister abgelegten Wertes einen Parallelablauf.

Ist der Inhalt des Ganzzahlregisters kleiner 1 oder größer als die Anzahl der Elemente der Namentabelle (hier im Beispiel: PROG-1, PROG-2), so wird ein Sprung auf die Marke, die nach ELSE definiert ist, durchgeführt. Nach fehlerfreier Ausführung des CANCEL-Befehls wird in der Zeile weitergearbeitet, die auf ELSE *Marke_e* folgt.

HINWEIS Der Befehl CASE.CANCEL kann auch auf Programme erfolgen, die nicht laufen (siehe dazu auch Abschnitt 1.6.5.3 auf Seite 76).

HINWEIS Ist das im CASE.CANCEL-Befehl angesprochene Programm nicht in der Steuerung vorhanden, bleibt die Steuerung mit dem Fehler E513, Programm nicht vorhanden, stehen.

Es gilt folgende Zuordnung:

<u>Wert im Register</u>	<u>Name in der Namentabelle</u>
1	1. Name
2	2. Name
...	...
n	n. Name

Befehlsform:

CASE.CANCEL.Ni

(Name1)

(Name2)

...

ELSE Marke_e

Anwendung:

Abhängig von z. B. unterschiedlichen Fertigungstypen können verschiedene Bearbeitungsabläufe auftreten, die parallel stattfinden oder ruhen. Mit CASE.CANCEL besteht die Möglichkeit, nicht gewünschte Parallelabläufe zu beenden.

Beispiel

1	<i>RUN PROG-1</i>	<i>;starten des Programms PROG-1</i>
2	<i>RUN PROG-2</i>	<i>;starten des Programms PROG-2</i>
3	<i>\$ANFANG</i>	
..	...	
19	<i>G11.0</i>	<i>;Ablaufanzeige ausschalten</i>
20	<i>G500.0</i>	<i>;LC-Anzeige ist aktueller Datenkanal</i>
21	<i>G501</i>	<i>;Anzeige löschen</i>
22	<i>\$FEHLER</i>	<i>;</i>
23	<i>G503.1.1</i>	<i>;Cursor positionieren</i>
24	<i>G510.EINGABE PROG.-NUMMER</i>	<i>;Ausgabe der Bedieneranforderung</i>
25	<i>G542.24.1.2.N20</i>	<i>;Wert über Tastatur eingeben</i>
26	<i>CASE.CANCEL.N20</i>	<i>;prüft den Inhalt von N20 und hält das entsprechende PAB-Programm bei N20 an:</i>
27	<i>(PROG-1)</i>	<i>;N20=1: hält das Programm PROG-1.PAB an</i>
28	<i>(PROG-2)</i>	<i>;N20=2: hält das Programm PROG-2.PAB an</i>
29	<i>ELSE FEHLER</i>	<i>;Sprung zur Marke „FEHLER“, wenn der Wert von N10 außerhalb der vorgesehenen Grenzen (hier wenn N10<1 und N10>2 ist.</i>
..	...	
100	<i>END</i>	

Programm: PROG-1.PNC

1	<i>\$ANFANG</i>	<i>;Sprungmarke</i>
2	<i>O1:=1</i>	<i>;setze Ausgang 1</i>
3	<i>I1.1</i>	<i>;warte bis Eingang 1 bestromt</i>
4	<i>O1:=0</i>	<i>;setze Ausgang 1 zurück</i>
5	<i>I2.1</i>	<i>;warte bis Eingang 2 bestromt</i>
6	<i>JMP ANFANG</i>	<i>;gehe zur Marke ANFANG</i>
7	<i>END</i>	

Programm: PROG-2.PAB

1	<i>\$ANFANG</i>	<i>;Sprungmarke</i>
2	<i>O2:=1</i>	<i>;setze Ausgang 2</i>
3	<i>T50</i>	<i>;warte 0,5s</i>
4	<i>O2:=0</i>	<i>;rücksetze Ausgang 2</i>
5	<i>T50</i>	<i>;warte 0,5s</i>
6	<i>JMP ANFANG</i>	<i>;gehe zur Marke ANFANG</i>
7	<i>END</i>	

1.6.5.7 PROGSTAT - Hole Programmstatus

Ni:=PROGSTAT

Ni:=PROGSTAT.xxx

Beschreibung:

Der Befehl überprüft, ob das Programm als Task (mit RUN oder CASE.RUN) aktiviert wurde. Ist das Programm nicht als laufende Ausgabe im Ablaufinterpreter zu finden wird zusätzlich geprüft, ob das Programm in der PA-COPNTROL überhaupt vorhanden ist. Im Register Ni wird nach der Prüfung folgende Information abgespeichert:

Inhalt von Ni		Status
0	→	Programm in der PA-CONTROL nicht vorhanden
1	→	Programm in der PA-CONTROL vorhanden, aber nicht als TASK aktiv
2	→	Als Task vorhanden, wird bearbeitet
3	→	Als Task vorhanden, wird nicht bearbeitet (SLEEP)

Befehlsform:

Ni:= PROGSTAT.Name
 N!i:=PROGSTAT.Name
 Ni:= PROGSTAT.S0
 Ni:= PROGSTAT.Sn
 Ni:= PROGSTAT.Nn
 Ni:= PROGSTAT.Nn.i

Beispiel

```

1      N10:=PROGSTAT.TUER_AUF      Ist das Programm „TUER_AUF“ aktiv, oder in der
                                     PA-CONTROL vorhanden. Abspeicherung des Er-
                                     gebnisses im Ganzzahlregister N10.

2      $A

3      N5:=128                      Lade die Zahl 128 in das Ganzzahlregister N5

...

10     N10:=PROGSTAT.S0             Erfrage den Status des Programms, dessen Name
                                     in der Zeichenkette S0 steht

...

20     N10:=PROGSTAT.S1             Erfrage den Status des Programms, dessen Name
                                     in der Zeichenkette S1 steht

...

30     N10:=PROGSTAT.N10           Erfrage den Status des Programms, dessen Name
                                     im Ganzzahlregister N10 steht

...

40     N10:=PROGSTAT.N5.7          Erfrage den Status des Programms 0000128. Der
                                     Befehl PROGSTAT fügt der dreistelligen Zahl aus
                                     dem Ganzzahlregister N5 noch vier Vornullen hinzu

41     END
  
```

1.6.6 Fehlermanagement

1.6.6.1 ERROROFF / ERRORON

ERROROFF / ERRORON

ab V5.07

Beschreibung:

Mit dem Befehl „ERROROFF“ und „ERRORON“ wird die Überwachung eines Zustandes der zu einem Systemfehler führt deaktiviert oder aktiviert.

HINWEIS Grundsätzlich (Defaultzustand) ist die Überwachung der PA-CONTROL aktiviert.

Befehlsform:

ERROROFF.<Fehlernummer>.<Achsnnummer>

ERRORON.<Fehlernummer>.<Achsnnummer>

Anwendungsbeispiel für eine PA-CONTROL MP

Bei einer PA-CONTROL MP wird eine Motorspannung von 24VDC von extern zugeführt. Die interne Überwachung dieser Spannung verlangt aber mindestens 60VDC.

Das Flag „ÜberwacheMotorSpannung“ für die PA-CONTROL-MP wird beim Einschalten (und nur dann!) der PA-CONTROL aktiviert.

Über den Befehl „ERROROFF.103.A1“ kann die Überwachung deaktiviert werden. Dies kann im „Grundstellungsprogramm“ erfolgen.

Für den Freigabemodus wäre die Einstellung „Bei Wechsel in Verfahrenbetriebsart“ die richtige

Liste der möglichen Systemfehler mit den jeweiligen Achstypen	
Fehlernummer	Achstypen
E103	PLS-MP

(siehe dazu folgendes Beispiel)

Beispiel

Grundstellung.pnc

```
1 ERROROFF.103.A1
2 T10
3 END
```

Deaktiviere Überwachung Fehler E103
Warte 100ms

Start.pnc

```
1 R1:=RA1.1
2 R4:=RA1.6
3 $A
4 G21 I9.1 REFG26
5 G21 I10.1 REFG25
6 G21 I7.1 FAHRNEGATIV
7 G21 I8.1 FAHRPOSITIV
8 G21 I6.1 FAHRE
9 JMP A
10 $REFG26
11 G26.A1.R1
12 I9.0
13 JMP A
... ;
14 $REFG25
15 G25.A1
16 I10.0
17 JMP A
18 $FAHRNEGATIV
19 A1:=0
20 I7.0
21 JMP A
... ;
22 $FAHRPOSITIV
23 G123.A1.I8.1
24 A1:=R2
25 I8.0
26 JMP A
... ;
27 $FAHRE
28 FA1:=R1
29 G100.A1.R4
30 A1:=100
31 A1:=0
32 JMP A
... ;
33 END
```

1.7 Achsbefehle

1.7.1 Allgemeines

1.7.1.1 Die Zustände einer Achse

Jede der möglichen 16 Achsen einer PA-CONTROL kann sich in unterschiedlichen Zuständen befinden (siehe *Abbildung 1, Seite 86, Erläuterungen siehe Tab.: 1, Seite 87*)

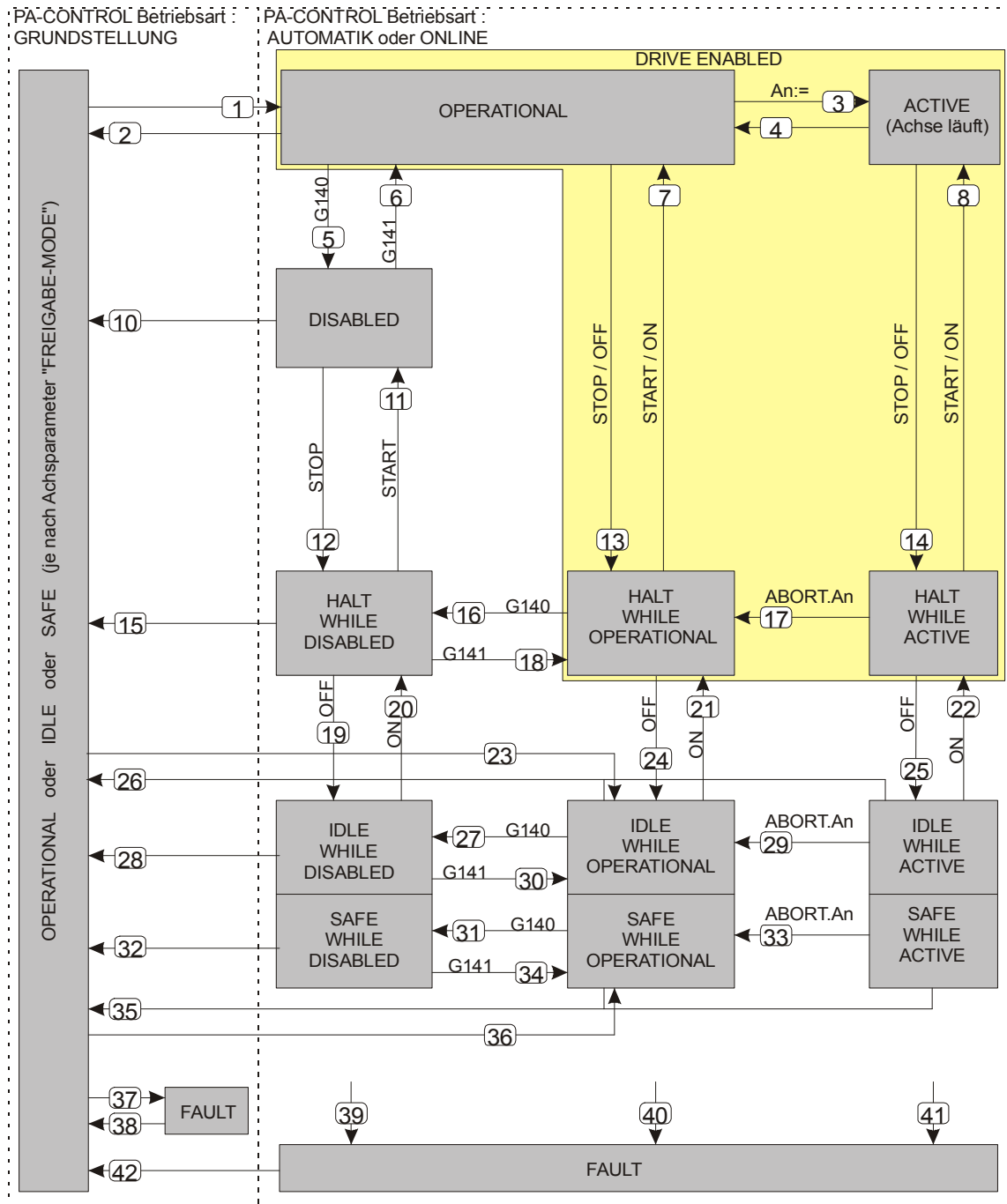


BILD 721D

Abbildung 1: Zustände einer Achse

Begriffserklärung zu *Abbildung 1: Zustände einer Achse*

Zustand	Beschreibung
OPERATIONAL (1), (4)	Der Motor ist bestromt und steht in Position.
ACTIVE (3), (8)	Der Motor dreht, die Achse fährt.
HALT WHILE OPERATIONAL (13), (17)	Der Motor ist bestromt und steht in Position.
HALT WHILE ACTIVE (14), (22)	Der Motor ist bestromt und steht. Ein Verfahrtauftrag wurde angehalten.
DRIVE ENABLED (1), (6), (18), (21), (22)	Der Motor ist bestromt. Die Achse ist entweder im Status OPERATIONAL, ACTIVE oder HALT.
IDLE WHILE OPERATIONAL (24), (29), (39)	Der Motor ist stromlos geschaltet, (DRIVE DISABLED).
IDLE WHILE ACTIVE (25), (26)	Ein Verfahrtauftrag der Achse wurde angehalten und danach der Motor stromlos geschaltet (DRIVE DISABLED).
SAFE WHILE OPERATIONAL (33), (34), (36)	Der Motor ist stromlos geschaltet. (DRIVE DISABLED). Über den Ausgang „PUT-SAFE“ wurde die Funktion „Sicherer Halt“ mit Anlaufsperrung auf dem Verstärker aktiviert.
SAFE WHILE ACTIVE	Ein Verfahrtauftrag der Achse wurde angehalten und danach der Motor stromlos geschaltet, (DRIVE DISABLED). Über den Ausgang „PUT-SAFE“ wurde die Funktion „Sicherer Halt“ mit Anlaufsperrung auf dem Verstärker aktiviert.
DISABLED (5), (11)	Der Motor ist stromlos geschaltet, (DRIVE DISABLED).
HALT WHILE DISABLED (12), (16), (20)	Der Motor ist stromlos geschaltet, (DRIVE DISABLED).
IDLE WHILE DISABLED (19), (27)	Der Motor ist stromlos geschaltet, (DRIVE DISABLED).
SAFE WHILE DISABLED (31)	Der Motor ist stromlos geschaltet, (DRIVE DISABLED). Über den Ausgang „PUT-SAFE“ wurde die Funktion „Sicherer Halt“ mit Anlaufsperrung auf dem Verstärker aktiviert.
FAULT (37), [(39), (40), (41)]	Die Achse oder der Antrieb hat einen Fehler.

Tab.: 1 Erläuterungen zu den Zuständen einer Achse

Zusätzliche Informationen entnehmen Sie bitte den Kapiteln:

1.7.2.1, G140 -, Seite 88,

1.7.2.2, G141 -, Seite 89,

1.7.2.3, OFF.An - Achse abschalten, Seite 90,

1.7.2.4, ON.An - Achse einschalten, Seite 91,

1.7.2.5, STOP.An - Unterbreche die Verfahrbereitschaft einer Achse, Seite 92,

1.7.2.6, START.An - Herstellen der Verfahrbereitschaft einer Achse, Seite 92,

1.7.2.7, ABORT.An - Abbruch eines gestoppten Fahrbefehls, Seite 93.

1.7.2 Modi

1.7.2.1 G140 - wechsele in den Zustand DISABLED

G140

Beschreibung:

Bei Ausführung des Befehls wechselt die Achse in den Zustand DISABLED. Die frühere Bedeutung des Befehles war die Aktivierung des Messmodus.

Funktion:

Der Befehl G140.An (A0, A[1..16]) aktiviert die Funktion "OFF-ACHSE". Die Achse wechselt aber nicht nach HALT und dann nach IDLE oder SAFE, sondern nach DISABLED ((5)).

HINWEIS

- Ist die Achse im Zustand ACTIVE, so wird der Ablauffehler „E523 - Achse laeuft noch“ gesetzt.
- Ist die Achse im Zustand HALT WHILE ACTIVE, so wird der Ablauffehler „E584 – Achse ist im Status HALT“ gesetzt.
- Ist die Achse im Zustand DISABLED, HALT WHILE DISABLED, IDLE WHILE DISABLED oder SAFE WHILE DISABLED und es wird ein Verfahrbefehl auf diese Achse aktiviert, so wird der Ablauffehler „E571 – Befehl mit dieser Achse nicht möglich“ erzeugt.

Aktionen beim Aktivieren:

- Bremse wird aktiviert (Option),
- Stromnullung in der Endstufe wird aktiviert,
- Überwachung für Drehgeber und Leistungsteilbereitschaft wird deaktiviert.

Ergebnis:

- Endstufe kann ausgeschaltet werden (Arbeitssicherheit),
- Achse kann manuell verschoben werden,
- Die Achse kann nicht mehr von der Steuerung verfahren werden (Positionierbefehle sind gesperrt und erzeugen eine Fehlermeldung).

Befehlsform:

G140.An [A1....A16]

Beispiel

siehe nächsten Befehl

HINWEIS Nicht bei PA-CONTROL mit PLS-6/ -7/ -8/ -9 sinnvoll.

1.7.2.2 G141 - wechsele in den Zustand OPERATIONAL

G141

Beschreibung:

Bei Ausführung des Befehls wechselt die Achse in den Zustand OPERATIONAL. Die frühere Bedeutung des Befehles war die Deaktivierung des Messmodus.

Aktionen bei PA-CONTROL MP:

- Motor wird wieder bestromt,
- Bremse deaktiviert (Option),
- Synchronisation auf Drehgeber wird durchgeführt (Option),
- Achsposition wird aus der Drehgeberposition übernommen (Option),
- Überwachung für Drehgeber und Leistungsteilbereitschaft wird aktiviert.

Aktionen bei LV-servoTEC:

- Regler freigeben,
- Werte für Drehgeberposition übernehmen

Ergebnis:

Die Achse kann wieder von der Steuerung verfahren werden.

HINWEIS Bei einer PA-CONTROL-MP und beim Verlassen des Zustandes DISABLED und Synchronisation auf den Drehgeber wird die Achse um maximal 4 Motorvollschritte in positive Richtung verfahren.

Befehlsform:

G141.An [A1...A16]

HINWEIS Nicht bei PA-CONTROL mit PLS-6/ -7/ -8/ -9 sinnvoll

Beispiel

1	G25.A1	;Referenzfahrt mit A1-Achse durchführen
2	A1:=100	;fahre auf Position 100
3	I5.1	;warte bis Einrichtbetrieb aktiv, Schutzkreis gebrückt
4	G140.A1	;A1-Achse in den Zustand DISABLED umschalten, Achse kann manuell bewegt werden.
5	I5.0	;Schutzkreis wieder aktiv, Einrichtbetrieb wird gesperrt
6	G141.A1	;Achse A1 führt Aktionen wie oben beschrieben aus. aus
7	A1:=1000	;mit der A1-Achse auf Position 1000 fahren
8	END	

1.7.2.3 OFF.An - Achse abschalten

OFF.An

ab V5.07

Beschreibung:

(siehe hierzu auch Applikationsschrift „APP5004_DE_1070300_PAC_OeffnenDerSchutztuer_servoTEC.pdf“)

Der Befehl OFF.An (A[0...16]) aktiviert die Funktion "OFF-ACHSE" für eine Achse.

Funktion:

Ausgehend von unterschiedlichen Ausgangssituationen müssen drei verschiedene Auswirkungen des Befehls unterschieden werden.

- Ist die Achse im Zustand OPERATIONAL wechselt sie in den Zustand HALT WHILE OPERATIONAL (siehe *Abbildung 1, Zustände einer Achse auf Seite 86 Pfad (13)*). Danach wird die Funktion "OFF-Achse" aktiv und die Achse in den Status "DISABLED / SAFE" geschaltet (siehe *Abbildung 1, Zustände einer Achse auf Seite 86 Pfad (24)*).
- Wenn die Achse fährt, wird die Positionierung sofort angehalten und die Achse wechselt in den Zustand HALT WHILE ACTIVE ((14)). Danach wird die Funktion "OFF-Achse" aktiv und die Achse in den Status "DISABLED / SAFE" geschaltet (siehe *Abbildung 1, Zustände einer Achse auf Seite 86 Pfad (25)*).
- Ist die Achse im Zustand HALT wechselt die Achse in den Status IDLE oder sogar SAFE.

Beispiel für die Befehle OFF.An und ON.An

```

1      $W_ON
2      I3.1
3      G21 I4.0 W_ON
4      ON.A1                Achse 1 ON
5      ON.A2                Achse 2 ON
6      ;
7      $W_OFF
8      G21 I3.0 M_OFF
9      G21 I4.0 M_OFF
10     JMP W_OFF
11     ;
12     $M_OFF
13     OFF.A1                Achse 1 OFF
14     OFF.A2                Achse 2 OFF
15     JMP W_ON

```

Im AUTOMATIK-Betrieb läuft ein Parallelprogramm, das den Eingang 3 überwacht. Ist der Eingang nicht bestromt, so werden über die Befehle „OFF.A1“ und „OFF.A2“ die Achsen ausgeschaltet (IDLE/SAFE). Diese Befehle bewirken, dass zuerst die laufenden Positionierungen gestoppt und danach die Achsen ausgeschaltet (IDLE) werden.

Durch den Achsparameter „PUT-SAFE“ wird über die Ausgänge 4 und 5 die personell sichere Wideranlaufsperr (KSI+/-) im servoTEC-Verstärker aktiviert. Dieser überrückt dann mit den Kontakten „KSO“ die Türschalter.

Aus Sicht der Achsen ist der Arbeitsraum sicher und die Tür kann geöffnet werden (SAFE).

1.7.2.4 ON.An - Achse einschalten

ON.An

ab V5.07

Beschreibung:

Der Befehl ON.An (A[0...16]) aktiviert die Funktion "ON-ACHSE" für eine im Befehl ausgewählte Achse (*siehe Beispiel auf Seite 90*)

Funktion:

Ausgehend von unterschiedlichen Ausgangssituationen müssen zwei verschiedene Auswirkungen des Befehls unterschieden werden.

- Wird der Befehl abgesetzt, wenn die Achse im Status IDLE oder SAFE ist, dann wird die Achse entsprechend dem Achsparameter Einschaltverfahrenmodus in den Zustand HALT geschaltet (*siehe Abbildung 1, Zustände einer Achse auf Seite 86 Pfad (21) und (22)*).
- Wurde bei der Funktion "OFF-ACHSE" eine Positionierung unterbrochen. Achse wechselte nach HALT WHIL ACTIVE (*siehe Abbildung 1, Zustände einer Achse auf Seite 86 Pfad (22)*), so wird, falls es der "Einschaltverfahrenmode" erlaubt, die Positionierung fortgeführt (*siehe Abbildung 1, Zustände einer Achse auf Seite 86 Pfad (8)*).

HINWEIS

- Die Funktion "ON-ACHSE" kann nur wirksam werden, wenn keine Aktivierung zur Funktion "OFF-ACHSE" ansteht (z.B.: Eingang "OFF-ACHSE" nicht bestromt), siehe dazu *Abbildung 2*.
- Wird der Befehl für eine Achse abgesetzt, die im Zustand HALT ACTIVE oder OPERATIONAL ist, so zeigt er keine Veränderung oder Wirkung.

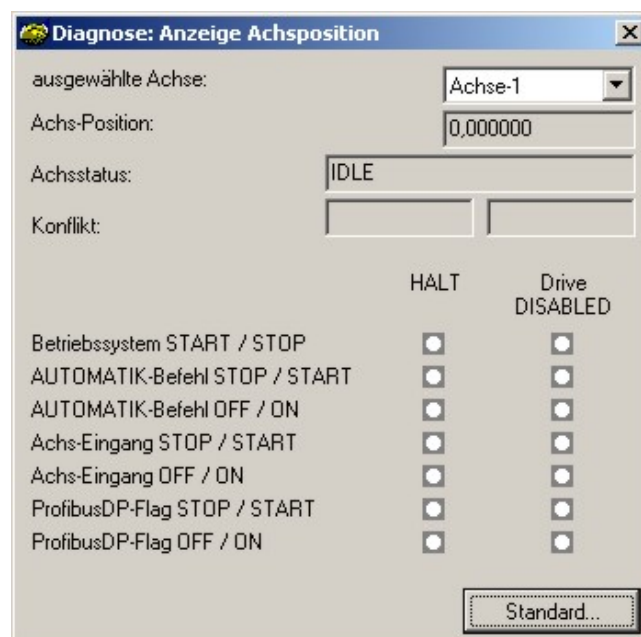


Abbildung 2: Diagnosefenster „Achspannung“

1.7.2.5 STOP.An - Unterbreche die Verfahrbereitschaft einer Achse

STOP.An

ab V5.07

Beschreibung und Funktion:

- Mit dem Befehl STOP.An (STOP.A[0...16]) kann ein Fahrbefehl unterbrochen werden. Die Achse wird angehalten und erhält den Status HALT WHILE ACTIVE (siehe *Abbildung 1, Zustände einer Achse auf Seite 86 Pfad (14)*).

HINWEIS Wird der Befehl STOP.An auf eine stehende Achse verwendet, so wechselt die Achse in den Status HALT WHILE OPERATIONAL ((13)). Danach wird ein Fahrbefehl für diese Achse erst dann gestartet, wenn die Achse durch ein START.An in den Status OPERATIONAL gewechselt hat (siehe *Abbildung 1, Zustände einer Achse auf Seite 86 Pfad (7)*).

Verwendung:

Die Verwendung ist in allen Programmtypen (*.PNC, *.PAB, *.PNX) möglich.

1.7.2.6 START.An - Herstellen der Verfahrbereitschaft einer Achse

START.An

ab V5.07

Beschreibung und Funktion:

- Der Befehl "START-An" (START.A1[...16]) bewirkt, daß eine Achse, die zuvor über den Befehl "STOP.An" angehalten wurde, die Achse ist dann im Zustand HALT WHILE ACTIVE, den unterbrochenen Fahrbefehl vollends ausführt (siehe *Abbildung 1, Zustände einer Achse auf Seite 86 Pfad (8)*).
- Ist die Achse im Zustand HALT WHILE OPERATIONAL so wechselt die Achse in den Zustand OPERATIONAL (siehe *Abbildung 1, Zustände einer Achse auf Seite 86 Pfad (7)*).

HINWEIS Wird der Befehl "START-An" auf eine Achse angewandt die gerade in Position ist (OPERATIONAL) oder läuft (ACTIVE), so hat der Befehl keine Auswirkung auf die Achse und den Status der Achse, siehe dazu *Abbildung 2, Seite 91*.

Verwendung:

Die Verwendung ist in allen Programmtypen (*.PNC, *.PAB, *.PNX) möglich.

1.7.2.7 ABORT.An - Abbruch eines gestoppten Fahrbefehls

ABORT.An

ab V5.07

Beschreibung und Funktion:

- Mit dem Befehl "ABORT.An" (A0,A1-A16) kann bei einer Achse, die über die Kommandos "STOP.An" oder „OFF.An“ gestoppt wurden, der Fahrbefehl abgebrochen werden. Dabei wird die STOP-Bedingung gelöscht und die aktuelle Position der Achse als Ist- und Zielposition übernommen. Die Achse wechselt aus dem Zustand HALT WHILE ACTIVE in den Zustand HALT WHILE OPERATIONAL (siehe *Abbildung 1, Zustände einer Achse auf Seite 86 Pfad (17), (29) und (33)*).

Beispiel

Start.pnc

```
1    $A
2    I9.1
3    RUN REFERENZ_A1
4    T20
5    $LOOP
6    G21 SM51.1 REF_OK
7    G21 I16.1 REF_ABORT
8    JMP LOOP
9    ;
10   ;
11   $REF_OK
12   T500
13   JMP A
14   ;
15   $REF_ABORT
16   STOP.A1
17   CANCEL_REFERENZ_A1
18   ABORT.A1
19   JMP A
20   ;
21   END
```

REFERENZ_A1.pnc

```
1    G25.A1
2    END
```

1.7.3 Fahren

1.7.3.1 G25 - Referenzfahrt

G25.An

G25.(Achse)

Beschreibung:

Nach Start des Befehls **G25.An** führt die PA-CONTROL mit der ausgewählten Achse entsprechend den eingestellten Parametern (Referenzfahrtart, Referenzgeschwindigkeit, ...) eine Referenzfahrt durch und übernimmt den Wert des Achsparameters Referenz-Offset als Position für die Achse.

HINWEIS Bei Schrittmotorachsen ist die Parametereinstellungen für die Referenzgeschwindigkeit und die Beschleunigung dem Überlaufweg (Abstand zwischen Referenzschalter und mechanischem Anschlag) anzupassen.

In der Version 4.xx konnte mit dem Befehl G25.A0 alle Achsen zum Fahren freigeschaltet werden, ohne die Achsen zu Referenzieren.

Besteht ab der Version 5.00 der Wunsch die Achsen zu verfahren, ohne eine Referenzfahrt durchzuführen, so kann dies mit dem Befehl G26 realisiert werden.

Über die System-Merker SM51 (A1) bis SM66 (A16) kann geprüft werden, ob die Achsen bereits referenziert sind

Befehlsform:

G25.An (nur in der V4.xx G25.A0), G25.A1 ... G25.A16

Anwendung:

Nach jedem Neubeginn, wie zum Beispiel nach einem Stromausfall oder dem Abschalten einer PA-CONTROL, besteht je nach Achstyp im Normalfall die Notwendigkeit eines Referenzlaufes. Nur dadurch ist gewährleistet, dass die Steuerung auch wirklich die Positionen anfährt, die im Programm vorgesehen sind.

Beispiel 1

```
1      G25.A1      ; Referenzlauf für die A1-Achse wird durchgeführt
2      G25.A2      ;Referenzlauf für die A2-Achse wird durchgeführt
4      END
```

1.7.3.2 A1 - Positionieren der Achsen

A1

A1:=(Operator)

Beschreibung:

Der Positionierbefehl lässt die Achsen in positiver bzw. negativer Richtung, vom Nullpunkt aus gesehen, verfahren. Der Position ist im Absolutmaßsystem (G90) der Absolutwert, im Kettenmaßsystem (**G91**) der relative Verfahrweg.

HINWEIS Der Endpunkt der Bewegung ist abhängig von dem programmierten Maßsystem, entweder Absolutmaß (G90) oder Kettenmaß (G91).

Befehlsform:

A1:=nnnn

A1:=Rn

Anwendung:

Positionierung von Achsen

Beispiel

1	G25.A1	;Referenzfahrt mit A1-Achse durchführen
2	G90.A1	;Absolutmaßsystem
3	A1:=200	;Achse A1 200 in positiver Richtung
4	A1:=198	;Achse A1 um 2 in negativer Richtung auf Absolutposition 198
5	R3:=678	
6	A1:=R3	;Achse A1 in positiver Richtung auf Absolutposition 678
7	G91.A1	;Kettenmaßsystem
8	R3:=-200	
9	A1:=-10	;Achse A1 in negativer Richtung um 10 verfahren
10	A1:=R3	;Achse A1 in negativer Richtung um -200 verfahren
11	R15:=3	
12	A1:=R!15	;Achse A1 in negativer Richtung um -200 verfahren
13	END	

1.7.3.3 G90 - Absolutmaßsystem

G90

G90.(Achse)

Beschreibung:

Nach dem Aufruf des Befehls **G90.An** erfolgen alle nachfolgenden Positionierungen im Absolutmaßsystem, bis durch den Befehl **G91.An** ins Kettenmaßsystem umgeschaltet wird. Das Absolutmaßsystem ist das Standard-Maßsystem.

Befehlsform:

G90.An

G90.A0

Anwendung:

Für Bemaßungen, die alle von einem Nullpunkt aus vermessen sind.

Beispiel

1	G25.A1	;Referenzfahrt
2	G90.A1	;nachfolgende Positionsbefehle werden für Achse 1 im Absolutmaßsystem ausgeführt
3	A1:=10	;mit der A1-Achse wird Position 10 angefahren, Weg = 10 mm [Skizze: 1.]
4	A1:=450	;mit der A1-Achse wird Position 450 angefahren, Weg = 440 mm [Skizze: 2.]
5	A1:=30	;mit der A1-Achse wird Position 30 angefahren, Weg = -420 mm [Skizze: 3.]
6	G90.A0	;nachfolgende Positionsbefehle werden bei allen Achsen im Absolutmaßsystem ausgeführt
7	END	

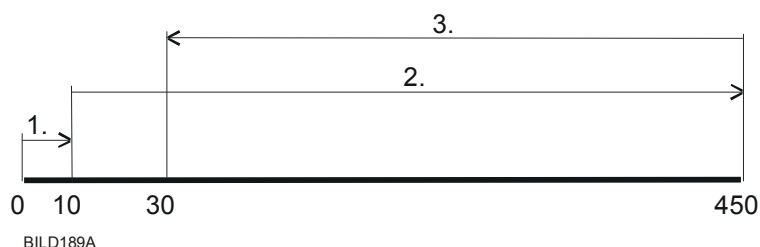


Abbildung 3: Absolutmaßsystem

1.7.3.4 G91 - Kettenmaßsystem

G91

G91.(Achse)

Beschreibung:

Nach dem Aufruf des Befehls **G91.An** erfolgen alle nachfolgenden Positionierungen im Kettenmaßsystem, bis durch den Befehl **G90.An** wieder ins Absolutmaßsystem umgeschaltet wird.

Befehlsform:

G91.An

G91.A0

Anwendung:

Für Bemaßungen, die nicht auf einen Nullpunkt bezogen sind.

Für Maßabstände, die hintereinander mehrfach vorkommen und mit einer Programmschleife (DEC.Ni) wiederholt werden können.

Beispiel

1	G25.A1	;Referenzfahrt
2	G91.A1	;nachfolgende Positionsbefehle werden für Achse 1 im Kettenmaßsystem ausgeführt
3	A1:=10	;mit der A1-Achse wird Position 10 angefahren, Weg = 10 mm [Skizze: 1.]
4	A1:=450	;mit der A1-Achse wird Position 460 angefahren, Weg = 450 mm [Skizze: 2.]
5	A1:=30	;mit der A1-Achse wird Position 490 angefahren, Weg = 30 mm [Skizze: 3.]
6	G90.A0	;nachfolgende Positionsbefehle werden bei allen Achsen im Absolutmaßsystem ausgeführt
7	END	

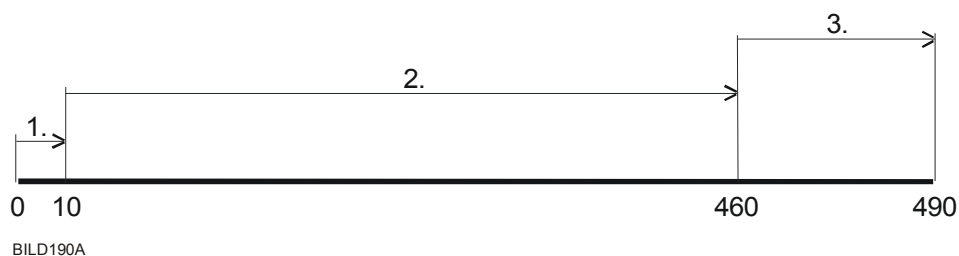


Abbildung 4: Kettenmaßsystem

1.7.3.5 G100 - Beschleunigung

G100

G100.(Achse).(Wert)

Beschreibung:

„An“ steht für die Bezeichnung der Achse und kann bei Mehrachssystemen den Namen aller vorhandenen Achsen annehmen (Anzahl der Achsen je nach Ausstattung unterschiedlich).

G100.B dient zur Einstellung der Bahnbeschleunigung bei der Interpolation. Die Bahngeschwindigkeit **FB** muss vorher eingestellt werden.

HINWEIS Ist die Achse im Zustand „HALT“, „IDLE“ oder „SAFE“, so wird der Befehl nicht ausgeführt. Der Programmablauf bleibt an diesem Befehl stehen und führt diesen Befehl erst aus, wenn die Achse sich im Zustand „OPERATIONAL“ befindet.

Ist der Wert größer oder kleiner als die Grenzwerte, erfolgt die Fehlermeldung „Wert zu groß“ oder „Wert zu klein“.

Befehlsform:

G100.An.15

G100.An.R7

Anwendung:

Mit Hilfe des Befehls G100 kann, abweichend von den in den Parameterwerten abgelegten Werten für die Beschleunigung, eine der Situation angepasste Arbeitsweise erreicht werden. Die getroffene Festlegung gilt bis zur nächsten Anwendung des Befehles G100 oder bis zum Verlassen des Automatikbetriebes. Wird der Befehl G100 nicht angewendet, so gelten die abgelegten Parameterwerte.

Beispiel

1	G25.A1	<i>;Referenzfahrt A1-Achse</i>
2	R7:=20	<i>;Beschleunigung in das Register 7 laden</i>
2	G91.A0	<i>;Kettenmaßsystem</i>
3	FA1:=250	
4	G100.A1.15	<i>;Festlegung der Beschleunigung für die A1-Achse</i>
5	A1:=250	<i>;Fahrbefehl für die A1-Achse</i>
6	G100.A2.R7	<i>;Festlegung der Beschleunigung für die A2-Achse nach dem Inhalt des Realzahlregisters 7</i>
7	A2:=500	<i>;Fahrbefehl für die A1-Achse</i>
8	END	

1.7.3.6 FAn - Verfahrensgeschwindigkeit

F

F(Achse):=(Geschwindigkeit)

Beschreibung:

Mit dem Befehl **F** wird die Steuerung veranlasst, im weiteren Programmverlauf mit der im Operator angegebenen Verfahrensgeschwindigkeit zu verfahren. Alle Achsen die mit dem Operanden **F** angesprochen werden, können mit verschiedenen Geschwindigkeiten verfahren werden.

Die in der Parameterliste festgelegten Achsparameter können ebenfalls über Register geladen werden.

Der Befehl **FB** dient der Einstellung der Bahngeschwindigkeit bei der Interpolation.

HINWEIS

Ist die Achse im Zustand „HALT“, „IDLE“ oder „SAFE“, so wird der Befehl nicht ausgeführt. Der Programmablauf bleibt an diesem Befehl stehen und führt diesen Befehl erst aus, wenn die Achse sich im Zustand „OPERATIONAL“ befindet.

Wird im Programmverlauf keine Geschwindigkeit **F** programmiert, dann werden alle Verfahrbefehle mit der im Parameterfeld definierten Verfahrensgeschwindigkeit ausgeführt.

Eine Geschwindigkeitsvorgabe, die höher ist als die im Parameterfeld festgelegte, führt zur Fehlermeldung „Wert außerhalb Bereich“ (E532).

Ist der Geschwindigkeitswert zu klein, führt dies zu keiner Fehlermeldung. Es wird die kleinste mögliche Geschwindigkeit eingestellt.

Befehlsform für die Änderung der Geschwindigkeit einer Achse:

FAn:=200

FAn:=R4

Befehlsform für die Änderung der Geschwindigkeit einer Bahn:

FBAAn:=200;

FBAAn:=R4

Anwendung:

Änderung der Geschwindigkeiten beim Verfahren und Bewegen von Achsen während des Programmablaufes.

Beispiel 1

1	G25.A1	;Referenzfahrt mit A1-Achse ausführen
2	G91.A1	;Kettenmaßsystem setzen
3	FA1:=200	;Verfahrgeschwindigkeit A1-Achse 200 AE/s
4	A1:=1000	
5	R4:=500	
6	FA1:=R4	;Verfahrgeschwindigkeit A1-Achse 500 AE/s
7	A1:=10	
8	R23:=4	
9	FA1:=R!23	;Verfahrgeschwindigkeit A1-Achse aus Register R4
10	END	

Besonderheiten bei der PA-CONTROL MP / Servomotorachsen:

Bei der PA-CONTROL MP und Servomotorachsen kann, im Gegensatz zu den anderen Varianten der Steuerung (Steuerungen mit PLS6-/7/8/9-Karte), während eines laufenden Positioniervorganges die Geschwindigkeit verändert werden.

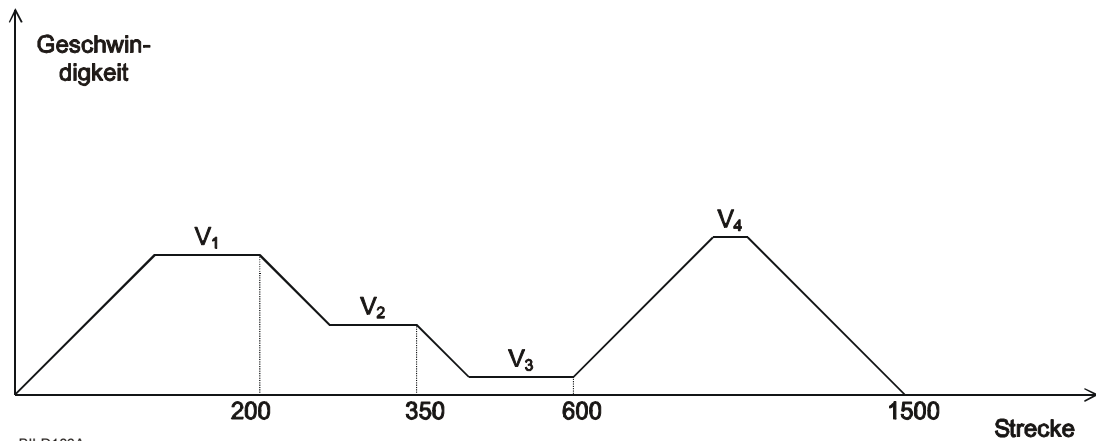


BILD186A

Abbildung 5: Geschwindigkeitsprofil

Beispiel

1	G90.A0	;Positionierung erfolgt im Absolutmaßsystem
2	G25.A1	;Referenzfahrt
3	G100.A1.100	;Beschleunigung der A1-Achse festlegen
4	G210.A0	;verarbeite Befehle über Zeilengrenze hinweg
5	FA1:=1000	;Verfahrgeschwindigkeit A1-Achse v_1 : 1000AE/s
6	A1:=1500	;Positionieren der A1-Achse
7	G230.1.A1.200	;warte bis die aktuelle Absolutposition größer 200 ist
8	FA1:=500	;Verfahrgeschwindigkeit A1-Achse v_2 : 500AE/s
9	G230.1.A1.350	
10	FA1:=150	;Verfahrgeschwindigkeit A1-Achse v_3 : 150AE/s
11	G230.1.A1.600	
12	FA1:=1200	;Verfahrgeschwindigkeit A1-Achse v_4 : 1200AE/s
13	G213.A0	
14	END	

1.7.3.7 G123

G123

G123.(Achse) (Bedingung)

Beschreibung:

Bei der Funktion **G123** wird der nächste Verfahrbefehl der entsprechenden Achse ausgeführt, solange die Bedingung **n.m** des Operanden gültig ist. Ist die Bedingung nicht erfüllt, wird der Verfahrbefehl abgebrochen und der nächste Befehl ausgeführt. **G123** wirkt nur auf die im Befehl definierte Achse. Ist die Bedingung beim Starten der Achse nicht erfüllt, so wird die Positionierung nicht gestartet. Es wird sofort mit dem nächsten Befehl fortgefahren.

HINWEIS

Bei abgebrochenen Verfahrbefehlen wird die aktuelle Absolutposition für die nachfolgende Positionierung berücksichtigt. Der Systemmerker (SM) für die entsprechende Achse wird dann gesetzt.

Es gilt folgende Zuordnung:

SM11: Achse 1 (SM = Systemmerker)

SM12: Achse 2

.....

SM26: Achse 16

Pro Achse kann nur ein G123 aktiv sein. Verknüpfungen von mehreren Bedingungen sind über die Befehle der logischen Verknüpfungen möglich

Befehlsform:

G123.An I3.1

G123.An M3.1

G123.An O4.1

Anwendung:

- Ein Werkstück wird solange verfahren, bis es erkannt wird. Ein Sensor o.ä. sendet bei Erkennung ein Signal an die PA-CONTROL zurück, so dass das Programm fortgeführt werden kann.
- Für das Teachen (manuelles Anfahren einer Position) Die Achse fährt solange die „Teachtaste“ betätigt ist

Beispiel

1	G25.A1 G90.A1	;Referenzfahrt mit A1-Achse durchführen und auf Absolutmaßsystem umschalten
2	R2:=1000	;Lade Geschwindigkeit in das Register R2
3	FA1:=R2	; langsame Geschwindigkeit wählen
4	G123.A1 I3.0	;nachfolgender Verfahrbefehl gilt, solange der Sensor des Eingangs 3 unbestromt ist
5	A1:=100	;Verfahrbefehl wird ausgeführt
6	G21SM11.0 KEIN_TEIL	; Kein Werkstück vorhanden → die Bedingung (Systemmerker der Achse 1, SM11, nicht gesetzt) ist erfüllt → es folgt ein bedingter Sprung zur Marke KEIN_TEIL. Ist ein Werkstück vorhanden → Bedingung nicht erfüllt → Programm wird in der nächsten Zeile fortgeführt.
7	R1:=A1	; Realzahlregister wird geladen (Wert 100 entspricht z.B. der Werkstückkante)
8	R1:= R1+10	; Realzahlregister wird um 10 erhöht (Wert 110 entspricht der Stelle des Werkstücks, an welcher die Bohrung gesetzt werden soll – z.B. Werkstückmitte)
9	A1:=R1	; fahre z.B. zur Werkstückmitte
10	SUB LOCHBOHR	; Unterprogramm aufrufen/abarbeiten
11	\$KEIN_TEIL	; Marke KEIN_TEIL
12	END	

Programm: LOCHBOHR

1	O10:=1	;Einschalten Bohrmaschine
2	O16:=1	;Einschalten Vorschub Bohrmaschine
3	I16.1	;Endposition erreicht
4	T50	;Warte 0,5 Sekunden
5	O16:=0 O15:=1	;Vorschub Bohrmaschine aus, Rückwärtsbewegung
6	I15.1	;Ruheposition erreicht
7	O15:=0	;Rückwärtsbewegung ausschalten
8	O10:=0	;Ausschalten Bohrmaschine
9	END	

Erläuterung des Programmbeispiels:

Im Beispiel wird das Werkstück solange verfahren, bis ein Sensor erkennt, dass es vorhanden ist. Wenn das Werkstück vorhanden ist, schaltet der Sensor ein und das Programm wird in der nächsten Zeile fortgeführt.

Wird kein Werkstück erkannt, erfolgt ein Sprung zur Marke \$KEIN_TEIL.

Der Systemmerker (SM) wird nur bei abgebrochenen Fahrbefehlen gesetzt.

1.7.3.8 G123Q - Fahre solange Bedingung erfüllt

G123Q

ab V5.16

G123Q.(Achse) (Bedingung)

Beschreibung:

Bei der Funktion **G123Q** wird der nächste Verfahrbefehl der entsprechenden Achse ausgeführt, solange die Bedingung **n.m** des Operanden gültig ist. Ist die Bedingung nicht erfüllt, wird der Verfahrbefehl abgebrochen und der nächste Befehl ausgeführt.

Dabei wird die Achse über den in den Achsparameter eingestellten Wert „OFF-Rampe“ abgebremst.

G123Q wirkt nur auf die im Befehl definierte Achse. Ist die Bedingung beim Starten der Achse nicht erfüllt, so wird die Positionierung nicht gestartet. Es wird sofort mit dem nächsten Befehl fortgefahren.

Anwendung:

Überwachen eines „Crash-Sensors“

HINWEIS

Bei *abgebrochenen Verfahrbefehlen* wird die aktuelle Absolutposition für die nachfolgende Positionierung berücksichtigt. Der Systemmerker (SM) für die entsprechende Achse wird dann gesetzt.

Es gilt folgende Zuordnung:

SM11: Achse 1 (SM = Systemmerker)

Pro Achse kann nur ein G123Q aktiv sein. Verknüpfungen von mehreren Bedingungen sind über die Befehle der logischen Verknüpfungen möglich

Der G123Q-Befehle kann nur für die Achstypen servoTEC und servoTEC S2 eingesetzt werden

Beispiel

1	G25.A1 G90.A1	;Referenzfahrt mit A1-Achse durchführen und auf Absolutmaßsystem umschalten
2	R2:=1000	;Lade Geschwindigkeit in das Register R2
3	FA1:=R2	; langsame Geschwindigkeit wählen
4	G123QA1 I3.0	;nachfolgender Verfahrbefehl gilt, solange der Sensor des Eingangs 3 unbestromt ist
5	A1:=100 END	;Verfahrbefehl wird ausgeführt

1.7.3.9 G 150 - Fahre Teilstrecke mit Start-Stop¹

G150

Beschreibung:

Der Befehl G150 gibt dem Programmierer die Möglichkeit, bei der nächsten Positionierung der entsprechenden Achse, den Bremsvorgang früher einzuleiten, und die mit G150 definierte Strecke mit der Start-Stoppgeschwindigkeit zu positionieren. Der Übergang von der Bremsrampe in die Start-Stoppgeschwindigkeit (Strecke) erfolgt kontinuierlich.

HINWEIS Der G150-Befehl kann bei einer servoTEC-Achse nicht angewendet werden. Der Syntaxcheck prüft den Achstyp und gibt eine entsprechende Meldung aus.

HINWEIS Die Zahl nach dem Achsnamen im Befehl G150 ist eine Strecke und wird vorzeichenlos und richtungsunabhängig interpretiert. Der eingestellte Getriebe-
faktor wird berücksichtigt. Der Befehl wird nur bei der nächsten Positionierung
dieser Achse berücksichtigt. Ist die zu verfahrenene Strecke bei der nächsten
Positionierung kleiner als die Strecke aus dem Befehl G150, so wird der Befehl
G150 ignoriert und die Positionierung verläuft wie gewohnt.

Befehlsform:

G150.An.200

G150.An.Rn

Anwendung:

Bei Positionierungen, bei denen die letzte Teilstrecke im Start-Stop-Betrieb positioniert wird.

Beispiel: Fügevorgang: schnellere Reaktion auf G123-Befehle.

¹ Nicht bei PAC-servoTEC

Beispiel

- | | | |
|---|--------------------------|---|
| 1 | <code>G25.A1</code> | <i>;Referenzfahrt mit A1-Achse durchführen</i> |
| 2 | <code>G91.A1</code> | <i>; auf Kettenmaßsystem umschalten</i> |
| 3 | <code>G150.A1.200</code> | <i>;fahre bei der nächsten Positionierung mit der A1-Achse die letzten 200 Inkremente mit der Start-Stoppgeschwindigkeit</i> |
| 4 | <code>A1:=1000</code> | <i>;mit der A1-Achse 1000 Inkremente positionieren. Der Bremsvorgang wird so eingeleitet, dass 200 Inkremente vor der Zielposition die Start-Stoppgeschwindigkeit erreicht wird. Die letzten 200 Inkremente werden dann mit der Start-Stoppgeschwindigkeit positioniert</i> |
| 5 | <code>END</code> | |

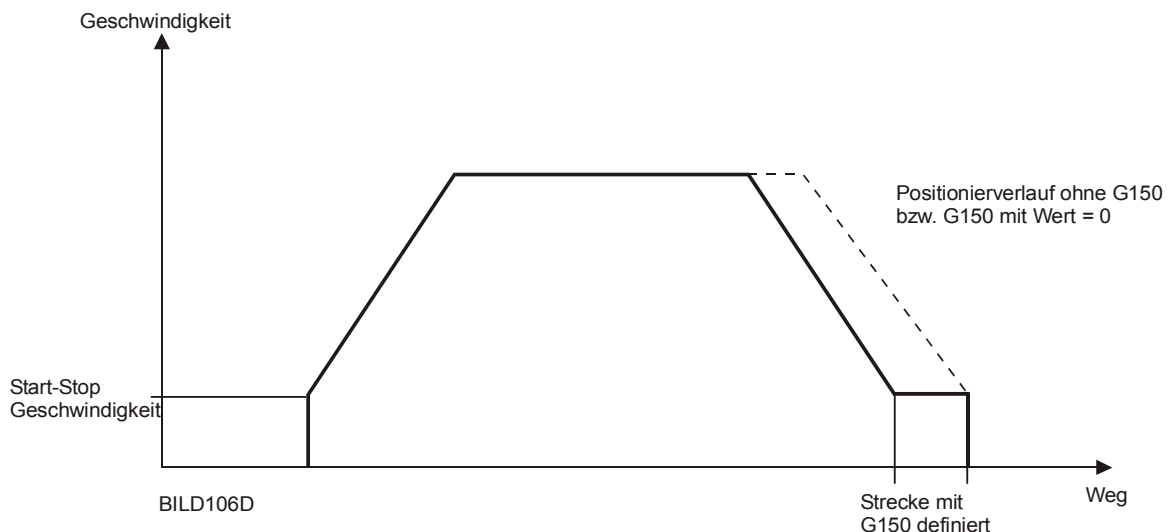


Abbildung 6: Geschwindigkeitsverlauf bei Anwendung des G150

1.7.3.10 Rn:=Fan - Letzte Verfahrensgeschwindigkeit einer Achse

Rn:=Fan

Befehlsform:

Rn:=Fan

Der Befehl liefert die zuletzt eingestellte Geschwindigkeit der Achse zurück. Eine Information ob die Achse aktuell mit dieser Geschwindigkeit fährt (Achse steht, Achse beschleunigt, Achse fährt oder Achse bremst) liefert dieser Befehl nicht.

1.7.4 Position und Parameter

1.7.4.1 G26 - Position auf Null setzen / auf Position setzen

G26

G26.(Achse)

G26.(Achse).konst

G26.Achse).Rn

Beschreibung:

Mit der Funktion **G26.An** kann die Position der Achse auf Null gesetzt, mit den Funktionen **G26.An.konst** und **G26.An.Rn** auf Position gesetzt werden. Die Variable **An** des Operanden gibt die angewählte Achse an. Es findet keine Bereichsüberprüfung statt. Die Achse kann somit auf eine Position außerhalb der Verfahrbereichsgrenzen gesetzt werden.

HINWEIS

Die Software-Endschalter, die in der Parameterebene durch den Bereich festgelegt wurden, werden unverändert übernommen. Bei entsprechender Wiederholung dieses Befehles kann praktisch unendlich in eine Richtung positioniert werden, da kein Zählerüberlauf und kein Überschreiten der Bereichsgrenzen möglich ist.

ACHTUNG: Bei der Version 4.xx und Servo-Achsen, die über den LV-servoTEC angetrieben werden, ist der Zählerüberlauf möglich!

Bei Getriebefaktoren mit Nachkommastellen ungleich 0 werden die Reste ebenfalls gelöscht, d.h. sie können bei nachfolgender Positionierung nicht berücksichtigt werden.

Bei der Ausführung des G26-Befehles wird das Referenzflag gesetzt. Die Achse gilt somit als referenziert

Befehlsform:

G26.An

Anwendung:

Ein sinnvoller Anwendungsfall für diesen Befehl liegt bei kontinuierlichen Bewegungen, z.B. Nutzung des Drehtisches DT140 in nur einer Drehrichtung, vor.

Als Ersatz für eine Referenzfahrt, wenn die Position von einem Maßsystem kommt.

Beispiel 1

```
1   G26.A1           ;Absolutposition der A1-Achse auf Null setzen
2   G26.A2           ;Absolutposition der A2-Achse auf Null setzen
3   END
```

Befehlsform:

G26.An.konst

G26.An.Rn

Anwendung:

Ein sinnvoller Anwendungsfall für diesen PAC-Befehl ist die Festlegung der Achsposition z.B. auf Grund abgespeicherter oder übernommener Werte.

Beispiel 2

1	<i>R113:=342.5</i>	
2	<i>G26.A1.234.7</i>	<i>;Absolutposition der A1-Achse auf 234.7 setzen</i>
3	<i>G26.A2.R113</i>	<i>;Absolutposition der A2-Achse auf 342.5 setzen</i>
4	<i>END</i>	

1.7.4.2 G29 - Position auf Maß setzen

G29

ab 4.72

G29.(Achse)

Beschreibung:

Mit der Funktion **G29** kann der Positionszähler jeder Achse auf einen neuen Wert gesetzt werden. Der Operator gibt die angewählte Achse an. Die Softwareschalter werden entsprechend mitverändert.

Befehlsform:

G29.Ai.Konstante	Setzt die aktuelle Achsposition auf den Wert der Konstante
G29.Ai.Rn	Setzt die aktuelle Achsposition auf den Wert des R-Registers
G29.Ai.Rn.Rm	Setzt die Achsposition so, dass bei der Position Rm die Achse auf den Wert des R-Registers Rn steht.

HINWEIS Die G29-Befehle der Versionen vor Version 4.71R werden komplett verworfen. Als Bezug dient die Achsposition nach der Referenzfahrt G25, G26).

Wird der Befehl G29.A1.Rn verwendet, geht der Bezugspunkt aus der Referenzfahrt (G25, G26) verloren. Der Anwender muss wissen wie oft und um was er die Position verschoben hat, damit er wieder auf den „Nullpunkt“ zurückfahren kann

Mit dem Befehl G29.Ai.Rn.Rm kann die Position einer Achse beliebig oft gesetzt oder verschoben werden. Wird dann als letzter der Befehl G29.Ai.Rm.Rm verwendet wird der Offset aus den G29-Befehlen verworfen und die Achse kann problemlos in der „Original-Koordinaten“ (aus / nach G25-Befehl) verfahren werden. Siehe hierzu das folgende Beispiel

Beispiel

Bearbeitung von drei Teilen

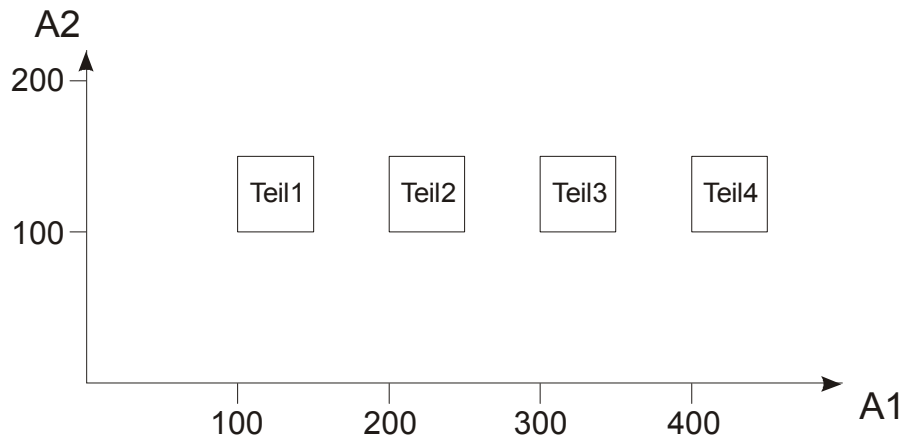


Abbildung 7: Maß setzen

Programm: START.PNC

```

1   G25.A1 G25.A2
2   R1:=100                               ;A1-Position von Teil 1
3   R2:=200                               ;A1-Position von Teil 2
4   R3:=300                               ;A1-Position von Teil 3
5   R4:=0
6   ;
7   $LOOP
8   G29.A1.R4.R1                           ; setze Achsposition für Teil 1 bearbeiten
9   SUB TEIL_BEARBEITEN
10  G29.A1.R4.R2                           ; setze Achsposition für Teil 2 bearbeiten
11  SUB TEIL_BEARBEITEN
12  G29.A1.R4.R3                           ; setze Achsposition für Teil 3 bearbeiten
13  SUB TEIL_BEARBEITEN
14  G29.A1.R4.R4                           ; setze Achsposition für die Fahrt auf Grundstel-
                                           ; lung, alle Verschiebungen durch G29 aus
                                           ; Zeile 8, 10 und 12 werden verworfen

15  A1:=0 A2:=0
16  ...
17  JMP LOOP
18  END
19

```

Programm: TEIL_BEARBEITEN

```

1   A1:=0 A2:=100                         Startposition zum Teil bearbeiten anfahren
2   ...
3   ...
4   END

```

1.7.4.3 Rn:=An / Nn:=An - Achsposition holen

Rn:=An
Nn:=An

Beschreibung:

Die aktuelle Achsposition wird in ein N- oder R-Register übernommen.

Im R-Register steht die Position entsprechend der eingestellten Maßeinheit zur Verfügung. In ein N-Register wird der Schrittzähler übernommen.

Befehlsform:

R1:=An

N1:=An

Anwendung:

Speichern von Achspositionen zum Beispiel im Teachprogramm.

Anzeigen der Achsposition und Schrittüberwachung im Programm.

Beispiel

1	R226:=0.01	;maximale Abweichung der Achsposition zur Encoderposition
2	\$LOOP	;Position 100 anfahren mit reduziertem Moment
3	R222:=A1	;Achsposition holen
4	R223:=ENC1	;Encoderposition holen
5	R225:=R222-R223	;Differenz bilden
6	M221:=R225>R226	;vergleichen
7	G21 M221.0 LOOP	
8	M220:=0	;STOP z.B. einer mit G123 gestarteten Achse
9	END	

1.7.5 Endschalter

1.7.5.1 G142 Endschalterüberwachung „AUS“

G142

G142.(Achse)

Beschreibung:

Nach dem Aufruf des Befehls G142.A0 oder G142.An wird die Endschalterüberwachung für (alle) eine Achsen ausgeschaltet, bis durch den Befehl G143.A0 oder G143.An (siehe Abschnitt 1.7.5.2, Seite 112) die Endschalterüberwachung für (alle) eine Achsen wieder eingeschaltet wird.

HINWEIS Die Endschalterüberwachung wird bei **jedem** Wechsel in den Automatikmodus aktiviert. Berücksichtigen Sie diese Tatsache beim Programmieren.

Befehlsform:

G142.A0

G142.An

Anwendung:

Der Befehl hat eine besondere Bedeutung bei Rundachsen, bei denen oftmals über die 360° Grenze hinaus positioniert wird.

Beispiel

1	G25.A1	;Referenzfahrt
2	G91.A1	;Nachfolgende Positionsbefehle werden für Achse 1 im Kettenmaßsystem ausgeführt
3	G142.A1	;Endschalterüberwachung für Achse 1 AUS
4	A1:=10	;mit der A1-Achse wird Position 10 angefahren
5	A1:=450	;mit der A1-Achse wird Position 460 angefahren
6	A1:=30	;Mit der A1-Achse wird Position 490 angefahren
7	G90.A1	;nachfolgende Positionsbefehle werden bei der Achse 1 im Absolutmaßsystem ausgeführt
8	G143.A1	;Endschalterüberwachung für Achse 1 EIN
9	END	

Achsenansteuerung	Befehl implementiert
PLS6	ja
PLS7	ja
MP	ja
servoTEC	nein
intelliMOT	Nein
flexmoTEC	nein

1.7.5.2 G143 - Endschalterüberwachung „EIN“

G143

G143.(Achse)

Beschreibung:

Nach dem Aufruf des Befehls G142.A0 oder G142.An (*siehe Abschnitt 1.7.5.1, Seite 111*) wird die Endschalterüberwachung für (alle) eine Achsen ausgeschaltet, bis durch den Befehl G143.A0 oder G143.An die Endschalterüberwachung für (alle) eine Achsen wieder eingeschaltet wird.

Befehlsform:

G143.A0

G143.An

Anwendung:

Der Befehl hat eine besondere Bedeutung bei Rundachsen, bei denen oftmals über die 360° Grenze hinaus positioniert wird.

Beispiel

1	G25.A1	<i>;Referenzfahrt</i>
2	G91.A1	<i>;nachfolgende Positionsbefehle werden für Achse 1 im Kettenmaßsystem ausgeführt</i>
3	G142.A1	<i>;Endschalterüberwachung für Achse 1 AUS</i>
4	A1:=10	<i>;mit der A1-Achse wird Position 10 angefahren</i>
5	A1:=450	<i>;mit der A1-Achse wird Position 460 angefahren</i>
6	A1:=30	<i>;mit der A1-Achse wird Position 490 angefahren</i>
7	G90.A1	<i>;Nachfolgende Positionsbefehle werden bei der Achse 1 im Absolutmaßsystem ausgeführt</i>
8	G143.A1	<i>;Endschalterüberwachung für Achse 1 EIN</i>
9	END	

Achsenansteuerung	Befehl implementiert
PLS6	ja
PLS7	ja
MP	ja
servoTEC	nein
intelliMOT	Nein
flexmoTEC	nein

1.7.6 Encoder und Schleppfehler

1.7.6.1 Ni:=PEAn / Ri:=PEAn - Schleppfehler holen

Ni:=PEAn
Ri:=PEAn

gilt nur für

PLS7-9 PLSMP servoTEC servoTEC S2

- **x** **x** **x**

Beschreibung:

Für die Achsen, die über einen LV-servoTEC, einen LV-servoTEC S2, eine PA-CONTROL MP angetrieben werden besteht die Möglichkeit, den aktuellen Schleppfehler dieser Achsen in ein N- oder R-Register zu übernehmen. Bei einer Servo-Achse wird der Schleppfehler aus dem LV-servoTEC, bei der PA-CONTROL MP aus dem internen Zähler in ein N- oder R-Register übernommen.

Es müssen dabei zwei grundsätzlich unterschiedliche Formen unterschieden werden:

- Schleppfehler der Achse n in SI-Einheiten, also in Inkrementen oder Schritten, werden in ein Ganzzahlregister übernommen.
- Schleppfehler der Achse n in Anzeigeeinheiten werden in Realzahlregister übernommen.

Befehlsform:

Ni:= PEAn

Ri:= PEAn

Anwendung:

Erfassen des „Schleppfehlers“ beim Schrauben oder Kraftbegrenzung

HINWEIS Für die Ausführung des Befehls werden etwa 8-10ms benötigt. Das bedeutet, dass ein Schleppfehler nur mit einer Datenrate von ca. 10ms erfasst werden kann.

Beispiel

```

1   N26:=150           ; Max.Schleppfehler
2   M22:=0            ; Abbruchmerker löschen
3   $LOOP
4   N25:=PEA1         ; Schleppfehler holen
5   N25:=ABS.N25     ; Absolutwert ohne Vorzeichen erzeugen
6   M21:=N25>N26    ; Differenz zu groß
7   G21 M21.0 LOOP
8   M22:=1           ; Abbruch
9   END

```

1.7.6.2 ENC - Encoderposition übernehmen²

```
Rn:=ENCn  
Ni:=ENCn
```

(ENCAchseNr)

Beschreibung:

Die Übernahme der Encoderposition kann in ein N- oder R-Register erfolgen.

HINWEIS

Wird der Encoderwert in einem R-Register abgelegt, so wird die Drehgeberposition, die vergleichbar mit der Achsposition ist, gespeichert.

Wird der Encoderwert in einem N-Register abgelegt, so werden die Drehgeberinkremente angepasst auf die Schritte des Schrittmotors gespeichert.

Befehlsform:

R1:=ENCn

N1:=ENCn

Anwendung:

Im Programm können Schrittabweichungen erkannt werden, z.B. beim Schrauben, bei dem man die Tatsache nutzt, dass einem bestimmten Drehmoment ein ganz bestimmter Strom zugeordnet werden kann. Wird das Drehmoment erreicht, bleibt der Motor stehen weil der vorher eingestellte Strom überschritten wird. Die Differenz zwischen der Encoderposition und dem Schrittzähler ist nun der Nachweis für die richtig angezogene Schraube.

² PA-CONTROL MP

Beispiel 1

1	R226:=0.01	;maximale Abweichung der Achsposition zur Encoderposition
2	\$LOOP	
3	R222:=A1	;Achspannung holen
4	R223:=ENC1	;Encoderposition holen
5	R224:=R222-R223	;Differenz bilden
6	R225:=ABS.R224	;Betrag bilden
7	M221:=R226>R225	;vergleichen
8	G21 M221.0 LOOP	
9	M220:=1	; zur Auslösung eines Stopps z.B.
10	END	

Beispiel 2

1	N226:=32	;maximale Abweichung des Schrittzählers zum Encoderpositionsgeber. 32 entspricht dem maximalen Schleppfehler eines 2-Phasen Schrittmotors von 2 Vollschritten (2 Vollschr. mit 16-facher Auflösung→32)
2	\$LOOP	
3	N222:=A1	;Schrittzähler holen
4	N223:=ENC1	;Encoder-Positionsgeber holen
5	N224:=N222-N223	;Differenz bilden
6	N225:=ABS.N224	;Betrag bilden
7	M221:=N225>N226	;vergleichen
8	G21 N221.0 LOOP	
9	M220:=0	; zur Auslösung eines Stopps z.B.
10	END	

1.7.6.3 SSIn.i.Rm / SSIn.i.Nm - Warte auf Position vom SSI-Interface

SSIn.i.Rm
SSIn.i.Nm

(SSIn.i.<Register>)

Beschreibung:

Wartet bis die Position oder der Zählwert des SSI-Moduls größer oder kleiner als der Wert des Registers (Real- oder Ganzzahlregister) ist.

Warte bis größer : i=1

Warte bis kleiner : i=0

Befehlsform:

SSIn.i.Rm

SSIn.i.Nm

Anwendung:

- Wird zum Setzen oder Rücksetzen von Ausgängen bei schnellen servoTEC-Achsen verwendet bei denen die Wiederholgenauigkeit kleiner 10ms sein soll

Beispiel

1 R12:=123.45

2 N33:=446

...

n SSI1.1.R12

;wartet bis die Position des SSI-Moduls größer als der Inhalt von R2, also größer 123.45

...

m SSI1.0.N33

; wartet bis der Zählerwert SSI-Moduls kleiner als der Inhalt von N33 ist, also kleiner als 446

1.7.7 Parameter-Befehle

1.7.7.1 PARAMETER.XX.Ai. - Parameter schreiben ³

PARAMETER.XX.Ai

Parameter.<Achstype>.<Achse>.<Parameter>:=<Wert>

Beschreibung:

Mit diesem Befehl können spezielle Parameter (Antriebsparameter) an einen bestimmten Verstärker geschickt werden.



VORSICHT

Für die Anwendung dieses Befehls sind umfangreiche Kenntnisse der Verstärker und Erfahrungen bezüglich ihrer Optimierung notwendig.

Die Fa. IEF Werner GmbH lehnt die Verantwortung für die Folgen einer nicht sachgemäßen Anwendung des Befehls ab.

HINWEIS

Die Liste der möglichen Parameter einschließlich ihrer Bedeutung entnehmen Sie bitte bei einem „LV servoTEC“ der Hilfefunktion des Programms „DRIVE.EXE“, dem Dokument „MAN_DE_EN_1048937_servoTEC_ascii20.chm“ bzw. dem PDF-Dokument „Liste der ASCII-Objekte“. Bei einem Verstärker „LV flexmoTEC“ benutzen Sie bitte die entsprechenden mitgelieferten Handbücher! Für den servoTEC S2 finden Sie die Parameterliste auf der nächsten Seite

Befehlsform:

PARAMETER.ST.Ai.Parametername:=Konstante

PARAMETER.ST.Ai.Parametername:=Nn

PARAMETER.ST.Ai.Parametername:=Rn

PARAMETER.FT.A1.Parametername:=Nn (flexmoTEC)

PARAMETER.S2.A1.Parametername:=Nn (servoTEC S2)

HINWEIS

Die Verwendung eines Ganzzahl- oder Realzahlregisters ergibt sich aus der Art des Parameters.

Programmbeispiel siehe nächste Seite

³ servoTEC, flexmoTEC

Parameterliste für den servoTEC S2:

SYNCHRONISATION_MAIN
 SYNCHRONISATION_ENCODER_SELECTION
 POWER_STAGE_TEMPERATURE
 MOTOR_TEMPERATURE
 ENCODER_EMULATION_RESOLUTION
 ENCODER_EMULATION_OFFSET
 FOLLOWING_ERROR_WINDOW
 FOLLOWING_ERROR_TIME_OUT
 CURRENT_ACTUAL_VALUE
 ENCODER_X10_RESOLUTION
 ENCODER_X10_NUMERATOR
 ENCODER_X10_DIVISOR
 ENCODER_X10_COUNTER

Die Verwendung und Skalierung der Parameter kann im CANopen Handbuch des servoTEC S2 nachgelesen werden.

Programm: start_send.pnc

```

1      $A
2      I9.1
3      N2:=1
4      R3:=0.5
5      PARAMETER.ST.A1.DIS:=N2      ; sende Parameter DIS an die servoTEC-Achse 1
                                   ; aus dem Ganzzahlregister N2
6      PARAMETER.ST.A1.IPEAK:=R3    ; sende Parameter IPEAK an die servoTEC-
                                   ; Achse 1 aus dem Realzahlregister R3
7      PARAMETER.ST.A1.O1:=1        ; sende Parameter O1 an servoTEC-Achse 1, die
                                   ; Definition erfolgt mit der Konstante „1“
8      T200
9      I10.1
10     N2:=0
11     R3:=1.5
12     PARAMETER.ST.A1.EN:=N2       ; sende Parameter EN an die servoTEC-Achse 1
                                   ; aus dem Ganzzahlregister N2
13     PARAMETER.ST.A1.IPEAK:=R3    ; sende Parameter IPEAK an die servoTEC-
                                   ; Achse 1 aus dem Realzahlregister R3
14     PARAMETER.ST.A1.O1:=0        ; sende Parameter O1 an servoTEC-Achse 1, die
                                   ; Definition erfolgt mit der Konstante „0“
15
16     PARAMETER.FT.A8.RunModeSelection :=N25
17                                     ;Sende Parameter RunModeSelection
                                   ; an die flexmoTEC-Achse 8 aus dem Ganzzahl-
                                   ; register N25
18     T200
19     JMP A
20     END
  
```

1.7.7.2 Ri:=PARAMETER.XX.Ai - Parameter lesen⁴

Ri:=PARAMETER.XX.Ai

<Register>:=Parameter.<Achstype>.<Achse>.<Parameter>

Beschreibung:

Mit diesem Befehl können spezielle Parameter (Antriebsparameter) von einem bestimmten Verstärker geholt werden.

HINWEIS Die Liste der möglichen Parameter einschließlich ihrer Bedeutung entnehmen Sie bitte bei einem „LV servoTEC“ der Hilfefunktion des Programms „DRIVE.EXE“, dem Dokument „MAN_DE_EN_1048937_servoTEC_ascii20.chm“ bzw. dem PDF-Dokument „Liste der ASCII-Objekte“. Bei einem Verstärker „LV flexmoTEC“ benutzen Sie bitte die entsprechenden mitgelieferten Handbücher! Für den servoTEC S2 finden Sie die Parameterliste beim Befehl Send-PARAMETER.S2.An.Parametername

Befehlsform:

Ni:= PARAMETER.ST.An.Parametername

Ri:= PARAMETER.ST.An.Parametername

Ni:= PARAMETER.FT.An.Parametername

Ni:= PARAMETER.S2.An.Parametername

HINWEIS Die Verwendung eines Ganzzahl- oder Realzahlregisters ergibt sich aus der Art des Parameters.

Programm: start_holen.pnc

```
1      $A
2      R1:=PARAMETER.ST.A1.PFB      ; hole Parameter PFB von der servoTEC-Achse 1
                                     und trage ihn in das Realzahlregister R1 ein
3      N2:=PARAMETER.ST.A1.IN1     ; hole Parameter IN1 von der servoTEC-Achse 1
                                     und trage ihn in das Ganzzahlregister N2 ein
4      N3:=PARAMETER.ST.A1.ACC     ; hole Parameter ACC von der servoTEC-Achse
                                     1 und trage ihn in das Ganzzahlregister N3 ein
5      R2:=PARAMETER.ST.A1.IPEAK   ; hole Parameter IPEAK von der servoTEC-
                                     Achse 1 und trage ihn in das Realzahlregister R2
                                     ein
6      T10
7      JMP A
8      END
```

⁴ servoTEC, flexmoTEC

1.7.7.3 RA_n.m:=R_i - Achsparameter laden aus einem Register

RA_n.m:=R_i

Beschreibung:

Mit Hilfe dieses Befehls können die Achsparameter mit Werten aus einem Register geladen werden.

Allgemeine Form : RA<Achsennummer>.<Parameternummer>:=R<Registernummer>

Die Auswahl des gewünschten Parameters „m“ erfolgt durch Eingabe nach der folgenden Tabelle:

7	→	Getriebefaktor
8	→	min. Verfahrbereich
9	→	max. Verfahrbereich
23	→	Referenzfahrtablauf
24	→	Drehrichtung
28	→	Achsendschalter ausgetauscht
62	→	Referenz-Offset

Befehlsform:

RA_n.m:=R_i

Beispiel

1	R2:=20	
2	RA1.7:=R2	<i>;setze den Getriebefaktor der Achse 1 auf den Inhalt des Registers 2</i>
3	R7:=10	
4	RA2.8:=R7	<i>;setze den min. Verfahrbereich der Achse 2 auf den Inhalt des Registers 7</i>
5	R23:=1000	
6	RA12.9:=R23	<i>;setze den max.. Verfahrbereich der Achse 12 auf den Inhalt des Registers 23</i>

HINWEIS Der Befehl überschreibt nicht nur den aktuellen (lokalen) Parameter, sondern auch den abgespeicherten Parameter, so dass beim Laden der Parameter von der PA-CONTROL der veränderte (neue) Parameter gelesen wird.

HINWEIS Bei einer Änderung des Getriebefaktors sollten, bevor die Achse verfahren wird, die Befehle „FAn:=...“ und „G100.An.“ aufgerufen werden, damit die Geschwindigkeit und die Beschleunigung dem neuen Getriebefaktor entsprechen

1.7.7.4 Ri:=RAn.m - Achsparameter in ein Register schreiben

Ri:=RAn.m

Beschreibung:

Mit Hilfe dieses Befehls können die Werte unterschiedlicher Achsparameter in ein R-Register geschrieben werden.

Dabei gilt:

Ri	→	Realzahlregister i
RA1.i	→	„i“-ter Achsparameter Achse 1
RA2.2	→	„2“-ter Achsparameter Achse 2
...	→	
RAn.i	→	„i“-ter Achseparameter Achse n

HINWEIS Die Nummer des Achs-Parameters entnehmen Sie bitte der Beschreibung der Achsparameter im Kapitel 5

Befehlsform:

Ri:=RAn.m

Beispiel

Programm: achs_para.pnc

```

1
2   R101:= RA1.1      ; „1“-ter Achsparameter Achse 1 (Verfahrgeschwindigkeit)
3   R102:=RA1.7      ; „7“-ter Achsparameter Achse 1 (Getriebefaktor)
4   R103:=RA1.8      ; „7“-ter Achsparameter Achse 1 (min. Verfahrbereich)
5
6   R201:= RA2.1      ; „2“-ter Achsparameter Achse 2 (Verfahrgeschwindigkeit)
7   R202:=RA2.7      ; „7“-ter Achsparameter Achse 2 (Getriebefaktor)
8   R203:=RA2.9      ; „7“-ter Achsparameter Achse 1 (max. Verfahrbereich)
9   END

```

1.7.8 Interpolation

1.7.8.1 G144 - Schalte die servoTEC Achse in den Interpolationsmode

G144	ab V5.13
------	----------

G144. A<Nr-servoTEC-Achse>.A<Nr-PLS7-Achse>

Beschreibung:

Die Interpolation in der PA-CONTROL erfolgt in der PLS7-Karte. Die Positionsvorgabe wird über das Puls-Richtungs-Interface von der PLS7-Karte auf die servoTEC-Verstärker (siehe Applikationsschrift „APP5014...“).realisiert

Der G144- Befehl schaltet zu Beginn der Interpolation die servoTEC Achse in den „Interpolationsmode“, und übernimmt die aktuelle Position der servoTEC Achse in die PLS-Achse (setzt PLS-Achse auf diese Position)

HINWEIS Der Befehl kann nur für servoTEC S2-Achsen und PLS7-Achsen verwendet werden.

Der G144-Befehl aktiviert den X10-Input auf dem servoTEC S2 Regler

Ein G29-Offset aus der servoTEC S2-Achse wird bei der Übernahme der Position in die PLS7-Achse mitübernommen und bei der Interpolation dann entsprechend berücksichtigt

Die Überwachung der Verfahrbereichsgrenzen erfolgt während der Interpolation nur in den PLS7-Achsen. Die PLS7-Achsen sollten die gleichen Verfahrbereichsgrenzen wie die dazugehörigen servoTEC-Achsen haben.

Beispiel

Siehe G145-Befehl oder Applikationsschrift „APP5014...“

1.7.8.2 G145 - Deaktiviere den Interpolationsmode der servoTEC Achse

G145

ab V5.13

G145. A<Nr-servoTEC-Achse>.A<Nr-PLS7-Achse>

Beschreibung:

Die Interpolation in der PA-CONTROL erfolgt in der PLS7-Karte. Die Positionsvorgabe wird über das Puls-Richtungs-Interface von der PLS7-Karte auf die servoTEC-Verstärker (siehe Applikationsschrift „APP5014...“).realisiert.

Der G145- Befehl deaktiviert am Ende der Interpolation den „Interpolationsmode“ der servoTEC Achse.

HINWEIS Der Befehl kann nur für servoTEC S2-Achsen und PLS7-Achsen verwendet werden.

Der G145-Befehl deaktiviert (sperrt) den X10-Input auf dem servoTEC S2 Regler

Ein G29-Offset, der bei dem G144-Befehl aus der servoTEC S2-Achse in die PLS7-Achse übernommen wurde, bleibt in der PLS7-Achse unverändert vorhanden.

Ein G29-Offset der mit einem G29-Befehl auf die PLS7-Achse ausgeführt wurde wird nicht in die servoTEC-Achse übernommen.

Beispiel

1	G25.A1 G25.A2 G90.A0	
2	A1:=10 A2:=20	;fahre A1-Achse auf Position 10 / 20
3	G100.B.100	;Bahnbeschleunigung für die komplette Kontur festlegen
4	G144.A1.A13	; schalte A1 in den Interpolationsmode, A13 dient als Quelle für Puls und Richtung
5	G144.A2.A14	; schalte A2 in Interpolationsmode, A14 dient als Quelle für Puls und Richtung
6	FB:=1000	;Bahngeschwindigkeit auf 1000AE/s einstellen
7	G01 A13:=40 A14:=30	;Linearinterpolation mit der A13- und A14-Achse von der aktuellen Position auf die Position A13=40 und A14=30
8	G01 A13:=100 A14:=100	;die Interpolation wird mit geänderter Geschwindigkeit ohne anzuhalten fortgesetzt
9	G01 A13:=10 A14:=20	Kontur wird beendet
10	G145.A1.A13	; deaktiviere den Interpolationsmode von A1
11	G145.A2.A14	; deaktiviere den Interpolationsmode von A2
12	END	

1.7.8.3 G100.B.n - IPO Bahnbeschleunigung⁵

G100.B.n

G100.B.n

Beschreibung:

G100.B.n dient zur Einstellung der Bahnbeschleunigung bei der Interpolation. Die Bahngeschwindigkeit **FB** muss vorher eingestellt werden.

HINWEIS Ist der Wert größer oder kleiner als die Grenzwerte, erfolgt die Fehlermeldung „Wert zu groß“ oder „Wert zu klein“.

Befehlsform:

G100.B.1500

G100.B.R2

Anwendung:

Mit Hilfe des Befehls G100.B.n kann, abweichend von den in den Parameterwerten abgelegten Werten für die Beschleunigung, eine der Situation angepasste Arbeitsweise erreicht werden. Die getroffene Festlegung gilt bis zur nächsten Anwendung des Befehls G100.B oder bis zum Verlassen des Automatikbetriebes.

HINWEIS Wird die Bahnbeschleunigung nicht festgelegt, so erfolgt die Ausgabe einer Fehlermeldung!

⁵ nur in Verbindung mit PA-CONTROL ausgestattet mit PLS7 / PLS9 möglich.

1.7.8.4 FB:=n - IPO Bahngeschwindigkeit⁶

FB:=n

FB:=1200

FB:=R4

Bei der Linearinterpolation wird mit der Bahngeschwindigkeit auf einer Geraden zur angegebenen Zielposition verfahren. Es können achsparallele und unter einem beliebigen Winkel verlaufende Bewegungen ausgeführt werden.

Die Bahngeschwindigkeit wird vor dem eigentlichen Interpolationsbefehl **G01** mit dem Befehl **FB:=n** eingestellt. Stehen mehrere Interpolationsbefehle hintereinander, so werden diese ohne anzuhalten abgearbeitet. Die Bahngeschwindigkeit kann während eines komplexen Ablaufs ohne Anhalten geändert werden.

HINWEIS Die Anwendung dieses Befehls ist nur in Verbindung mit PA-CONTROL ausgestattet mit PLS7 / PLS9 möglich.

Folgen Interpolationsbefehle im Programm unmittelbar aufeinander (keine anderen Befehle dazwischen), so werden diese Interpolationsbefehle an einem Stück abgearbeitet. Dabei können Bahngeschwindigkeits- und Linearinterpolationsbefehle sowie die Befehle der Gruppe G16i (Bedienung von Ausgängen während der Interpolation) gemischt werden.

HINWEIS Vor Ausführung der Interpolation muss unbedingt die Bahnbeschleunigung und die Bahngeschwindigkeit festgelegt werden (siehe Beispiel 1 auf Seite 127).

1.7.8.5 IPO Bahngeschwindigkeit definieren, eine Achse läuft frei mit

FB.i:=n
FB.i:=Rn

FB.2:=1234

FB.3:=R7

In Applikationen mit der PA-CONTROL wird bei Interpolationen oft gefordert, dass eine Achse parallel zur Interpolation mitläuft. Diese Achse soll aber die resultierende "IPO-Bahn" der anderen Achsen nicht beeinflussen.

Anwendung:

- die Flußmittelförderung soll in einem festen Verhältnis zur Bahngeschwindigkeit realisiert werden.

während der Interpolation soll eine Achse an bestimmten Punkten verfahren werden

⁶ nur in Verbindung mit PA-CONTROL ausgestattet mit PLS7 / PLS9 möglich.

1.7.8.6 G01 - Linearinterpolation mit 4 aus 16 Achsen⁷

G01

G01 A1:=(POS) A2:=(POS).....

Beschreibung:

Bei der Linearinterpolation wird mit der Bahngeschwindigkeit auf einer Geraden zur angegebenen Zielposition verfahren. Es können achsparallele und unter einem beliebigen Winkel verlaufende Bewegungen ausgeführt werden. Die Bahngeschwindigkeit wird vor **G01** mit dem Befehl **FBn** eingestellt. Stehen mehrere Interpolationsbefehle hintereinander, so werden diese ohne anzuhalten abgearbeitet. Die Bahngeschwindigkeit kann während eines komplexen Ablaufs ohne Anhalten geändert werden.

Der Übergang auf die neue Bahngeschwindigkeit erfolgt mit der voreingestellten Bahnbeschleunigung. Sie wird mit dem Befehl **G100.B.n** eingestellt und kann innerhalb einer Kontur nicht verändert werden.

HINWEIS Die Anwendung dieses Befehls ist nur in Verbindung mit PA-CONTROL ausgestattet mit PLS7 / PLS9 möglich.

Alle bei der Interpolation beteiligten Achsen müssen über eine PLS-Karte angesteuert werden.

Folgen Interpolationsbefehle im Programm unmittelbar aufeinander (keine anderen Befehle dazwischen), so werden diese Interpolationsbefehle an einem Stück abgearbeitet. Dabei können Bahngeschwindigkeits- und Linearinterpolationsbefehle sowie die Befehle der Gruppe G16i (Bedienung von Ausgängen während der Interpolation) gemischt werden.

HINWEIS Vor Ausführung der Interpolation muss unbedingt die Bahnbeschleunigung und die Bahngeschwindigkeit festgelegt werden (siehe Beispiel 1 auf Seite 127).

Befehlsform:

G01 → AiAn

G01 → Ari ARn

Anwendung:

Gleichzeitiges Verfahren von 2 bis 4 Achsen auf einer Gerade zwischen Start- und Zielposition.

⁷ nur in Verbindung mit PA-CONTROL ausgestattet mit PLS7 / PLS9 möglich.

Beispiel 1

1	G25.A1 G25.A2 G90.A0	
2	A1:=10	;fahre A1-Achse auf Position 10
3	A2:=20	;fahre A2-Achse auf Position 20
4	G100.B.100	;Bahnbeschleunigung für die komplette Kontur festlegen
5	FB:=1000	;Bahngeschwindigkeit auf 1000AE/s einstellen
6	G01 A1:=40 A2:=30	;Linearinterpolation mit der A1- und A2-Achse von der aktuellen Position auf die Position A1=40 und A2=30
7	FB:=800	;ändern der Bahngeschwindigkeit auf 800AE/s für die kommenden Interpolationsabschnitte mit der Rampe der vorgegebenen Beschleunigung
8	G01 A1:=100 A2:=100	;die Interpolation wird mit geänderter Geschwindigkeit ohne anzuhalten fortgesetzt
9	G01 A1:=10 A2:=20	Kontur wird beendet
10	END	

Beispiel 2

Komplexes Beispiel mit mehreren Linearinterpolationen nacheinander, um eine Kontur mit konstanter Bahngeschwindigkeit abzufahren. Dadurch, dass die Interpolationsbefehle unmittelbar aufeinander folgen (kein anderer Befehl dazwischen), werden die Strecken „a“ bis „f“ an einem Stück abgefahren.

1	O6:=0 I5.1	;Werkzeug hoch
2	G25.A1 G25.A2. G90.A0	;Referenz fahren mit Achse 1 und 2
3	A1:=30 A2:=20	;Werkstückkante anfahren
4	O6:=1	;Werkzeug senken
5	I6.1	;warten bis Werkzeug unten
6	FB:=500	;Bahngeschwindigkeit auf 500AE/s einstellen
7	G100.B.30	;Bahnbeschleunigung auf 30AE/ss einstellen
8	G01 A1:=40 A2:=40	;Seitenkante „a“ abfahren
9	G01 A1:=60 A2:=40	;Seitenkante „b“ abfahren
10	FB:=300	;Bahngeschwindigkeit auf 300AE/s einstellen ;hier wird nicht angehalten, es erfolgt mit der Rampe der voreingestellten Beschleunigung der Übergang in die neue Bahngeschwindigkeit
11	G01 A1:=70 A2:=30	;Seitenkante „c“ abfahren
12	G01 A1:=70 A2:=25	;Seitenkante „d“ abfahren
13	G01 A1:=60 A2:=20	;Seitenkante „e“ abfahren
14	G01 A1:=30 A2:=20	;Seitenkante „f“ abfahren
15	O6:=0	;Werkzeug hoch
16	A1:=5 A2:=5	;Grundstellung anfahren
17	END	

1.7.8.7 Linearinterpolation mit mitlaufender /mitlaufenden Achse(n)

G01 A1:=(POS) A2:=(POS).....

G01 A1:=100 A2:=100 A3:=50

G01 A1:=10 A2:=5 A3:=FB*R7

G01 A1:=20 A2:=10 A3:=0

HINWEIS Der Weg der Achsen, die nicht die Bahn beeinflussen (FB.2:=..., An:=FB*Ri), sollte kleiner gleich der resultierenden Bahnlänge sein, da sonst das IPO-Intervall verlängert wird, bis der Weg dieser Achsen fertig verfahren wurde.
In diesem Fall wird der Systemmerker SM8 gesetzt.

HINWEIS Die Beeinflussung der Bahn (FB.i:=...) kann nur ab der ersten Achse in aufsteigender Folge eingestellt werden.
z.B.: A1, oder A1 & A2, oder A1 & A2 & A3, oder alle.
Nicht möglich ist z.B. A2 & A3, oder A1 & A3.

HINWEIS Bei FB*Rn wird die Richtung für diese Achse aus Rn übernommen
Rn > 0 → Richtung positiv
Rn < 0 → Richtung negativ

HINWEIS Ergibt sich bei der Multiplikation des Achswegs aus dem Bahnweg, dem Faktor und dem Getriebefaktor (An:=FB*Ri) ein Anzahl Achsschritte, die größer sind als der Bahnschritte, so wird der Systemmerker SM8 gesetzt.

HINWEIS Ist der Getriebefaktor einer Achse, die mit Bahngeschwindigkeit verfahren wird (An:=FB**Ri), größer als der größte einer an der Interpolation beteiligten Achse, so kann der Achsweg größer als der Bahnweg werden. In diesem Fall wird auch der Systemmerker SM8 gesetzt.

Beispiel 1

1 R7:=0.5

2 R8:=-0.5

3

4 G01 A1:=10 A2:=5 A3:=FB*R7 ; Die Achse 3 wird mit halber Bahngeschwindigkeit in positiver Richtung verfahren

5 G01 A1:=20 A2:=10 A3:=FB*R8 ; Die Achse 3 wird mit halber Bahngeschwindigkeit in negativer Richtung verfahren

6

Beispiel 2

1	G25.A1 G25.A2 G25.A3	<i>; Referenzfahrt mit allen drei Achsen</i>
2	G90.A0	<i>; Absolutmaßsystem für alle drei Achsen</i>
3	FB.2:=1000	<i>; Bahngeschwindigkeit auf 1000mm/s einstellen, nur die beiden ersten Achsen beeinflussen die Bahn</i>
4	R4:=0.1	<i>; Geschwindigkeit für Achse 3 auf ein Zehntel der Bahngeschwindigkeit einstellen</i>
5	G01 A1:=100 A2:=100 A3:=50	<i>; die resultierende Bahnlänge = 141,42mm, die Achse 3 wird während dieses Intervalls um 50mm verfahren</i>
6	G01 A1:=200 A2:=200 A3:=0	<i>; die resultierende Bahnlänge = 141,42mm, die Achse 3 wird während dieses Intervalls nicht verfahren</i>
7	G01 A1:=300 A2:=300 A3:=FB*R4	<i>; die resultierende Bahnlänge = 141,42mm, die Achse 3 wird während dieses Intervalls um 100mm in positiver Richtung verfahren</i>
8	

1.7.8.8 IPOEND - Linearinterpolation abbrechen

IPOEND

IPOEND.Zähler.Grenzwert

IPOEND.Ni.Nn

IPOEND.N!i.N!n

Beschreibung:

Der Befehl IPOEND erlaubt dem Bediener eine Interpolation abzubrechen.

Funktion:

Bei jedem IPO-Befehl (G01) wird geprüft ob die Funktion IPOEND aktiviert wurde. Wenn nein, dann wird das Programm wie notiert fortgeführt. Wenn ja, dann wird geprüft, ob der Zähler kleiner als der Grenzwert ist ($N_i < N_n$). Für diesen Fall wird dieser IPO-Befehl ausgeführt.

Ist der Zähler gleich oder größer dem Grenzwert, so wird der IPO-Befehl nicht mehr bearbeitet und das Programm wird nach dem letzten IPO-Befehl fortgeführt.

Zu beachten:

- IPOEND ist im Normalfall inaktiv und wird am Ende einer Interpolation deaktiviert.
- Der Zähler und der Grenzwert können während der Interpolation nicht verändert werden.
- IPOEND setzt (zu Beginn der Bearbeitung) den Zähler auf 0.
- Es wird mindestens ein IPO-Befehl (G01) ausgeführt.

Beispiel

Programm 1

```

1      N7:=3
2      SUB FAHRE_IPO
      ...
6      N7:=5
7      SUB FAHRE_IPO
8      END

```

Unterprogramm

```

1      $ FAHRE_IPO
2      IPOEND.N2.N7
3      G100.B.100
4      FB:=100
5      G01 A1:=R1 A2:=R2
6      G01 A1:=R3 A2:=R4
7      G01 A1:=R5 A2:=R6
8      G01 A1:=R7 A2:=R8
9      G01 A1:=R9 A2:=R10
10     G01 A1:=R11 A2:=R12
11     END

```

Aktiviere IPOEND, d.h. es werden 3 oder 5 IPO-Befehle ausgeführt

;Bahnbeschleunigung für die komplette Kontur festlegen

;Bahngeschwindigkeit auf 100AE/s einstellen

;Linearinterpolation mit der A1- und A2-Achse

1.7.8.9 Rn:=FB - Aktuelle Bahngeschwindigkeit holen

Rn:=FB

Befehlsform:

Rn:=FB

Beschreibung:

Mit diesem Befehl kann parallel zu einer laufenden Interpolation aus einer anderen Task heraus (Parallelablauf) die aktuelle Bahngeschwindigkeit geholt und in ein Realzahlregister abgespeichert werden.

Diese Information stellt für die Applikation eine Möglichkeit dar, auf Grund der aktuellen gefahrenen Bahngeschwindigkeit Elemente der Anlage zu beeinflussen.

Aktion	Ergebnis
Interpolation aktiviert und es wird gefahren	Aktuelle Bahngeschwindigkeit
Beschleunigen oder bremsen während der Interpolation	Aktuelle Bahngeschwindigkeit in der Rampe
Interpolation wurde gestoppt (STOP PA-CONTROL)	0
Keine Interpolation aktiv	-1

Beispiel

```
1      G26.A1 G25.A2
...
11     RUN GET_AKTUELLE_FB      Start PAB-Programm GET_AKTUELLE_FB
12     R1:=1000
13     R2:=5000
14     R3:=3000
15     R4:=1000
16     $A
17     I8.1
18     G100.B.10000
19     FB:=R1
20     G01 A1:=1000 A2:=2000
21     FB:=R2
22     G01 A1:=2000 A2:=4000
23     FB:=R3
24     G01 A1:=3000 A2:=6000
25     FB:=R4
26     G01 A1:=4000 A2:=8000
27     T100
28     A1:=0 A2:=0
29     T100
30     JMP A
31     END
```

Programm GET_AKTUELLE_FP

```
1      $A
2      R5:=FB
3      JMP A
4      END
```

Hole die aktuelle Bahngeschwindigkeit und lege den Wert im Realzahlregister 5 ab.

1.7.8.10 Rn:=FBAn - Aktuelle Verahrgeschwindigkeit während einer Interpolation

Rn:=FBAn

Befehlsform:

Rn:=FBAn

Beschreibung:

Der Befehl liefert die aktuelle Geschwindigkeit der ausgewählten Achse während einer Interpolation. Die Geschwindigkeit wird entsprechend der Bahngeschwindigkeit und dann mit dem anteiligen Verhältnis der Achse zurückgeliefert.

1.7.8.11 G16x - Bedienung von Ausgängen bei der Interpolation

G16?

Beschreibung:

Mit den G160-Befehlen können synchronisiert zu den Interpolationsbefehlen Ausgänge oder Merker beeinflusst werden.

Allgemein:

G16x.[Zeit].[Element].[Zustand]

[Zeit] : Zeitangabe für die Aktion in n*10ms
 [Element] : O → Ausgang, M → Merker
 [Zustand] : 0 → wird zurückgesetzt, 1 → wird gesetzt

HINWEIS

- In einem Interpolationsintervall ist nur Speicherplatz für jeweils einen G160-, G161- oder G162-Befehl vorgesehen.
- Müssen mehrere Ausgänge beeinflusst werden, so kann das nur über logische Befehle und parallele Abläufe realisiert werden.
- Ist die Zeitdauer eines Interpolationsintervalls kürzer als in den G161- oder G162-Befehlen angegeben, so wird bei der Anwendung des G161-Befehl am Ende des Intervalls und beim G162-Befehl am Anfang des Intervalls das Element beeinflusst.

1.7.8.12 G160 - Bediene Ausgänge vor dem Start der Interpolation

Beispiel 1

<p>n G160.20.O123.1</p> <p>n+1 G01 A1:=10 A2:=20</p> <p>... G01 A1:=30 A2:=30</p>	<p><i>Setze den Ausgang O123 auf 1 und warte 200ms mit dem Start der nachfolgenden Interpolation</i></p> <p><i>; Linearinterpolation mit der A1- und A2-Achse von der aktuellen Position auf die Position A1=10 und A2=20</i></p> <p><i>; Linearinterpolation mit der A1- und A2-Achse von der aktuellen Position auf die Position A1=30 und A2=30. Die zweite Interpolation schließt sich unmittelbar und ohne Unterbrechung an die erste an!</i></p>
---	--

1.7.8.13 G161 / G162 - Bediene Ausgänge während der Interpolation

Beispiel 2

n G01 A1:=10 A2:=20
n+1 G161.1.O333.1

n+2 G01 A1:=30 A2:=30

 ... G01 A1:=60 A2:=40

 ... G162.5.O333.0

 ... G01 A1:=90 A2:=50

Setze den Ausgang O333 auf „1“, 10ms nach dem das nachfolgenden Interpolationsintervall begonnen wurde.

; Linearinterpolation mit der A1- und A2-Achse von der aktuellen Position auf die Position A1=30 und A2=30

; Linearinterpolation mit der A1- und A2-Achse von der aktuellen Position auf die Position A1=60 und A2=40

Setze den Ausgang O333 auf „0“, 50ms bevor das nächste Interpolationsintervall fertig ist.

Beispiel 3

n G01 A1:=10 A2:=20
n+1 G161.1.M34.1

 ... G162.5.M34.0

 ... G01 A1:=300 A2:=250

Setze den Merker M34 auf „1“, 10ms nach dem das nachfolgenden Interpolationsintervall begonnen wurde.

Setze den Merker M34 auf „0“, 50ms bevor das nächste Interpolationsintervall fertig ist.

Beispiel 4

n G01 A1:=10 A2:=20
n+1 G161.1.O333.1

n+2 G01 A1:=30 A2:=30

 ... G162.5.O334.1

 ... G01 A1:=60 A2:=40
 ... G162.5.M3.1

 ... G01 A1:=90 A2:=50

Setze den Ausgang O333 auf „1“, 10ms nach dem das nachfolgenden Interpolationsintervall begonnen wurde.

; Linearinterpolation mit der A1- und A2-Achse von der aktuellen Position auf die Position A1=30 und A2=30

Setze den Ausgang O334 auf „1“, 50ms bevor das nächste Interpolationsintervall fertig ist.

Setze den Merker M3 334 auf „1“, 50ms bevor das nächste Interpolationsintervall fertig ist.

1.7.8.14 JMP-LINE-IPO.Ni - Start einer Interpolation bei einer bestimmten Programmzeile

JMP-LINE-IPO.Ni

Beschreibung:

Mit diesem Befehl kann zu einer bestimmten Zeile im Programm gesprungen und von dort aus mit der Interpolation begonnen werden.

Voraussetzung:

- Es darf keine Interpolation aktiv sein
- In der Programmzeile, die als Sprungziel angegeben wird, muss ein G01-, FB- oder G160-Befehl stehen. Ist diese Bedingung nicht erfüllt, dann wird der Fehler "E577"="falsches Sprungziel" erzeugt.

HINWEIS Die Bahnbeschleunigung (G100.B...) und die Bahngeschwindigkeit (FB...) müssen vor Ausführung des Befehls „JMP-LINE-IPO“ auf sinnvolle Werte eingestellt werden, da im anderen fall die Ausgabe eines Fehlers Exxx erfolgt.

HINWEIS Ein FB-Befehl wird bei JMP-IPO-LINE bearbeitet und dann zum nächsten (IPO-)Befehl gewechselt.

HINWEIS G16x-Befehle werden nicht bearbeitet und es wird zum nächsten (IPO-)Befehl gewechselt. Die Wahrscheinlichkeit, dass eine Zeilennummer eines G16x-Befehls erkannt wird ist sehr gering, da bei der Bearbeitung der IPO-Befehle die G16x-Befehle im IPO-Intervall implementiert sind.
Ein G160-Befehl würde über eine Zeilennummer nur erkannt, wenn die Interpolation während der Startphase (Ladephase der G01-Intervalle) abgebrochen wird.

1.7.9 Motorstrom Verändern oder Begrenzen G101

G101

(G100.Achse.Wert)

Beschreibung:

Mit dem G101-Befehl kann der Motorstrom einer Achse verändert oder begrenzt werden.

Das maximale Drehmoment des Motors verhält sich nahezu linear zum Motorstrom.

Der G101-Befehl kann nur bei den folgenden Achstypen verwendet werden	
Gerät / Achstyp	Wertebereich [%]
PA-CONTROL-MP	10 bis 100
PA-CONTROL-servoTEC , LV-servoTEC	1 bis 300
LV-servoTEC S2	1 bis 300

Befehlsform:

G101.An.n

G101.An.Nn

Anwendung:

- Schrauben mit Schrittmotor oder Servomotor, bei definiertem Drehmoment.
- Kraftbegrenzung von Spindelachsen oder motorisch betriebenen Pressen.
- Kraftbegrenzung bei Achsbewegungen mit Kollisionsgefahr

HINWEIS Wird die Betriebsart AUTOMATIK nicht verlassen, so bleibt die Veränderung durch den G101-Befehl erhalten.
Erst nach verlassen der Betriebsart AUTOMATIK und danach beim Wiedereinstieg in die Betriebsart AUTOMATIK wird die Veränderung durch den G101-Befehl verworfen und die Achse auf den Parameterwert aus den Achs- oder Antriebsparametern eingestellt.

1.7.9.1 G101 - Motorstrom verändern bei einer PA-CONTROL-MP

G101

(G100.Achse.Wert)

Beschreibung:

Bei einer PA-CONTROL-MP kann der Motorstrom prozentual zwischen 10% und 100% zu dem eingestellten Parameterwert „Phasenstrom“ verändert werden.

HINWEIS Die Stromeinstellung ist nur während der Fahrt aktiv. Im Stillstand ist der Motor an der PA-CONTROL MP mit dem in den Motorparametern eingestellten „Phasenstrom im Stillstand“ bestromt.

Beispiel 1

1	G101.A1.50	<i>;50% vom Phasenstrom</i>
2	A1:=100	<i>;Position 100 anfahren mit reduziertem Moment</i>
3	END	

Beispiel 2

1	N1:=50	<i>;50% vom Phasenstrom</i>
2	G210.A1	<i>;aktiviert den Startpositioniermode für Achse 1</i>
3	A1:=100	<i>;Position 100 anfahren</i>
4	G230.1.A1.50	<i>;warte bis die Position >=50 erreicht ist</i>
5	G101.A1.N1	<i>;den Motorstrom entsprechend N1 einstellen</i>
6	G213.A1	<i>;warte bis Achsen in Position sind und aktiviere den Normalpositioniermode</i>
7	END	

1.7.9.2 G101 - Spitzenstrom verändern bei einer servoTEC-Achse

G101

(G100.Achse.Wert)

Beschreibung:

Bei einer servoTEC-Achse verändert der G101-Befehl den Spitzestrom IPEAK. Der Spitzenstrom wird auf den prozentualen Wert des Nennstromes IRMS eingestellt. Die mögliche Veränderung kann zwischen 1% bis 300% erfolgen.

HINWEIS Soll nach der Veränderung durch den G101-Befehl der Spitzenstrom IPEAK wieder auf den Wert aus den Parametern eingestellt werden, so muss dies durch einen weiteren G101-Befehl realisiert werden (siehe Beispiel).

Ob der G101-Befehl mit 300% (G101.A1.300) dem Motor schadet, liegt im Verantwortungsbereich des Anwenders.

Einstellung der Achsparameter für das Beispiel:

Nennstrom IRMS : 1,5 A
Spitzenstrom IPEAK : 2,0 A

Beispiel

1	<i>N1:=20</i>	<i>;IPEAK auf 20% von IRMS → 0,3 A</i>
2	<i>G210.A1</i>	<i>;aktiviert den Startpositioniermode für Achse 1</i>
3	<i>A1:=100</i>	<i>;Position 100 anfahren</i>
4	<i>G230.1.A1.50</i>	<i>;warte bis die Position >=50 erreicht ist</i>
5	<i>G101.A1.N1</i>	<i>;den Motorstrom entsprechend N1 auf 0,3 A begrenzen</i>
6	<i>G213.A1</i>	<i>;warte bis Achse in Position ist und aktiviere den Normalpositioniermode</i>
7	<i>G101.A1.133</i>	<i>; IPEAK auf 133% vom IRMS → 2,0 A einstellen (Begrenzung aufheben)</i>
8	<i>END</i>	

1.7.9.3 G101 - Strombegrenzung verändern bei einer servoTEC S2

G101

(G100.Achse.Wert)

Beschreibung:

Bei einer servoTEC S2 Achse gibt es für den Strom die Parameter (Antriebsparameter)

- Maximalstrom [A]
- Nennstrom [A]
- Strombegrenzung (current limitation) [A]

Der G101-Befehl wirkt auf den Parameter „Strombegrenzung“. Der G101-Befehl setzt beim servoTEC S2 den Parameter „Strombegrenzung“ auf den prozentualen Wert des Nennstromes. Die mögliche Veränderung kann zwischen 1% bis 300% erfolgen.

HINWEIS Soll nach der Veränderung durch den G101-Befehl die Strombegrenzung wieder inaktiv werden so muss dies durch einen weiteren G101-Befehl realisiert werden (siehe Beispiel). Da der G101-Befehl die Parameter Nennstrom und Maximalstrom nicht verändert, kann dies durch die Übergabe von 300% geschehen. Dabei nimmt der servoTEC S2 den prozentualen Wert oder den Maximalstrom des Verstärkers wenn der prozentuale Wert größer als der Maximalstrom sein sollte.

Einstellung der Achsparameter für das Beispiel servoTEC S2105:

Nennstrom	: 2,5 A
Maximalstrom	: 4,0 A
Strombegrenzung (current limitation)	: 5,0 A (servoTEC S2105)

Beispiel

1	<i>N1:=20</i>	<i>;Strombegrenzung auf 20% vom Nennstrom → Spitzenstrom = 0,5 A</i>
2	<i>G210.A1</i>	<i>;aktiviert den Startpositioniermode für Achse 1</i>
3	<i>A1:=100</i>	<i>;Position 100 anfahren</i>
4	<i>G230.1.A1.50</i>	<i>;warte bis die Position >=50 erreicht ist</i>
5	<i>G101.A1.N1</i>	<i>;den Motorstrom entsprechend N1 auf 0,5 A begrenzen</i>
6	<i>G213.A1</i>	<i>;warte bis Achse in Position ist und aktiviere den Normal- positioniermode</i>
7	<i>G101.A1.300</i>	<i>;Strombegrenzung auf 300% vom Nennstrom → 7,5 A einstellen da der Verstärker nur 5,0 A Maximalstrom treiben kann wird die Strombegrenzung auf 5,0 A eingestellt</i>
8	<i>END</i>	

1.8 Eingänge, Ausgänge und Merker

1.8.1 Eingänge

1.8.1.1 In.m - Warten auf log. Zustand des Eingangs

In.m

I (Nummer).(Zustand)

Beschreibung:

Bei dem Befehl I (Input) überprüft die Steuerung, ob der im Operand angegebene Eingang **n** den Zustand **m** hat. Ist die Abfrage nicht wahr, wartet die Steuerung bis die Bedingung im Operator **m** erfüllt ist.

Befehlsform:

I12.1

I12.0

Anwendung:

Abfrage von Näherungsschaltern, Tastern, Schaltern, usw. ...

Beispiel:

1 I8.1
2 I4.0
3 END

*Steuerung wartet bis Eingang 8 logisch 1 ist
Steuerung wartet bis Eingang 4 logisch 0 ist*

1.8.2 Ausgänge

1.8.2.1 On.m - Warten auf log. Zustand des Ausgangs

On.m

O (Nummer).(Zustand)

Beschreibung:

Bei dem Befehl **O** (Output) überprüft die Steuerung, ob der im Operand angegebene Ausgang **n** den Zustand **m** hat. Ist die Abfrage nicht wahr, wartet die Steuerung bis die Bedingung im Operator **m** erfüllt ist.

HINWEIS Dieser Befehl ist nur sinnvoll, wenn mehrere parallele Ablaufprogramme gestartet sind. Denn nur ein anderer paralleler Ablauf kann den Ausgang verändern, während dieser Ablauf auf den Ausgang wartet.

Befehlsform:

O12.1

O12.0

Anwendung:

Abfrage von Ausgängen

Beispiel:

1 i O8:=1

*;Steuerung setzt den Ausgang 8
(z.B. für den zeitgenauen Start eines Parallelablaufes)*

2 O7.1

*; Steuerung wartet bis Ausgang 7 logisch 1 ist
(wird von einem parallel arbeitenden Programm gesetzt)*

3 END

1.8.2.2 On:=m - Ausgang setzen / rücksetzen

On:=m

O (Nummer):=(Zustand)

Beschreibung:

Mit dem Befehl **O** (Output) können die zur Verfügung stehenden Ausgänge beeinflusst werden. Wird der Ausgang **n** auf den Zustand **m = 1** gesetzt, schaltet dieser durch, d.h. der Transistor ist leitend und bleibt solange eingeschaltet, bis dieser mit **m = 0** zurückgesetzt und der Transistor gesperrt wird.

Befehlsform:

On:=1

On:=0

On:=Mm Zustand des Merker m wird für Ausgang n übernommen

On:=Im Zustand des Eingangs m wird für Ausgang n übernommen

On:=Om Zustand des Ausgangs m wird für Ausgang n übernommen

Anwendung:

Schalten von Relais oder pneumatischen Ventilen, Signalweitergabe an andere elektrische Geräte.

Beispiel:

1	O8:=1	<i>;Ausgang 8 wird auf 1 gesetzt</i>
2	O7.0	<i>;Steuerung wartet bis Ausgang 7 logisch 0 ist</i>
3	END	

1.8.3 Merker

1.8.3.1 Mn:=m - Merker setzen / rücksetzen

Mn:=m

M (Nummer):=(Zustand)

Beschreibung:

Mit dem Befehl **M** können die zur Verfügung stehenden Merker beeinflusst werden. Wird der Merker **n** in den logischen Zustand $m=1$ gesetzt, so bleibt er so lange in diesem Zustand, bis er mit $m=0$ zurückgesetzt wird (siehe auch Befehlsbeschreibung Vergleichsoperationen).

Befehlsform:

M1:=0		Rücksetzen Merker 1
M1:=1		Setzen Merker 1
Mn:=Mm		Zustand des Merker m wird für Merker n übernommen
Mn:=Im		Zustand des Eingangs m wird für Merker n übernommen
Mn:=Om		Zustand des Ausgangs m wird für Merker n übernommen
Mn:=SMm		Zustand des Systemmerkers wird für Merker übernommen (ab V5.17)
M!n:=M!m		Indirekte Adressierung

Anwendung:

- Abspeichern von Zuständen, Verarbeitungssituationen
- Synchronisieren von Parallelabläufen

1.8.3.2 M!2:=0 - Direkte oder Indirekte Adressierung von Merkern

Ab der Version 3.72 ist es möglich die Merker indirekt zu adressieren. Der Merker für die indirekte Adressierung wird durch ein Ausrufezeichen nach dem Bezeichner **M** gekennzeichnet. Dadurch zeigt der Merker auf den Inhalt des Ganzzahlregisters mit dem gleichen Index.

Befehlsform:

M!2:=0	→	Indirekte Adressierung beim Merkerersetzen
--------	---	--

Vor der Anwendung muss unbedingt eine Deklaration der indirekten Adressierung erfolgen!

N15:=10	→	der Inhalt des Ganzzahlregisters N15 ist 10
M!15:=1	→	Die Zahl 15 nach dem Ausrufezeichen zeigt auf den Inhalt des Ganzzahlregisters N15 , dessen Inhalt =10 ist. Es wird der Merker M10 auf "1" gesetzt.

Anwendung:

- Parametrierung von Programmen

Beispiel

1	<i>I8.1</i>	<i>;Steuerung wartet bis Eingang 8 logisch 1 ist</i>
2	<i>M12:=1</i>	<i>;Merker 12 wird auf logisch 1 gesetzt</i>
3	<i>M10:=0</i>	<i>;Merker 10 wird auf logisch 0 gesetzt</i>
4	<i>N15:=10</i>	<i>;Register N15 für indirekte Adressierung laden</i>
5	<i>M!15:=1</i>	<i>;M10 wird durch indirekte Adressierung von M15 auf logisch 1 gesetzt</i>
6	<i>T100</i>	<i>;warte 1 Sekunde</i>
7	<i>N16:=10</i>	<i>;Register N16 für indirekte Adressierung laden</i>
8	<i>M!16:=0</i>	<i>;M10 wird durch indirekte Adressierung von M16 auf logisch 0 gesetzt</i>
9	<i>END</i>	

1.8.3.3 Mn.m - Warten auf log. Zustand des Merkers

Mn.m

M (Nummer).(Zustand)

Beschreibung:

Bei dem Befehl **M** (Merker) überprüft die Steuerung, ob der im Operand angegebene Merker **n** den Zustand **m** hat. Ist die Abfrage nicht wahr, wartet die Steuerung bis die Bedingung **m** im Operator erfüllt ist.

Es ist möglich die Merker indirekt zu adressieren. Der Merker für die indirekte Adressierung wird durch ein Ausrufezeichen nach dem Bezeichner **M** gekennzeichnet. Die Zahl **n** nach dem Ausrufezeichen weist auf das Ganzzahlregister mit der gleichen Adresse hin. Der Inhalt des Registers ergibt die eigentliche Merkeradresse. Siehe dazu das folgende Beispiel.

N10:=5	→	der Inhalt des Ganzzahlregisters N10 ist 5
M!10.1	→	Die Zahl 10 nach dem Ausrufezeichen zeigt auf den Inhalt des Ganzzahlregisters N10 , dessen Inhalt =5 ist. Es wird nun gewartet, bis der Zustand des Merkers M5 gleich "1" ist
M5.1	→	Der Befehl M5.1 ist identisch zum vorigen Beispiel mit indirekter Adressierung

Befehlsform:

M1.1	→	Abfragen des Merkers 1 auf den Zustand logisch 1
M1.0	→	Abfragen des Merkers 1 auf den Zustand logisch 0
M!3.1	→	Indirekte Adressierung bei Merkerabfrage

Anwendung:

Synchronisieren von Parallelabläufen.

Beispiel

1	O8:=1	<i>;Steuerung setzt den Ausgang 8 (z.B. Triggerung für Parallelablaufsteuerung).</i>
2	M7.1	<i>;Steuerung wartet bis Eingang 7 logisch 1 ist (von Parallelablaufsteuerung gesetzt wurde)</i>
3	N10:=5	<i>;N-Register für indirekte Adressierung laden</i>
4	M!10.1	<i>;warte bis M5 den logischen Zustand 1 hat</i>
5	END	

1.8.3.4 > = < Vergleichsergebnisse Merker

> = <

Befehlsform: siehe folgende Tabelle

Direkte Adressierung		indirekte Adressierung	
Mn:=Rm=K	Mn:=Nm=K	M!n:=Rm=K	M!n:=Nm=K
Mn:=Rm>K	Mn:=Nm>K	M!n:=Rm>K	M!n:=Nm>K
Mn:=Rm<K	Mn:=Nm<K	M!n:=Rm<K	M!n:=Nm<K
Mn:=Rm=Ri	Mn:=Nm=Ni	M!n:=Rm=Ri	M!n:=Nm=Ni
Mn:=Rm>Ri	Mn:=Nm>Ni	M!n:=Rm>Ri	M!n:=Nm>Ni
Mn:=Rm<Ri	Mn:=Nm<Ni	M!n:=Rm<Ri	M!n:=Nm<Ni
Mn:=R!m=Ri	Mn:=N!m=Ni	M!n:=R!m=Ri	M!n:=N!m=Ni
	Mn:=Nn<>K		

HINWEIS Siehe dazu auch Abschnitt 1.10.3, Seite 165.

1.8.3.5 Schreibe Merker-Nummer nach Vergleich in ein N-Register

GETM

ab V5.07

Befehlsform:

Ni:=GETM.n.m.l

Ni:=GETM.<Ab Merker-Nr>.<Bis Merker-Nr>.<Vergleichstatus>

N3:=GETM.1.128.1

Anwendung:

Der Befehl schreibt in das N-Register (Ni) die Nummer des ersten Merkers aus dem zu überprüfenden Merkerbereich (<Ab Merker-Nr>.<Bis Merker-Nr>) der dem Vergleichsstatus entspricht

HINWEIS Es können maximal 128 Merker abgeprüft werden
(<Ab Merker-Nr>.<Bis Merker-Nr>)

Wird im abgeprüften Bereich kein Merker gefunden, der dem Vergleichsstatus entspricht, so wird das N-Register auf „0“ gesetzt

Beispiel

```
1 LD SM10.0
2 RES M1.M48
3 $A
4 N3:=GETM.2.39.1
5 O1:=1
6 T10
7 O1:=0
8 T10
9 JUMP A
10 END
```

1.9 Zeiten, Datum und Uhrzeit

1.9.1 Warten

1.9.1.1 Tn - Verweilzeit

T

T (Zeit in 10/Millisekunden)

Beschreibung:

Der Befehl **T** lässt die Steuerung um die im Operand angegebene Zeit verweilen.

Hierbei ist:

die kürzeste Zeit	n	=	1	=	10 ms
die längste Zeit	n	=	$2^{31} = 2.147.483.648$	=	21.474.836.480 ms

Befehlsform:

T1000

TN12

Anwendung:

Wartezeiten, Ausgänge für definierte Zeiten gesetzt halten.

Beispiel

1	T1	;Verweilzeit 10ms
2	T300	;Verweilzeit 3s
3	TN2	;Verweilzeit entsprechend des Inhaltes von N2
3	END	

1.9.2 Datum und Uhrzeit

1.9.2.1 DATE:= - Setzen des Datums aus dem Anwenderprogramm

DATE:=

DATE:=[Tag].[Monat].[Jahr].[Wochentag]

Beschreibung:

Mit Hilfe dieses Befehls kann aus dem Anwenderprogramm heraus das Datum und der Wochentag der Real-Zeit-Uhr auf der CPU4 gesetzt werden.

Befehlsform:

Date:=Ni.Nj.Nn.Nm

HINWEIS Bei der Ausführung des Befehls wird überprüft, ob die Werte für das Datum gültig sind. Ist das nicht der Fall so wird der Fehler E532 „Wert außerhalb Bereich“ gesetzt.

Ziffer	Wochentag
0	Sonntag
1	Montag
2	Dienstag
3	Mittwoch
4	Donnerstag
5	Freitag
6	Samstag

Beispiel

1 N1:=15
2 N2:=8
3 N3:=2003
4 N4:=5
5 DATE:=N1.N2.N3.N4

Setzt das Datum auf Freitag, den 15.August 2003

1.9.2.2 TIME:= - Setzen der Zeit aus dem Anwenderprogramm

TIME:=

TIME:=[Stunden].[Minuten].[Sekunden]

Beschreibung:

Mit Hilfe dieses Befehls kann aus dem Anwenderprogramm heraus die Uhrzeit der Real-Zeit-Uhr auf der CPU4 gesetzt werden.

Befehlsform:

TIME:=Ni.Nj.Nn

HINWEIS Bei der Ausführung des Befehls wird überprüft, ob die Werte für die Zeit gültig sind. Ist das nicht der Fall so wird der Fehler E532 „Wert außerhalb Bereich“ gesetzt.

Beispiel

1 N1:=14

2 N2:=50

3 N3:=23

4 TIME:=N1.N2.N3

Setzt die Uhrzeit auf 14:50:23 Uhr

1.9.2.3 S0:=TIME - Auslesen der Echtzeit-Uhr

S0:=TIME

S0:=TIME[.YEAR][.MONTH][.DAY][.HOUR][.MIN][.SEC][.WD][.YD][.W]

Beschreibung:

Mit diesem Befehl kann auf die Uhrzeit der Steuerung zugegriffen werden. Bei der CPU4 ist diese als Real-Time-Clock Baustein auf der Leiterplatte vorhanden.

Ist am CANopen-Bus ein IEF-Touch-Screen Bedienkonsole angeschlossen (siehe Bedienungsanleitung PA-CONTROL Kapitel Optionen), so kann die Uhrzeit von der Bedienkonsole zyklisch an die PA-CONTROL werden und mit diesem Befehl weiterverarbeitet werden.

Die Ausgabe der Informationen erfolgt grundsätzlich in die globale S0-Zeichenkette. Die Elemente in den [] sind optional und können gemischt werden. Als Trennzeichen zwischen den Elementen sind Punkt, ein Bindestrich oder ein Doppelpunkt möglich. Das verwendete Trennzeichen wird an der gleichen Stelle in die S0-Zeichenkette übernommen

Die folgenden Eingabebeispiele beziehen sich auf den 04.Dezember 2002, 08:41:53 Uhr

Befehlsform	Ausgabe
S0:=TIME.DAY.MONTH.YEAR	04.12.2002
S0:=TIME.DAY.MONTH.YEAR-HOUR:MIN:SEC	04.12.02-08:41:53
S0:=TIME.HOUR:MIN:SEC	08:41:53
S0:=TIME.W	49
S0:=TIME.W-WD	49-3

Beispiel

1	S0:=TIME.DAY.MONTH.YEAR	<i>;Hole Tag, Monat und Jahr aus der Real-Zeit-Uhr und schreibe diese Information in die globale Zeichenkette S0</i>
2	<i>G11.0</i>	<i>;Ablaufanzeigen ausschalten</i>
3	<i>G500.0</i>	<i>;schaltet die G500-Befehle auf das aktuelle Anzeigemedium um (LC-Anzeige oder bei „Simulation der Frontplatte“ die serielle Schnittstelle)</i>
4	<i>G501</i>	<i>;löscht die Anzeige und stellt den Cursor in die linke obere Ecke (1. Spalte, 1. Zeile)</i>
5	<i>G510.AKTUELLES DATUM</i>	<i>;Ausgabe des Textes „AKTUELLES DATUM“</i>
6	<i>G503.17.1</i>	<i>;Positioniere den Cursor in die 17.Spalte, 1.Zeile</i>
7	<i>G520.S0</i>	<i>;Gib den Inhalt der globalen Zeichenkette aus</i>
8	<i>T500</i>	<i>;Warte 5 Sekunden</i>
9	<i>END</i>	

1.10 Mathematik

1.10.1 Organisation der Befehle

1.10.1.1 Direkte und indirekte Adressierung

Beschreibung:

Register können bei Wertzuweisungen und den Grundrechenarten direkt und indirekt adressiert werden.

Direkte Adressierung:

- Dem Buchstaben R(N) folgt eine Zahl (1-8192), die das gewünschte Register kennzeichnet.

Indirekte Adressierung:

- Dem Buchstaben R(N) folgt ein „!“ und dann eine Zahl (1-8192), die auf das Register verweist, dessen Inhalt das gewünschte Register kennzeichnet.

1.10.1.2 Zuweisungen

Es gilt allgemein bei Zuweisungen:

Ziel	Quelle
Register	→ Konstante (K)
	→ Register
	→ Absolutposition einer Achse (nur bei Realzahlregistern)

Es gilt allgemein bei Rechenoperationen:

Zielloperand	Quelloperand 1	Operation	Quelloperand 2
Register	→ Register	→ + - * /	→ vorzeichenlose Konstante Register

HINWEIS Das Ergebnis ist von der Operation und dem Vorzeichen der Registerinhalte abhängig.

K (Konstante) wird sowohl bei der Beschreibung für Ganzzahlen, als auch für Realzahlen benutzt. Bei der realen Anwendung kann nur der zulässige Zahlentyp benutzt werden.

Die Quelloperanden bleiben unverändert. Ausnahme ist, wenn der Zielloperand auch der Quelloperand ist.

Beispiele

Beispiel 1:

R1:=5
R2:=3
R3:= R1+R2

Danach haben die Register folgende Werte : R1=5, R2=3, R3=8

Beispiel 2:

R1:=R1+R2

Danach haben die Register folgende Werte : R1=8, R2=3.

1.10.2 Mathematische Befehle

1.10.2.1 Rn/Nn:= Register laden

Rn:=
Nn:=

Beschreibung:

Mit Hilfe dieser Befehle können Werte in die Register geladen werden.

HINWEIS Innerhalb des Befehls sind keine Leerstellen zugelassen.

Befehlsform: (Auszug aus den Kombinationsmöglichkeiten)

Nn:=K	N!n:=K	Nn:=Nm	Nn:=Rm	Nn:=Ai	Nn:=SNn
Rn:=K	R!n:=K	Rn:=Rm	Rn:=Nm	Rn:=Ai	Rn:=SRn

Beispiel

1	R2:=45	;Realzahlregister 2 mit 45 laden
2	R3:=R2	;Realzahlregister 3 mit Wert von Realzahlregister 2 laden
3	R!2:=100.79	;Realzahlregister 45 mit 100.79 laden
5	R5:=A1	;Realzahlregister 5 mit Absolutposition der A1-Achse laden
6	N7:=87	;Ganzzahlregister 7 mit Wert 87 laden
7	N65:=N7	;Ganzzahlregister 65 mit Wert aus Ganzzahlregister 7 laden
8	N1:=R45	;Ganzzahlregister 1 mit Wert aus Realzahlregister 45 laden, Nachkommastellen werden abgeschnitten. In unserem Beispiel bedeutet das, der Inhalt von N1= 100!
9	R25:=N65	;Realzahlregister 25 mit Wert aus Ganzzahlregister 65 laden
10	N2:=4	;Ganzzahlregister 2 mit 4 laden
11	N!2:=17	;Ganzzahlregister 4 (N2=4) mit 17 laden
12	N24:=SN12	;Übernahme der Nummer der fehlerhaften Achse
13	R12:=SR32	;Übernahme der Absolutposition der Achse A2 in das Realzahlregister R11
14	END	

1.10.2.2 Lade Register mit einem binärem Wert

Nn:=2#xxxxxxx

Beschreibung:

Mit Hilfe dieser Befehle kann eine Konstante in Form eines binären Wertes in ein Ganzzahlregister geladen werden.

Befehlsform:

N2:=2#0011000

N3:=2#1010

N4:=N2+2#1010

HINWEIS Innerhalb des Befehls sind keine Leerstellen zugelassen.

1.10.2.3 Lade Register mit einem hexadezimalen Wert

Nn:=16#yyy

Beschreibung:

Mit Hilfe dieser Befehle kann eine Konstante in Form eines binären Wertes in ein Ganzzahlregister geladen werden.

Befehlsform:

N2:=16#64

N3:=16#3E8

N4:=N2+16#0A00

N5:=N2&16#000A

N6:=N4&N5

HINWEIS Innerhalb des Befehls sind keine Leerstellen zugelassen.

1.10.2.4 Rn:=Rm+K - Addition

+

Befehlsform:

Rn:=Rm+K

R!n:=Rm+K

Rn:=Rm+Ri

Rn:=R!m+Ri

Nn:=Nm+K

Nn:=Ni+Nm

N!n:=Nm+K

Nn:=N!m+Ni

Beispiel

1	<i>R2:=R2+57</i>	<i>;addiert 57 zum Inhalt von R2</i>
2	<i>R2:=R4+R3</i>	<i>;addiert den Inhalt von R4 und den Inhalt von R3 und weist das Ergebnis R2 zu</i>
3	<i>R1:=5</i>	
4	<i>R5:=100</i>	
5	<i>R24:=R!1+R2</i>	<i>;addiert den Inhalt von R5 und den Inhalt von R2 und weist das Ergebnis Register R24 zu Inhalt von R1 = 5. Durch die indirekte Adressierung wird also auf den Inhalt von R5 zugegriffen. Der Inhalt von R5 und R2 wird addiert und das Ergebnis R24 zugewiesen. R1, R5 und R2 bleiben unverändert.</i>
6	<i>N3:=N3+87</i>	<i>;addiert 87 zum Inhalt von N3</i>
7	<i>N3:=N3+N5</i>	<i>;addiert zu dem Inhalt von N3 den Inhalt von N5</i>
8	<i>END</i>	

1.10.2.5 Rn:=Rm-K - Subtraktion

-

Befehlsform:

Rn:=Rm-K

R!n:=Rm-K

Rn:=Rm-Ri

Rn:=R!m-Ri

Nn:=Nm-K

Nn:=Nm-Ni

N!n:=Nm-Ni

Nn:=N!m-Ni

Nn:=K-Nm

Rn:=K-Rm

Beispiel

1	<i>R2:=R2-57</i>	<i>;subtrahiert 57 vom Inhalt von R2</i>
2	<i>R2:=R2-R3</i>	<i>;subtrahiert vom Inhalt von R2 den Inhalt von R3</i>
3	<i>R1:=5</i>	
4	<i>R5:=100</i>	
5	<i>R24:=R!1-R2</i>	<i>;subtrahiert vom Inhalt von R5 den Inhalt von R2 und weist das Ergebnis Register R24 zu. Der Inhalt von R1 = 5. Durch die indirekte Adressierung wird also auf den Inhalt von R5 zugegriffen. Von diesem Wert wird nun der Inhalt von R2 subtrahiert und das Ergebnis R24 zugewiesen. R1, R5 und R2 bleiben unverändert.</i>
6	<i>N23:=N23-99</i>	<i>;subtrahiert 99 vom Inhalt von N23</i>
7	<i>N23:=N23-N4</i>	<i>;subtrahiert vom Inhalt von N23 den Inhalt von N4</i>
8	<i>END</i>	

1.10.2.6 Rn:=Rm*K - Multiplikation

*

Befehlsform:

Rn:=Rm*K

R!n:=Rm*K

Rn:=Rm*Ri

Rn:=R!m*Ri

Nn:=Nm*K

Nn:=Nm*Ni

N!n:=N!m*Ni

Beispiel

1	$R2:=R2*4$	<i>;multipliziert den Inhalt von R2 mit 4 und weist das Ergebnis R2 zu</i>
2	$R4:=R2*R3$	<i>;multipliziert den Inhalt von R2 mit dem Inhalt von R3 und weist das Ergebnis R4 zu</i>
3	$R1:=5$	
4	$R5:=100$	
5	$R24:=R!1*R2$	<i>;multipliziert den Inhalt von R5 mit dem Inhalt von R2 und weist das Ergebnis Register R24 zu</i>
6	$N2:=N2*4$	<i>;multipliziert den Inhalt von N2 mit 4 und weist das Ergebnis N2 zu</i>
7	$N2:=N3*N6$	<i>;multipliziert den Inhalt von N3 mit dem Inhalt von N6 und weist das Ergebnis N2 zu</i>
8	END	

1.10.2.7 Rn:=Rm/K - Division

/

HINWEIS Bei Division durch 0 entsteht ein Systemfehler, der durch die Meldung „E005“ auf dem Display ausgegeben wird.

Befehlsform:

Rn:=Rm/K

R!n:=Rm/K

Rn:=Rm/Ri

Rn:=R!m/Ri

Nn:=Nm/K

N!n:=Nm/K

Nn:=Nm/Ni

Nn:=N!m/Ni

Rn:=K/Rm

Nn:=K/Nm

Beispiel

- | | | |
|---|---------------------|--|
| 1 | <i>R2:=R2/3</i> | <i>;dividiert den Registerinhalt von R2 durch 3 und weist das Ergebnis R2 zu</i> |
| 2 | <i>R4:=R2/R3</i> | <i>;dividiert den Registerinhalt von R2 durch den Registerinhalt von R3 und weist das Ergebnis R4 zu</i> |
| 3 | <i>R1:=5</i> | |
| 4 | <i>R5:=100</i> | |
| 5 | <i>R24=R!1/R2</i> | <i>;dividiert den Registerinhalt von R5 durch den Registerinhalt von R2 und weist das Ergebnis R24 zu</i> |
| 6 | <i>N34:=N34/2</i> | <i>;dividiert den Inhalt von N34 durch 2 und weist das Ergebnis N34 zu. Stellen nach dem Komma werden abgeschnitten.</i> |
| 7 | <i>N25:=N67/N88</i> | <i>;dividiert den Inhalt von N67 durch den Inhalt von N88 und weist das Ergebnis N25 zu. Stellen nach dem Komma werden abgeschnitten</i> |
| 8 | <i>END</i> | |

1.10.2.8 Winkelfunktionen

Einleitung:

Die Winkelfunktionen Sinus, Cosinus und Tangens sowie ihre Umkehrfunktionen Arcussinus, Arcuscossinus und Arcustangens sind im Befehlsvorrat der PA-CONTROL verfügbar. Die Eingabe des Wertes erfolgt in Grad. Die Winkelfunktionen sind in Verbindung mit R-Register und konstanten Werten einsetzbar.

1.10.2.9 SIN/ASIN - Sinusfunktionen

SIN / ASIN

Befehlsform:

Rn:=SIN.K

Rn:=SIN.Ri

Rn:=SIN.R!i

Rn:=ASIN.K

Rn:=ASIN.Ri

Rn:=ASIN.R!i

Beispiel

1	R43:=70	
2	R70:=45	
3	R100:=SIN.10	<i>;Sinus vom Winkel 10° in das Register R100 schreiben (R100= 0.174)</i>
4	R101:=SIN.R43	<i>;Sinus vom Winkel 70° in das Register R101 schreiben (R101=0.940)</i>
5	R102:=SIN.R!43	<i>;Sinus vom Winkel 45° in das Register R102 schreiben (R102=0.707)</i>
6	R200:=ASIN.R100	<i>;Arcussinus vom Registerinhalt von R100 in Register R200 schreiben (R200:=10)</i>
7	END	

1.10.2.10 COS/ACOS - Cosinusfunktionen

COS / ACOS

Befehlsform: (Auszug aus den Kombinationsmöglichkeiten)

Rn:=COS.K

Rn:=COS.Ri

Rn:=COS.R!i

Rn:=ACOS.K

Rn:=ACOS.Ri

Rn:=ACOS.R!i

Beispiel

1	R43:=70	
2	R70:=-210	
3	R100:=COS.10	<i>;Cosinus vom Winkel 10° in das Register R100 schreiben (R100= 0.985)</i>
4	R101:=COS.R43	<i>;Cosinus vom Winkel 70° in das Register R101 schreiben (R101=0.342)</i>
5	R102:=COS.R!43	<i>;Cosinus vom Winkel -210° in das Register R102 schreiben (R102=-0.866)</i>
6	R200:=ACOS.R101	<i>;Arcuscosinus vom Registerinhalt von R101 in Register R200 schreiben (R200:=70)</i>
7	END	

1.10.2.11 TAN/ATAN - Tangensfunktionen

TAN / ATAN

HINWEIS Der Wertebereich für den Tangens ist durch den Anwender selbst zu definieren. Nicht zulässige Werte ($90^\circ + x * 180^\circ$, für $x = 0, 1, 2, \dots$) führen zu einem undefinierten Registerwert.

Befehlsform: (Auszug aus den Kombinationsmöglichkeiten)

Rn:=TAN.K

Rn:=TAN.Ri

Rn:=TAN.R!i

Rn:=ATAN.K

Rn:=ATAN.Ri

Rn:=ATAN.R!i

Beispiel

1	R43:=70	
2	R70:=-210	
3	R100:=TAN.10	<i>;Tangens vom Winkel 10° in das Register R100 schreiben (R100= 0.176)</i>
4	R101:=TAN.R43	<i>;Tangens vom Winkel 70° in das Register R101 schreiben (R101=2.747)</i>
5	R102:=TAN.R!43	<i>;Tangens vom Winkel -210° in das Register R102 schreiben (R102=-0.577)</i>
6	R200:=ATAN.R102	<i>;Arcustangens vom Registerinhalt von R101 in Register R200 schreiben (R200:=-30)</i>
7	END	

1.10.2.12 SQRT - Wurzelfunktion

SQRT

HINWEIS Der Zahlenwert unter der Wurzel darf nicht negativ werden! Der Zahlenwert für die Berechnung muss immer aus einem Register genommen werden!

Befehlsform:

Rn:= SQRT.Ri

Rn:= SQRT.R!i

Beispiel

1 R43:=70

2 R70:=210

3 R101:=SQRT.R43

4 R102:=SQRT.R!43

5 END

;Wurzel der Zahl 70 in das Register R101 schreiben (R101=8.367)

;Wurzel der Zahl 210 in das Register R102 schreiben (R102=14.491)

1.10.2.13 INT - Ganzzahliger Anteil

INT

Befehlsform:

Rn:=INT.Rn

Beispiel

1 R43:=70

2 R70:=210.23445

3 R10:=INT.R70

4 END

;speichert den ganzzahligen Anteil des Realzahlregister R70 in Realzahlregister R10 (Ergebnis: 210)

1.10.2.14 FRAC - Nachkomma-Anteil einer Realzahl

FRAC

Befehlsform:

Rn:=FRAC.Rn

Beispiel

```
1   R43:=70
2   R70:=210.23445
3   R10:=FRAC.R70

4   END
```

;speichert den Nachkomma-Anteil des Realzahlregister R70 in Realzahlregister R10 (Ergebnis: 0,23445)

1.10.2.15 ABS - Betrag einer Real- / Ganzzahl

ABS

Befehlsform:

Rn:=ABS.Rn

Nn:=ABS.Nn

Beispiel

```
1   R70:=-210.23445
2   R10:=ABS.R70

3   N23:=-452
4   N11:=ABS.N23

5   END
```

;speichert den Betrag des Realzahlregister R70 in Realzahlregister R10 (Ergebnis: |210,23445|)

;speichert den Betrag des Ganzzahlregisters N23 im Ganzzahlregister N11 (Ergebnis: |452|)

1.10.3 Vergleiche

> = <

1.10.3.1 Vergleiche: Befehlsformen

Direkte Adressierung		indirekte Adressierung	
Mn:=Rm=K	Mn:=Nm=K	M!n:=Rm=K	M!n:=Nm=K
Mn:=Rm>K	Mn:=Nm>K	M!n:=Rm>K	M!n:=Nm>K
Mn:=Rm<K	Mn:=Nm<K	M!n:=Rm<K	M!n:=Nm<K
Mn:=Rm=Ri	Mn:=Nm=Ni	M!n:=Rm=Ri	M!n:=Nm=Ni
Mn:=Rm>Ri	Mn:=Nm>Ni	M!n:=Rm>Ri	M!n:=Nm>Ni
Mn:=Rm<Ri	Mn:=Nm<Ni	M!n:=Rm<Ri	M!n:=Nm<Ni
Mn:=R!m=Ri	Mn:=N!m=Ni	M!n:=R!m=Ri	M!n:=N!m=Ni
	Mn:=Nn<>K		

Beispiel

- | | | |
|----|-------------------------|--|
| 1 | <i>M34:=R1=56.77</i> | <i>;Merker 34 wird gesetzt, wenn der Inhalt von R1 gleich 56.77 ist, ansonsten wird Merker 34 zurückgesetzt</i> |
| 2 | <i>M35:=R1>56.34</i> | <i>;Merker 35 wird gesetzt, wenn der Inhalt von R1 größer 56.34 ist, ansonsten wird Merker 35 zurückgesetzt</i> |
| 3 | <i>M36:=R1<56</i> | <i>;Merker 36 wird gesetzt, wenn der Inhalt von R1 kleiner 56 ist, ansonsten wird Merker 36 zurückgesetzt</i> |
| 4 | <i>R2:=10</i> | |
| 5 | <i>M12:=R!2=0</i> | <i>;Merker 12 wird gesetzt, wenn der Inhalt von R10 gleich 0 ist, ansonsten wird Merker 12 zurückgesetzt</i> |
| 6 | <i>M34:=N1=56</i> | <i>;Merker 34 wird gesetzt, wenn der Inhalt von N1 gleich 56 ist, ansonsten wird Merker 34 zurückgesetzt</i> |
| 7 | <i>M35:=N1>56</i> | <i>;Merker 35 wird gesetzt, wenn der Inhalt von N1 größer 56 ist, ansonsten wird Merker 35 zurückgesetzt</i> |
| 8 | <i>M36:=N1<56</i> | <i>;Merker 36 wird gesetzt, wenn der Inhalt von N1 kleiner 56 ist, ansonsten wird Merker 36 zurückgesetzt</i> |
| 9 | <i>N37:=255</i> | |
| 10 | <i>M!37:=N100=510</i> | <i>Merker 255 wird durch indirekte Adressierung von Merker37 gesetzt, wenn der Inhalt von N100 gleich 510 ist, ansonsten wird der Merker 100 zurückgesetzt</i> |
| 11 | <i>END</i> | |

1.10.3.2 Vergleiche: komplexe Beispiele

Beispiel 1

Lade Realzahlregister 10 bis 20 beim Programmstart mit 0 (Rücksetzen von Zählern)

1	R1:=10	;R1 dient zur indirekten Adressierung
2	\$SCHLEIFE	
3	R!1:=0.0	;Register, auf das der Inhalt von R1 zeigt, den Wert zuweisen
4	R1:=R1+1	;Nächstes Register
5	M1:=R1<20	;Alle 10 Register (R10-R20) mit dem Wert geladen?
6	G21 M1.1 SCHLEIFE	;Sprung, wenn noch nicht alle Register geladen
7	END	

Beispiel 2

Produktionsablauf für Palette mit ungleichen Abständen

1	R11:=1	
2	G11.0	;Ablaufanzeige ausschalten
2	G500.0	;LC-Anzeige ist der aktuelle Datenkanal
3	G501	;Anzeige löschen
4	\$EINLESEN	
5	G503.1.1	;Cursor positionieren
6	G510. EINGABE PALETTENPOS.:	;Textausgabe „EINGABE PALETTENPOS.“
	G542.25.1.6.R!11	;Eingabe der Palettenposition über Frontplatte in das indirekt adressierte Register R11. Das Eingabefeld erscheint in der 25. Spalte der 1. Zeile und ist 6 Zeichen lang
7	R11:=R11+1	;Zeiger auf nächste Palettenposition
8	M1:=R11>5	;alle Palettenpositionen eingelesen ?
9	G21 M1.0 EINLESEN	;Nein!
10	R11:=1	;Positionszeiger für Bearbeitung auf 1. Position
11	\$NAECHST	
12	A1:=100	;Holposition anfahren
13	SUB TEILHOLE	;Teil aufnehmen
14	A1:=R!11	;Ablegeposition anfahren
15	SUB TEILABLE	;Teil ablegen
16	R11:=R11+1	;Positionszeiger auf nächste Palettenposition
17	M1:=R1>6	;alle Palettenpositionen belegt ?
18	G21 M1.0 NAECHST	;Nein !

Programm: TEILHOLE

1	I1.1	;warte bis Teil in Abholposition
...		Ablauf für Teil aufnehmen
n	END	

Programm : TEILABLE

1	I7.1	Palette vorhanden
...		Ablauf für Teil ablegen
N	END	

1.10.4 Logische Verknüpfungen mit Eingängen, Ausgängen und Merkern

1.10.4.1 Einführung

Eine Logische Verknüpfung muss immer mit einem LD-Befehl beginnen und mit einem OUT-Befehl enden. Innerhalb einer logischen Kette (beginnend mit LD, endend mit OUT) dürfen nur Befehle der logischen Verknüpfungen angewendet werden.

Logische Verknüpfungen werden nach folgendem Funktionsmechanismus bearbeitet.

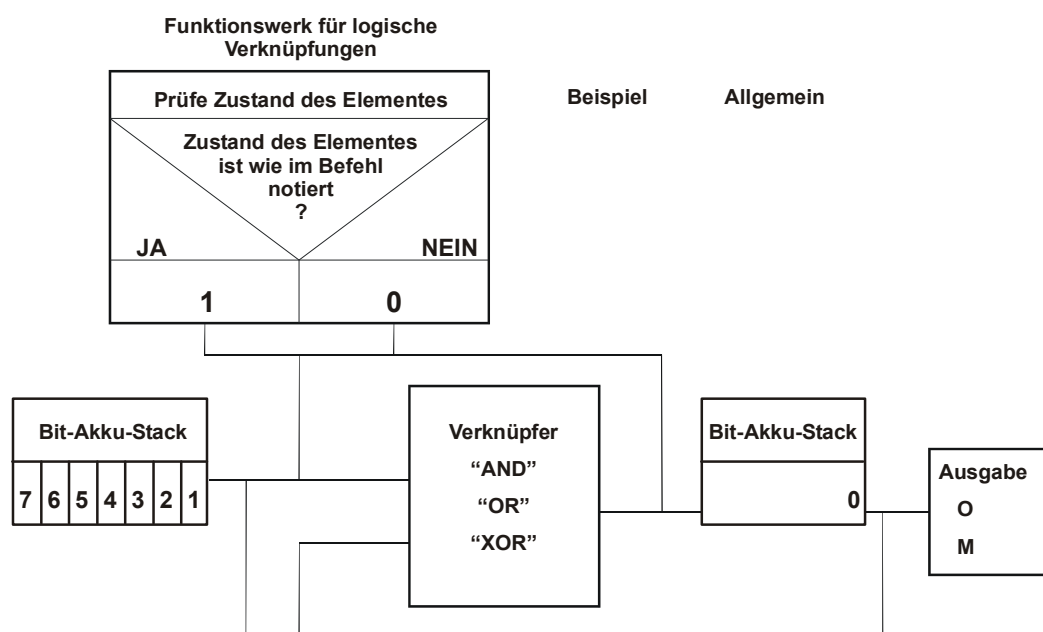


BILD192D

Abbildung 8: Allgemeines Verknüpfungsschema

Um die Schreibweise aus den Befehlen „Warte auf den Zustand eines Elementes“ (z.B. I3.1, M12.0) auch bei den logischen Verknüpfungen anzuwenden, wurden den **LD**-, **AND**- und **OR**-Befehlen nicht nur das Element und die Nummer wie Eingang, Merker, Systemmerker oder Ausgang, sondern auch noch der gewünschte Zustand des Elementes zugeordnet. Entspricht der aktuelle Zustand des Elementes (I, M, SM, O) dem Zustand des Elementes beim logischen Befehl, so wird bei der Umsetzung des Befehls (LD, AND, OR) mit einer logischen „1“ weitergearbeitet. Ansonsten wird mit einer logischen „0“ weitergearbeitet.

Das Ergebnis der logischen Operationen LD, AND und OR wird im Bit-Akku gespeichert. Bei einem LD-Befehl wird zuerst der Zustand des Bit-Akku in den Bit-Akku-Stack (7 Bit tief) übernommen und dann erst das Ergebnis der Überprüfung des Elementes im Bit-Akku gespeichert.

Die logischen Operationen **AND-LD** und **OR-LD** führen eine Verknüpfung zwischen Bit-Akku und Bit-Akku-Stack durch und legen das Ergebnis im Bit-Akku ab.

Der **OUT**-Befehl übergibt den Zustand des Bit-Akkus an das angesprochene Element. Ist der Bit-Akku auf logisch „1“ so wird der Ausgang oder Merker gesetzt, ansonsten zurückgesetzt. Es können mehrere OUT-Befehle direkt hintereinander folgen.

Mit dem **NOT**-Befehl hat der Programmierer die Möglichkeit, den Zustand des Bit-Akkus invertierend an das zu beeinflussende Element Ausgang oder Merker zu übergeben. Der logische Zustand „1“ führt also zu einem RESET des angesprochenen Elementes

Jeder Parallelablauf hat seinen eigenen Bit-Akku und Bit-Akku-Stack mit einer Tiefe von 8 Bit für komplexe Verknüpfungen.

Befehl: LD

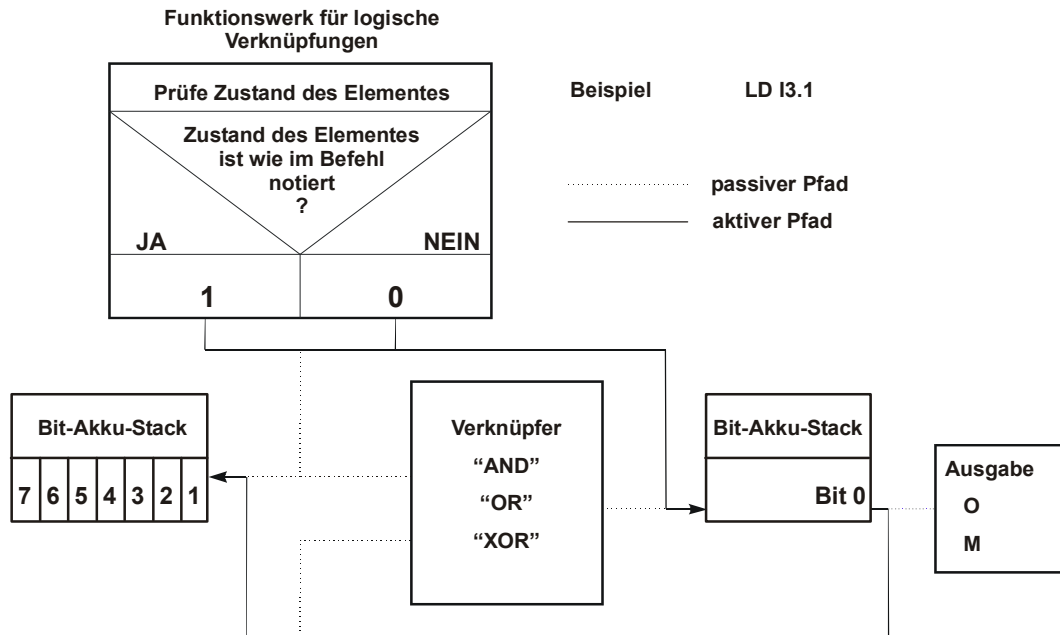


BILD193D

Abbildung 9: Verknüpfungsschema für den Befehl LD

Befehl: AND oder OR

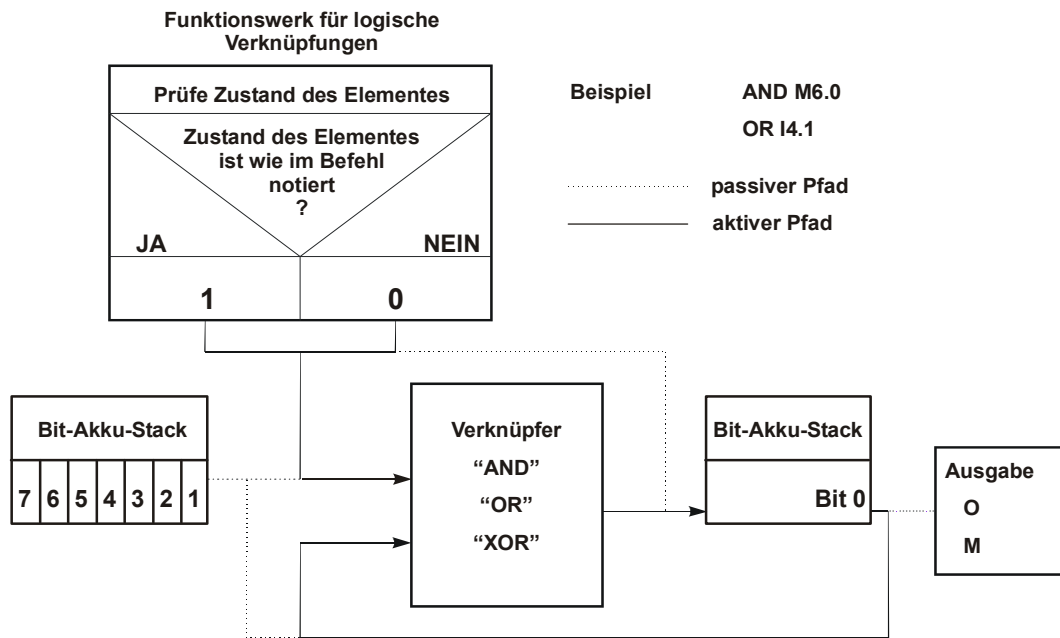


BILD194D

Abbildung 10: Verknüpfungsschema für die Befehle AND und OR

Befehl: AND-LD oder OR-LD

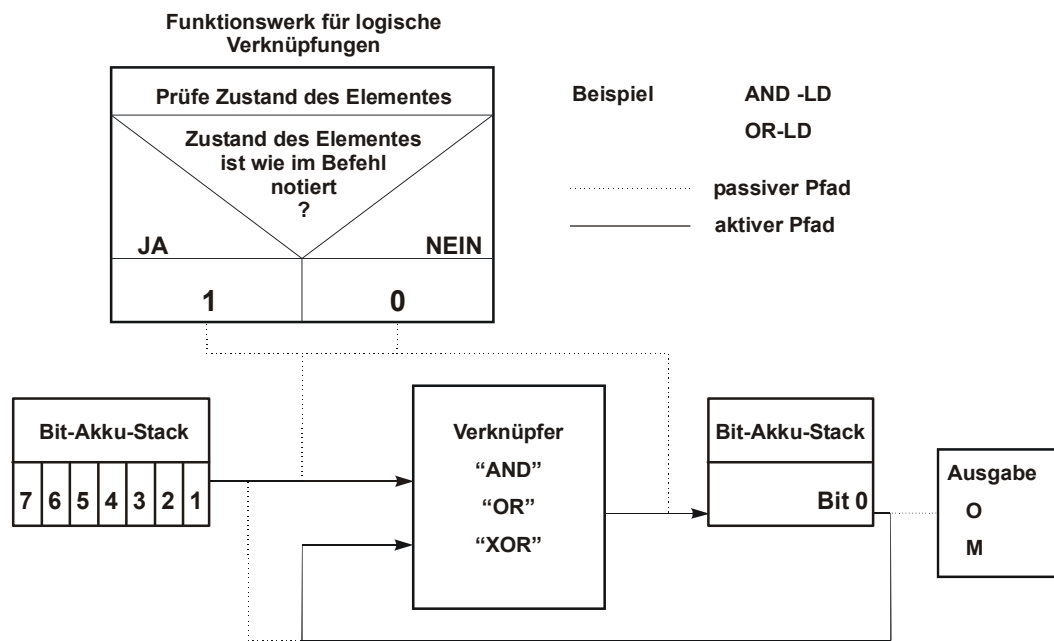


BILD195D

Abbildung 11: Verknüpfungsschema für die Befehle AND-LD und OR-LD

Die logischen Operationen AND-LD und OR-LD holen das zuletzt gespeicherte Bit aus dem Bit-Akku-Stack, d.h. es wird das Bit 1 des Bit-Akku-Stacks in den Verknüpfen geschoben. Dieses Bit wird nun mit dem Bit 0 des Bit-Akku-Stacks verknüpft. Das Resultat dieser Verknüpfung steht dann im Bit-Akku-Stack Bit 0.

HINWEIS Beim Einlesen des Bit 1 aus dem Bit-Akku-Stack wird der gesamte Bit-Akku-Stack um eine Position nach rechts geschoben. Da maximal 7 Bit im Bit-Akku-Stack gespeichert sein können, können die Befehle AND-LD oder OR-LD hintereinander nur maximal 7 mal sinnvoll angewendet werden!

Befehl: OUT

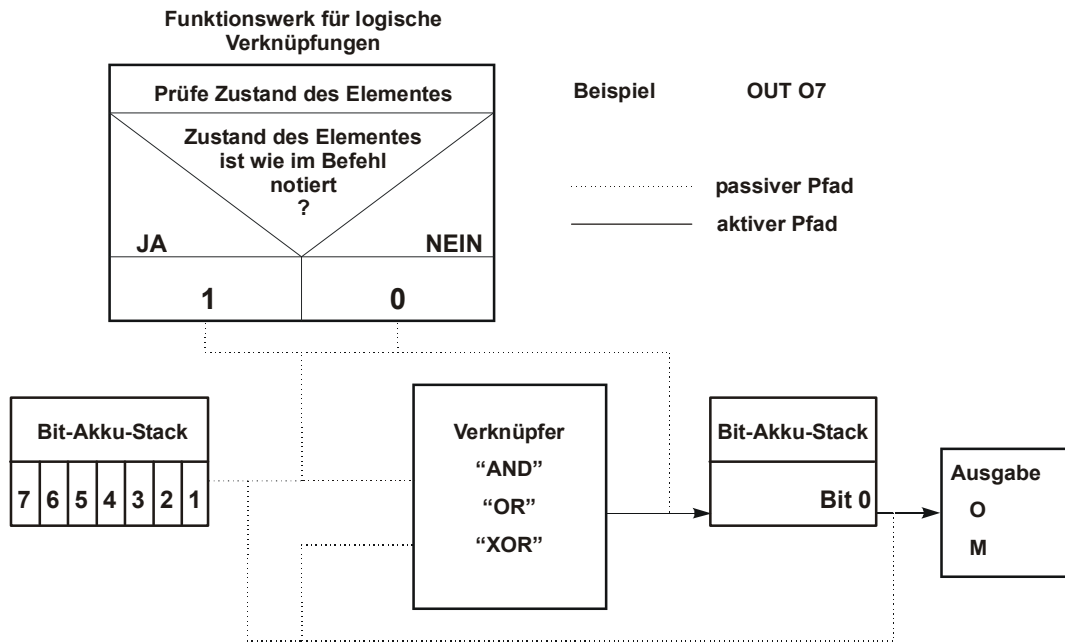


BILD196D

Abbildung 12: Verknüpfungsschema für den Befehl OUT

1.10.4.2 LD/AND/OUT/NOT - Logische UND-Verknüpfung

LD, AND, OUT, NOT

Befehlsform:

AND Ii.n
 AND Oi.n
 AND Mi.n
 AND Mli.n
 AND SMi.n

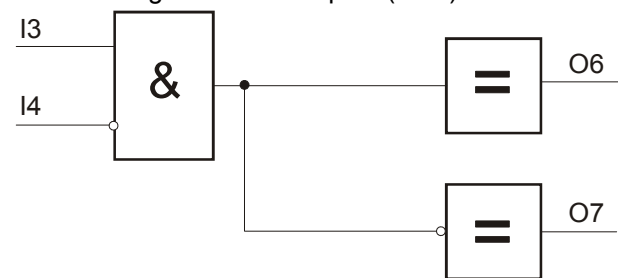
Anwendung:

Aufbauen von logischen UND-Verknüpfungen zwischen Eingängen, Merkern, Systemmerkern und Ausgängen. Das Ergebnis der logischen Verknüpfung kann einem Merker oder einem Ausgang zugewiesen werden.

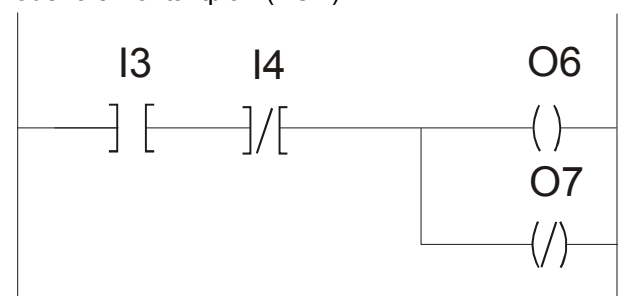
Beispiel

1 LD I3.1
 2 AND I4.0
 3 OUT O6
 4 NOT O7

Realisierung als Funktionsplan (FUP)



oder als Kontaktplan (KOP)



5 END

1.10.4.3 LD/OR/OUT/NOT - Logische ODER-Verknüpfung

LD, OR, OUT, NOT

Befehlsform:

OR Ii.n
 OR Oi.n
 OR Mi.n
 OR Mi.n
 OR Smi.n

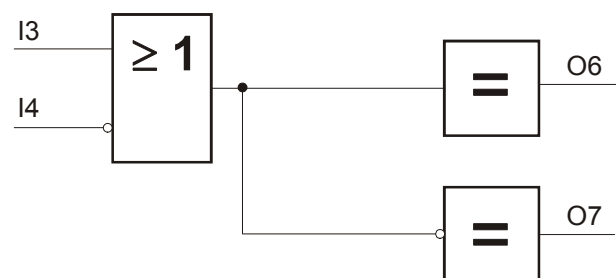
Anwendung:

Aufbauen von logischen ODER-Verknüpfungen zwischen Eingängen, Merkern, Systemmerkern und Ausgängen. Das Ergebnis der logischen Verknüpfung kann einem Merker oder einem Ausgang zugewiesen werden.

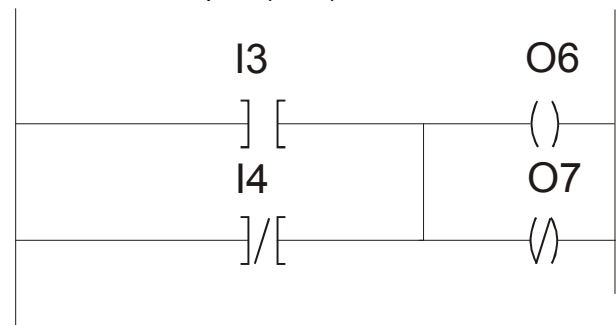
Beispiel

1 LD I3.1
 2 OR I4.0
 3 OUT O6
 4 NOT O7
 5 END

Realisierung als Funktionsplan (FUP)



oder als Kontaktplan (KOP)



1.10.4.4 SET/RES - Ergänzungsbefehle zu logischen Verknüpfungen

SET / RES

ab V5.07

Befehlsform:

SET On

RES On

SET Mn

RES Mn

Erweiterte Funktionen (von Ausgang/Merker bis Ausgang/Merker # Es können maximal 1024 Ausgänge oder Merker beeinflusst werden.)

SET On.Om

RES On.Om

SET Mn.Mm

RES Mn.Mm

Anwendung:

Einfachere Handhabung mit Verknüpfungsergebnissen aus logischen Befehlen.

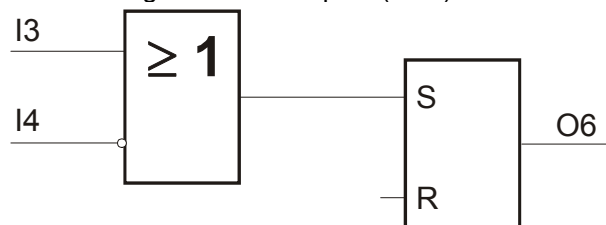
Beispiel 1

```

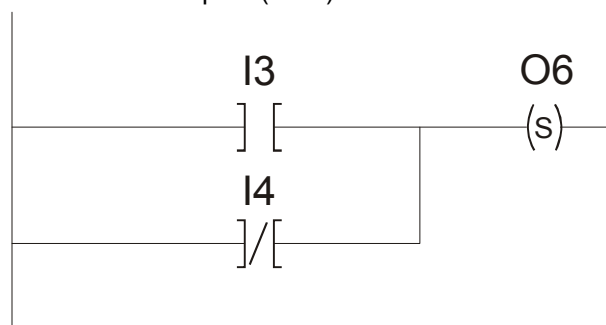
1   LD I3.1
2   OR I4.0
3   SET O6
4   END

```

Realisierung als Funktionsplan (FUP)



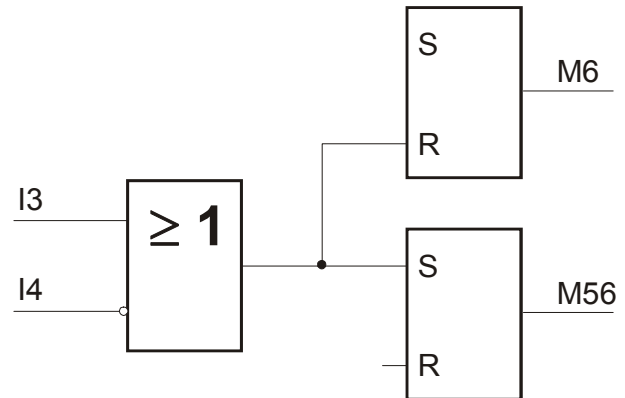
oder als Kontaktplan (KOP)



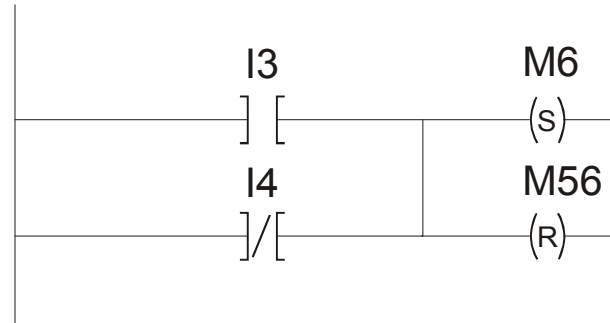
Beispiel 2

- 1 LD I3.1
- 2 OR I4.0
- 3 RES M6
- 4 SET M56
- 5 END

Realisierung als Funktionsplan (FUP)



oder als Kontaktplan (KOP)



Beispiel 3

Im folgenden Beispiel wird gezeigt, wie ganze Merkerblöcke beeinflusst werden können.

```
1      $A
2      G21 I9.1 SETM3M9
3      G21 I10.1 SETM3M39
4      G21 I11.1 RESM3M9
5      G21 I12.1 RESM3M39
6      JMP A
7      ;
8      $ SETM3M9
9      LD I16.1
10     SET M3.M9           Setze die Merker von M3 bis M9
11     T10 I9.0
12     JMP A
13     ;
14     $ SETM3M39
15     LD I16.1
16     SET M3.M39         Setze die Merker von M3 bis M39
17     T10 I10.0
18     JMP A
19     ;
20     $ RESM3M39
21     LD I16.1
22     RES M3.M9         Setze die Merker von M3 bis M9 zurück
23     T10 I11.0
24     JMP A
25
26     $ RESM3M39
27     LD I16.1
28     RES M3.M39       Setze die Merker von M3 bis M39 zurück
29     T10 I12.0
30     JMP A
31     ;
32     END
```

1.10.4.5 XOR-Verknüpfung (Exklusiv-ODER)

XOR

Befehlsform:

XOR In.i

XOR On.i

XOR Mn.i

XOR SMn.i

Beispiel 1 mit Wahrheitstabelle

	Zustände			
LD I1.1	unwahr	wahr	unwahr	wahr
XOR I2.1	unwahr	unwahr	wahr	wahr
OUT O1	0	1	1	0

Beispiel 2 mit Wahrheitstabelle

HINWEIS Bei XOR-Verknüpfungen mit mehr als zwei Variablen wird intern zuerst die XOR-Verknüpfung der beiden ersten Variablen und danach mit diesem Ergebnis die nächste XOR-Verknüpfung realisiert (siehe dazu das nächste Beispiel) und so weiter.

	Zustände							
LD I1.1	unwahr	wahr	unwahr	wahr	unwahr	wahr	unwahr	wahr
XOR I2.1	unwahr	unwahr	wahr	wahr	unwahr	unwahr	wahr	wahr
XOR I3.1	unwahr	unwahr	unwahr	unwahr	wahr	wahr	wahr	wahr
OUT O1	0	1	1	0	1	0	0	1

1.10.4.6 Mehrstufige logische UND-Verknüpfung

AND-LD

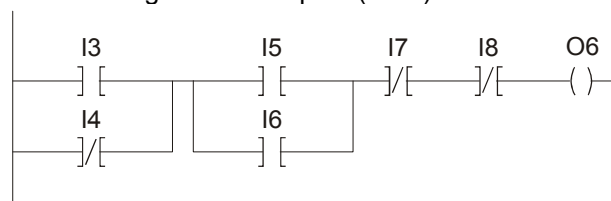
Befehlsform:

AND-LD

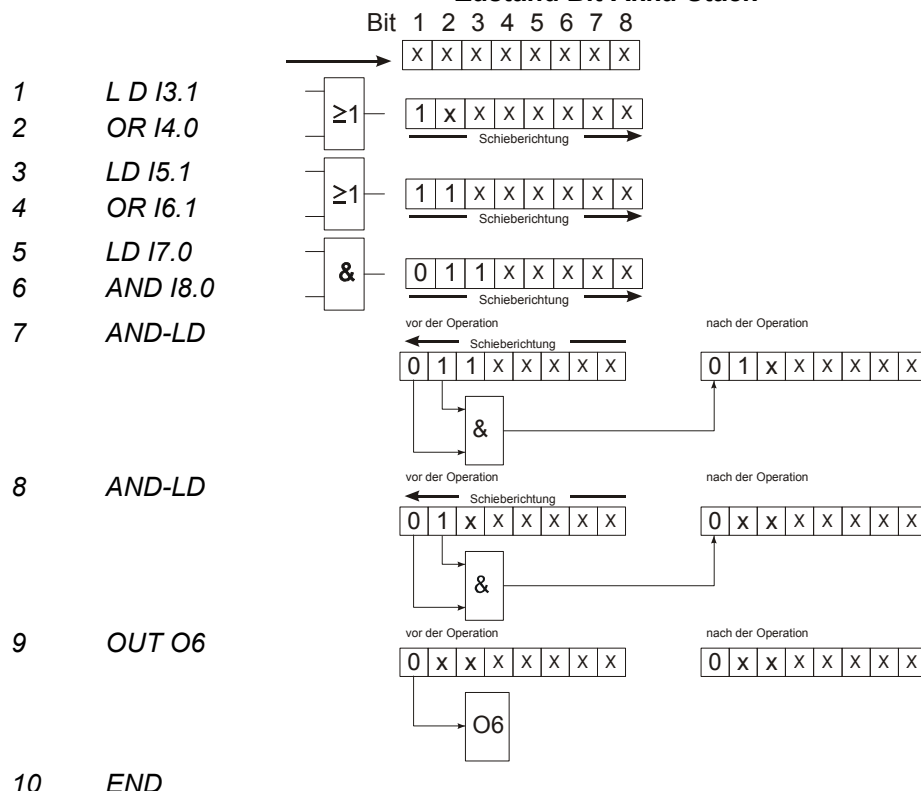
Beispiel

Eingang	I3	I4	I5	I6	I7	I8
Log. Zustand	1	0	1	0	1	0

Realisierung als Kontaktplan (KOP)



Zustand Bit-Akku-Stack



Anwendung:

Aufbauen von komplexen (mehrstufigen) logischen UND-Verknüpfungen zwischen Eingängen, Merkern, Systemmerkern und Ausgängen. Der Bit-Akku-Stack hat eine Tiefe von 7 Bit. Das Ergebnis der logischen Verknüpfung kann einem Merker oder einem Ausgang zugewiesen werden.

1.10.4.7 OR-LD - Mehrstufige logische ODER-Verknüpfung

OR-LD

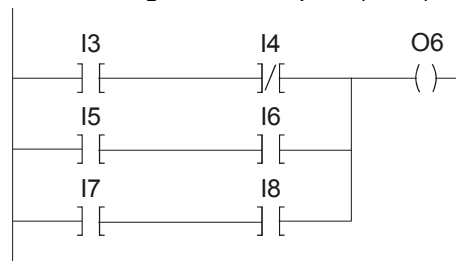
Befehlsform:

OR-LD

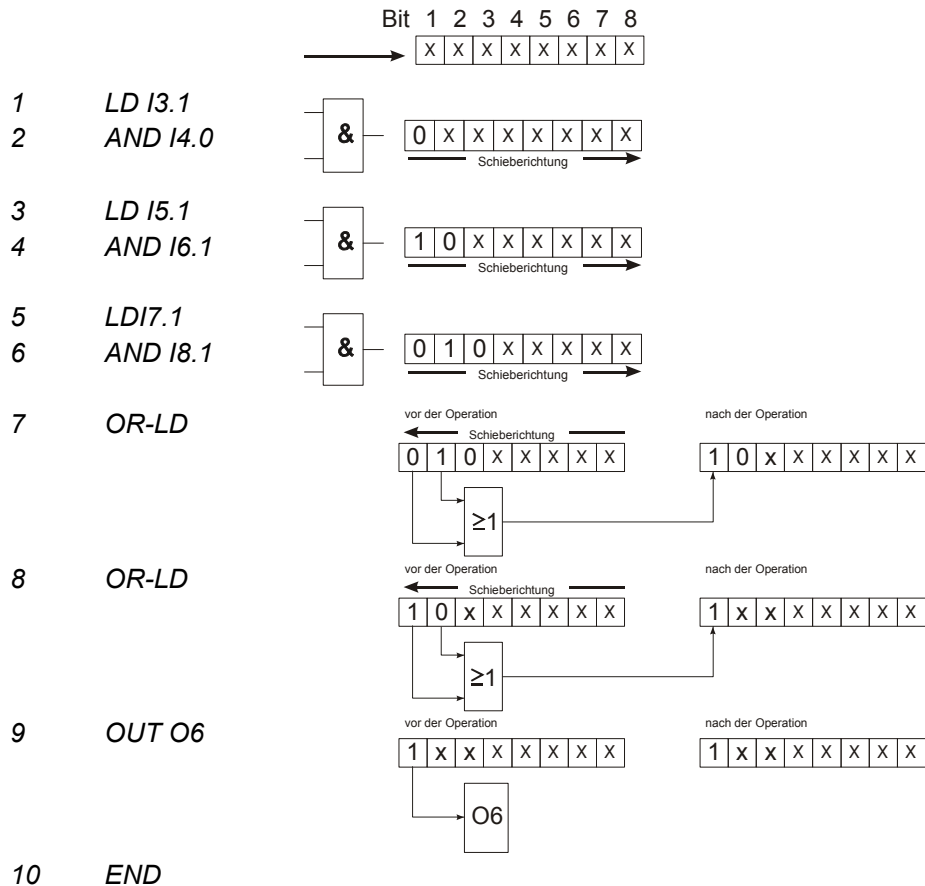
Beispiel

Eingang	I3	I4	I5	I6	I7	I8
Log. Zustand	1	1	1	1	1	0

Realisierung als Kontaktplan (KOP)



Zustand Bit-Akku-Stack



Anwendung:

Aufbauen von komplexen (mehrstufigen) logischen ODER-Verknüpfungen zwischen Eingängen, Merkern, Systemmerkern und Ausgängen. Der Bit-Akku-Stack hat eine Tiefe von 7 Bit.

Das Ergebnis der logischen Verknüpfung kann einem Merker oder einem Ausgang zugewiesen werden.

1.10.4.8 Komplexe logische Verknüpfung

Beschreibung:

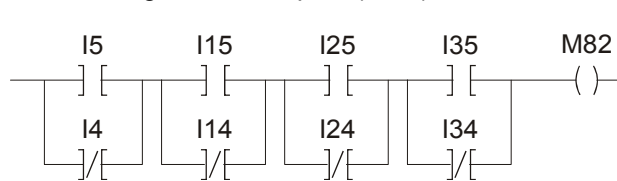
Wird eine mehrstufige Verknüpfung durchgeführt, muss das Ergebnis einer Verknüpfung zwischengespeichert werden, da das Verknüpfungsergebnis im Bit-Akku-Stack wiederholt überschrieben wird. Der Bit-Akku-Stack hat 8 Bit Tiefe.

Das Beispiel zeigt eine mehrstufige Verknüpfung. Wird die Programmierung ohne das Zwischenspeichern im Merker 82 durchgeführt, wird der Bit-Akku-Stack nach dem zweiten LD-Befehl überschrieben und das Verknüpfungsergebnis wird falsch.

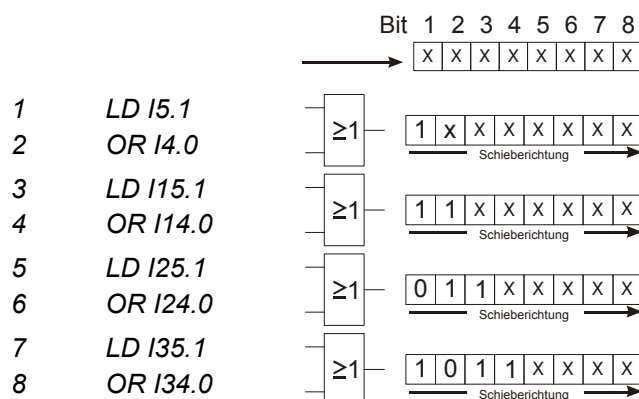
Beispiel

Eingang	I4	I5	I14	I15	I24	I25	I34	I35
Log. Zustand	1	1	0	1	1	0	1	1

Realisierung als Kontaktplan (KOP)

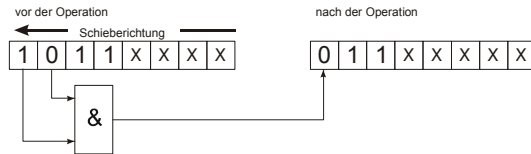


Zustand Bit-Akku-Stack

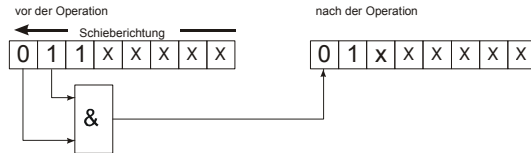


Fortsetzung

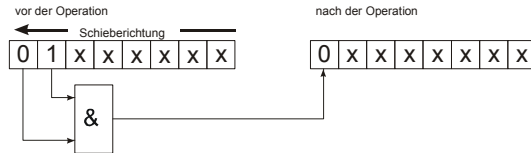
9 *AND-LD*



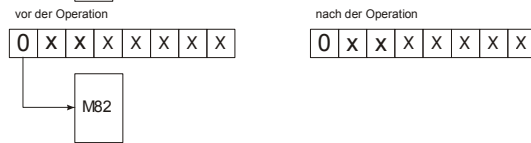
10 *AND-LD*



11 *AND-LD*



12 *OUT M82*



13 *END*

1.10.4.9 Bitweise Verarbeitung von N-Registern

UND, ODER, EXKLUSIV-ODER

Die Möglichkeiten für die Anwendung von N-Register wurde um folgende bitweise arbeitenden Funktionen ergänzt:

Funktion		Operator
Bitweise UND	UND	&
Bitweise ODER	ODER	
Bitweise EXKLUSIV-ODER	EXKLUSIV-ODER	^

Die Erweiterung der Befehle erfolgte für:

Befehlsformen

Ni:=Nn&Nm
 Ni:=Nn&Konstante
 Ni:=Nn|Nm
 Ni:=Nn|Konstante
 Ni:=Nn^Nm
 Ni:=Nn^Konstante

HINWEIS Die N-Register sind in der PA-CONTROL mit 32 Bit hinterlegt (signed long int). Die bitweise Bearbeitung erfolgt über alle 32 Bit!

Daraus ergibt sich folgende Darstellung:

Dez	Hex	Binär
0	0000 0000	0000 0000 0000 0000 0000 0000 0000 0000
1	0000 0001	0000 0000 0000 0000 0000 0000 0000 0001
100	0000 0064	0000 0000 0000 0000 0000 0000 0000 0100
-1	FFFF FFFF	1111 1111 1111 1111 1111 1111 1111 1111
-2	FFFF FFFE	1111 1111 1111 1111 1111 1111 1111 1110

Beispiel 1

N2:=6 ; Binär 0000 0000 0000 0000 0000 0000 0000 0110
 N3:=3 ; Binär 0000 0000 0000 0000 0000 0000 0000 0011
 N12:=N2&N3 ; Binär (Ergebnis=2) 0000 0000 0000 0000 0000 0000 0000 0010

Beispiel 2

N4:=3 ; Binär 0000 0000 0000 0000 0000 0000 0000 0011
 N6:=10 ; Binär 0000 0000 0000 0000 0000 0000 0000 1010
 N3:=N4|N6 ; Binär (Ergebnis=11) 0000 0000 0000 0000 0000 0000 0000 1011

Beispiel 3

N4:=3 ; Binär 0000 0000 0000 0000 0000 0000 0000 0011
 N6:=10 ; Binär 0000 0000 0000 0000 0000 0000 0000 1010
 N3:=N4^N6 ; Binär (Ergebnis=9) 0000 0000 0000 0000 0000 0000 0000 1001

1.10.5 Register-Operationen

1.10.5.1 Ni.n - Warten auf den Status eines N-Registers

Ni.Wert

N (Nummer).(Wert)

Beschreibung:

Wartet mit der Bearbeitung des folgenden Programmteils bis der Status des N-Registers gleich der Konstanten ist. Ist die Abfrage nicht wahr, wartet die Steuerung bis die Bedingung im Operator n erfüllt ist.

Die Veränderung des Registerwertes erfolgt z.B. in einem parallel arbeitenden Programm.

Befehlsform:

Ni.n

Ni.Nm

N!i.n

N!i.N!n

Anwendung:

Abfrage von zyklisch ablaufenden Programmteilen

Beispiel

1	\$A	
2	N2.5	<i>Steuerung wartet bis der Inhalt vom Register N2 gleich 5 ist</i>
3	O1:=1 T10 O1:=0 T10	<i>Blinken einer Lampe</i>
4	JMP A	<i>Sprung nach A</i>
5	END	

1.10.5.2 Nn:=SNn - Lade Inhalt eines System-N-Registers in ein Ganzzahlregister

Nn:=SNn

Beschreibung:

In der PA-CONTROL sind 108 System-Ganzzahlregister implementiert, die für unterschiedlichste Aufgaben reserviert sind (siehe Seite 24).

Um bei Fehlern, Störungen oder besonderen Situation eine Aussage über das System zu erhalten, können diese Register ausgelesen und dann weiter verarbeitet, z. B. im Display angezeigt werden.

HINWEIS Die systeminternen Ganzzahlregister können vom Anwender nur abgefragt werden!

Befehlsform:

N100:=SN6

N!20:=SN21

Beispiel 1

1	N2:=21	;
2	N124:=SN102	; Überträgt die Anzahl der über die Schnittstelle COM 2 empfangenen Zeichen in das Ganzzahlregister N124
3	N!2:=SN1	;Überträgt die Systemlaufzeit der PA-CONTROL (in Sekunden) in das Ganzzahlregister N21
4		
5	END	

1.10.5.3 Rn:=SRn - Lade Inhalt eines System-R-Registers in ein Realzahlregister

Rn:=SRn

Beschreibung:

In der PA-CONTROL sind gegenwärtig 46 System-Realzahlregister implementiert, die für unterschiedlichste Aufgaben reserviert sind (siehe *Seite 27*).

Um bei Fehlern, Störungen oder besonderen Situation eine Aussage über das System zu erhalten, können diese Register ausgelesen und dann weiter verarbeitet, z. B. im Display angezeigt werden.

HINWEIS Die systeminternen Realzahlregister können vom Anwender nur abgefragt werden!

Befehlsform:

R1028:=SR10

R!27:=SR31

Beispiel 1

1	R27:=524	;
2	R12:=SR10	; Überträgt die Interpolationssollgeschwindigkeit in das Realzahlregister 12
3	R!27:=SR31	;Überträgt die Absolutposition der Achse 1 in das Realzahlregister R524
4		
5	END	

1.10.5.4 Bitweise Verarbeitung von N-Registern

UND, ODER, EXKLUSIV-ODER

Die Möglichkeiten für die Anwendung von N-Register wurde um folgende bitweise arbeitenden Funktionen ergänzt:

Funktion		Operator
Bitweise UND	UND	&
Bitweise ODER	ODER	
Bitweise EXKLUSIV-ODER	EXKLUSIV-ODER	^

Die Erweiterung der Befehle erfolgte für:

Befehlsformen

Ni:=Nn&Nm
 Ni:=Nn&Konstante
 Ni:=Nn|Nm
 Ni:=Nn|Konstante
 Ni:=Nn^Nm
 Ni:=Nn^Konstante

HINWEIS Die N-Register sind in der PA-CONTROL mit 32 Bit hinterlegt (signed long int) . Die bitweise Bearbeitung erfolgt über alle 32 Bit!

Daraus ergibt sich folgende Darstellung :

Dez	Hex	Binär
0	0000 0000	0000 0000 0000 0000 0000 0000 0000 0000
1	0000 0001	0000 0000 0000 0000 0000 0000 0000 0001
100	0000 0064	0000 0000 0000 0000 0000 0000 0000 0100
-1	FFFF FFFF	1111 1111 1111 1111 1111 1111 1111 1111
-2	FFFF FFFE	1111 1111 1111 1111 1111 1111 1111 1110

Beispiel 1

N2:=6 ; Binär 0000 0000 0000 0000 0000 0000 0000 0110
 N3:=3 ; Binär 0000 0000 0000 0000 0000 0000 0000 0011
 N12:=N2&N3 ; Binär (Ergebnis=2) 0000 0000 0000 0000 0000 0000 0000 0010

Beispiel 2

N4:=3 ; Binär 0000 0000 0000 0000 0000 0000 0000 0011
 N6:=10 ; Binär 0000 0000 0000 0000 0000 0000 0000 1010
 N3:=N4|N6 ; Binär (Ergebnis=11) 0000 0000 0000 0000 0000 0000 0000 1011

Beispiel 3

N4:=3 ; Binär 0000 0000 0000 0000 0000 0000 0000 0011
 N6:=10 ; Binär 0000 0000 0000 0000 0000 0000 0000 1010
 N3:=N4^N6 ; Binär (Ergebnis=9) 0000 0000 0000 0000 0000 0000 0000 1001

1.11 Kommunikation

1.11.1 Kommunikation über MODBUS

Der AnyBus Communicator kann an den CANopen-Bus der PA-CONTROL angeschlossen werden.

Mit diesem Umsetzer kann die PA-CONTROL mit Geräten kommunizieren die mit einer MODBUS-Schnittstelle ausgestattet sind (z.B.: Temperaturregler, ...)

weiterführende Informationen können Sie der Beschreibung „APP5002_DE_1070171_AnyBusCommunicator_MODBUS.pdf“ entnehmen.

1.11.2 Lesen Werte von einem MODBUS-Teilnehmer

Ni/Ri:=MODBUS

ab V5.07

Befehlsform:

Ni:=MODBUS.<Adresse>.<Function>.<Register-Nummer>

Ri:=MODBUS.<Adresse>.<Function>.<Register-Nummer>

Anwendung:

Lesen von Prozessdaten, Parametern oder Konfigurationsdaten

Beispiel

1	<i>N23:=MODBUS.2.3.3180</i>	<i>;holt den Inhalt (16Bit-Integer) des Registers 3180 des MODBUS-Teilnehmers mit der Adresse 2</i>
2	<i>R4:=MODBUS.1.3.39128</i>	<i>;holt den Inhalt (32Bit-Float) des Registers 39128 des MODBUS-Teilnehmers mit der Adresse 1</i>
3	<i>END</i>	

1.11.3 Schreiben eines Wertes in einen MODBUS-Teilnehmer

MODBUS

ab V5.07

Befehlsform:

MODBUS.<Adresse>.<Function>.<Register-Nummer>:= Ni / Konstante

MODBUS.<Adresse>.<Function>.<Register-Nummer>:= Ri / Konstante

Anwendung:

Schreiben von Prozessdaten, Parametern oder Konfigurationsdaten

Beispiel

1	<i>MODBUS.1.6.3180:=5</i>	<i>;schreibt den Wert 5 in das Register 3180 in den MODBUS-Teilnehmers mit der Adresse 1</i>
2	<i>MODBUS.1.16.39128:=12.3</i>	<i>;schreibt den Wert 12.3 in das Register 39128 in den MODBUS-Teilnehmers mit der Adresse 1</i>
3	<i>MODBUS.5.6.3180:=N4</i>	<i>;schreibt den Inhalt von N4 in das Register 3180 in den MODBUS-Teilnehmers mit der Adresse 5</i>
4	<i>MODBUS.4.16.39128:=R5</i>	<i>;schreibt den Inhalt von R5 in das Register 39128 in den MODBUS-Teilnehmers mit der Adresse 4</i>
5	<i>END</i>	

1.11.4 Kommunikation über Display PA-CONTROL

1.11.4.1 G11 - Anzeige ein- / ausschalten

G11

G11.n

Befehlsform:

G11.0

G11.1

Anwendung:

Für Abläufe, bei denen mit den Befehlen der G500-Befehlsgruppe das Display für Bedienerführung sowie Meldungen verwendet werden soll.

Beschreibung:

Mit der Funktion **G11** wird die Anzeige der Befehle während des Programmlaufs gesteuert. Mit **G11.0** wird die Anzeige ausgeschaltet, mit **G11.1** wird die Anzeige wieder eingeschaltet.

HINWEIS Der zuletzt abgearbeitete Befehl wird unabhängig vom Programmstand angezeigt.

Folgende Befehle werden im Display nie angezeigt : JMP, M1:=, usw.!

Nur so kann bei der Programmierung von Schleifen eine vorherige Programmzeile angezeigt werden.

Beispiel

1	G11.0	<i>;Anzeige ausschalten</i>
2	G500.0	<i>;wählt das LC-Display der PA-CONTROL aus</i>
3	G501	<i>;löscht die Anzeige und stellt den Cursor in die 1. Spalte der 1. Zeile</i>
4	G510.Maschinenfehler!	<i>;gibt den Text „Maschinenfehler!“ auf dem aktuellen Anzeigemedium, dem LC-Display aus.</i>
5	G11.1	<i>;Anzeige einschalten</i>
6	END	

1.11.4.2 Nn:=CHN - Prüfe ob Zeichenübernahme im Hintergrund abgeschlossen

Nn:=CHN

Beschreibung:

Mit dem Befehl Nn:=CHN wird der Empfangszustand eines G533- oder G534-Befehls geholt. Auf Grund des Ergebnisses kann dann im Programm verzweigt werden. Das Ergebnis im Ganzzahlregister wird wie folgt interpretiert:

<u>Inhalt von Ni</u>		<u>Beschreibung</u>
0	→	Endekriterium noch nicht empfangen
1	→	Endekriterium empfangen
2	→	Übertragungsfehler auf der seriellen Schnittstelle (Parity,...)
3	→	Empfangspuffer voll, aber Endekriterium noch nicht empfangen

Befehlsform:

Nn:=CHN

1.11.5 String-Befehle

1.11.5.1 Sn - Kopieren von Zeichenketten

Sn:= Si

Beschreibung:

Zeichenketten (Strings) dienen zur Speicherung von bis 80 ASCII-Zeichen, die z.B. über eine serielle Schnittstelle empfangen wurden. Es gibt S1 bis S16 als globale Zeichenpuffer (Strings) auf die jeder Ablauf zugreifen kann. Jeder Ablauf hat zusätzlich noch seinen eigenen lokalen Zeichenpuffer S0.

Der Inhalt des S0 kann mit dem Befehl Sn:=S0 in einen globalen Zeichenpuffer kopiert werden. Auch der umgekehrte Weg, aus einem globalen Zeichenpuffer in den lokalen Zeichenpuffer S0, ist möglich .

Befehlsform:

S0:=Sn

Sn:=S0

1.11.5.2 S0:=CHN - Kopieren von Zeichenketten

S0:= CHN

Beschreibung:

Zeichenketten (Strings) dienen zur Speicherung von bis 80 ASCII-Zeichen, die z.B. über eine serielle Schnittstelle empfangen wurden. Es gibt S1 bis S16 als globale Zeichenpuffer (Strings) auf die jeder Ablauf zugreifen kann. Jeder Ablauf hat zusätzlich noch seinen eigenen lokalen Zeichenpuffer S0.

Der lokale Zeichenpuffer S0 dient zur Bearbeitung und Untersuchung von Zeichenketten (POS, COPY, usw.). Der Befehl S0:=CHN kopiert den Inhalt des seriellen Empfangspuffers des aktuellen Datenkanals in den lokalen Zeichenpuffer. Das Endekriterium für die Übertragung ist das Zeichen 0.

Die beiden Befehlsformen unterscheiden sich dadurch, dass bei Anwendung des Befehls S0:=CHN der Inhalt des gesamten seriellen Eingangspuffers, im anderen Fall eine definierte Anzahl von Zeichen kopiert werden.

Befehlsform:

S0:=CHN

S0:=CHN.i

Sn:=S0

1.11.5.3 POS - Suche Position eines Zeichens im lok. Zeichenpuffer S0

POS

POS.(Suchposition).(Zeichen 0-255)

Beschreibung:

Mit dem Befehl **POS.** kann im lokalen Zeichenpuffer S0 nach dem Vorkommen eines bestimmten Zeichens („m“) gesucht werden.

„m“ ist eine Zahl größer 0 und kleiner 255 und repräsentiert letztlich das entsprechende ASCII-Zeichen.

Die Suche im lokalen Zeichenpuffer S0 beginnt ab der Stelle „n“. Wurde das Zeichen gefunden, so wird die Position (1 bis n) in dem Ganzzahlregister abgelegt. Wurde das Zeichen nicht gefunden, so wird der Wert „0“ im Ganzzahlregister abgelegt.

Befehlsform:

Ni:=POS.n.m

Ni:=POS.Nn.m

Ni:=POS.Nn.Nm

Beispiel

1	G500.2.6.1.0	;Initialisierung der seriellen Schnittstelle 2
2	G532.13 U_FEHLER	;empfängt Zeichen an der seriellen Schnittstelle und legt diese im lokalen Zeichenpuffer S0 ab, bis das Endekriterium 13 = 0Dhex = CR empfangen wurde
3	N2:=POS.1.35	;suche im lokalen Zeichenpuffer S0 nach dem Zeichen 35= 23hex = # und lege die Position in N2 ab
4	M1:=N2<1	;Zeichen gefunden ?
5	G21 M1.1 Z_NEIN	;Nein !
6	N3:=COPY.N2.5 C_FEHLER	;wandle ab dem Wert von N2 die nächsten 5 Zeichen aus dem lokalen Zeichenpuffer S0 in eine Zahl und lege diese im Ganzzahlregister 3 ab
7	JMP ENDE	
8	\$U_FEHLER	
9	SUB Fehler_1	
10	JMP ENDE	
11	\$C_FEHLER	
12	SUB Fehler_2	
13	JMP ENDE	
14	\$Z_NEIN	
15	SUB Fehler_3	
16	\$ENDE	
17	END	

1.11.5.4 COPY - Wandle Zeichen aus dem lokalen Zeichenpuffer

COPY

Copy.(Startposition).(Anzahl Zeichen)[(Marke)]

Beschreibung:

Mit dem Befehl COPY können maximal 80 Zeichen aus dem lokalen Zeichenpuffer (S0) in eine Zahl gewandelt werden und in einem Ganzzahl- oder Realzahlregister abgelegt werden.

COPY beginnt ab dem „n“-ten Zeichen und nimmt die nächsten „m“ Zeichen soweit Zeichen im lokalen Zeichenpuffer (S0) vorhanden sind. Sind weniger als „m“-Zeichen vorhanden wird ohne Fehler der Wandelvorgang beendet.

Tritt ein Fehler auf, so wird der Wandelvorgang abgebrochen. Das Register bleibt unverändert und das Programm wird an der Marke fortgeführt.

mögliche Fehler:

- „n“ ist kleiner 1 oder größer 80
- „m“ ist kleiner 1 oder größer 80

es wurde beim Wandelvorgang ungültige Zeichen gefunden.

Gültige Zeichen	N-Register	R-Register
Vorlaufende Leerzeichen	X	X
0...9	X	X
Dezimaltrennzeichen „.“	X	X
E	-	X, wenn davor mindestens eine Zahl zwischen 0...9, oder zwei Zahlen getrennt durch das Dezimaltrennzeichen „.“ stehen. Nach dem E muss mindestens eine Zahl (1...9) folgen! (Zahlenformat für das R-Register)

HINWEIS Vorlaufende Leerzeichen werden bei der Ermittlung des „n“-ten Zeichens mitgezählt!

Befehlsformen:

Ni:=COPY.n.m Marke
Ri:=COPY.n.m Marke

Ni:=COPY.Nn.Nm Marke
Ri:=COPY.Nn.Nm Marke

Ni:=COPY.N!n.N!m Marke
Ri:=COPY.N!n.N!m Marke

Beispiel

1	G500.2.6.1.0	
2	G532.13 U_FEHLER	<i>;Initialisierung der seriellen Schnittstelle 2 ;empfängt Zeichen an der seriellen Schnittstelle und legt diese im lokalen Zeichenpuffer S0 ab, bis das Endekriterium 13 = 0Dhex = CR empfangen wurde</i>
3	N3:=COPY.2.5 C_FEHLER	<i>;wandelt ab dem 2. Zeichen die nächsten 5 Zeichen aus dem lokalen Zeichenpuffer S0 in eine Zahl und legt diese Zahl im Ganzzahlregister 3 ab</i>
4	JMP ENDE	
5	\$U_FEHLER	
6	SUB Fehler_1	
7	JMP ENDE	
8	\$C_FEHLER	
9	SUB Fehler_2	
10	\$ENDE	
11	END	

1.11.5.5 Sn:=COPY - Hole Teilstring

Sn:=COPY

Sn:=Copy.(Startposition).(Anzahl Zeichen)

Beschreibung:

Das Kopieren eines S0-Strings in einen anderen String (S0 ... S16) erfolgt ebenfalls mit dem COPY-Befehl..

Befehlsformen:

Sn:=COPY.(Startposition).(Anzahl der Zeichen)

Beispiele:

S1:=COPY.4.5
S3:=COPY.N5.N7
S5:=COPY.N!3.N!9

Beispiel

1	G500.2.6.1.0	<i>;Initialisierung der seriellen Schnittstelle 2</i>
2	G532.13 U_FEHLER	<i>;empfängt Zeichen an der seriellen Schnittstelle und legt diese im lokalen Zeichenpuffer S0 ab, bis das Endekriterium 13 = 0Dhex = CR empfangen wurde</i>
3	S3:=COPY.2.5	<i>;wandelt ab dem 2. Zeichen die nächsten 5 Zeichen aus dem lokalen Zeichenpuffer S0 in eine Zahl und legt diese Zahl im String S3 ab</i>
4	JMP ENDE	
5	\$U_FEHLER	
6	SUB Fehler_1	
7	\$ENDE	
8	END	

1.11.5.6 Si:=Sn+Sm - String zusammenfügen

Si:=Sn+Sm

Die Befehle der folgenden Befehlsgruppe dienen der Erstellung eines Strings bzw. dem Zusammenfügen von zwei Strings.

Beschreibung:

- Zwei Strings werden zu einem String zusammengefügt
- Ein String und eine Zahl werden zu einem String zusammengefügt

Für die Konvertierung von Zahlen können keine Formatangaben gemacht werden. N-Register werden einfach gewandelt, R-Register werden mit der Funktion „WandleFloatNach Ascii“ (siehe G500-Befehle) gewandelt.

Ergibt sich bei der Anwendung des Befehls ein String, der länger als 80 Zeichen lang ist, dann wird der Systemfehler „E754, String zu lang“ angezeigt.

Befehlsform:

Si:=Sn+Sm
Si:=SN+Nm
Si:=Sn+Rm
Si:=Nm+Sn
Si:=Rm+Sn

Beispiele:

S0:=S1+S3
S0:=S0+S7
S4:=S3+S4

S1:=S3+N6
S3:=S0+R7
S0:=N4+S0

1.11.5.7 LENGTH - Hole die Länge eines Strings

LENGTH

Mit dem folgenden Befehl kann die Länge eines Strings, also die aktuelle Anzahl Zeichen, ermittelt werden.

Befehlsform:

Ni:=LENGTH.Sn

Beispiele:

N6:=LENGTH.S0
N!7:=LENGTH.S4

1.11.5.8 Erzeuge XOR-Quersumme von einem String

Ni:=XOR.Sn

ab V5.07

Bei Ausführung dieses Befehls wird von einem String (S0, S1 bis S16) die XOR-Quersumme gebildet und in dem festgelegten N-Register abgelegt.

Befehlsform:

Ni:=XOR.Sn

N23:=XOR.S0

N34:=XOR.S3

1.11.5.9 Sn:=Ni / Sn:=Ri - Schreibe den Inhalt eines Registers (Zahl) in einen String

Schreibe Konstante in einen String

Sn:=Ni / Sn:=Ri

Mit diesem Befehl wird ein numerischer Wert in einen String konvertiert.

- Anzahl der Stellen und die Anzahl der Nachkommastellen ist optional (wie G520)
- Ist die Zahl kleiner als die Anzahl der vorgegebenen Stellen, dann werden führende Leerzeichen ausgegeben
- Ist die Anzahl größer als die Anzahl der Stellen, dann wird der String länger

Befehlsform:

Sn:=Ni

Sn:= Ni.(Anzahl Stellen)

SN:=Ri

SN:=Ri.(Anzahl Stellen).(Anzahl Nachkommastellen)

Beispiele:

S0:=N7

S0:=N8.5

S2:=N!7.N9

S4:=R2

S5:=R5.9.2

S4:=R6.N3.N7

S7:=R1.N!3.N!7

1.11.5.10 GET - Übertrage den Inhalt des lokalen Zeichenpuffers S0 in ein Ganzzahlregister

GET

GET.(Startposition).(Anzahl Zeichen)

Beschreibung:

Mit dem Befehl **GET** können bis zu 4 Zeichen aus dem lokalen Zeichenpuffer S0 in ein Ganzzahlregister übernommen werden.

GET beginnt ab der im Befehl festgelegten "**Startposition**" und nimmt die im Befehl definierte "**Anzahl Zeichen**".

mögliche Fehler:

- „**Startposition**“ ist kleiner 1 oder größer 80
- „**Anzahl Zeichen**“ ist kleiner 1 oder größer 4
- "**Startposition**" + "**Anzahl Zeichen**" ist größer 80

Befehlsform:

Ni:=GET.n.m

Ni:=GET.Nn.Nm

Ni:=GET.N!n.N!m

Beispiel

1 N1:=GET.1.2

;übernehme ab der ersten Stelle 2 Zeichen des lokalen Zeichenpuffers S0 und lege sie in N1 ab

n-1
n END

Lokaler Zeichenpuffer S0	Byte 1	Byte 2	Byte 3	Byte 4	Inhalt N1
3E8 _{hex}	03	E8	00	00		1000
64 _{hex}	00	64	00	00		100

1.11.5.11 GETI - Übertrage Inhalt des lokalen Zeichenpuffer S0 im INTEL-Format in ein Ganzzahlregister

GETI

Ni:=GETI.(Startposition).(Anzahl Zeichen)

Beschreibung:

Mit dem Befehl **GETI** können bis zu 4 Zeichen aus dem lokalen Zeichenpuffer S0 in ein Ganzzahlregister übernommen werden.

GETI beginnt ab der im Befehl festgelegten **"Startposition"** und nimmt die im Befehl definierte **"Anzahl Zeichen"**.

mögliche Fehler:

- „**Startposition**“ ist kleiner 1 oder größer 80
- „**Anzahl Zeichen**“ ist kleiner 1 oder größer 4
- **"Startposition" + "Anzahl Zeichen"** ist größer 80

Anwendung:

Die Übertragung von Prozessdatenobjekten über den CANopen-Bus kann im INTEL-Format erfolgen. Für eine weitere Bearbeitung ist die Wandlung der abgelegten Information notwendig.

Befehlsform:

Ni:=GETI.n.m

Ni:=GETI.Nn.Nm

Ni:=GETI.N!n.N!m

Beispiel

1 *N1:=GETI.1.2*

;übernehme ab der ersten Stelle 2 Zeichen des lokalen Zeichenpuffers S0 aus der INTEL-Darstellung und lege sie in N1 ab

n-1 ...
n END

Lokaler Zeichenpuffer S0	Byte 1	Byte 2	Byte 3	Byte 4	Inhalt N1
3E8 _{hex}	E8	03	00	00		1000
64 _{hex}	64	00	00	00		100

1.11.5.12 PUT - Übertrage Inhalt eines Registers in den lokalen Zeichenpuffer

PUT

PUT.(Startposition).(Anzahl Zeichen).(N-Register)

Beschreibung:

Mit dem Befehl **PUT** kann der Inhalt eines Registers byteweise in den lokalen Zeichenpuffer übertragen werden. Die Zeichen werden ab der "**Startposition**" im lokalen Zeichenpuffer abgelegt. Es werden die als "Anzahl Zeichen" festgelegten Zeichen abgelegt.

mögliche Fehler:

- „**Startposition**“ ist kleiner 1 oder größer 80
- „**Anzahl Zeichen**“ ist kleiner 1 oder größer 4
- "**Startposition**" + "**Anzahl Zeichen**" ist größer 80

Befehlsform:

PUT.n.m.Ni

PUT.Nn.Nm.Ni

PUT.N!n.N!m.Ni

PUT.n.m.Ri

Beispiel

```
1   N1:=100
2   PUT.1.4.N1
```

*;lade die Zahl 100 in das Ganzzahlregister 1
;schreibe den Inhalt von N1 ab der 1. Stelle mit
der Länge 4 Stellen (4Byte) als Binärzahl in
den lokalen Zeichenpuffer S0*

```
n-1 .....
n   END
```

Beispiel:

Lokaler Zeichenpuffer S0	1	2	3	4
100 _{dez} → 64 _{hex}	00	00	00	64
1000 _{dez} → 3E8 _{hex}	00	00	03	E8
100000 _{dez} → 186A0 _{hex}	00	01	86	A0

1.11.5.13 PUTI - Übertrage Inhalt eines Registers im INTEL-Format in den lokalen Zeichenpuffer S0

PUTI

PUTI.(Startposition).(Anzahl Zeichen).(N-Register)

Beschreibung:

Mit dem Befehl **PUTI** kann der Inhalt eines Registers byteweise in den lokalen Zeichenpuffer übertragen werden. Die Zeichen werden ab der "**Startposition**" im lokalen Zeichenpuffer abgelegt. Es werden die als "Anzahl Zeichen" festgelegten Zeichen abgelegt. Dabei wird die Ablage von Zahlen im Speicher im INTEL-Format berücksichtigt.

mögliche Fehler:

- „**Startposition**“ ist kleiner 1 oder größer 80
- „**Anzahl Zeichen**“ ist kleiner 1 oder größer 4
- "**Startposition**" + "**Anzahl Zeichen**" ist größer 80

Anwendung:

Da die Übertragung von Prozessdatenobjekten über den CANopen-Bus möglicherweise im INTEL-Format erfolgt, muss dies beim Zugriff auf den S0-String berücksichtigt werden.

Befehlsform:

PUTI.n.m.Ni

PUTI.Nn.Nm.Ni

PUTI.N!n.N!m.Ni

PUTI.n.m.Ri

Beispiel

```

1      N1:=100                ;lade die Zahl 100 in das Ganzzahlregister 1
2      PUTI.1.4.N1           ;schreibe den Inhalt von N1 ab der 1.Stelle mit
                             ;der Länge 4 Stellen (4Byte) als Binärzahl in den
                             ;lokalen Zeichenpuffer S0

```

```

n-1    .....
n      END

```

Beispiel :

Lokaler Zeichenpuffer S0	1	2	3	4
100 _{dez} → 64 _{hex}	64	00	00	00
1000 _{dez} → 3E8 _{hex}	E8	03	00	00
100000 _{dez} → 186A0 _{hex}	A0	86	01	00

1.12 Analog / Digital

1.12.1 Analog-Digital-Wandler - Allgemeine Beschreibung

Die PA-CONTROL kann optional mit AD-Wandlern ausgestattet werden, dabei muss zwischen den als IEF-Modul ausgeführten und den über CANopen-Bus angeschlossenen Analog-Digital-Wandlern unterschieden werden.

- (1) IEF-Modul-AD-Wandler 1-fach oder 8-fach. Es kann in der PA-CONTROL ein AD-Wandler-Modul eingesetzt werden.
- (2) Einsatz von bis zu maximal 32 AD-Wandlern über CANopen-Bus
 - ID 17: AD1...AD4
 - ID 18: AD5...AD8
 - ID24: AD31...AD32
 (Weitere Informationen entnehmen Sie bitte der Dokumentation APP5019_DE_1085270_PAC_mit_Ein-Ausgaengen_am_CAN-Bus.pdf)

1.12.1.1 AD-Werte holen

ADi

Beschreibung:

Der Wert des AD-Wandlers kann direkt übernommen werden. Der Anwender kann den Bit-Wert des AD-Wandlers in ein Ganzzahlregister oder das auf den Endausschlag parametrisierte Ergebnis in ein Realzahlregister übergeben.

Befehlsform:

Nn:=Adj

Rn:=Adj

	IEF-Modul 527451 (12-Bit-Übertragung)	IEF-Modul 1077914 (12-Bit-Übertragung)	BK 5120 (15-Bit-Übertragung)
Ni:=Adj	-4096...0...+4096	-4096...0...+4096	-32767 0 +32767
Ri:=Adj	-10VDC...0...+10VDC	-5VDC...0...+5VDC	-10VDC...0...+10VDC

Anwendung:

Erfassen von Temperaturen, Kräften, Geschwindigkeiten usw.

Beispiel

1	\$ANFANG	
2	N100:=2250	;Grenzwert festlegen
3	\$SCHLEIFE	;
4	N101:=AD1	;Der Bitwert wird in das Register N101 übernommen.
5	M100:=N100<N101	
3	G21 M100.1 SCHLEIFE	
4	O8:=1	;Fehlerlampe ein
5	I8.1 I8.0	;Quittungstaste für Fehler
6	O8:=0	;Fehlerlampe aus
7	R1:=AD1	;Der auf den vorgegebenen Endausschlag umgerechnete Wert (Spannung, die einem Druckwert, Temperatur, usw. entspricht) wird in das Realzahlregister R1 übernommen
8	JMP ANFANG	;
9	END	

1.12.2 Digital-Analog-Wandler - Allgemeine Beschreibung

Die PA-CONTROL kann optional mit bis zu 8 DA-Wandlern am CANOpen-Bus ausgestattet werden. Jeweils 4 DA-Wandler werden zu einer Gruppe (Hardwaremodul) zusammen gefasst.

CAN-ID 17 :	DA 1 bis	DA 4
CAN-ID 18 :	DA 5 bis	DA 8

Die ganze Gruppe wird über den CANOpen-Bus in einem Telegramm angesprochen und belegt feste PDO-Adressen (PDO, Process Data Objects). Weitere Informationen können der „APP5019_DE_1085270_PAC_mit_Ein-Ausgaengen_am_CAN-Bus.pdf“ entnommen werden.

HINWEIS Wird die Betriebsart Automatik verlassen, so werden die DA-Wandler auf den Wert 0 gesetzt!

Diagnose

Über die PA-CONTROL Frontplatte bzw. IEF-Bedienkonsole ist eine Diagnose der DA-Wandler möglich (siehe Kapitel 2 Bedienungsanleitung, Abschnitt CANopen-Bus)

HINWEIS Bei jedem Eintreten in diesen Menüpunkt werden die Ausgabewerte der DA-Wandler auf 0 gesetzt.

Mit der Enter-Taste wird das Eingabefeld geöffnet und es kann ein neuer DA-Wert eingegeben werden.

1.12.2.1 DAi - DA-Werte ausgeben

DAi

Befehlsform:

DAi:=Nn

DAi:=Rn

Anwendung:

Ausgabe von Analogwerten

Beispiel

1	\$ANFANG	
2	N2:=32767	<i>; Lade die Zahl 32767 in das N2-Register</i>
3	DA1:=N2	<i>; Inhalt von N2 als Analogwert ausgeben, dabei ergibt die Zahl 32767 bei einer Auflösung von 15 Bit/10 V eine Ausgabespannung von +10VDC</i>
4		<i>;</i>
5	R2:=5.2	<i>; Lade den Zahlenwert 5,2 ins Realzahlregister R2</i>
6	DA2:=R2	<i>; Inhalt von R2 als Analogwert ausgeben, dabei ergibt der Zahlenwert 5,2 bei einer Skalierung von 10V als maximaler Ausgabewert des Wandlers eine Ausgabespannung von 5,2V</i>
n	END	

1.13 Datei-Befehle

1.13.1 Befehle der G17x-Gruppe

1.13.1.1 G170 - Zeichen aus PTX-File lesen

G170

G170.(Zeile).(Spalte).(Zeichen).(File)

Beschreibung:

Der Befehl G170 liest aus einem PTX-File ein Zeichen aus und legt den ermittelten ASCII-Wert des Zeichens in das Ganzzahlregister **Nm ab** (siehe Tabelle „PAC-Tastencode + ASCII-Zeichensatz“ im Handbuch). Über die Ganzzahlregister **Ni** und **Nn** wird die genaue Position des Zeichens im File bestimmt. Hierbei gibt **Ni** die Zeilennummer und **Nn** die Stelle in der ausgewählten Zeile an.

HINWEIS

Der Wert der Ganzzahlregister Ni und Nn muss größer 0 sein.

Bei nicht vorhandenem Programmfile erfolgt die Ausgabe „Programm nicht vorhanden“. Ist die Zeile Ni in dem PTX-File nicht vorhanden oder die ausgewählte Zeile kürzer als Nn, wird das Register Nm auf 0 und der Systemmerker SM6 gesetzt.

Über den Systemmerker kann danach geprüft werden, ob der Befehl fehlerfrei ausgeführt wurde, SM6=0 → Befehl wurde fehlerfrei abgearbeitet, SM6=1 Befehlsabarbeitung fehlerhaft.

Befehlsform:

G170.Ni.Nn.Nm.TEXTE

G170.N!i.N!n.N!m.NAMEN

G170.Ni.Nn.Nm.Sx

G170.N!i.N!n.N!m.Sx

Beispiel

1	<i>N1:=5</i>	<i>;Zeilennummer in der Datei TEXTE.PTX</i>
2	<i>N2:=3</i>	<i>;drittes Zeichen in der 5. Zeile</i>
3	<i>; N10</i>	<i>;im Register N10 steht der eingelesene ASCII-Wert</i>
4	<i>G170.N1.N2.N10.TEXTE</i>	<i>;lese aus dem File TEXTE.PTX, aus der 5.Zeile das 3.Zeichen und schreibe den ASCII-Wert in Register N10</i>
5	<i>END</i>	

Programm: TEXTE.PTX

1	<i>MONTAG</i>
2	<i>DIENSTAG</i>
3	<i>MITTWOCH</i>
4	<i>DONNERSTAG</i>
5	<i>FREITAG</i>
6	<i>SAMSTAG</i>
7	<i>SONNTAG</i>

Ende der Belegung

1.13.1.2 G171 - Zeichen in PTX-File schreiben

G171

G171.(Zeile).(Spalte).(Zeichen).(File)

Beschreibung:

Der Befehl **G171** schreibt in ein PTX-File ein Zeichen. Über die Ganzzahlregister **Ni** und **Nn** wird die genaue Position des Zeichens im File bestimmt. Hierbei gibt **Ni** die Zeilennummer und **Nn** die Stelle in der ausgewählten Zeile an.

Der Zahlenwert aus dem Ganzzahlregister **Nm** wird in ein ASCII-Zeichen gewandelt (*siehe Tabelle „PAC-Tastencode + ASCII-Zeichensatz“ im Handbuch*). Unerlaubte Zeichen sind: 0x00=NUL, 0x0A=LF und 0x0D=CR.

HINWEIS Der Wert der Ganzzahlregister Ni und Nn muss größer 0 sein.

Bei nicht vorhandenem Programmfile erfolgt die Ausgabe „Programm nicht vorhanden“. Ist die Zeile Ni in dem PTX-File nicht vorhanden oder die ausgewählte Zeile kürzer als Nn, wird der Systemmerker SM6 gesetzt.

Über den Systemmerker kann danach geprüft werden, ob der Befehl fehlerfrei ausgeführt wurde, SM6=0 → Befehl wurde fehlerfrei abgearbeitet, SM6=1 Befehlsabarbeitung fehlerhaft.

Befehlsform:

G171.Ni.Nn.Nm.TEXTE

G171.N!i.N!n.N!m.NAMEN

G171.Ni.Nn.Nm.Sx

G171.N!i.N!n.N!m.Sx

Beispiel

1	N1:=5	;Zeilennummer in der Datei TEXTE.PTX
2	N2:=3	;drittes Zeichen in der 5. Zeile
3	; N10	;im Register N10 steht der eingelesene ASCII-Wert
4	G171.N1.N2.N10.TEXTE	;überschreibt in der 5.Zeile des Files TEXTE.PTX das 3.Zeichen, mit dem ASCII-Zeichen (Wert aus N10)
5	END	

Programm: TEXTE.PTX

```

1   MONTAG
2   DIENSTAG
3   MITTWOCH
4   DONNERSTAG
5   FREITAG
6   SAMSTAG
7   SONNTAG

```

Ende der Belegung

1.13.1.3 G172 - Zeile aus PTX-File in String (Sn) schreiben

G172

G172.(Zeile).(String).(File)

Beschreibung:

Der Befehl **G172** kopiert aus einem PTX-File die im Befehl angegebene Zeile in eine Zeichenkette. Die Zeile wird dabei direkt im Befehl angegeben oder über ein Register adressiert. Die Zieladresse kann eine globale aber auch die lokale Zeichenkette sein.

HINWEIS Bei nicht vorhandenem File erfolgt die Ausgabe der Fehlermeldung „Programm nicht vorhanden“. Ist die adressierte Zeile nicht vorhanden oder die Zeile länger als die maximale Länge eines Strings, dann wird der Systemmerker „SM6“ gesetzt. Über den Systemmerker kann danach geprüft werden, ob der Befehl fehlerfrei ausgeführt wurde, SM6=0 → Befehl wurde fehlerfrei abgearbeitet, SM6=1 Befehlsabarbeitung fehlerhaft.

Befehlsform:

G172.4.S2.FEHLERTEXTE

G172.N8.S7. MENUETEXTE

G172.N!5.S6. MELDETEXTE

G172.4.S2.S0

G172.N8.S7.S1

G172.N!5.S6.S2

Beispiel 1

```
1   G172.5.S2.MENUETEXTE           ;Kopiert aus dem File „MENUETEXTE.PTX“ die
                                     fünfte Zeile in die Zeichenkette „S2“
2   END
```

Beispiel 2

```
1   N8:=12                          ; Zeilennummer der Datei FEHLERTEXTE.PTX
2   G172.N8.S0.FEHLERTEXTE         ; Kopiert aus dem File „FEHLERTEXTE“ die Zeile
                                     12 in den lokalen Zeichenpuffer S0
3   END
```

Beispiel 3

```
1   S1:="EINSTELLUNGEN_00"
2   N8:=12                          ; Zeilennummer der Datei *.PTX
3   G172.N8.S0.FEHLERTEXTE         ; Kopiert aus dem File „EINSTELLUNGEN_00“
                                     die Zeile 12 in den lokalen Zeichenpuffer S0
4   END
```

1.13.1.4 G173 - String (Sn) in einer Zeile eines PTX-Files speichern

G173

G173.(Zeile).(String).(File)

Beschreibung:

Der Befehl **G173** speichert den Inhalt des Strings „Sn“ in eine ausgewählte Zeile eines PTX-Files. Die Zeile wird dabei direkt im Befehl angegeben oder über ein Register adressiert. Die Zieladresse kann eine globale aber auch die lokale Zeichenkette sein.

HINWEIS

Bei nicht vorhandenem File erfolgt die Ausgabe der Fehlermeldung „Programm nicht vorhanden“. Ist die adressierte Zeile nicht vorhanden oder der String länger als die Zeile, dann erfolgt die Ausgabe der Fehlermeldung „STORE-Befehl konnte nicht richtig umgesetzt werden“.

Ist die Zeile des PTX-Files länger, so wird der Rest mit Leerzeichen überschrieben. Eine Zeile sollte nicht länger als 80 Zeichen sein, da sonst bei der Anwendung des Befehls G172 die Ausgabe der Meldung „Zeile zu lang“ erfolgt.

Befehlsform:

G173.6.S0.MITTEILUNGEN

G173.N2.S1.FEHLER_1

G173.N!9.S2.FEHLER_2

G173.6.S0.S1

G173.N2.S1.S0

G173.N!9.S2.S11

Beispiel 1

```
1 G173.6.S0.MITTEILUNGEN
2 END
```

;Speichert den Inhalt des lokalen Strings „S0“ in der 6. Zeile des Files „MITTEILUNGEN“ ab.

Beispiel 2

```
1 N2:=21
2 G173.N2.S1.FEHLER_1
3 END
```

; Zeilennummer der Datei FEHLER_1.PTX
; Speichert den Inhalt des globalen Strings „S1“ in der 21. Zeile des Files „FEHLER_1.PTX“ ab.

Beispiel 3

```
1 S0:="FEHLER_INFO"
2 N2:=21
3 G173.N2.S1.S0
4 END
```

; Zeilennummer der Datei *.PTX
; Speichert den Inhalt des globalen Strings „S1“ in der 21. Zeile des Files „FEHLER_INFO.PTX“ ab.

1.13.1.5 STORE - Speichern aktueller Werte in einen Programm-File

STORE

Store.(Name)

Beschreibung:

Mit dem Befehl **STORE** werden die aktuellen Werte von N-Registern, R-Registern, Strings und Merkern in einem PNC-, PNX- oder PAB-Programm abgelegt.

Beim Ausführen des **STORE**-Befehls wird das aufgerufene Programm nach Zuweisungen durchsucht. Wird dabei eine Register-, Merker- oder Stringzuweisung gefunden, so wird der aktuelle Wert in die Zuweisung geschrieben. Hierzu müssen genügend Leerstellen in den PNX-, PNC- oder PAB-Programmen zur Verfügung stehen.

- minimal benötigte Anzahl von Platzhaltern für Register: 9 Stellen
- minimal benötigte Anzahl von Platzhaltern für Merker: 1 Stelle
- minimal benötigte Anzahl von Platzhaltern für Strings: aktuelle Stringlänge + 1 Stelle

HINWEIS

Der **STORE-Befehl** darf nicht das Programm aufrufen in dem er selbst steht.

Der **STORE-Befehl** darf nur Programme aufrufen, die **nicht** zum Zeitpunkt aktiv sind.

Steht der **STORE-Befehl** in einem Unterprogramm, darf das übergeordnete Programm nicht aufgerufen werden.

Vor dem Ausführen des STORE-Befehls

1	R10:=000000000	; A1 Abholposition
2	R11:=000000000	; A1 Zwischenposition
3	N10:=000000000	; Anzahl der zu bearbeiteten Produkte
4	M101:=0	; Produktmerker für Produkt 1
5	S1:="000000"	; Begrüßung
6	END	

Nach dem Ausführen des STORE-Befehls

1	R10:=145.345	; A1 Abholposition
2	R11:=34565.789	; A1 Zwischenposition
3	N10:=645	; Anzahl der zu bearbeiteten Produkte
4	M101:=1	; Produktmerker für Produkt 2
5	S1:="HALLO"	; Begrüßung
6	END	

HINWEIS Das aufgerufene PNX/PNC/PAB-Programm muss vorhanden sein. In einer Programmzeile darf nur eine Zuweisung stehen.

Es müssen genügend Nullen oder Leerzeichen zum Überschreiben nach der Zuweisung bis zum ";" (Kommentar) oder "CR" (Zeilenende) vorhanden sein.

Rechenzuweisungen sind in dem zu überschreibendem Programm **nicht** erlaubt.

Ist die STORE Anweisung auf Grund nicht ausreichendem freien Platz nicht erfolgreich, so wird der Fehler E569 gesetzt.

Befehlsform:

STORE.(Name)

Anwendung:

Ab speichern von aktuellen Werten in unterschiedlichen Programm-Dateien. Für verschiedene Produkttypen können z.B. Verfahrspositionen (R-Register) oder Zustandsmerker abgespeichert und je nach Anforderung wieder aufgerufen werden.

Beispiel

```

1   R10:=R13           ;Wert über Register einlesen z.B. 4678.123
2   N1:=N13           ;Wert über Register einlesen z.B. 967
3   M25:=N1>11       ;M25 wird gesetzt, wenn Inhalt von N1 größer
                       als 11 ist.
4   S1:="GREIFER"     ; GREIFER wird in den String S1 geladen
5   STORE.TA2_PE_001  ;überschreibe Zuweisungen im PNC-
                       Programm TA2_PE_001 mit aktuellen Werte
                       von Registern und Merkern

6   END

```

Programm TA2_PE_001 vor dem Ausführen des STORE-Befehls:

```

1   R10:=000000000    ; A1 Holposition für Produkttyp 1
2   N1:=000000000    ; Anzahl der zu bearbeiteten Produkte
3   M25:=0            ; Produktmerker für Produkt 1
4   S1:="SAUGER1"     ; Beschriftung für spätere Ausgabe
5   END

```

Programm TA2_PE_001 nach dem Ausführen des STORE-Befehls:

```

1   R10:=4678.123     ; A1 Abholposition
2   N1:=967           ; Anzahl der zu bearbeiteten Produkte
3   M25:=1           ; Produktmerker für Produkt 1
4   S1:="GREIFER"     ; Beschriftung für spätere Ausgabe
5   END

```

HINWEIS Im Beispiel sind Leerzeichen durch einen Unterstrich "_" dargestellt.

1.13.1.6 STOREEND / STORESTART - Zuweisung mit Konstanten deaktivieren / aktivieren

STOREEND STORESTART

ab V5.07

Beschreibung:

Bei der Ausführung des STORE-Befehls wurden bisher alle „Zuweisungen mit Konstanten“ durch den STORE-Befehl bearbeitet.

Sollen in einem Programm einige „Zuweisungen mit Konstanten“ nicht bearbeitet werden (also der Wert soll unverändert erhalten bleiben) so kann mit den Befehlen „STOREEND“ und „STORESTART“ die Bearbeitung deaktiviert und wieder aktiviert werden.

Beispiel:

Im folgenden Beispiel sollen die Konstanten für die Zuweisung der Variablen „N2“ und „R1“ unverändert erhalten bleiben.

Beispiel

Programm: Start.pnc

```
1   $A
2   G21 I9.1 SPEICHERE_W_TYP1
3   G22 I10.1 LADE_TYP1
4   ;
5   $ SPEICHERE_W_TYP1
6   STRORE LADE_TYP1
7   JMP A
8   ;
9   END
```

Programm: LADE_TYP1

```
1   N1:=1
2   STOREEND
3   N2:=2
4   R1:=0.000000
5   STORESTART
6   R2:=0.000000
7   M1:=0
8   M2:=0
9   S2:="OTTO IST GROSS"
10  END
```

STOREEND schaltet die Bearbeitung vor „N2:=...“ aus

STORESTART schaltet die Bearbeitung nach „R1:=...“ wieder ein

1.13.1.7 CASE.STORE.N - Speichern wenn Inhalt von N-Register

CASE.STORE.N

CASE.STORE.(Variable)

Beschreibung:

Der **CASE.STORE**-Befehl ist eine Programmverzweigung in Abhängigkeit von dem Inhalt eines Ganzzahlregisters **Ni**. Die PA-CONTROL prüft den Inhalt des Ganzzahlregisters und führt entsprechend dem Wert ein Unterprogrammaufruf durch. Ist der Inhalt des Ganzzahlregisters kleiner 1 oder größer der Anzahl der Elemente der Namentabelle (hier im Beispiel: TA2:=PE_001, TA2:=PE_002), so wird ein Sprung auf die Marke, die hinter ELSE definiert ist, durchgeführt. Ansonsten wird das entsprechende Unterprogramm aufgerufen und abgearbeitet und anschließend mit der Programmzeile nach dem ELSE-Zweig weitergemacht.

Mit dem Befehl **CASE.STORE** werden die aktuellen Werte von N-Registern, R-Registern und Merkern in einem PNC/PAB-Programm abgelegt (siehe STORE-Befehl).

Befehlsform:

CASE.STORE.Ni

(Name1)

(Name2)

...

ELSE Marke_e

Anwendung:

Programmverzweigung zum Abspeichern von aktuellen Registerwerten in unterschiedlichen PNC/PAB-Dateien. Für verschiedene Produkttypen können z.B. Verfahrspositionen (R-Register) oder Zustandsmerker abgespeichert und je nach Anforderung wieder aufgerufen werden.

Beispiel

<pre> 1 \$FEHLER 2 N8:=N13 3 CASE.STORE.N8 4 (TA2_PE_001) 5 (TA2_PE_002) 6 ELSE FEHLER 7 END </pre>	<pre> ;Wert über Register einlesen ;prüfe den Inhalt von N8 und Aufruf von Unter- programm bei: ;N8=1: Unterprogramm „ TA2:=PE_001“ auf Zu- weisungen durchsuchen und aktualisieren N8=2 : Unterprogramm „ TA2:=PE_002“ auf Zu- weisungen ;durchsuchen und aktualisieren ;Sprung zur Marke „FEHLER“, wenn der Wert von N8 außerhalb der vorgesehenen Grenzen (hier wenn N8<1 oder N8>2) ist </pre>
Programm: TA2_PE_001	
<pre> 1 R10:=0 2 R11:=0 3 N10:=0 4 M101:=0 5 END </pre>	<pre> ;A1-Holposition für Produkttyp 1 ;A1-Zwischenposition für Produkttyp 1 ;Anzahl der zu bearbeitenden Produkte ;Produktmerker für Produkt 1 </pre>
Programm: TA2_PE_002	
<pre> 1 R10:=0 2 R11:=0 3 N10:=0 4 M102:=0 5 END </pre>	<pre> ;A1-Holposition für Produkttyp 2 ;A1-Zwischenposition für Produkttyp 2 ;Anzahl der zu bearbeitenden Produkte ;Produktmerker für Produkt 2 </pre>

1.14 Zähler

1.14.1 Allgemeine Grundlagen

Ab der Version 5.04E kann die PA-CONTROL über den CAN-Bus mit bis zu 16 Hardware-Zählern (CNT1 bis CNT16) ausgestattet werden. Bei der PA-CONTROL smart ist am Stecker X1 ein weiterer Zähler (CNT0) vorhanden.

Von den Funktionen (Betriebsarten) der Busklemmen (KL5101, KL5111) wurden die Betriebsart „Inkremental-Encoder“ als „Quadraturdecoder“ mit 4-fach Auswertung und der Vor/Rückwärtszähler implementiert. Der Hardwarezähler (CNT0) an der PA-CONTROL smart ist nur mit der Funktion „Quadraturdecoder“ mit 4-fach Auswertung implementiert.

Nach dem Einschalten und beim Neuinitialisieren der PA-CONTROL werden die Zähler immer auf 0 und auf die Betriebsart „ENCODER“ gesetzt.

Beschreibung	CNT 0 (nur PA-CONTROL smart)	CNT1 bis CNT16 (über CAN-Bus, KL5101, KL5111)
Eingangspegel	RS422	siehe KL51xx
Maximale Eingangsfrequenz	200000 Inkremente / s	siehe KL51xx, über den CAN-Bus wird der Zählerstand ca. alle 2ms aktualisiert
Vor- / Rückwärtszähler	Nein	Ja
Quadraturdecoder	Ja	Ja
Zählerwert	31Bit plus Vorzeichen	31Bit plus Vorzeichen
Diagnose mit WINPAC	SN110	SN111 bis SN126

1.14.2 Hardware

An der PA-CONTROL kommt die CANopen-Busklemme von Beckhoff BK5120 mit den Klemmen „KL5101“ oder „KL511“ zur Anwendung.

Es wurden die folgenden Festlegungen getroffen :

- maximal 2 Klemmen pro Busknoten
- maximal 16 Busknoten mit Zählerklemmen im CANopen-System

Nach dem Einschalten werden die vorhandenen Zählerklemmen der Busknoten in aufsteigender Folge den Zählern der PA-CONTROL zugeordnet.

Beispiel:

CAN-ID – Busklemme	Bestückt mit Klemmen	Zähler (CNT) in der PA-CONTROL
ID17 – BK5120	4 x KL2134 1 x KL5111	CNT1
ID18 – BK5120	4 x KL1104 1 x KL4032 1 x KL3064	-
ID19 – FESTO CPV	-	-
ID20 – BK5120	1 x KL1104 1 x KL5111 1 x KL5101	- CNT2 CNT3

1.14.3 Befehle

1.14.3.1 Zähler initialisieren

CNTn.INIT.i

CNT4.INIT.0

CNT2.INIT.1

Beschreibung:

Mit Hilfe des Befehls **CNTn.INIT.i** erfolgt die Initialisierung der Zähler über den CANopen-Bus angeschlossenen Zähler.

Befehlsform:

CNT4.INIT.0 ; Zähler 4 arbeitet in der Betriebsart „Encoder“

CNT2.INIT.1 ; Zähler 2 arbeitet in der Betriebsart „Vor/Rückwärtszähler“

Betriebsart : „Inkremental-Encoder“ als „Quadraturdecoder“ mit 4-fach Auswertung		
Eingang	Funktion des Einganges	
Input A	Encoder-A	
Input B	Encoder-B	
Input C	Keine Funktion	

Betriebsart : Vor / Rückwärtszähler		
Eingang	Funktion des Einganges	
Input A	Count, gezählt werden die positiven Flanken der Eingangsimpulse	
Input B	Vor / Rückwärtseingang 0 : Zählrichtung vorwärts 1 : Zählrichtung rückwärts	
Input C	Gate-Eingang : 0 : Zähler ist freigegeben 1 : Zähler ist gesperrt	

Anwendung:

Festlegung der „Betriebsart“ der erkannten Zähler am CANopen-Bus (Betriebsart Encoder- oder Vorwärts/Rückwärtszähler).

1.14.3.2 Zähler setzen

CNTn:=Ni

CNTn:=<Konstante>

CNTn:=Ni

Beschreibung:

Mit Hilfe des Befehls **CNTn.INIT.i** kann einem über CANopen-Bus angeschlossenen Zähler ein Wert zugewiesen werden..

Befehlsform:

CNT1:=1234

CNT3:=N67

Anwendung:

Festlegung von Ausgangswerten für Zählvorgänge, z.B. Positionen

1.14.3.3 Zähler auslesen

Ni:=CNTn

Ni:=CNTn

Beschreibung:

Mit Hilfe des Befehls **Ni:=CNTn** kann aus einem über CANopen-Bus angeschlossenen Zähler ein Wert ausgelesen werden. Der ausgelesene Wert wird in das Ganzzahlregister Ni abgelegt.

Befehlsform:

N34:=CNT2

Anwendung:

Abfragen von Zählerständen

1.14.3.4 Zähler vergleichen

CNTn.i.Nm

CNTn.i.Nm

Beschreibung:

Mit Hilfe des Befehls **CNTn.i.Nm** kann der Inhalt eines über CANopen-Bus angeschlossenen Zählers mit einer Konstante verglichen werden.

Befehlsform:

CNT2.1.1234	warte bis der Zählerstand des Zählers 2 größer Wert 1234
CNT2.0.-345	warte bis der Zählerstand des Zählers 2 kleiner Wert -345
CNT2.1.N4	warte bis der Zählerstand des Zählers 2 größer als der Inhalt von N4
CNT2.0.N4	warte bis der Zählerstand des Zählers 2 kleiner als der Inhalt von N4

1.15 Temperaturregelung

1.15.1 Allgemeine Grundlagen

Ab Version 5.07 kann über die PA-CONTROL eine Temperaturregelung erfolgen. Zum Einsatz kommt dafür das Modulare Mehrkanal-Regelsysteme „KS VARIO“, dass über den CAN-Bus angeschlossen wird. Die Feldbusklemme kann mit einem 4-Kanal oder 30-Kanaltemperaturregler ausgestattet sein (*siehe dazu im Bedienerhandbuch der PA-CONTROL Kapitel Optionen*).

Die Kommunikation mit den Temperaturreglern „KS VARIO“ erfolgt ausschließlich über SDOs. Dafür wurden für den Automatikbetrieb der PA-CONTROL die Befehle:

- Parameterwert schreiben (siehe Abschnitt 1.15.2.1, Seite 223) und
- Parameterwert lesen (siehe Abschnitt 1.15.2.2, Seite 224)

geschaffen.

Die Inbetriebnahme des Temperaturreglers und die Einstellung der Schnittstellendaten erfolgt mit dem Programm „Blue Control“ der Fa. PMA.

1.15.1.1 Parameter

Für die Programmierung sind die Parameter der nachstehenden Tabelle wichtig:

Bezeichnung im Befehl <Parameter-name>	Verwendung für		Bezeichnung Fa. PMA	Beschreibung
SP	N,R	R/W	SP	Sollwert des Reglers, (Temperaturvorgabe in °C)
SP_LO	N,R,	R/W	SP.LO	Minimaler Grenzwert für die Sollwertvorgabe
SP_HI	N,R	R/W	SP.HI	Maximaler Grenzwert für die Sollwertvorgabe
X_EFF	N,R	R	X.Eff	Wirksamer Istwert
X_PHYS	N,R,	R	Xphys	physikalischer Istwert direkt von der Klemme
C_OFF	N	R/W	X.Off	Regler ausschalten 1 = Regler aus, 0 = Regler an
YPID	N,R	R	Ypid	Stellgröße des Reglers
DIGOUT1	N	R	DIGOUT1	Signal am digital Ausgang 1
DIGOUT2	N	R	DIGOUT2	Signal am digital Ausgang 1
C_STA	N	R	C.Sta	Status des Reglers
C_STA2	N	R	C.Sta2	Status des Reglers
BOOST	N	R/W	BOOST	Die Boost-Funktion bewirkt eine kurzzeitige Erhöhung des Sollwertes

Die detaillierte Beschreibung der einzelnen Parameter entnehmen Sie bitte der Dokumentation „KSvario CON.pdf“ der Fa. PMA.

1.15.2 Befehle für die Temperaturregelung

1.15.2.1 Parameterwert schreiben

TCn.Parametername:=

ab V5.07

TC3.Sp:=25.5

TC2.C_OFF:=1

Beschreibung:

Mit Hilfe des Befehls **TCn.Parametername:=Wert** wird ein Parameterwert aus der Steuerung in den Regler übertragen

Befehlsform:

TC<KanalNummer>.<Parametername>:=<Wert>

TC<KanalNummer>.<Parametername>:=Ni

TC<KanalNummer>.<Parametername>:=Rn

TC<KanalNummer>.<Parametername>:=INTEGER_KONSTANTE

TC<KanalNummer>.<Parametername>:=REAL_KONSTANTE

Beispiel:

Siehe komplexes Beispiel auf Seite 224.

1.15.2.2 Parameterwert lesen

Rn:=TCi.Parametername

ab V5.07

R27:=TC1.X_EFF

N35:=TC2.C_STA

Beschreibung:

Mit Hilfe des Befehls **Rn:=Tci. Parametername** wird ein aktueller Wert vom Regler in die PA-CONTROL geholt

Befehlsform:

Register:=TC<KanalNummer>.<Parametername>

Rn:=TC<KanalNummer>.<Parametername>

Nn:=TC<KanalNummer>.<Parametername>

Beispiel

```
1                                     ; Temperatur Controller
2   N10:=0
3   TC1.C_OFF:=N10                   ; Temperatur Controller einschalten
4
5   R10:=100.0                       ; Sollwert laden
6   TC1.SP:=R10                      ; Sollwert setzen
7
8   $LOOP
9   R11:=TC1.X_PHYS                  ; Istwert auslesen
10  M10:=R11<R10
11  G21 M10.1 LOOP                   ; warten, bis Ende Temperatur Regelung
12  M10:=0                           ; Reset, Ende Temperatur Regelung
13
14  N10:=1
15  TC1.C_OFF:=N10                   ; Temperatur Controller ausschalten
16  END
```

1.16 Befehlsgruppen der PA-CONTROL

1.16.1 Befehle der G18x-Gruppe

1.16.1.1 G18x. - Erfassen von mehreren Werten

G18x.

Einleitung:

Die **G180** Befehlsgruppe dient zum Erfassen von Drehmoment- oder AD-Wandlerwerten. Die Werte werden nach den Erfassungskriterien Position/Weg oder Zeit über den CAN-Bus empfangen oder vom AD-Wandler geholt und in den N-Registern abgelegt. Die abgelegten Werte können dann weiter verarbeitet werden.

1.16.1.2 G180. - AD-Werte synchron zur Achsbewegung erfassen

G180.

G180.(Achse).(Position).(Auflösung).(AD-Nr).(1.N-Register).(Anzahl)

Befehlsform:

G180.A1.R27.R31.5.100.25 (siehe folgende Tabelle)

G180.A3.R23.R2.N7.N2.N3

G180.A1.R!23.R!2.N!7.N!2.N!3

- Achse** / z.B. A1 : Achse, die überwacht wird, immer als Konstante
- Position** : In diesem R-Register ist hinterlegt, ab welcher Achsposition die AD-Werte erfasst werden. Ob die Erfassung gestartet wird, wenn die Position größer oder kleiner ist, wird vom Vorzeichen der Auflösung abgeleitet. Es muss ein R-Register verwendet werden.
z.B. R-Reg. 27
- Auflösung** / : In diesem R-Register ist die Auflösung, also der Weg nach welchem der nächste AD-Wert geholt werden soll, abgelegt. Das Vorzeichen der Auflösung gilt für das Starten der Erfassung ob größer oder kleiner als Merkmal. Es muss ein R-Register verwendet werden.
z.B. R-Reg. 31
- AD-Nr.** : Gibt die Nummer (von 1 bis 8) des AD-Wandlers an, von welchem die Werte geholt werden.
z.B. AD-Wandler 5
- 1.N-Register** / : Gibt die Nummer des ersten N-Registers an, ab welchem die AD-Werte abgelegt werden. Mit jedem neuen Messwert wird die Adresse des N-Registers um 1 erhöht. Die Angabe kann als Konstante oder über ein N-Register erfolgen.
z.B. N-Reg. 100
- Anzahl** / : Gibt die maximale Anzahl der AD-Werte an, die erfasst werden sollen. Diese Zahl bestimmt auch die Anzahl der für die Ablage der Messwerte notwendigen Ganzzahlregister. Die Angabe kann als Konstante oder über ein N-Register erfolgen.
z.B.
max. Anzahl = 25

HINWEIS In den System-N-Registern SN51 – 66 (SN51 für Achse 1, SN52 für Achse 2 usw.) wird die Anzahl der erfassten AD-Werte abgelegt und steht für Abfragen zur Verfügung. Wird die Achse gestoppt oder ausgeschaltet so wird der G180-Befehl beendet.

Maximale Abtastrate:

HINWEIS	PA-CONTROL MP mit AD-Wandler 1-fach	:	0,2ms
	PA-CONTROL MP mit AD-Wandler 8-fach	:	1ms
	PA-CONTROL SINGLE - " -	:	1ms
	PA-CONTROL COMPACT - " -	:	1ms
	PA-CONTROL STEUER - " -	:	1ms

Beispiel 1

```

1  G25.A1 G90.A1
2  R1:=10                ;Erfassung ab Achsposition 10mm
3  R2:=0.5              ;alle 0,5mm soll ein AD-Wert erfasst werden
4  G180.A1.R1.R2.1.100.200 ;erfasse bei der nächsten Positionierung der
                           ;Achse 1, ab der Position 10mm, alle 0,5mm
                           ;vom AD-Wandler 1 maximal 200 Werte und
                           ;lege diese ab dem N-Register 100 im Spei-
                           ;cher ab
5  N1000:=SN51          ;lade System-N-Register 51 ins N-Register
                           ;1000
6  M55:=N1000>180      ;Vergleiche Inhalt N-Reg. 1000 mit 180
7  G22 M55.1 FEHLER    ; springe nach UP FEHLER, wenn <SN51>
                           ;>180
8
9  A1:=1000            ; fahre auf Position 1000
10 END

```

Programm: FEHLER

```

2  O32:=1              ;Fehlerlampe einschalten
3  I32.1              ;Warte auf Quittierung
4  I32.0              ;
5  O32:=0             ;Fehlerlampe ausschalten
6  END

```

1.16.1.3 G181 - AD-Werte im definierten Zeitraster erfassen

G181

(G181.Zeitraster.AD-Nr.Startregister.Anzahl)

Beschreibung:

Die AD-Werte werden im angegebenen Zeittakt erfasst.

Befehlsform:

G181.5.1.100.50

G181.N1.1.N2.N3

Anwendung:

Erfassen von dynamischen AD-Werten, z.B. Drehmomentwerte beim Schrauben oder Kraftwerte beim Fügen und Pressen.

Beispiel 1

1	G210.A1	<i>;aktiviert den Startpositioniermode für Achse 1</i>
2	A1:=100	<i>;Position 100 anfahren</i>
3	G230.1.A1.50	<i>;warte bis die Position >50.0 erreicht ist</i>
4	G181.5.1.100.50	<i>;im 5ms Takt werden Werte des AD-Wandlers 1 gelesen und ab N-Register 100 bis N-Register 149 abgelegt.</i>
5	G213.A1	<i>;gehe in den Normalpositioniermode</i>
6	END	

1.16.1.4 G182 - AD-Werte synchron zum Counterwert (CNT0) holen

G182

ab V5.13

(G182. <Zähler-Nr>. <Anfangswert>. <Auflösung>. <AD-Nr>. <1.N-Register>. <Anzahl>)

Beschreibung:

Die AD-Werte werden in Abhängigkeit des Zählerstandes des Zähler 0 erfasst und in N-Registern in der PA-CONTROL abgelegt

Befehlsform:

G182.0.N10.N11.1.100.10

G182.0.N10.N11.1.N11.N12

G182.0.N!10.N!11.1.N!11.N!12

Anwendung:

Erfassen von dynamischen AD-Werten beim Verfahren einer Achse : z.B. Drehmomentwerte beim Schrauben oder Kraftwerte beim Fügen und Pressen oder Hüllkurve eines Magnetes.

Zähler-Nr.: Zähler, der überwacht wird, immer 0, nur für CNT0 möglich,

Anfangswert : In diesem N-Register ist hinterlegt, ab welchem Zählerwert die AD-Werte erfasst werden. Ob die Erfassung gestartet wird, wenn die Position größer oder kleiner ist, wird vom Vorzeichen der Auflösung abgeleitet. Es muss ein N-Register verwendet werden. Der Anfangswert muss zwischen -32768 und 32767 liegen (15Bit).

Auflösung: In diesem N-Register ist die Auflösung, also die Differenz des Zählerstandes nach welchem der nächste AD-Wert geholt werden soll, abgelegt. Das Vorzeichen der Auflösung gilt für das Starten der Erfassung ob größer oder kleiner als Merkmal. Es muss ein N-Register verwendet werden. Die Auflösung muss zwischen -32768 und 32767 liegen.

AD-Nr. : Gibt die Nummer (von 1 bis 8) des AD-Wandlers an, von welchem die Werte geholt werden.

1.N-Register : Gibt die Nummer des ersten N-Registers an, ab welchem die AD-Werte abgelegt werden. Mit jedem neuen Messwert wird die Adresse des N-Registers um 1 erhöht. Die Angabe kann als Konstante oder über ein N-Register erfolgen.

Anzahl: Gibt die maximale Anzahl der AD-Werte an, die erfasst werden sollen. Diese Zahl bestimmt auch die Anzahl der für die Ablage der Messwerte notwendigen Ganzzahlregister. Die Angabe kann als Konstante oder über ein N-Register erfolgen.

HINWEIS Der Befehl G182 ist nur bei der PA-CONTROL smart möglich

Die Parameter <Anfangswert> und <Auflösung> sind auf 15-Bit begrenzt (+/- 326767)

Die Einstellung für das Zählereingangssignal sollten so gewählt werden, dass die maximale Eingangsfrequenz des Zähler 0 mit 200000 Inkremente pro Sekunde nicht überschritten wird.

In dem System-N-Registern SN130 wird die Anzahl der erfassten AD-Werte abgelegt und steht für Abfragen zur Verfügung.

Beispiel 1

Mit der Achse 1 wird eine Bewegung (rotatorisch oder translatorisch) durchgeführt. Synchron zu dieser Bewegung sollen AD-Werte (ca. alle 1ms ein AD-Wert) mit hoher Wiederholgenauigkeit erfasst werden

Um dies zu realisieren wurde folgender Hardwareaufbau gewählt:

- PA-CONTROL smart mit
 - AD-Wandler 1-fach (8-fach)
 - CNT0
- LV servoTEC S2
- Verdrahtung LV-servoTEC S2 Ausgang „Encoderemulation“ X11 auf den Zähler 0 der PA-CONTROL smart

1	N1:=20	<i>; Starte Erfassung der AD-Werte ab dem Zählerwert 20 (größer gleich)(siehe G182-Befehl)</i>
2	N2:=10	<i>; hole wenn der Zählerstand um 10 größer geworden ist den nächsten AD-Wert (siehe G182-Befehl)</i>
3	CNT0:=0	<i>;Zähler 0 auf einen definierten Anfangswert setzen</i>
4	G182.0.N1.N2.1.10.1000	<i>Initialisiere G182-Befehl :</i> <i>; Zählernummer : 0</i> <i>; Zähler Anfangswert : N1:=20</i> <i>; Zähler Auflösung : N2:=10</i> <i>; AD-Wandler-Nr : 1</i> <i>; 1.N-Register : 10</i> <i>; max. Anzahl : 1000</i>
5	A1:=100	<i>;Position 100 anfahren und AD-Werte während dieser Bewegung erfassen</i>
6	N3:=SN130	<i>Hole die Anzahl der erfassten AD-Werte</i>
7	END	

1.16.1.5 G183 - AD-Werte synchron zum Counterwert (CNT0) holen

G183

ab V5.13

(G183. <Zähler-Nr>. <Anfangswert>. <Auflösung>. <AD-Nr>. <CAN-ID>. <Anzahl>)

Beschreibung:

Die AD-Werte werden in Abhängigkeit des Zählerstandes des Zähler 0 erfasst und sofort als TxPDO über den CAN-Bus versendet.

Befehlsform:

G183.0.N10.N11.1.33.10

G183.0.N10.N11.1.N23.N12

G183.0.N!10.N!11.1.N!23.N!12

Anwendung:

Erfassen von dynamischen AD-Werten beim Verfahren einer Achse : z.B. Drehmomentwerte beim Schrauben oder Kraftwerte beim Fügen und Pressen oder Hüllkurve eines Magnetes.

Zähler-Nr: Zähler, der überwacht wird, immer 0, nur für CNT0 möglich,

Anfangswert: In diesem N-Register ist hinterlegt, ab welchem Zählerwert die AD-Werte erfasst werden. Ob die Erfassung gestartet wird, wenn die Position größer oder kleiner ist, wird vom Vorzeichen der Auflösung abgeleitet. Es muss ein N-Register verwendet werden. Der Anfangswert muss zwischen -32768 und 32767 liegen (15Bit).

Auflösung: In diesem N-Register ist die Auflösung, also die Differenz des Zählerstandes nach welchem der nächste AD-Wert geholt werden soll, abgelegt. Das Vorzeichen der Auflösung gilt für das Starten der Erfassung ob größer oder kleiner als Merkmal. Es muss ein N-Register verwendet werden. Die Auflösung muss zwischen -32768 und 32767 liegen.

AD-Nr.: Gibt die Nummer (von 1 bis 8) des AD-Wandlers an, von welchem die Werte geholt werden.

CAN-ID: Gibt die CAN-ID über welche die erfassten AD-Messwerte als TxPDO1 versendet werden.
Die Angabe kann als Konstante oder über ein N-Register erfolgen.

Anzahl: Gibt die maximale Anzahl der AD-Werte an, die erfasst werden sollen. Diese Zahl bestimmt auch die Anzahl der für die Ablage der Messwerte notwendigen Ganzzahlregister. Die Angabe kann als Konstante oder über ein N-Register erfolgen.

HINWEIS Der Befehl G183 ist nur bei der PA-CONTROL smart möglich

Die Parameter <Anfangswert> und <Auflösung> sind auf 15-Bit begrenzt (+/-326767)

Die Einstellung für das Zählereingangssignal sollten so gewählt werden, dass die maximale Eingangsfrequenz des Zähler 0 mit 200000 Inkremente pro Sekunde nicht überschritten wird.

Bei einer Baudrate von 500KBaud sollten nicht mehr als 2 AD-Werte pro Millisekunde über den CAN-Bus übertragen werden.

In dem System-N-Registern SN130 wird die Anzahl der erfassten AD-Werte abgelegt und steht für Abfragen zur Verfügung.

Aufbau des TxPDO1:

Über den CAN-Bus werden die erfassten AD-Werte und der Zählerstand bei der Erfassung in folgender Form übertragen

Byte 0	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5
Zählerstand (signed int, 32 Bit)				AD-Wert (signed short, 16-Bit)	

Beispiel 1

Mit der Achse 1 wird eine Bewegung (rotatorisch oder translatorisch) durchgeführt. Synchron zu dieser Bewegung sollen AD-Werte (ca. alle 1ms ein AD-Wert) mit hoher Wiederholgenauigkeit erfasst und gleich über den CAN-Bus versendet werden

Um dies zu realisieren wurde folgender Hardwareaufbau gewählt:

- PA-CONTROL smart mit
 - AD-Wandler 1-fach (8-fach)
 - CNT0
- LV servoTEC S2
- Verdrahtung LV-servoTEC S2 Ausgang „Encoderemulation“ X11 auf den Zähler 0 der PA-CONTROL smart

1	<i>N1:=20</i>	<i>; Starte Erfassung der AD-Werte ab dem Zählerwert 20 (größer gleich)(siehe G182-Befehl)</i>
2	<i>N2:=10</i>	<i>; hole wenn der Zählerstand um 10 größer geworden ist den nächsten AD-Wert (siehe G182-Befehl)</i>
3	<i>CNT0:=0</i>	<i>;Zähler 0 auf einen definierten Anfangswert setzen</i>
4	<i>G183.0.N1.N2.1.33.4</i>	<i>Initialisiere G183-Befehl : ; Zählernummer :0 ; Zähler Anfangswert :N1:=20 ; Zähler Auflösung :N2:=10 ; AD-Wandler-Nr :1 ; CAN-ID RxPDO1 :33 → 0x221 ; max. Anzahl :10</i>
5	<i>A1:=100</i>	<i>;Position 100 anfahren und AD-Werte während dieser Bewegung erfassen</i>
6	<i>N3:=SN130</i>	<i>Hole die Anzahl der erfassten AD-Werte</i>
7	<i>END</i>	

1.16.2 Befehle der G2xx-Gruppe

1.16.2.1 Grundsätzlich Ausführungen zu den Befehlen der G2xx-Gruppe

G2xx

Einleitung:

Für das Abarbeiten von Positionierbefehlen kennt die PA-CONTROL zwei Modi.

- Normalpositioniermodus:** Der Positioniervorgang wird gestartet und es wird gewartet, bis die Achse ihre Zielposition erreicht hat. Erst dann wird der nächste Befehl im Programm bearbeitet. Folgen mehrere Positionierbefehle in einer Zeile unmittelbar aufeinander, so werden diese parallel zueinander abgearbeitet.
- Startpositioniermodus:** Der Positioniervorgang wird gestartet um dann sofort mit dem nächsten Befehl im Programm fortzufahren. Der Positioniervorgang wird im Hintergrund gesteuert.

Die Voreinstellung ist der Normalpositioniermode.

Die Umschaltung und das Verhalten sind achsselektiv organisiert.

Befindet sich eine Achse im Startpositioniermode so kann über verschiedene Befehle aus der Gruppe G200 auf die Achse zugegriffen werden. Bevor die Achse erneut gestartet wird sollte über einen Befehl (G213, G211, G221, usw.) geprüft werden, ob die Achse noch fährt.

1.16.2.2 G210 - Aktiviere den Startpositioniermodus

G210

G210.(Achse)

Beschreibung:

Die PA-CONTROL kennt zwei Betriebsarten. Im Normalpositioniermodus, siehe Seite 235, wird nach dem Start einer Achse das Ende der Bewegung abgewartet, bevor der nächste Befehl gestartet wird. Im Startpositioniermodus kann nach dem Start einer Achse sofort die Fahrt der zweiten und weiterer Achsen ausgelöst werden. Die Achsen fahren im Hintergrund.

Der Befehl G210 aktiviert den Startpositioniermodus und deaktiviert den Normalpositioniermodus. Man könnte auch sagen, dass durch die Befehle G210 bzw. G213 die Verfahrensweise des Ablaufinterpreters bezüglich der Behandlung von Fahrbefehlen beeinflusst wird.

Bei der Programmierung muss unbedingt berücksichtigt werden, dass die Bewegung einer Achse vor einem weiteren Start beendet sein muss. Die Nichtbeachtung dieser Voraussetzung führt zu einem Fehler. Bei Anwendung des Startpositioniermodus sollte eine Kontrolle erfolgen, ob der Start der Achse zu dem Zeitpunkt erlaubt ist.

Mit dem Befehl G213 erfolgt der Übergang in den Normalpositioniermodus

Befehlsform:

G210.A0

G210.A1

Anwendung:

Die Positionierung einer Achse wird gestartet. Die Steuerung bearbeitet das Programm weiter ohne zu warten, bis die Achse in der Endposition angekommen ist.

Beispiel

siehe nächste Befehle

1.16.2.3 G211 - Positionsbedingter Sprung

G211

G211.(Achse).(Bedingung.) (Maske)

Zu beachten:

Der Befehl wertet nur aus, ob die Achse(n) in Position oder nicht in Position ist bzw. sind.

Befehlsform:

G211.A0.j Marke

G211.An.j Marke

Anwendung:

Ausführung eines Sprunges in Abhängigkeit davon, ob die ausgewählte Achse oder alle Achsen fahren oder stehen. **A** steht für die Achsennummer und **j** für die Bedingung.

Es gilt:

- A0 → alle Achsen
- A1 → Achse 1
- A2 → Achse 2
- An → Achse n

- j = 0 → springe, wenn die Position der selektierten Achsen noch nicht erreicht ist
- j = 1 → springe, wenn die Position der selektierten Achsen erreicht ist

Beispiel 1

1	G25.A1 G25.A2	;Referenzfahrten der A1- und A2-Achse
2	G90.A0	;Absolutmaßsystem
3	G210.A0	;aktiviere den Startpositionsmode für alle Achsen
4	A1:=10000 A2:=6000	;Start der Achsen
5	\$SETZE	
6	O1:=1 T50 O1:=0	;Ausgang 1 für 500ms setzen
7	T50	;500ms warten
8	G211.A0.0 SETZE	;springe nach Marke „SETZE“, wenn alle Achsen noch nicht ihre Position erreicht haben
9	G213.A0	;warte bis alle Achsen in Position sind und aktiviere den Normalpositioniermode (deaktiviere den Startpositioniermode)
10	O2:=1	
11	END	

Beispiel 2

1	G25.A1 G25.A2	;Referenzfahrten der A1- und A2-Achse
2	G90.A0	;Absolutmaßsystem
3	G210.A0	;aktiviere den Startpositioniermode für alle Achsen
4	A1:=10000 A2:=6000	;Start der Achsen
5	\$SCHLEIFE	
6	O1:=1 T50 O1:=0	;Ausgang 1 für 500ms setzen
7	T50	;500ms warten
8	G211.A2.1 FERTIG	;springe nach „FERTIG“, wenn die A2-Achse ihre Position erreicht hat
9	JMP SCHLEIFE	;springe nach „SCHLEIFE“
10	\$FERTIG	
11	G213.A0	;warte bis alle Achsen in Position sind und aktiviere den Normalpositioniermode (deaktiviere den Startpositioniermode)
12	O2:=1	
13	END	

Beispiel 3

1	G25.A1 G25.A2	;Referenzfahrten der A1- und A2-Achse
2	G90.A0	;Absolutmaßsystem
3	G210.A0	;aktiviere den Startpositioniermode für alle Achsen
4	A1:=10000 A2:=6000	;Start der Achsen
5	\$SCHLEIFE	
6	O1:=1 T50 O1:=0	;Ausgang 1 für 500ms setzen
7	T50	;500ms warten
8	G211.A1.0 SCHLEIFE	;Springe nach „SCHLEIFE“, wenn die A1-Achse die Position noch nicht erreicht hat
9	G213.A0	;warte bis alle Achsen in Position sind und aktiviere den Normalpositioniermode (deaktiviere den Startpositioniermode)
10	O2:=1	
11	END	

1.16.2.4 G212 - Positionsbedingter Unterprogrammaufruf

G212

G212.(Achse).(Bedingung) (Programm)

HINWEIS Der Befehl wertet nur aus, ob die Achse(n) in Position oder nicht in Position ist bzw. sind.

Befehlsform:

G212.A0.j Name

G212.An.j Name

Anwendung:

Ausführung eines Unterprogrammaufrufes in Abhängigkeit davon, ob die selektierte Achse oder alle Achsen fahren oder stehen. **A** steht für die Achsnamen und **j** für die Bedingung.

Es gilt:

- A0 → alle Achsen
- A1 → Achse 1
- A2 → Achse 2
- An → Achse n

- j = 0 → springe ins Unterprogramm, wenn die Position der selektierten Achsen noch nicht erreicht ist.
- j = 1 → springe ins Unterprogramm, wenn die Position der selektierten Achsen erreicht ist.

Beispiel

1	G25.A1 G25.A2	<i>;Referenzfahrten der A1- und A2-Achse</i>
2	G210.A0	<i>;aktiviere den Startpositioniermode für alle Achsen</i>
3	A1:=100 A2:=500	<i>;Start der Achsen</i>
4	\$SCHLEIFE	<i>;Sprungmarke</i>
5	G212.A0.0 BLINKEN	<i>;springe ins Unterprogramm BLINKEN, wenn alle Achsen noch nicht ihre Position erreicht haben</i>
6	G211.A1.0 SCHLEIFE	<i>;springe nach Marke „SCHLEIFE“, wenn die Achse 1 noch nicht ihre Position erreicht hat</i>
7	G213.A0	<i>;warte bis alle Achsen in Position sind und aktiviere den Normalpositioniermode (deaktiviere den Startpositioniermode)</i>
8	END	

Programm: BLINKEN.PNC

1	O9:=1	<i>;Setze Ausgang 1</i>
2	T50	<i>;Warte 500ms</i>
3	O9:=0	<i>; Setze Ausgang 1 zurück</i>
4	T20	<i>; Warte 200ms</i>
5	END	

1.16.2.5 G213 - Aktiviere den Normalpositioniermodus

G213

G213.(Achse)

Beschreibung:

Der Befehl G213 aktiviert den Normalpositioniermodus und deaktiviert den Startpositioniermodus. Man könnte auch sagen, dass durch die Befehle G210 bzw. G213 die Verfahrensweise des Ablaufinterpreters bezüglich der Behandlung von Fahrbefehlen beeinflusst wird.

Die Steuerung wartet, bis für alle Achsen (G213.A0) die laufenden Positioniervorgänge abgeschlossen sind oder für eine bestimmt angesprochene Achse (G213.Ai) der laufende Positioniervorgang beendet ist, um dann den nächsten Befehl zu starten.

Befehlsform:

G213.A0

G213.An

Beispiel

siehe andere G200-Befehle

1.16.2.6 G221 - Positionsbedingter Sprung (aktuelle Position)

G221

G221.(Bedingung).(Achse).(Position) (Marke)

Befehlsform:

G221.j.An.Pos Marke

G221.j.ARn.Pos Marke

Anwendung:

Durch Anwendung des Befehles **G221** kann ein bedingter Sprung, in Abhängigkeit einer aktuellen Achsposition, realisiert werden. **j** steht für Sprungbedingung, **An** für die Achse und die Position und **Marke** für das Ziel, an das gesprungen werden soll.

Es gilt:

j = 0	→	springe, wenn die aktuelle Achsposition kleiner als die Position (An) ist
j = 1	→	springe, wenn die aktuelle Achsposition größer als die Position (An) ist

HINWEIS Dieser Befehl wertet die Absolutposition der angegebenen Achse, unabhängig vom gewählten Maßsystem (G90.An / G91.An) aus.

Beispiel 1

1	G25.A1 G25.A2	;Referenzfahrten der A1- und A2-Achse
2	G90.A0	;Absolutmaßsystem
3	G210.A0	;aktiviere den Startpositioniermode für alle Achsen
4	A1:=10000 A2:=6000	;Start der Achsen
5	\$SCHLEIFE	
6	O1:=1 T50 O1:=0 T50	
7	G221.1.A1.8000 ROT	;springe nach „ROT“, wenn die A1-Position größer 8000 ist
8	O2:=1 T10 O2:=0	
9	\$ROT	
10	G211.A0.0 SCHLEIFE	;springe nach „SCHLEIFE“, wenn noch nicht alle Achsen in Position sind
11	G213.A0	;warte bis alle Achsen in Position sind und aktiviere den Normalpositioniermode (deaktiviere den Startpositioniermode)
12	END	

Beispiel 2

1	G25.A1 G25.A2	;Referenzfahrten der A1- und A2-Achse
2	G90.A0	;Absolutmaßsystem
3	G210.A0	;aktiviere den Startpositioniermode für alle Achsen
4	A1:=10000 A2:=6000	;Start der Achsen
5	\$SCHLEIFE	
6	O1:=1 T50 O1:=0 T50	
7	G221.1.A2.2000 ROT	;springe nach „ROT“, wenn die A2-Position größer 2000 ist
8	O2:=1 T10 O2:=0	
9	\$ROT	
10	G211.A0.0 SCHLEIFE	;springe nach „SCHLEIFE“, wenn noch nicht alle Achsen in Position sind
11	G213.A0	;warte bis alle Achsen in Position sind und aktiviere den Normalpositioniermode (deaktiviere den Startpositioniermode)
12	END	

Beispiel 3

1	G25.A1 G25.A2	;Referenzfahrten der A1- und A2-Achse
2	G90.A0	;Absolutmaßsystem
3	G210.A0	;aktiviere den Startpositioniermode für alle Achsen
4	A1:=10000 A2:=6000	;Start der Achsen
5	\$SCHLEIFE	
6	O1:=1 T50 O1:=0 T50	
7	G221.0.A1.8000 ROT	;springe nach „ROT“, wenn die A1-Position kleiner 8000 ist
8	O2:=1 T10 O2:=0	
9	\$ROT	
10	G211.A0.0 SCHLEIFE	;springe nach „SCHLEIFE“, wenn noch nicht alle Achsen in Position sind
11	G213.A0	;warte bis alle Achsen in Position sind und aktiviere den Normalpositioniermode (deaktiviere den Startpositioniermode)
12	END	

1.16.2.7 G222 - Positionsbedingter Unterprogramm-Aufruf (akt. Pos.)

G222

G222.(Bedingung) (Achse).(Position) (Programm)

HINWEIS Dieser Befehl wertet die Absolutposition der angegebenen Achse unabhängig vom gewählten Maßsystem (G90.An / G91.An) aus.

Befehlsform:

G222.j.An.Pos Name

Anwendung:

Durch Anwendung des Befehles **G222** kann ein bedingter Unterprogrammaufruf in Abhängigkeit einer aktuellen Achsposition realisiert werden.

j steht für Sprungbedingung, **An** für die Position der Achse und **Name** für das Unterprogramm, das aufgerufen werden soll.

Es gilt:

j = 0	→	springe, wenn die aktuelle Achsposition kleiner als die Position (An) ist
j = 1	→	springe, wenn die aktuelle Achsposition größer als die Position (An) ist

Beispiel 1

1	G25.A1 G25.A2	;Referenzfahrten der A1- und A2-Achse
2	G90.A0	;Absolutmaßsystem
3	G210.A0	;aktiviert den Startpositioniermode für alle Achsen
4	A1:=10000 A2:=6000	;Start der Achsen
5	\$SCHLEIFE	
6	O1:=1 T50 O1:=0 T50	
7	G222.1.A1.8000 ROT	;Aufruf des Unterprogramms „ROT“, wenn die A1-Position größer 8000 ist
8	G211.A0.0 SCHLEIFE	;springe nach „SCHLEIFE“, wenn noch nicht alle Achsen in Position sind
9	G213.A0	;warte bis alle Achsen in Position sind und aktiviere den Normalpositioniermode (deaktiviere den Startpositioniermode)
10	END	

Programm: ROT

```
1 O2:=1 T10 O2:=0
2 END
```

Beispiel 2

1	G25.A1 G25.A2	<i>;Referenzfahrten der A1- und A2-Achse</i>
2	G90.A0	<i>;Absolutmaßsystem</i>
3	G210.A0	<i>;aktiviert den Startpositioniermode für alle Achsen</i>
4	A1:=10000 A2:=6000	<i>;Start der Achsen</i>
5	\$SCHLEIFE	
6	O1:=1 T50 O1:=0 T50	
7	G222.1.A2.1000 ROT	<i>;Aufruf des Unterprogramms „ROT“, wenn die A2:=Position größer 1000 ist</i>
8	G211.A0.0 SCHLEIFE	<i>;springe nach „SCHLEIFE“, wenn noch nicht alle Achsen in Position sind</i>
9	G213.A0	<i>;warte bis alle Achsen in Position sind und aktiviere den Normalpositioniermode (deaktiviere den Startpositioniermode)</i>
10	END	

Programm: ROT

1	O2:=1 T10 O2:=0
2	END

Beispiel 3

1	G25.A1 G25.A2	<i>;Referenzfahrten der A1- und A2-Achse</i>
2	G90.A0	<i>;Absolutmaßsystem</i>
3	G210.A0	<i>;aktiviert den Startpositioniermode für alle Achsen</i>
4	A1:=10000 A2:=6000	<i>;Start der Achsen</i>
5	\$SCHLEIFE	
6	O1:=1 T50 O1:=0 T50	
7	G222.0.A1.4000 ROT	<i>;Aufruf des Unterprogramms „ROT“, wenn die A1-Position kleiner 4000 ist</i>
8	G211.A0.0 SCHLEIFE	<i>;Springe nach „SCHLEIFE“, wenn noch nicht alle Achsen in Position sind</i>
9	G213.A0	<i>;warte bis alle Achsen in Position sind und aktiviere den Normalpositioniermode (deaktiviere den Startpositioniermode)</i>
10	END	

Programm: ROT

1	O2:=1 T10 O2:=0
2	END

1.16.2.8 G230 - Warten bis aktuelle Position </> als Wert

G230

G230.(Bedingung).(Achse).(Vergleichs-Position)

HINWEIS Dieser Befehl wertet die Absolutposition der angegebenen Achse unabhängig vom gewählten Maßsystem (G90.An / G91.An) aus.

Befehlsform:

G230.j.An.Konstante

G230.j.An.Rn

G230.j.An.R!n

Anwendung:

Durch Anwendung des Befehls **G230** kann erreicht werden, dass bis zu einer bestimmten Position die weitere Abarbeitung des Programms ausgesetzt wird.

j steht für die Bedingung und **An** für die Achse und die Position.

Es gilt:

j = 0	→	warte bis die aktuelle Position < angegebene Vergleichs-Position
j = 1	→	warte bis die aktuelle Position > angegebene Vergleichs-Position

Beispiel

1	G25.A1	<i>;Referenzfahrten der A1-Achse</i>
2	G90.A0	<i>;Absolutmaßsystem</i>
3	R10:=300	
4	G210.A0	<i>;aktiviere den Startpositioniermode für alle Achsen</i>
5	A1:=1000	<i>;Start der Achse</i>
6	O1:=1 T10 O1:=0	
7	G230.1.A1.300	<i>;Ausgang 2 ist von A1:=300 bis A1:=500 gesetzt</i>
8	O2:=1	
9	G230.1.A1.500	
10	O2:=0	
11	G213.A0	
12	G210.A0	<i>;aktiviere den Startpositioniermode für alle Achsen</i>
13	A1:=0	
14	O1:=1	
15	G230.0.A1.R10	<i>;Ausgang 1 ist von A1:=1000 bis A1:=300 gesetzt (in R10 steht der Wert 300)</i>
16	O1:=0	
17	G213.A0	<i>;warte bis alle Achsen in Position sind und aktiviere den Normalpositioniermode (deaktiviere den Startpositioniermode)</i>
18	END	

1.16.2.9 G231 - Warte bis der Restweg der akt. Positionierung erreicht oder unterschritten ist

G231

ab V5.07

G231.(Achse).(Restweg)

Befehlsform:

G231.An.Ri

G231.An.R!i

G231.An.Konstante

Beschreibung:

Es wird mit der Abarbeitung des Ablaufes bei diesem Befehl gewartet, bis der noch zu verfahren Restweg der Achse (laufend Positionierung) kleiner als der Vergleichsrestweg ist

HINWEIS Dieser Befehl wertet die Absolutposition unabhängig vom gewählten Maßsystem (G90 oder G91) aus.

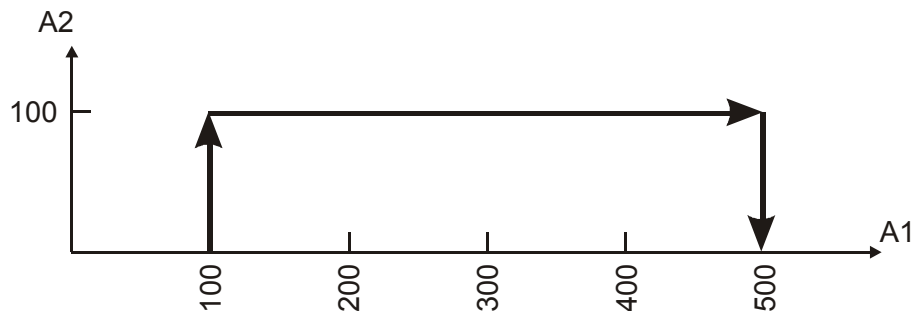
Anwendung:

Durch Anwendung des Befehls **G231** kann erreicht werden, dass die Abarbeitung des weiteren Programms während der Ausführung eines Restweges erfolgt.

HINWEIS Hat die Achse ihre Zielposition schon erreicht, steht also in Position, oder ist keine Positionierung aktiv (noch nicht aktiv), so wird nicht gewartet. Der Restweg wird als Betrag, also ohne Vorzeichen angegeben. Negative Vergleichswerte haben keinen Sinn. Die Steuerung hält in einem solchen Fall an (E532 - Wert außerhalb Bereich).

Beispiele

ohne G231

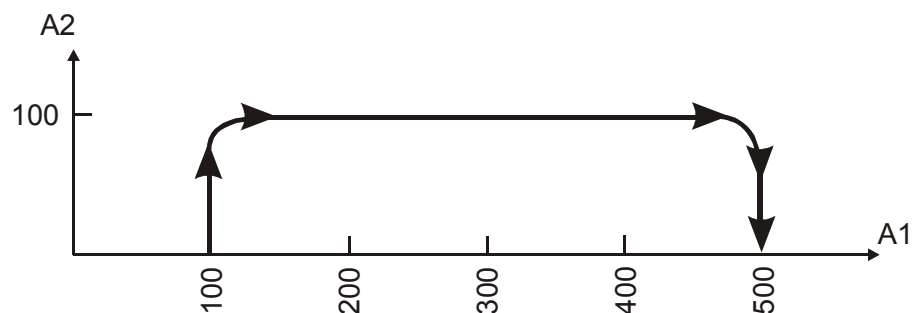


```

1   G25.A1 G25.A2           ;Referenzfahrten der A1 / A2-Achse
2   G90.A0                  ;Absolutmaßsystem
3   A2:=100                 ;Fahre Achse 2 auf absolute Position 100
4   A1:=500                 ;Fahre Achse 1 auf absolute Position 500
5   A2:=0                   ;Fahre Achse 2 auf absolute Position 0
6   END

```

mit G231



```

1   G25.A1 G25.A2           ;Referenzfahrten der A1/ A2-Achse
2   G90.A0                  ;Absolutmaßsystem
3   G210.A0                 ;aktiviere den Startpositioniermode für alle Achsen,
4   A2:=100                 ;Start Achse 2 und fahre auf absolute Position
                           100mm
5   G231.A2.20              ;wartet bis der Restweg der Achse 2 kleiner als
                           20 mm ist
6   A1:=500                 ;Starte Achse 1, während Achse 2 noch fährt
7   G231.A1.20              ;Überprüfe Restweg der Achse 1 bis er kleiner
                           als 20 mm ist
8   A2:=0                   ;Starte Achse 2, wenn der Restweg der Achse 1
                           unter 20 mm ist
9   G213.A0                 ;warte bis alle Achsen in Position sind und akti-
                           viere den Normalpositioniermode (deaktiviere
                           den Startpositioniermode)
10  END

```

1.16.3 Befehle der G4xx-Gruppe

G4xx

1.16.3.1 Einleitung, Zeitüberwachungsbefehle der PA-CONTROL-Familie

Die Benutzung der nachstehend beschriebenen Befehle bietet sich dann an, wenn Vorgänge bezüglich ihres definierten Endes kontrolliert werden müssen und bei Unregelmäßigkeiten Hilfsprogramme zur Fehlerbeseitigung oder Fehlerregistrierung gestartet werden müssen. Diese Situationen können immer dann auftreten, wenn durch die Steuerungen Abläufe, z.B. Ansteuerungen von Pneumatikzylindern gestartet werden, die parallel und unabhängig zu Programmen der PA-CONTROL ablaufen.

Nach abgelaufener Zeit kann je nach Befehl ein bedingter Sprung oder ein Unterprogrammaufruf erfolgen. Gesetzte Zeitbedingungen können zurückgesetzt oder abgebrochen werden. Vor jedem Befehl, den die PA-CONTROL ausführt, wird die Zeitbedingung überprüft und gegebenenfalls reagiert.

1.16.3.2 G421 - Zeitüberwachung mit bedingtem Sprung

G421.1.

G421.1.(Zeit) (Marke)

Beschreibung:

Die Überwachungszeit wird mit dem Befehl **G421.1.200 Marke** gestartet. Wird vor Ablauf der Überwachungszeit der Rücksetzbefehl G401.1 erkannt, dann wird die Überwachung zurückgesetzt. Im anderen Fall erfolgt die Fortsetzung des Programms an der spezifizierten Marke.

Ein im Programm laufender Befehl **G421** wird durch einen weiteren **G421** oder **G422** abgebrochen und gleichzeitig der neue Zeitüberwachungsbefehl gestartet. Die Restzeit des ersten Überwachungsauftrags wird nicht weiter bearbeitet.

HINWEIS Nach Start einer Überwachungszeit sollte kein Unterprogrammaufruf erfolgen, da dann das Sprungziel für den G421-Befehl nicht mehr erreichbar ist.

Befehlsform:

G421.1.nnn Marke

G421.1.Ni Marke

HINWEIS Die maximale Überwachungszeit beträgt:
 bei der Festlegung über Konstante 65535 x 10ms
 bei der Festlegung über N-Register 2^{31} x 10ms

Beispiel

```

1   $ANFANG
2   O1:=1                               ;setze Ausgang 1
3   G421.1.200 FEHLER_1                 ;springe zur Marke „FEHLER_1“, wenn 2 Sekunden
                                        abgelaufen sind und die Zeitbedingung nicht
                                        zurückgesetzt ist
4   I1.1 O1:=0                           ;warte auf Eingang 1 = „1“ und setze Ausgang 1
                                        zurück
5   G401.1                               ;setze die Zeitbedingung zurück
6   END
7   $FEHLER_1
8   O2:=1                               ;Störungssignal setzen
9   I2.1                                 ;warte auf Fehlerquittungssignal
10  O1:=0 O2:=0                          ;Ausgang 1 und Störungssignal zurücksetzen
11  JMP ANFANG

```

1.16.3.3 G422 - Zeitüberwachung mit bedingtem Unterprogrammaufruf

G422.1.

G422.1.(Zeit) (File)

Beschreibung:

Die Überwachungszeit wird mit dem Befehl **G422.1.200 Name** gestartet. Wird vor Ablauf der 2 Sekunden Überwachungszeit (200*10ms) der Rücksetzbefehl G401.1 erkannt, dann wird die Überwachung zurückgesetzt. Im anderen Fall erfolgt der Sprung in das spezifizierte Unterprogramm.

HINWEIS Der Rücksprung aus dem Zeitüberwachungsunterprogramm (im Beispiel FEHLER) erfolgt auf den PAC-Befehl, der durch die Zeitüberwachung unterbrochen wurde.

Befehlsform:

G422.1.nnn Name

G422.1.Ni Marke

HINWEIS Die maximale Überwachungszeit beträgt:
 bei der Festlegung über Konstante 65535 x 10ms
 bei der Festlegung über N-Register 2^{31} x 10ms

Beispiel

1	O1:=1	;setze Ausgang 1
2	G422.1.200 FEHLER	;Unterprogrammaufruf, wenn 2 Sekunden abgelaufen sind und die Zeitbedingung nicht zurück gesetzt ist.
3	I1.1 O1:=0	;warte auf Eingang 1 = „1“ und setze Ausgang 1 zurück
4	G401.1	
5	END	

Programm: FEHLER

1	O3:=1	;Störung an ...
2	I5.1	;Quittung durch den Bediener
3	I5.0	;
4	END	

1.16.3.4 G423 - mit bedingtem Unterpr.aufruf (zurück in gleicher Zeile)

G423.1.

G423.1.(Zeit) (File)

Beschreibung:

Die Überwachungszeit wird mit dem Befehl **G423.1.200 Name** gestartet. Wird vor Ablauf der 2 Sekunden Überwachungszeit (200*10ms) der Rücksetzbefehl G401.1 erkannt, dann wird die Überwachung zurückgesetzt. Im anderen Fall erfolgt der Sprung in das spezifizierte Unterprogramm.

HINWEIS Bei Rückkehr aus dem Unterprogramm wird das Programm in der Zeile wieder aufgenommen, in welcher der G423-Befehl steht. Das Programm in dem der G423-Befehl aktiviert wurde, muss noch aktuell aktiv sein. Es darf also nicht mit dem Befehl CANCEL an anderer Stelle beendet werden.

Befehlsform:

G423.1.nnn Name

G423.1.Ni Marke

HINWEIS Die maximale Überwachungszeit beträgt:
 bei der Festlegung über Konstante 65535 x 10ms
 bei der Festlegung über N-Register 2^{31} x 10ms

Beispiel

<p>1 O1:=1</p> <p>2 G423.1.200 FEHLER</p> <p>3 I1.1 O1:=0</p> <p>4 G401.1</p> <p>5 END</p> <p>Programm: FEHLER</p> <p>1 O3:=1</p> <p>2 I5.1</p> <p>3 END</p>	<p>;setze Ausgang 1</p> <p><i>;Unterprogrammaufruf, wenn 2 Sekunden abgelaufen sind und die Zeitbedingung nicht zurück gesetzt ist.</i></p> <p><i>Bei Rückkehr aus dem Unterprogramm wird das Programm in der Zeile wieder aufgenommen, in welcher der G423-Befehl steht. Die Zeitüberwachung wird noch einmal durchlaufen und damit sichergestellt, dass der Eingang 1 wirklich den logischen Zustand 1 hat. Das Programm in dem der G423-Befehl aktiviert wurde muss noch aktuell aktiv sein</i></p> <p><i>;warte auf Eingang 1 = „1“ und setze Ausgang 1 zurück</i></p> <p>;Störung an ...</p> <p><i>;Quittung durch den Bediener</i></p>
--	---

1.16.3.5 G401 - Zurücksetzen der Zeitbedingung

G401.1.

Beschreibung:

Durch diesen Befehl wird die Zeitüberwachung, die durch die Befehle **G421**, **G422** oder **G423** gestartet wurden, gestoppt. Damit ist die Zeitüberwachung nicht mehr aktiv.

Befehlsform:

G401.1

Beispiel

siehe Beispiele der Befehle **G421** oder **G422** oder **G423**

1.16.4 Befehle der G5xx-Gruppe

G5xx

1.16.4.1 Einleitung, Datenkanäle der PA-CONTROL

Mit dieser Befehlsgruppe können während des Programmablaufs (Automatikbetrieb) Texte, Registerinhalte, Eingangs-, Ausgangs- und Merkerzustände über den aktuellen Datenkanal ausgegeben werden.

Folgende Datenkanäle sind möglich:

- Datenkanal 0 : LC-Display auf der IEF-Bedienkonsole der PA-CONTROL
- Datenkanal 1 : 1. serielle Schnittstelle der PA-CONTROL (COM 1)
- Datenkanal 2 : 2. serielle Schnittstelle der PA-CONTROL (COM 2)
- Datenkanal 3 : 3. serielle Schnittstelle der PA-CONTROL (COM 3)
- Datenkanal 4 : 4. serielle Schnittstelle der PA-CONTROL (COM 4)

HINWEIS

Die Information über den aktuellen Datenkanal ist jeder Task (jedem Parallelablauf) zugeordnet und muss vor der Benutzung der Befehle initialisiert werden.

Wurde eine serielle Schnittstelle bereits Initialisiert und die Kommunikation ist schon aktiv, dann kann nur durch Angabe der Datenkanalnummer (z.B. G500.1) umgeschaltet werden. Der Empfangs- und Sendepuffer bleiben dann unverändert.

1.16.4.2 G500 - Wahl des Datenkanals / Initialisierung der Schnittstellen

G500.

G500.(Kanalnummer).(Baudrate).(Datenformat).(Handshake)

Beschreibung:

Mit dem Befehl „G500.“ wird der aktuelle Datenkanal angewählt. Außerdem kann bei den seriellen Schnittstellen die Initialisierung (Baudrate, Anzahl Datenbits, Parity,..) erfolgen.

Wird die Anwahl einer seriellen Schnittstelle mit Initialisierung (z.B. G500.1.6.1.0) durchgeführt, so wird auch der Empfangspuffer der seriellen Schnittstelle gelöscht. Wird dagegen die Anwahl der seriellen Schnittstelle ohne Initialisierung (z.B. G500.1) durchgeführt, bleibt der Empfangspuffer der seriellen Schnittstelle unverändert.

Bedeutung der Übergabeparameter siehe Auflistung.

HINWEIS Ausgaben auf das LC-Display der PA-CONTROL sind nur ohne CR & LF sinnvoll.

Wird nach einem G533.n-Befehl der G500-Befehl ausgeführt, so wird das Endekriterium gelöscht!

Befehlsform:

G500.Nr.Baudrate.Datenformat.Handshake

G500.Nr

G500.0

Beispiel

siehe Beispiele der folgenden Befehle

Bedeutung der Übergabeparameter :

Nr.:		
0	→	LC-Display auf der PAC Frontplatte wird aktiv
1	→	Schnittstelle 1 wird aktiv (COM 1)
2	→	Schnittstelle 2 wird aktiv (COM 2)
3	→	...

Baudrate:		
1	→	110 Baud
2	→	300 Baud
3	→	1200 Baud
4	→	2400 Baud
5	→	4800 Baud
6	→	9600 Baud
7	→	19200 Baud
8	→	38400 Baud (zur Zeit noch nicht verfügbar)

Datenformat:		
1	→	8 Databits, 1 Stoppbit, no Parity
2	→	7 Databits, 1 Stoppbit, no Parity
3	→	7 Databits, 1 Stoppbit, even Parity
4	→	7 Databits, 1 Stoppbit, odd Parity
5	→	8 Databits, 1 Stoppbit, even Parity (ab V4.49)
6	→	8 Databits, 1 Stoppbit, odd Parity (ab V4.49)
7		8 Databits, 2 Stoppbit, no Parity (ab V4.55 K)
8		7 Databits, 2 Stoppbit, no Parity (ab V4.55 K)
9		7 Databits, 2 Stoppbit, even Parity (ab V4.55 K)
10		7 Databits, 2 Stoppbit, odd Parity (ab V4.55 K)
11		8 Databits, 2 Stoppbit, even Parity (ab V4.55 K)
12		8 Databits, 2 Stoppbit, odd Parity (ab V4.55 K)

Handshake:		
0	→	Hardwarehandshake (CTS) inaktiv
1	→	Hardwarehandshake (CTS) aktiv

1.16.4.3 G501 - Lösche die Anzeige

G501

Beschreibung:

Der Befehl G501 löscht die Anzeige des aktuellen Anzeigemediums und stellt den Cursor in die linke obere Ecke (1.Spalte, 1. Zeile).

HINWEIS Bevor der Befehl G501 benutzt wird, sollte mit dem Befehl „G500.“ die Umschaltung auf das aktuelle Anzeigemedium durchgeführt sein.

Dieser Befehl ist nur für die Ausgabe auf dem Display sinnvoll!

Befehlsform:

G501

Beispiel

1	<i>G11.0</i>	<i>;Ablaufanzeigen ausschalten</i>
2	<i>G500.0</i>	<i>;schaltet die G500-Befehle auf das aktuelle Anzeigemedium um (LC-Anzeige oder bei „Simulation der Frontplatte“ die serielle Schnittstelle)</i>
3	<i>G501</i>	<i>;löscht die Anzeige und stellt den Cursor in die linke obere Ecke (1. Spalte, 1. Zeile)</i>
4	<i>G510.Hallo</i>	<i>;gibt den Text „Hallo“ auf dem aktuellen Anzeigemedium aus</i>
5	<i>END</i>	

1.16.4.4 G502 - Lösche bis zum Zeilenende

G502

Beschreibung:

Der Befehl G502 löscht die Anzeige auf dem Anzeigemedium von der aktuellen Cursorposition an bis zum Zeilenende. Die Position des Cursors bleibt unverändert erhalten.

Der Befehl kann benutzt werden, um bei der Ausgabe von Registerwerten den vorher angezeigten Wert zu löschen, um anschließend den neuen Registerwert auszugeben.

HINWEIS Bevor der Befehl G502 benutzt wird, sollte mit dem Befehl „G500.“ die Umschaltung auf das aktuelle Anzeigemedium durchgeführt sein.

Dieser Befehl ist nur für die Ausgabe auf dem Display sinnvoll!

Befehlsform:

G502

Beispiel

1	G11.0	;Ablaufanzeigen ausschalten
2	G500.0	;schaltet die G500-Befehle auf das aktuelle Anzeigemedium um (LC-Anzeige oder Bei „Simulation der Frontplatte“ die serielle Schnittstelle)
3	G501	;löscht die Anzeige und stellt den Cursor in die linke obere Ecke (1. Spalte, 1. Zeile)
4	N1:=234	
5	\$SCHLEIFE	
6	G503.1.2	;positioniere den Cursor in die 1. Spalte der 2. Zeile
7	G510.WERT N1	
8	G503.11.2	;positioniere Cursor in die 11. Spalte der 2. Zeile
9	G502	;lösche ab der Cursorposition bis zum Zeilenende
10	G520.N1	;gib den Wert von N1 auf der Anzeige aus (linksbündig)
11	T100	
12	DEC.N1 SCHLEIFE	
13	END	

1.16.4.5 G503/G504 - Positioniere den Cursor

G503 / 504

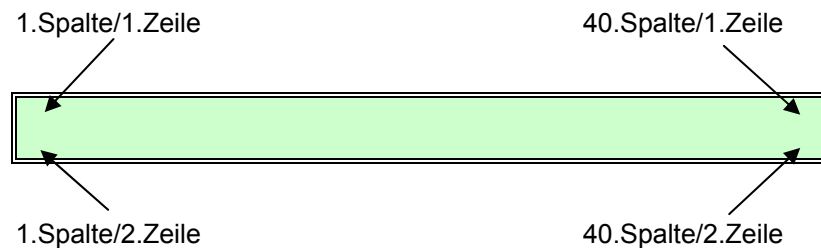
G50X.(Spalte).(Zeile)

Beschreibung:

Der Befehl G503 dient zum Positionieren des Cursors auf der Anzeige, damit die Ausgabe von Zeichen an der richtigen Stelle auf dem Display geschieht.

Der Befehl G504 dient ebenfalls zum Positionieren des Cursors auf der Anzeige. Der Unterschied zum Befehl G503 besteht darin, dass der blinkende Cursor an der vorgesehenen Stelle des Displays zum Zwecke einer Tastatureingabe blinkend fixiert wird. Mit der Eingabe des ersten Zeichens oder durch den Befehl G504 mit den Parametern 0 (G504.0.0) wird der Cursor wieder deaktiviert.

Das LC-Display ist in 40 Spalten und 2 Zeilen aufgeteilt.



HINWEIS Bevor der Befehl G503 benutzt wird, sollte mit dem Befehl „G500.“ die Umschaltung auf das aktuelle Anzeigemedium durchgeführt sein.

Dieser Befehl ist nur für die Ausgabe auf dem Display sinnvoll!

Befehlsform:

G503.n.m

G503.Nn.m

G503.n.Nm

G503.Nn.Nm

G504.n.m

G504.Nn.m

G504.n.Nm

G504.Nn.Nm

Beispiel

1	G11.0	;Ablaufanzeigen ausschalten
2	G500.0	;schaltet die G500-Befehle auf das aktuelle Anzeigemedium um (LC-Anzeige oder bei „Simulation der Frontplatte“ die serielle Schnittstelle)
3	N23:=5	
4	N35:=2	
5	G501	;löscht die Anzeige und stellt den Cursor in die linke obere Ecke (1. Spalte, 1. Zeile)
6	G503.4.1	;stellt den Cursor in die 4. Spalte der 1. Zeile
7	G510.HALLO	;gibt den Text „HALLO“ an der aktuellen Cursor-Position auf dem aktuellen Anzeigemedium aus
8	G503.N23.N35	;stellt den Cursor in die 5. Spalte der 2. Zeile, siehe Werte von N23 und N35
9	G504.N23.N35	
10	G510.BEDIENER	;gibt den Text „BEDIENER“ an der aktuellen Cursorposition auf dem aktuellen Anzeigemedium aus
11	END	

1.16.4.6 G510 - Textausgabe

G510

G510.(Text)

Beschreibung:

G510. gibt den Text über die angewählte serielle Schnittstelle oder auf das Display aus.

HINWEIS

Am Textende wird kein CR und kein LF gesendet. Nach G510 darf kein weiterer Befehl in dieser Zeile stehen.

Der Text beginnt nach „G510.“ und endet mit dem Zeilenende (Eingabe beenden mit ENTER im Programmeditor).

Befehlsform:

G510.Text

Beispiel

1	G500.0	;wählt das LC-Display der PA-CONTROL aus
2	G501	;löscht die Anzeige und stellt den Cursor in die 1. Spalte, 1. Zeile
3	G510.Maschinenfehler!	;gibt den Text „Maschinenfehler!“ auf dem aktuellen Anzeigemedium, dem LC-Display, aus.
3	END	

1.16.4.7 G511 - Textausgabe mit CR und LF

G511.

G511.(Text)

Beschreibung:

G511. gibt den Text über die angewählte serielle Schnittstelle aus und sendet nach dem Text ein CR und LF und ist deshalb für die Ausgabe auf dem Display der PA-CONTROL nicht sinnvoll.

HINWEIS Die Übertragung wird mit CR und LF beendet. Nach G511 darf kein weiterer Befehl in dieser Zeile stehen.

Der Text beginnt nach „G511.“ und endet mit dem Zeilenende (Eingabe beenden mit ENTER im Programmeditor).

Befehlsform:

G511.Text

Beispiel

1	<i>G500.1.6.1.0</i>	<i>;Initialisierung der aktuellen Schnittstelle (COM1)</i>
2	<i>G511.Maschinenfehler !</i>	<i>;gibt den Text „Maschinenfehler !“ über die serielle Schnittstelle COM1 aus und schließt die Übertragung mit CR und LF ab</i>
3	<i>END</i>	

1.16.4.8 G512 - Steuerzeichenausgabe

G512.

G512.(0-255)

Beschreibung:

G512. interpretiert n als n-tes Zeichen (Ordinalzahl) der ASCII-Tabelle und gibt das entsprechende Zeichen über die serielle Schnittstelle aus. Die Auswahl des entsprechenden Zeichens kann außerdem durch den Inhalt eines Ganzzahlregisters oder durch indirekte Adressierung eines Ganzzahlregisters erfolgen.

Es sind alle Zahlen zwischen dezimal 0 und 255 zulässig.

HINWEIS Nach dem Zeichen wird kein CR (Wagenrücklauf) und kein LF (Zeilenvorschub) gesendet.

Befehlsform:

G512.n

G512.Ni

G512.N!i

Anwendung:

Ausgabe von Steuerzeichen an einen Drucker. Die Bedeutung der einzelnen Zeichen entnehmen Sie bitte der ASCII-Tabelle Ihres Druckers.

Beispiel

1	G500.2.6.3.0	;Initialisierung der aktuellen Schnittstelle (COM2) 7 Databits, 1Stoppbit, evenParity, CTS inaktiv
2	G512.02	;gibt „STX“ über die serielle Schnittstelle aus
3	G512.07	;gibt „BEL“ über die serielle Schnittstelle aus
4	G512.52	4
5	G512.03	;gibt „ETX“ über die serielle Schnittstelle aus
6	G512.49	1
7	END	

1.16.4.9 G515 - Textausgabe aus einem PTX-File

G515.

G515.(Zeile).(PTX-File)

Beschreibung:

G515. gibt einen Text, den Inhalt der Zeile *i* einer Programmdatei (*.PTX) auf dem aktuellen Datenkanal aus. Dieser Befehl ist gut geeignet, die an einer Stelle abgelegten Meldungen oder Bedienerhinweise in unterschiedlichen Programmteilen zu verwenden.

HINWEIS

Die Datei muss vom Typ *.PTX sein und zur Laufzeit des Programms vorhanden sein. Wird die entsprechende Textzeile nicht gefunden (Zahl ist größer als die Anzahl der Zeilen in der Datei), so wird die letzte Programmzeile angegeben.

Befehlsform:

G515.i.Name

G515.Ni.Name

Beispiel

Programm : MELDUNG.PTX

```
1 Maschine bereit
2 Stoerung Teilezuführung
3 Teilemagazin leer
4 Text nicht definiert
5 END
```

Programm : BEISPIEL.PNC

```
1 G500.0 ;Initialisierung, LC-Display ist aktueller Datenkanal
2 G501 ;Löschen der Anzeige und Positionierung des Cursors
auf die 1.Stelle der 1.Zeile
3 G515.2.MELDUNG ;gibt die 2. Zeile des Programms „MELDUNG.PTX“
auf dem aktuellen Datenkanal aus
4 N5:=3
5 G503.1.2 ;Positioniere den Cursor auf die 1 Stelle der 2. Zeile
6 G515.N5.MELDUNG ;gibt die 3. Zeile des Programms „MELDUNG.PTX“
auf dem aktuellen Datenkanal aus
6 END
```

1.16.4.10 G520 - Wertausgabe eines Operanden

G520

G520.(Operand).(Feldbreite).(Nachkommastellen)

Beschreibung:

G520. ermittelt den Zustand des Operanden (Register, Eingang, Ausgang, Merker) und gibt den entsprechenden Wert über die aktuelle serielle Schnittstelle, oder auf das Display aus.

HINWEIS Die Ausgabe kann unformatiert oder formatiert erfolgen.

Befehlsform unformatiert:

G520.Rn

G520.Nn

G520.In

G520.On

G520.Mn

G520.Sn

Befehlsform formatiert (mit vorlaufenden Leerzeichen):

G520.Nn.m

G520.Rn.m.i

G520.Rn.m.0

Grenzwerte für die Formatierung

Formatiert : (G520.Nn.m)

Feldbreite max. = 20

Formatiert: (G520.Rn.m.i)

Feldbreite

Nachkommastellen min. = 0 max. = 6

max. = 20

HINWEIS Bei der Ausgabe wird kein CR und kein LF gesendet.

Ausgabewerte für Eingänge, Ausgänge und Merker

Operand	Zustand	Ausgabe (ASCII-Zeichen)
Eingang	bestromt	1
Eingang	unbestromt	0
Ausgang	gesetzt	1
Ausgang	zurückgesetzt	0
Merker	gesetzt	1
Merker	zurückgesetzt	0

Beispiel

1	<i>O10:=1</i>	Setze Ausgang 10
2	<i>O20:=0</i>	Setze Ausgang 20 zurück
3	<i>M25:=1</i>	Setze Merker 25
4	<i>M35:=0</i>	Setze Merker 35 zurück
5	<i>R2:=546.067</i>	Lade die Zahl 546,06 in das Realzahlregister R2
6	<i>N3:= 34</i>	Lade die Zahl 34 in das Ganzzahlregister N3
7	<i>G500.2.6.1.0</i>	;Initialisierung der aktuellen Schnittstelle (COM2)
8	<i>G520.I7</i>	<i>;gibt den Zustand des Eingang 7 über den aktuellen Datenkanal (0 oder 1, zustandsabhängig) aus</i>
9	<i>G520.O10</i>	<i>;gibt den Zustand des Ausgang 10 über den aktuellen Datenkanal (Ausgabe 1) aus</i>
10	<i>G520.O20</i>	<i>;gibt den Zustand des Ausgang 20 über den aktuellen Datenkanal (Ausgabe 0) aus</i>
11	<i>G520.M25</i>	<i>;gibt den Zustand des Merkers 25 über den aktuellen Datenkanal (Ausgabe 1) aus</i>
12	<i>G520.M35</i>	<i>;gibt den Zustand des Merkers 35 über den aktuellen Datenkanal (Ausgabe 0) aus</i>
13	<i>G520.N3.10</i>	<i>;gibt den Inhalt von N3 in rechtsbündig in formatierter Ausgabe (10 Stellen mit 8 vorlaufenden Leerzeichen) über den aktuellen Datenkanal aus</i>
14	<i>G520.R2</i>	<i>;gibt den Inhalt von R2 linksbündig unformatiert über den aktuellen Datenkanal aus</i>
15	<i>G520.R2.10.2</i>	<i>;gibt den Inhalt von R2 rechtsbündig und formatiert über den aktuellen Datenkanal (10 Zeichen breites Feld mit 2 Stellen nach dem Komma) aus</i>
16	<i>G520.R2.10.0</i>	<i>;gibt den Inhalt von R2 rechtsbündig und formatiert über den aktuellen Datenkanal (10 Zeichen breites Feld ohne Nachkommastellen) aus</i>
17	<i>END</i>	

1.16.4.11 G521 - Wertausgabe eines Operanden mit CR und LF

G521.

G521.(Operand).(Feldbreite).(Nachkommastellen)

Beschreibung:

G521. ermittelt den Zustand des Operanden (Register, Eingang, Ausgang, Merker) und gibt den entsprechenden Wert über die aktuelle serielle Schnittstelle, oder auf das Display aus.

HINWEIS Die Ausgabe kann unformatiert oder formatiert erfolgen.

Befehlsform unformatiert:

G521.Rn

G521.Nn

G521.In

G521.On

G521.Mn

G521.Sn

Befehlsform formatiert (mit vorlaufenden Leerzeichen):

G521.Nn.m

G521.Rn.m.i

G521.Rn.m.0

Grenzwerte für die Formatierung

Formatiert : (G521.Nn.m)

Feldbreite max. = 20

Formatiert: (G521.Rn.m.i)

Feldbreite

Nachkommastellen min. = 0 max. = 6

max. = 20

HINWEIS Außer dem eigentlichen Wert wird noch CR und LF gesendet.

Ausgabewerte für Eingänge, Ausgänge und Merker

Operand	Zustand	Ausgabe (ASCII-Zeichen)
Eingang	bestromt	1
Eingang	unbestromt	0
Ausgang	gesetzt	1
Ausgang	zurückgesetzt	0
Merker	gesetzt	1
Merker	zurückgesetzt	0

Beispiel

1	O10:=1	Setze Ausgang 10
2	O20:=0	Setze Ausgang 20 zurück
3	M25:=1	Setze Merker 25
4	M35:=0	Setze Merker 35 zurück
5	R2:=546.067	Lade die Zahl 546,06 in das Realzahlregister R2
6	N3:= 34	Lade die Zahl 34 in das Ganzzahlregister N3
7	G500.2.6.1.0	;Initialisierung der aktuellen Schnittstelle (COM2)
8	G521.I7	<i>;gibt den Zustand des Eingang 7 über den aktuellen Datenkanal (0 oder 1, zustandsabhängig sowie CR und LF) aus</i>
9	G521.O10	<i>;gibt den Zustand des Ausgang 10 über den aktuellen Datenkanal (Ausgabe 1 sowie CR und LF) aus</i>
10	G521.O20	<i>;gibt den Zustand des Ausgang 20 über den aktuellen Datenkanal (Ausgabe 0 sowie CR und LF) aus</i>
11	G521.M25	<i>;gibt den Zustand des Merkers 25 über den aktuellen Datenkanal (Ausgabe 1 sowie CR und LF) aus</i>
12	G521.M35	<i>;gibt den Zustand des Merkers 35 über den aktuellen Datenkanal (Ausgabe 0 sowie CR und LF) aus</i>
13	G521.N3.10	<i>;gibt den Inhalt von N3 in rechtsbündig in formatierter Ausgabe (10 Stellen mit 8 vorlaufenden Leerzeichen sowie CR und LF) über den aktuellen Datenkanal aus</i>
14	G521.R2	<i>;gibt den Inhalt von R2 sowie CR und LF linksbündig unformatiert über den aktuellen Datenkanal aus</i>
15	G521.R2.10.2	<i>;gibt den Inhalt von R2 rechtsbündig und formatiert über den aktuellen Datenkanal (10 Zeichen breites Feld mit 2 Stellen nach dem Komma sowie CR und LF)aus</i>
16	G521.R2.10.0	<i>;gibt den Inhalt von R2 rechtsbündig und formatiert über den aktuellen Datenkanal (10 Zeichen breites Feld ohne Nachkommastellen sowie CR und LF) aus</i>
17	END	

1.16.4.12 G531 - Wertübernahme

G531.

G531.(Operand) (Marke)

Beschreibung:

G531. wartet an der aktuellen seriellen Schnittstelle auf den Empfang von Zeichen. Mit dem Empfang von CR wird die Übertragung als beendet betrachtet und die empfangenen Zeichen entsprechend dem Operanden weiterverarbeitet.

Eine Zeitüberwachung ist im G531-Befehl nicht implementiert und kann bei Bedarf mit den Befehlen G401, G421, G422 und G423 realisiert werden.

Wird während des Empfanges ein Übertragungsfehler (Parity-Error,...) festgestellt oder eine ungültige Zeichenfolge empfangen, so wird das Programm an der Marke fortgeführt.

Befehlsform:

G531.Rn Marke

G531.Nn Marke

G531.On Marke

G531.Mn Marke

G531.Sn Marke

Beispiel

1	G500.2.6.1.0	;Initialisierung der aktuellen Schnittstelle
2	G531.R2 FEHLER	;wartet auf den Empfang einer Nachricht an der aktuellen seriellen Schnittstelle und weist die Information dem Register R2 zu, bei Übertragungsfehler wird das Programm an der Marke „FEHLER“ fortgeführt
3	G531.M3 FEHLER	;wartet auf den Empfang einer Nachricht an der aktuellen seriellen Schnittstelle und setzt den Merker 3 entsprechend der Information
4	G531.S2 FEHLER	;wartet auf den Empfang einer Nachricht an der aktuellen seriellen Schnittstelle und weist die Information dem Zeichenpuffer (in diesem Fall, dem globalen Zeichenpuffer) S2 zu, bei Übertragungsfehler wird das Programm an der Marke „FEHLER“ fortgeführt
5	JMP ENDE	
6	\$FEHLER	;Fehlermeldungsprogrammteil
7	
8	
9	\$ENDE	
10	END	

Operand	Zustand	Ausgabe (ASCII-Zeichen)
Ausgang	Ausgang setzen	1
Ausgang	Ausgang rücksetzen	0
Merker	Gesetzt	1
Merker	Gesetzt	1
Merker	Zurückgesetzt	0
Realzahlregister	546.09	546.09
Ganzzahlregister	34	34

1.16.4.13 G532 - Zeichenübernahme in den lokalen Zeichenpuffer S0

G532.

G532.(Endekriterium).(Marke)

Beschreibung:

Der Befehl **G532.n** übernimmt solange Zeichen (max. 80) von der aktuellen seriellen Schnittstelle, bis das Endekriterium „n“ empfangen wurde oder ein Fehler aufgetreten ist und legt diese Zeichen im lokalen Zeichenpuffer S0 ab. Das Endekriterium „n“ ist eine Zahl größer 0 und kleiner 255 und repräsentiert letztlich das entsprechende ASCII-Zeichen. Tritt während der Übernahme der Zeichen ein Fehler auf, wird die Übernahme abgebrochen und das Programm an der Marke fortgeführt.

Fehlermöglichkeiten:

- Parity-Error, Overun-Error, Framing-Error bei der Zeichenübertragung
- Zeichenpuffer voll

HINWEIS Das Programm bleibt solange in der Programmzeile (im folgenden Beispiel Zeile 2) stehen, bis das Endekriterium erkannt wird oder ein Übertragungsfehler auftritt.

Befehlsform:

G532.n Marke

Beispiel

1	G500.2.6.1.0	;Initialisierung der seriellen Schnittstelle COM2
2	G532.13 U_FEHLER	;empfängt Zeichen an der seriellen Schnittstelle
3		und legt diese im lokalen Zeichenpuffer S0 ab,
		bis das Endkriterium 13 = 0Dhex = CR empfangen wurde
4	N3:=COPY.2.5 C_FEHLER	;wandle ab dem 2. Zeichen die nächsten 5 Zeichen aus dem Zeichenpuffer in eine Zahl und lege diese Zahl im Ganzzahlregister 3 ab
5	JMP ENDE	
6	\$U_FEHLER	
7	SUB Fehler_1	
8	JMP ENDE	
9	\$C_FEHLER	
10	SUB Fehler_2	
11	\$ENDE	
12	END	

1.16.4.14 G533 - Zeichenübernahme im Hintergrund bis zum Endekriterium

G533

G533.(Endekriterium)

Beschreibung:

Der Befehl **G533.n** übernimmt im Hintergrund solange Zeichen (max. 80) von der aktuellen seriellen Schnittstelle, bis das Endekriterium „n“ empfangen wurde oder ein Fehler aufgetreten ist. Die Zeichen verbleiben im Empfangspuffer der seriellen Schnittstelle.

Das Endekriterium „n“ ist eine Zahl größer 0 und kleiner 255 und repräsentiert letztlich das entsprechende ASCII-Zeichen.

Der Zusatz „im Hintergrund“ bedeutet, dass die PA-CONTROL eine Systemroutine startet, welche die Zeichen übernimmt. Anschließend wird der nächste Befehl des Programms ausgeführt.

Ob die Zeichenkette komplett empfangen wurde, kann mit dem Befehl Ni:=CHN abgeprüft werden.

Fehlermöglichkeiten (siehe dazu auch Abschnitt 1.16.4.14 , Seite 273):

- Parity-Error, Overun-Error, Framing-Error bei der Zeichenübertragung
- Zeichenpuffer voll

Mit dem Befehl „S0:=CHN“ können die Zeichen dann aus dem Empfangspuffer der seriellen Schnittstelle in den lokalen Zeichenpuffer kopiert werden.

HINWEIS Bei Kommunikationsaufbau über die serielle Schnittstelle kann es u.U. zu Informationsverlust kommen.

G510.Bitte Senden!

...

;Achtung: Sendet Kommunikationspartner
;sofort, besteht unter Umständen Daten
;verlust !

G533.13

;Ab hier ist PAC bereit zum Empfangen

Die bessere Lösung ist:

G533.13

;Die PAC ist bereit zum Empfangen von
;Daten

...

G510.Bitte Senden!

;PAC ist bereits empfangsbereit

Befehlsform:

G533.n

Beispiel

<pre> 1 G500.2.6.1.0 2 G533.13 3 \$SCHLEIFE 4 SUB ARBEIT 5 N1:=CHN 6 CASE.JMP.N1 7 (ZEI_DA) 8 (U_FEHLER) 9 (P_VOLL) 10 ELSE SCHLEIFE 11 \$ZEI_DA 12 S0:=CHN 13 N3:=COPY.2.5 C_FEHLER 14 JMP ENDE 15 \$U_FEHLER 16 O16:=1 17 I16.1 18 O16:=0 19 JMP ENDE 20 \$P_VOLL 21 G510.PUFFER VOLL 22 \$ENDE 23 END </pre>	<pre> ;Initialisierung der seriellen Schnittstelle COM2 empfangt Zeichen im Hintergrund an der seriel- len Schnittstelle und legt diese im Empfangspuf- fer ab, bis das Endekriterium 13 = 0Dhex = CR empfangen wurde ;Zeichenkette komplett empfangen ? ;Auswertung der Übernahme ;Endekriterium wurde empfangen ;Fehler bei der Übernahme eines Zeichens ;Zeichenpuffer ist voll und das Endekriterium wurde noch nicht empfangen ;Endekriterium wurde noch nicht empfangen ;kopiere Inhalt des Empfangspuffers des aktuel- len Datenkanal in den lokalen Zeichenpuffer S0 ;wandle ab dem 2. Zeichen die nächsten 5 Zei- chen aus dem Zeichenpuffer in eine Zahl und lege diese Zahl im Ganzzahlregister 3 ab ;Fehleranzeige ein ;Fehler quittiert ;Fehleranzeige aus ;Ausgabe der Fehlernachricht </pre>
--	---

1.16.4.15 G534 - Zeichenübernahme einer bestimmten Menge von Zeichen im Hintergrund

G534.

G534.(Anzahl Zeichen)

Beschreibung:

Der Befehl **G534.n** übernimmt im Hintergrund n Zeichen (max. 80) von der aktuellen seriellen Schnittstelle. Die Zeichen verbleiben im Empfangspuffer der seriellen Schnittstelle.

Das Endekriterium „n“ ist eine Zahl größer 0 und kleiner 81 und repräsentiert die Anzahl der zu empfangenden Zeichen.

Der Zusatz „im Hintergrund“ bedeutet, dass die PA-CONTROL eine Systemroutine startet, welche die Zeichen übernimmt. Anschließend wird der nächste Befehl des Programms ausgeführt.

Ob die Zeichenkette komplett empfangen wurde, kann mit dem Befehl Ni:=CHN abgeprüft werden.

Fehlermöglichkeiten:

- Parity-Error, Overun-Error, Framing-Error bei der Zeichenübertragung
- Zeichenpuffer voll

Danach können mit dem Befehl „S0:=CHN“ die Zeichen aus dem Empfangspuffer der seriellen Schnittstelle in den lokalen Zeichenpuffer kopiert werden.

Befehlsform:

G534.n

Beispiel

<pre> 1 G500.2.6.1.0 2 G534.4 3 \$SCHLEIFE 4 SUB ARBEIT 5 N1:=CHN 6 CASE.JMP.N1 7 (ZEI_DA) 8 (U_FEHLER) 9 (P_VOLL) 10 ELSE SCHLEIFE 11 \$ZEI_DA 12 S0:=CHN 13 N3:=COPY.1.4 C_FEHLER 14 JMP ENDE 15 \$U_FEHLER 16 SUB Fehler_1 17 JMP ENDE 18 \$P_VOLL 19 SUB Fehler_2 20 \$ENDE 21 END </pre>	<pre> ;Initialisierung der seriellen Schnittstelle 2 ;empfängt vier Zeichen im Hintergrund an der seriellen Schnittstelle und legt diese im Emp- fangspuffer ab ;Zeichenkette komplett empfangen ? ;Auswertung der Übernahme ;vier Zeichen wurden empfangen ;Fehler bei der Übernahme eines Zeichens ;Zeichenpuffer ist voll ;Endekriterium wurde noch nicht empfangen ;kopiere Inhalt des Empfangspuffers des aktuel- len Datenkanal in den lokalen Zeichenpuffer S0 ;wandle ab dem 2. Zeichen die nächsten 5 Zei- chen aus dem lokalen Zeichenpuffer S0 in eine Zahl und lege diese Zahl im Ganzzahlregister 3 ab </pre>
---	--

1.16.4.16 G540 - Zeichenübernahme von der Tastatur, Prüfe ob eine Taste betätigt

G540

G540.(N-Register)

Beschreibung:

Der Befehl G540 prüft, ob eine Taste auf der Frontplatte der PA-CONTROL betätigt ist.

Ist dies der Fall, so wird der Tastencode im N-Register abgelegt. Ist keine Taste betätigt, wird das N-Register auf 0 gesetzt.

Der Befehl dient zum Abfragen der Tastatur, während ein Programmteil bearbeitet wird, um auf mögliche Anforderungen eines Bedieners zu reagieren.

HINWEIS Der Befehl arbeitet nur mit der Frontplatte der PA-CONTROL (LC-Display, Folientastatur).

Die Tastencodes entnehmen Sie bitte der Tabelle „PAC-Tastencode“ im Kapitel „Technischer Anhang“.

Befehlsform:

G540.Nn

Beispiel

1	G11.0	;Ablaufanzeigen ausschalten
2	G500.0	;schaltet die G500-Befehle auf das aktuelle Anzeigemedium um (LC-Anzeige oder bei „Simulation der Frontplatte“ die serielle Schnittstelle)
3	G501	;Anzeige löschen
4	N1:=0	
5	\$SCHLEIFE	
6	O1:=1 T10	
7	O1:=0 T10	
8	G540.N1	;prüft, ob eine Taste betätigt ist und legt den Tasten-Code in N1 ab, ansonsten wird N1 zu 0
9	M1:=N1>0	;war eine Taste betätigt ?
10	G21 M1.0 SCHLEIFE	;Nein !
11	G503.4.1	;positioniere Cursor in der 4. Spalte der 1. Zeile
12	G502	;lösche bis zum Zeilenende
13	G510.N1	
14	G512.32	
15	G520.N1	
16	JMP SCHLEIFE	
17	END	

1.16.4.17 G541 - Hole ein Zeichen von der Tastatur

G541

G541.(N-Register)

Beschreibung:

Der Befehl **G541** wartet, bis eine Taste auf der Frontplatte der PA-CONTROL betätigt ist und legt den Tastencode im N-Register ab.

Der Befehl G541 kann verwendet werden, um eigene Menüs auf der PA-CONTROL zu realisieren.

HINWEIS Der Befehl arbeitet nur mit der Frontplatte der PA-CONTROL (LC-Display, Folientastatur).

Die Tastencodes entnehmen Sie bitte der Tabelle „PAC-Tastencode“ im Kapitel „Technischer Anhang“.

Befehlsform:

G541.Nn

Beispiel

1	G11.0	;Ablaufanzeigen ausschalten
2	G500.0	;schaltet die G500-Befehle auf das aktuelle Anzeigemedium um (LC-Anzeige oder bei „Simulation der Frontplatte“ die serielle Schnittstelle)
3	G501	;Anzeige löschen
4	N1:=0	
5	\$SCHLEIFE	
6	G541.N1	;wartet bis eine Taste betätigt ist und legt den Tastencode in N1 ab
7	G503.4.1	;positioniere Cursor in der 4. Spalte der 1. Zeile
8	G502	;lösche bis zum Zeilenende
9	G510.N1	
10	G512.32	
11	G520.N1	
12	JMP SCHLEIFE	
13	END	

1.16.4.18 G542 - Eingabe eines Wertes über Tastatur

G542

G542.(Spalte).(Zeile).(Länge).(Ziel)

Beschreibung:

Der Befehl G542.s.z.l.Ni ermöglicht dem Bediener die Eingabe eines Wertes während des Programmablaufs über die Tastatur.

Der Befehl eröffnet ein Eingabefeld (siehe Kapitel „Bedienoberfläche“). Die Länge und die Position des Eingabefeldes werden durch die Operatoren (Parametern) des Befehls festgelegt.

Durch Abfrage des Systemmerkers **SM5** kann festgestellt werden, ob der G542-Befehl mit der Enter-Taste (d.h. Übernahme des Wertes) oder mit der ESC-Taste (d.h. Abbruch der Eingabe und Rückgabe des alten Wertes) verlassen wurde.

Es gilt:

s	→	steht für die Spalte
z	→	steht für die Zeile
l	→	steht für die Länge des Eingabefeldes
Ziel	→	Register oder Zeichenpuffer (Ablage)

Befehlsform:

G542.s.z.l.Ni

G542.s.z.l.Ri

G542.s.z.l.Si *

G542.Ns.Nz.Nl.Ni

G542.Ns.Nz.Nl.Ri

G542.Ns.Nz.Nl.Si *

* Si kann S0, S1 ...S16 sein

Beispiel

1	G11.0	;Ablaufanzeigen ausschalten
2	G500.0	;schaltet die G500-Befehle auf das aktuelle Anzeigemedium um (LC-Anzeige oder bei „Simulation der Frontplatte“ die serielle Schnittstelle)
3	G501	;Anzeige löschen
4	\$SCHLEIFE	
5	G503.1.1	
6	G510.Eingabe Zeit	
7	G542.30.1.3.N1	;Eingabe des N-Registers 1 über die Frontplatte. Das Eingabefeld erscheint in der 30.Spalte der 1.Zeile und ist 3 Zeichen lang
8	G501	
9	G503.1.2	
10	G510.Zeit fuer Ausgang 1 ist	
11	G512.32	
12	G520.N1	
13	G512.32	
14	G510. * 10ms	
15	O1:=1 TN1 O1:=0	
16	JMP SCHLEIFE	
17	END	

1.16.5 Befehle der G6xx-Gruppe

G6xx.

1.16.5.1 Möglichkeiten und Befehlsaufbau der G6xx-Befehlsgruppe

Mit dieser Befehlsgruppe können während des Automatikbetriebes Registerinhalte auf Ausgänge oder Merker abgebildet werden, oder es kann das Abbild von Eingängen oder Merken in ein Register geladen werden.

Befehlsaufbau allgemein:

G60?.Rn.m.i

G60?.Nn.m.i

Rn / Nn:	Quelle / Ziel
m :	erstes Element der Ziel- / Quelladresse (entspricht dem niederwertigsten Bit)
i :	Anzahl der Elemente auf denen / von denen abgebildet werden soll ($1 \leq i \leq 32$)

HINWEIS $m + i$ muss kleiner gleich der maximalen Anzahl der verwendeten Elemente sein.

Der Registerinhalt wird wie folgt benutzt:

- Nachkommastellen werden abgeschnitten
- maximal 32 Bit werden abgebildet

Es stehen unterschiedliche Formate zur Verfügung:

- Binär-Abbild
- BCD-Abbild

1.16.5.2 G600 - Binärdarstellung auf Ausgänge

G600.

G600.(Register).(1.Ausgang).(Anzahl Ausgänge)

Befehlsform:

G600.Rn.m.i

G600.Nn.m.i

Beispiel 1

```
1   R3:=10.7
2   G600.R3.5.8
3   END
```

;Inhalt von R3 wird im Binärcode ab Ausgang 5 mit 8 Bit dargestellt

Zustände der Ausgänge nach Ausführung der obigen Befehle:

Ausgang 5	→	zurückgesetzt (0)
Ausgang 6	→	gesetzt (1)
Ausgang 7	→	zurückgesetzt (0)
Ausgang 8	→	gesetzt (1)
Ausgang 9	→	zurückgesetzt (0)
Ausgang 10	→	zurückgesetzt (0)
Ausgang 11	→	zurückgesetzt (0)
Ausgang 12	→	zurückgesetzt (0)

Beispiel 2

```
1   N3:=18
2   G600.N3.2.4
3   END
```

;Inhalt von N3 wird im Binärcode ab Ausgang 2 mit 4 Bit dargestellt

Zustände der Ausgänge nach Ausführung der obigen Befehle:

Ausgang 2	→	zurückgesetzt (0)
Ausgang 3	→	gesetzt (1)
Ausgang 4	→	zurückgesetzt (0)
Ausgang 5	→	zurückgesetzt (0)

1.16.5.3 G601 - BCD-Darstellung auf Ausgänge

G601.

G601.(Register).(1.Ausgang).(Anzahl Ausgänge)

Befehlsform:

G601.Rn.m.i

G601.Nn.m.i

Beispiel 1

```
1   R3:=10.7
2   G601.R3.5.8
3   END
```

;Inhalt von R3 wird im BCD-Code ab Ausgang 5 mit 8 Bit dargestellt

Zustände der Ausgänge nach Ausführung der obigen Befehle:

Ausgang 5	→	zurückgesetzt (0)
Ausgang 6	→	zurückgesetzt (0)
Ausgang 7	→	zurückgesetzt (0)
Ausgang 8	→	zurückgesetzt (0)
Ausgang 9	→	gesetzt (1)
Ausgang 10	→	zurückgesetzt (0)
Ausgang 11	→	zurückgesetzt (0)
Ausgang 12	→	zurückgesetzt (0)

Beispiel 2

```
1   N3:=12
2   G601.N3.2.4
3   END
```

;Inhalt von N3 wird im BCD-Code ab Ausgang 2 mit 4 Bit dargestellt

Zustände der Ausgänge nach Ausführung der obigen Befehle:

Ausgang 2	→	zurückgesetzt (0)
Ausgang 3	→	gesetzt (1)
Ausgang 4	→	zurückgesetzt (0)
Ausgang 5	→	zurückgesetzt (0)

1.16.5.4 G602 - Binärdarstellung auf Merker

G602

G602.(Register).(1.Merker).(Anzahl Merker)

Befehlsform:

G602.Rn.m.i

G602.Nn.m.i

Beispiel 1

```
1   R3:=11.3
2   G602.R3.4.8           ;Inhalt von R3 wird im Binär-Code ab Merker 4
                           mit 8 Bit dargestellt
3   END
```

Zustände der Merker nach Ausführung der obigen Befehle:

```
Merker 4   →   gesetzt (1)
Merker 5   →   gesetzt (1)
Merker 6   →   zurückgesetzt (0)
Merker 7   →   gesetzt (1)
Merker 8   →   zurückgesetzt (0)
Merker 9   →   zurückgesetzt (0)
Merker 10  →   zurückgesetzt (0)
Merker 11  →   zurückgesetzt (0)
```

Beispiel 2

```
1   R3:=65535
2   G602.R3.17.32        ;Inhalt von R3 wird im Binär-Code ab Merker 17
                           mit 32 Bit dargestellt
3   END
```

Zustand der Merker 17 bis 49 nach der Ausführung der obigen Befehle:

```
Merker 17 – 32   →   sind gesetzt
Merker 33 – 49   →   sind zurückgesetzt
```

Beispiel 3

1 N3:=11
2 G602.N3.1.4
3 END

*;Inhalt von N3 wird im Binär-Code ab Merker 1
mit 4 Bit dargestellt*

Zustände der Merker nach Ausführung der obigen Befehle:

Merker 1 → gesetzt (1)
Merker 2 → gesetzt (1)
Merker 3 → zurückgesetzt (0)
Merker 4 → gesetzt (1)

1.16.5.5 G603 - Eingänge im Binärformat in Register

G603.

G603.(Register).(1.Eingang).(Anzahl Eingänge)

Befehlsform:

G603.Rn.m.i

G603.Nn.m.i

Beispiel 1

1 G603.R3.2.8

*;die Eingänge 2 bis 9 (2=niederwertigstes Bit)
werden als Binär-Zahl interpretiert und der Wert
ins Register 3 übernommen*

2 END

Zustände der Eingänge bei der Ausführung des obigen Befehls:

Eingang 2	→	unbestromt (0)	→	Wert = 1
Eingang 3	→	bestromt (1)	→	Wert = 2
Eingang 4	→	unbestromt (0)	→	Wert = 4
Eingang 5	→	bestromt (1)	→	Wert = 8
Eingang 6	→	unbestromt (0)	→	Wert = 16
Eingang 7	→	unbestromt (0)	→	Wert = 32
Eingang 8	→	unbestromt (0)	→	Wert = 64
Eingang 9	→	bestromt (1)	→	Wert = 128

Im Binärcode ergibt dies folgende Zahl „10001010“ und im Register 3 den Wert 138.

1.16.5.6 G604 - Merker im Binärformat in Register

G604.

G604.(Register).(1.Merker).(Anzahl Merker)

Befehlsform:

G604.Rn.m.i

G604.Nn.m.i

Beispiel 1

1 G604.R3.2.9

;die Merker 2 bis 9 (2=niederwertigstes Bit) werden als Binär-Zahl interpretiert und der Wert ins Register R3 übernommen

2 END

Zustände der Merker bei der Ausführung des obigen Befehls:

Merker 2	→	1	→	Wert = 1
Merker 3	→	1	→	Wert = 2
Merker 4	→	1	→	Wert = 4
Merker 5	→	1	→	Wert = 8
Merker 6	→	0	→	Wert = 16
Merker 7	→	0	→	Wert = 32
Merker 8	→	0	→	Wert = 64
Merker 9	→	1	→	Wert = 128

Im Binärcode ergibt dies folgende Zahl „10001111“ und im Register 3 den Wert 143.

1.16.6 Befehle der G7xx-Gruppe

1.16.6.1 Allgemeines

Alle PA-CONTROL Geräte sind ab Version 4.60 mit dem CANopen-Interface ausgestattet. Am CANopen-Bus können unterschiedliche Teilnehmer, die in Gruppen eingeteilt sind, angeschlossen werden. Den Teilnehmern sind zu ihrer Identifikation Teilnehmeradressen, die ID-Nummern, zugewiesen.

ID-Nummer	Teilnehmer	
ID 1 ... ID16	Antriebsachse 1 bis Antriebsachse 16	LV-servoTEC, PA-CONTROL MP
ID17 ... ID48	Digitale Ein- / Ausgangsmodule Analoge Ein- / Ausgangsmodule	Murr Beckhoff FESTO
ID49 ... ID52	Beliebiger CANopen-Teilnehmer	Beispiele
ID52 ... ID59	Reserve	Reserve
ID60 ... ID63	4. Bedienkonsole bis 1. Bedienkonsole	SÜTRON

Als Teilnehmer mit den ID-Nummern 1 bis 48 können nur Geräte entsprechend der obigen Tabelle angeschlossen werden. Diese Geräte werden von der PA-CONTROL beim Umladen automatisch erkannt und vom Betriebssystem bedient.

Als Teilnehmer 49 bis 52 kann ein beliebiges CANopen-Gerät angeschlossen werden. Für die Bedienung dieser Teilnehmer werden für den Automatik-Betrieb Befehle bereit gestellt. Mit diesen Befehlen kann dann in der jeweiligen Anwendung durch PNC/PTX- oder PAB-Programme die Kommunikation realisiert werden.

Für die Inbetriebnahme und für den Service sind im Programm WINPAC, Menü Diagnose, Möglichkeiten zur Überprüfung und für die einfache Bedienung vorhanden.

HINWEIS Als Teilnehmer 60 bis 63 kann eine Bedienkonsole angeschlossen werden. Die Kommunikation erfolgt über PDOs direkt durch das Betriebssystem. Es kommen Bedienkonsolen der Fa. SÜTRON zum Einsatz.

HINWEIS Beachten Sie unbedingt die Hinweise und Vorschriften der Hersteller von Geräten, die am CANopen-Bus in Ihrem Fall zur Anwendung kommen. Diese Dokumentationen stellen Ihnen auch nötige Zusatzinformationen zu den in den nächsten Abschnitten beschriebenen Befehlen zur Verfügung.

Der Befehlssatz für CANopen-Teilnehmer besteht aus den folgenden Befehlsgruppen:

- Netzwerk-Management Befehle,
- Emergency-Befehle,
- SDO-Befehle und
- PDO-Befehle.

Der Befehlssatz wird durch die Befehlsgruppe G7xx realisiert.

1.16.6.2 G 70x - Netzwerk-Management Befehle

Mit den 5 Befehlen dieser Befehlsgruppe können die unterschiedlichen Betriebszustände (siehe *Abbildung 13, unten*) eines Teilnehmers am CANopen-Bus erreicht werden.

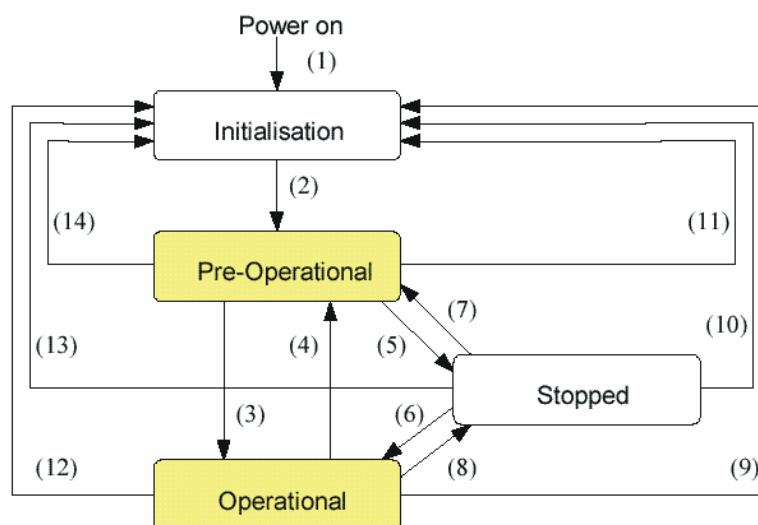


Abbildung 13: Betriebszustände eines CANopen-Teilnehmers

1.16.6.3 G701 - Start Remote Node

G701

G701.n

Befehlsformen:

G701.49

G701.50

Beschreibung:

- Mit diesem Befehl erfolgt der Wechsel vom Betriebszustand „Pre-Operational“ in den Zustand „Operational“ (siehe *Abbildung 13, oben, Pfad 3*).
- Für die Ausführung dieses Befehls sendet der Master das NMT-Telegramm „Start-Remote-Node“ über den CANopen-Bus.

Beispiel: Am Ende des Abschnittes finden Sie im Rahmen eines Komplexbeispiels eine Anwendung für diesen Befehl.

1.16.6.4 G702 - Stop Remote Node

G702

G702.n

Befehlsformen:

G702.49

G702.50

Beschreibung:

- Mit diesem Befehl erfolgt der Wechsel vom Betriebszustand „Operational“ in den Zustand „Stopped“ (siehe *Abbildung 13, Seite 289, Pfad 8*).
- Für die Ausführung dieses Befehls sendet der Master das NMT-Telegramm „Stop-Remote-Node“ über den CANopen-Bus.

Beispiel:

Am Ende des Abschnittes finden Sie im Rahmen eines Komplexbeispiels eine Anwendung für diesen Befehl.

1.16.6.5 G703 - Reset Remote Node

G703

G703.n

Befehlsformen:

G703.49

G703.50

Beschreibung:

- Mit diesem Befehl erfolgt der Wechsel vom Betriebszustand „Operational“ in den Zustand „Initialisation“ (siehe *Abbildung 13, Seite 289, Pfad 9*). Nach erfolgter Initialisierung wechselt der Teilnehmer selbständig in den Zustand „Pre-Operational“.
- Für die Ausführung dieses Befehls sendet der Master das NMT-Telegramm „Reset-Remote-Node“ über den CANopen-Bus.

Beispiel:

Am Ende des Abschnittes finden Sie im Rahmen eines Komplexbeispiels eine Anwendung für diesen Befehl.

1.16.6.6 G704 - Enter Pre-Operational-State

G704

G704.n

Befehlsformen:

G704.49

G704.50

Beschreibung:

- Mit diesem Befehl erfolgt der Wechsel vom Betriebszustand „Operational“ in den Zustand „Pre-Operational“ (siehe *Abbildung 13, Seite 289, Pfad 4*).
- Für die Ausführung dieses Befehls sendet der Master das NMT-Telegramm „Enter-Pre-Operational-State“ über den CANopen-Bus.

Beispiel:

Am Ende des Abschnittes finden Sie im Rahmen eines Komplexbeispiels eine Anwendung für diesen Befehl.

1.16.6.7 G705 - Reset-Communication

G705

G705.n

Befehlsformen:

G705.49

G705.50

Beschreibung:

- Mit diesem Befehl erfolgt der Wechsel vom Betriebszustand „Operational“ in den Zustand „Initialisation“ (siehe *Abbildung 13, Seite 289, Pfad 9*). Nach erfolgter Initialisierung wechselt der Teilnehmer selbständig in den Zustand „Pre-Operational“.
- Für die Ausführung dieses Befehls sendet der Master das NMT-Telegramm „Reset-Communication“ über den CANopen-Bus.

Beispiel:

Am Ende des Abschnittes finden Sie im Rahmen eines Komplexbeispiels eine Anwendung für diesen Befehl.

1.16.6.8 G711 - Funktionsüberwachung der Teilnehmer

G711

G711.(ID).Nn (Marke)

Befehlsformen:

G711.49.12 FEHLER

Anwendung:

Das Node Guarding Protokoll dient der Funktionsüberwachung der Teilnehmer. Dazu soll der Teilnehmer in gleichen Zeitabständen von der PA-CONTROL, dem CANopen-Master, mit den "NODE-GUARDING-Telegramm" angesprochen und eine Nachricht des Teilnehmers angefordert werden.

Diese Art der Überwachung eines Teilnehmers ist vorgesehen, wenn die Kommunikation nach der Initialisierung nur über PDOs abgewickelt wird. Der maximale zeitliche Abstand zwischen zwei Anforderungen an den Teilnehmer, auch Node Guard Telegramm genannt, ergibt sich aus dem Produkt der Überwachungszeit (Guard Time) und eines Faktors (Life Time Factor). Ist einer dieser Werte 0, dann wird die Ansprechüberwachung deaktiviert.

Wird der Teilnehmer innerhalb der Überwachungszeit nicht angesprochen, ist die Reaktion des Teilnehmers aus der Gerätebeschreibung zu entnehmen.

Beschreibung:

Der G711-Befehl fordert mit einem Datenanforderungstelegramm (Remote Frame) die Überwachungsnachricht eines ausgewählten Teilnehmers an. Die Antwort wird im N-Register, das im Befehl angegeben wurde, abgelegt. Der Inhalt dieser codierten Nachricht kann aus dem Handbuch des Teilnehmers entnommen werden.

Wird innerhalb von 200ms keine Überwachungsnachricht vom Teilnehmer empfangen, so verzweigt das PA-CONTROL-Programm zur programmierten Marke.

Beispiel:

Am Ende des Abschnittes finden Sie im Rahmen eines Komplexbeispiels eine Anwendung für diesen Befehl.

1.16.6.9 G72x - Befehle für die Kontrolle von Gerätefehlern, Allgemeines

Gerätefehlernachrichten („Emergency Messages“) werden durch interne Gerätefehler ausgelöst. Sie haben eine hohe Priorität, um eine schnelle Bearbeitung zu sichern.

Eine Gerätefehlernachricht hat einen vorgeschriebenen Aufbau:

- Fehlerfeld mit vordefinierter Fehlernummer (2 Bytes)
- Fehlerregister (1 Byte)
- Fehlerkategorie (1 Byte)
- Zusätzliche Informationen.

Das höherwertige Byte der Fehlernummer gibt die Fehlerklasse, das niederwertige Byte die Fehlernummer in der Klasse an. Im Kommunikationsprofil sind die Fehlernummern von xx00h bis xx7Fh definiert.

Die Fehlernummern von xx80h bis xxFFh ergeben sich aus Festlegungen der Hersteller. Die Fehlerkategorien erlauben eine Klassifizierung der aufgetretenen Fehler nach ihrer Bedeutung.

1.16.6.10 G721 / G722 - Prüfe ob eine Gerätefehlernachricht vorhanden ist

G721 / G722

G721.(ID) (Marke)

G722.(ID) (Name)

Es wird das Bit „Neue Gerätefehlernachricht empfangen“ geprüft. Ist es gesetzt, wurde eine neue Gerätefehlernachricht empfangen.

In diesem Fall wird die Nachricht in den S0-Zeichpuffer kopiert und das Bit zurückgesetzt. Danach wird entweder zu der im Befehl angegebenen "Marke" (G721) verzweigt oder das Unterprogramm „Name“ (G722) aufgerufen.

Liegt keine neue Gerätefehlernachricht vor, wird mit dem nächsten Befehl das Programm fortgesetzt.

Darstellung des S0-Strings nach einem Gerätefehler (Definition CiA Draft 301)

Byte	1	2	3	4	5	6	7	8
Inhalt	LowByte	HighByte	Fehler-Register (Objekt 1001)	Fehlerfeld, durch den Hersteller spezifiziert				
	Gerätefehler-Code							

Der mögliche Inhalt der Gerätefehlernachricht ist dem Handbuch des fehlerhaften CANopen-Teilnehmers zu entnehmen.

Befehlsformen:

G721.49 EMERGENCY

G722.50 GREIFER

Beispiel für die Auswertung von Fehlern:

...

N1:=GETI.1.2 ; Fehler-Code

...

N2:=GETI.3.1 ; Fehler-Register (Objekt 1001hex)

1.16.6.11 G73x - Befehle für die Bearbeitung von Servicedatenobjekten, Allgemeines

Analog zu anderen bekannten Feldbussystemen unterscheidet CANOpen zwei grundsätzliche Datenübertragungsmechanismen, den schnellen Austausch kurzer Prozessdaten über so genannte Prozessdatenobjekte (PDOs) sowie den Zugriff auf Eintragungen des Objekt-datenverzeichnisses über Servicedatenobjekte (SDOs). Die Servicedatenobjekte dienen in erster Linie zur Übertragung von Parametern während der Gerätekonfiguration sowie zur Übertragung längerer Datenbereiche.

Der Zugriff auf die Servicedatenobjekte erfolgt mit variabler Länge

HINWEIS Für das Lesen oder Beschreiben eines Servicedatenobjektes sind grundsätzlich ein Anforderungstelegramm und ein Antworttelegramm nötig. Es ist daher notwendig, dass auf einen Teilnehmer aus nur einem parallelen Ablauf zugegriffen werden sollte.

Bei Nichtbeachten dieses Hinweises kann es zu Datenfehlern kommen, wenn das Anforderungstelegramm aus dem Ablauf 1, das Antworttelegramm aus dem Ablauf 2 kommt.

1.16.6.12 G730 - Liest den Inhalt eines Servicedatenobjektes (Initiate Domain Upload Rq)

G730

Befehlsform:

G730.<ID>.<Index(hex)>.<Subindex>.<Anzahl-Daten-Bytes>.<Fehler>.<Read Data> <Marke>

Tritt während des Lesevorganges ein Fehler auf, z.B. ein falscher Index oder falscher Subindex, so wird der Fehlercode im festgelegten Ganzzahlregister abgespeichert. Das Programm verzweigt zur <Marke>. Bei einem regulär ablaufenden Befehl werden die Daten im Register für die gelesenen Daten abgelegt und der Programmablauf mit dem nächsten Befehl fortgesetzt.

Wird die Anforderung zum Lesen nicht innerhalb 1 Sekunde quittiert, dann verzweigt die PA-CONTROL mit dem Fehler 9, „IEF-Time-Out“, nach der Marke.

HINWEIS Der Index wird als Hex-Zahl übergeben

Beispiel:

G730.49.1000.0.4.N1.N2 FEHLER_SDO

In diesem Beispielbefehl wird die Gerätedefinition (Device Type) des CANOpen-Teilnehmers 49 eingelesen und im Ganzzahlregister N2 abgelegt. In einem Fehlerfall würde der Fehler im Register N1 abgelegt und das Programm an der Marke „FEHLER_SDO“ fortgesetzt.

1.16.6.13 G731 - Schreibe den Inhalt eines Servicedatenobjektes (Initiate Domain Upload Rq)

G731

Befehlsform:

G731.<ID>.<Index(hex)>.<Subindex>.<Anzahl-Daten-Bytes>.<Fehler>.<Write Data> <Marke>

Tritt während des Schreibvorganges ein Fehler auf, so wird der Fehlercode im festgelegten Ganzzahlregister abgespeichert. Das Programm verzweigt zur <Marke>. Bei einem regulär ablaufenden Befehl wird der Programmablauf mit dem nächsten Befehl fortgesetzt.

Wird die Anforderung nicht innerhalb 1 Sekunde quittiert, dann verzweigt die PA-CONTROL mit dem Fehler 9, „IEF-Time-Out“, nach der Marke.

HINWEIS Der Index wird als Hex-Zahl übergeben

Befehlsform:

G731.49.100C.0.2.N1.N2 FEHLER_SDO

In diesem Beispiel wird die Überwachungszeit (Guard Time) des CANopen-Teilnehmers 49 mit dem Inhalt des Registers N2 überschrieben. Bei einem Fehler wird der Fehlercode im Register N1 angelegt und das Programm an der Marke „FEHLER_SDO“ fortgesetzt.

1.16.6.14 G74x, G75x - Befehle für die Bearbeitung von Prozessdatenobjekten

Wie im vorhergehenden Abschnitt beschrieben, unterscheidet CANopen zwei grundsätzliche Datenübertragungsmechanismen, den schnellen Austausch kurzer Prozessdaten über so genannte Prozessdatenobjekte (PDOs) sowie den Zugriff auf Eintragungen des Objekt-datenverzeichnisses über Servicedatenobjekte (SDOs).

Prozessdatenobjekte lassen sich wahlweise durch Ereignisse gesteuert oder synchronisiert übertragen. Der ereignisgesteuerte Modus erlaubt es darauf zu verzichten, ständig das gesamte Prozessabbild zu übertragen. Es genügt, dass nur die Änderungen übertragen werden. Auf diese Weise lassen sich die Busbelastung als auch die Reaktionszeit auf ein Minimum reduzieren. Bei vergleichsweise niedriger Baudrate wird eine hohe Kommunikationsleistung erreicht.

Prozessdatenobjekte können gesendet (Transmit) und empfangen (Receive) werden. Senden und empfangen ist dabei immer aus der Sicht des Slaves, z.B. eines I/O-Moduls zu sehen.

Für jeden Teilnehmer am CANopen-Bus sind 4 Prozessdatenobjekte zugeordnet, für die in der PA-CONTROL ein Befehl und je ein Speicherplatz existiert.

Das CANopen-Protokoll basiert auf dem CAN Application Layer (CAL) und stellt standardisierte Geräteprofile zur Verfügung. Die Grundlage der Kommunikation auf dem CANopen-Bus bildet das Kommunikationsprofil DS-301. Es definiert die zulässigen Kommunikationsmechanismen zwischen den Geräten am Bus. Das Kommunikationsprofil ist die Basis für diverse Geräteprofile, die den standardisierten Zugriff auf alle Parameter eines Gerätes ermöglichen. Die nachstehenden Profile wurden bisher implementiert:

Profil	Funktion
DS-301	CANopen-Kommunikationsprofil
DS-401	Geräteprofil für digitale und analoge E/A-Baugruppen
DS-402	Geräteprofil für Antrieb
DS-403	Geräteprofil für Bediengeräte
DS-404	Geräteprofil für Sensoren und Regler
DS-405	Schnittstelle zu programmierbaren Systemen (IEC1131)
DS-406	Geräteprofil für Drehgeber und Encoder
DS-407	Applikationsprofil für den öffentlichen Nahverkehr

Nach der Definition DEVICE-PROFILE 401 „I/O-Module“ werden die Prozessdatenobjekte wie folgt zugeordnet. Die Anordnung kann durch „variables Mapping“ verändert werden.

Receive-PDO (Empfang)	
PDO-Nr.	Daten
1	Maximal 64 digitale Ausgänge (O1 – O64)
2	Maximal 4 analoge Ausgänge (DA1 – DA4)
3	Frei
4	Frei

Transmit-PDO (Senden)	
PDO-Nr.	Daten
1	Maximal 64 digitale Eingänge (I1 – I64)
2	Maximal 4 analoge Eingänge (AD1 – AD4)
3	Frei
4	Frei

Das dritte und vierte Prozessdatenobjekt ist im Standard nicht fest definiert. Die Hersteller von I/O-Modulen legen auf diese PDOs in Abhängigkeit der Hardwarekonfiguration des I/O-Moduls die nächsten digitale und analoge Eingänge und Ausgänge. Weitere Informationen sind im Anwendungsfall der Technischen Dokumentation des Herstellers zu entnehmen.

Beispiel: Murr-I/O-Modul mit

- 16 Digitale Eingänge
- 8 digitale Ausgänge
- 4 analoge Eingänge (12 Bit, +/-10V)
- 4 analoge Ausgänge (12 Bit, +/-10V)

Receive-PDO (Empfang)	
PDO-Nr.	Daten
1	Maximal 64 digitale Ausgänge (O1 – O8)
2	Maximal 4 analoge Ausgänge (DA1 – DA4)
3	Frei
4	Frei

Transmit-PDO (Senden)	
PDO-Nr.	Daten
1	Maximal 64 digitale Eingänge (I1 – I16)
2	Maximal 4 analoge Eingänge (AD1 – AD4)
3	Frei
4	Frei

Für den Zugriff auf die Prozessdatenobjekte (PDO) sind Befehle für alle vier Objekte vorhanden:

- PDO1
- PDO2
- PDO3
- PDO4

Die Anzahl der Daten-Bytes eines PDO kann zwischen 1 und 8 variieren.

1.16.6.15 G74x - Prozessdaten lesen

G74x

G74x.(ID). (Anzahl-Daten-Bytes)

Der CANopen-Teilnehmer sendet in Abhängigkeit der Konfiguration, also bei einer Änderung eines Einganges, zyklisch oder nach einer Synchronisation, seine Prozessdaten an die PA-CONTROL. Dort erfolgt die Abspeicherung der empfangenen Daten in dem zugeordneten PDO-Empfangspuffer (PDO1, PDO2,...).

Der Befehl „Prozessdaten lesen“ bewirkt, dass die Daten aus dem Empfangspuffer in den lokalen Zeichenpuffer S0 kopiert werden.

HINWEIS Bei der Auswertung der Daten ist unbedingt zu berücksichtigen, dass die Übertragung der Prozessdaten über den CANopen-Bus im INTEL-Format erfolgt. Die Daten im lokalen Zeichenpuffer S0 werden mit dem „GETI-Befehl“ (GET-INTEL-Format), *Seite 200*, ausgelesen und stehen dann für eine Auswertung zur Verfügung.

Befehlsformen:

G741.(ID).(Anzahl-Daten-Bytes)	PDO1
G742.(ID).(Anzahl-Daten-Bytes)	PDO2
G743.(ID).(Anzahl-Daten-Bytes)	PDO3
G744.(ID).(Anzahl-Daten-Bytes)	PDO4

Beispiele:

G741.49.4	Es werden die ersten 4 Bytes der empfangenen Daten des ersten Prozessdatenobjektes (PDO1) vom CANopen-Teilnehmer 49 in den S0-String kopiert.
G742.50.2	Es werden die ersten 2 Bytes der empfangenen Daten des zweiten Prozessdatenobjektes (PDO1) vom CANopen-Teilnehmer 50 in den S0-String kopiert.

1.16.6.16 G75x - Prozessdaten senden

G75x

G75x.(ID). (Anzahl-Daten-Bytes)

Für das Senden von Prozessdaten sind zwei Schritte notwendig. Im ersten Schritt erfolgt die Eintragung der zu sendenden Daten in den S0-String. Danach erfolgt das eigentliche Senden an den CANopen-Teilnehmer.

HINWEIS Die Prozessdaten werden über den CANopen-Bus im INTEL-Format übertragen. Beim Eintragen der Daten in den lokalen Zeichenpuffer S0 muss dies unbedingt berücksichtigt werden.

Der PUTI-Befehl (PUT-INTEL-Format), *Seite 202*, schreibt den Inhalt eines N-Registers in den S0-String.

Befehlsformen:

G751.(ID).(Anzahl-Daten-Bytes)	PDO1
G752.(ID).(Anzahl-Daten-Bytes)	PDO2
G753.(ID).(Anzahl-Daten-Bytes)	PDO3
G754.(ID).(Anzahl-Daten-Bytes)	PDO4

Beispiel:

G751.49.4	Sendet die ersten 4 Bytes aus dem S0-String an das erste Prozessdatenobjekt (PDO1) des CANopen-Teilnehmers 49.
-----------	--

1.16.6.17 Applikationsbeispiele G7xx- Befehlsgruppe

Beispiel 1 für MURR-IO-Modul

Im Beispiel wird der CANopen-Modul 55900 der Fa. Murr verwendet.

55920: D18 ⇒ 8 Eingänge ⇒ auf Merker M1 bis M8
 55922: DO8/0,5A ⇒ 8 Ausgänge ⇒ auf Merker M9 bis M16

Programmname: Start-SDO.pnc

```

1      G701.49                               ; START-REMOTE-NODE
2      T50                                    ; warte bis CAN-Module im
                                           OPERATION-Mode
3      ;
4      $A
5      G721.49 EMERGENCY                     ; Fehlermeldung da?
6      ;
7      G730.49.6200.2.1.N1.N2 READ_SDO_FEHLER ; Lese SD= ⇒ Eingänge
8      G602.N2.1.8                           ; Abbild auf Merker M1 – M8
9      ;
10     G604.N3.9.8                            ; Abbild von Merker M9 – M16
11     G731.49.6200.1.1.N1.N3 WRITE_SDO_FEHLER ; Schreibe SDO ⇒ Ausgänge
12     JMP A
13     ;
14     $READ_SDO_FEHLER
15     $WRITE_SDO_FEHLER
16     O1:=1                                  ; Fehler melden
17     I1.1                                    ; SDO-Fehler
18     BREAK                                  ;
19     ;
20     $EMERGENCY
21     N5:=GETI.1.2                            ; Error-Code
22     N6:=GETI.3.1                            ; Error-Register Objekt 0x1001
23     M100:=N5=0                              ; Fehler <>0?
24     G21 M100.1 A                            ; NEIN, kein Fehler !
25     I1.1                                    ; Emergency-Fehler
26     Break                                    ;
27     ;
28     END
  
```

HINWEIS im obigen Beispiel kann auf das NODE-GUARDING verzichtet werden, mit dem der Teilnehmer über SDOs angesprochen wird

Beispiel 2 für MURR-IO-Modul

Im Beispiel wird der CANopen-Modul 55900 der Fa. Murr verwendet.

55920: D18 ⇒ 8 Eingänge ⇒ auf Merker M1 bis M8
 55922: DO8/0,5A ⇒ 8 Ausgänge ⇒ auf Merker M9 bis M16

Programmname: Start-pdo.pnc

```

1      G701.49                ; START-REMOTE-NODE
2      T50                    ; warte bis CAN-Module im OPERATION-
                               ; Mode
3      ;
4      $A
5      G721.49 EMERGENCY      ; Fehlermeldung da?
6      G711.49 NODE_GUARD    ;
7      G741.49.2              ; Lese PDO1 → S0
8      N2:=GETI.2.1           ; hole 2 Bytes aus S0
9      G602.N2.1.8           ; Abbild auf Merker M1 – M8
10     ;
11     G604.N3.9.8            ; Abbild von Merker M9 – M16 holen
12     PUTI.1.1.N3            ; in den S0 einfügen
13     G751.49.1              ; S0 → Schreibe PDO1
14     JMP A
15     ;
16     $EMERGENCY
17     N5:=GETI.1.2           ; Error Code
18     N6:=GETI.3.1           ; Error-Register Objekt 0x1001
19     M100:=N5=0             ; Fehler <>0 ?
20     G21 M100.1 A           ; Nein kein Fehler!
21     I1.1                   ; Emergency-Fehler
22     BREAK                  ;
23     ;
24     $NODE_GUARD           ; Teilnehmer hat sich auf NODE-GUARD nicht
                               ; gemeldet

25     I1.1
24     BREAK
23     ;
24     END

```

2 PA-CONTROL-Tastencode

Dez	Hex	Zeichen	PAC Tastatur
0	0	NUL	
1	1	SOH	
2	2	STX	
3	3	ETX	
4	4	EOT	
5	5	ENQ	
6	6	ACK	
7	7	BEL	
8	8	BS	Shift+DEL
9	9	HT	
10	A	LF	
11	B	VT	
12	C	FF	
13	D	CR	ENTER
14	E	SO	
15	F	SI	
16	10	DLE	
17	11	DC1	
18	12	DC2	
19	13	DC3	
20	14	DC4	
21	15	NAK	
22	16	SYN	
23	17	ETB	
24	18	CAN	
25	19	EM	
26	1A	SUB	
27	1B	ESC	ESC/Shift+ESC
28	1C	FS	
29	1D	GS	
30	1E	RS	
31	1F	US	
32	20	SP	LEERTASTE
33	21	!	Shift+1
34	22	„	Shift+2
35	23	#	
36	24	\$	Shift+4
37	25	%	Shift+5
38	26	&	Shift+6
39	27	'	
40	28	(Shift+8
41	29)	Shift+9
42	2A	*	Shift++
43	2B	+	+
44	2C	,	,
45	2D	-	-
46	2E	.	.

Dez	Hex	Zeichen	PAC Tastatur
47	2F	/	Shift+7
48	30	0	0
49	31	1	1
50	32	2	2
51	33	3	3
52	34	4	4
53	35	5	5
54	36	6	6
55	37	7	7
56	38	8	8
57	39	9	9
58	3A	:	Shift+ .
59	3B	;	Shift+ ,
60	3C	<	<
61	3D	=	Shift+0
62	3E	>	Shift+<
63	3F	?	Shift+3
64	40	@	Ctrl+Q
65	41	A	Shift+A
66	42	B	Shift+B
67	43	C	Shift+C
68	44	D	Shift+D
69	45	E	Shift+E
70	46	F	Shift+F
71	47	G	Shift+G
72	48	H	Shift+H
73	49	I	Shift+I
74	4A	J	Shift+J
75	4B	K	Shift+K
76	4C	L	Shift+L
77	4D	M	Shift+M
78	4E	N	Shift+N
79	4F	O	Shift+O
80	50	P	Shift+P
81	51	Q	Shift+Q
82	52	R	Shift+R
83	53	S	Shift+S
84	54	T	Shift+T
85	55	U	Shift+U
86	56	V	Shift+V
87	57	W	Shift+W
88	58	X	Shift+X
89	59	Y	Shift+Y
90	5A	Z	Shift+Z
91	5B	[Ctrl+8
92	5C	\	
93	5D]	Ctrl+9

Dez	Hex	Zeichen	PAC Tastatur
94	5E	^	
95	5F	_	Shift+ -
96	60	`	
97	61	a	A
98	62	b	B
99	63	c	C
100	64	d	D
101	65	e	E
102	66	f	F
103	67	g	G
104	68	h	H
105	69	i	I
106	6A	j	J
107	6B	k	K
108	6C	l	L
109	6D	m	M
110	6E	n	N
111	6F	o	O
112	70	p	P
113	71	q	Q
114	72	r	R
115	73	s	S
116	74	t	T
117	75	u	U
118	76	v	V
119	77	w	W
120	78	x	X
121	79	y	Y
122	7A	z	Z
123	7B	{	Ctrl+7
124	7C		
125	7D	}	Ctrl+0
126	7E	~	
127	7F	DEL	
128	80	PAD	
129	81	HOP	
130	82	BPH	
131	83	NBH	
132	84	IND	
133	85	NEL	
134	86	SSA	
135	87	ESA	
136	88	HTS	
137	89	HTJ	
138	8A	VTS	
139	8B	PLD	
140	8C	PLU	

Dez	Hex	Zeichen	PAC Tastatur
141	8D	RI	
142	8E	SS2	
143	8F	SS3	
144	90	DCs	
145	91	PU1	
146	92	PU2	
147	93	STS	
148	94	CCH	
149	95	MW	
150	96	SPA	
151	97	EPA	
152	98	SOS	
153	99	SGCI	
154	9A	SCI	
155	9B	CSI	
156	9C	ST	
157	9D	OSC	
158	9E	PM	
159	9F	APC	
160	A0	NBSP	
161	A1	ı	
162	A2	ø	
163	A3	£	
164	A4	€	
165	A5	¥	
166	A6	Š	
167	A7	§	
168	A8	š	
169	A9	©	
170	AA	ª	
171	AB	«	
172	AC	¬	
173	AD	SHY	
174	AE	®	
175	AF	—	
176	B0	°	
177	B1	±	
178	B2	²	
179	B3	³	
180	B4	Ž	
181	B5	μ	
182	B6	¶	
183	B7	·	
184	B8	ž	
185	B9	¹	
186	BA	º	
187	BB	»	

Ergänzung

Dez	Hex	PAC Tastatur
222	DE	Starttaste
294	126	Alt+L
327	147	Shift+Pfeil links
328	148	Pfeil auf
329	149	Shift+Pfeil auf
331	14B	Pfeil links
333	14D	Pfeil rechts
335	14F	Shift+Pfeil rechts
336	150	Pfeil ab
337	151	Shift+Pfeil ab
338	152	INS Shift+INS
339	153	DEL

INDEX

-	157
*	158
/	159
+	156
> = <	147, 165
A1	95
Abbruch	44
Abfrage	
aktuelle Bahngeschwindigkeit	131
aktuellen Zeilennummer	55
Ablaufdiagramm	
PA-CONTROL in Grundstellung	47
Start Automatikbetrieb	45
Ablaufinterpreter	42
ABORT.An	93
ABS	164
ABS	164
Achsparameter laden aus einem Register	32, 121
ACOS	160
ADi	203
AD-Wandler	
Abtastrate	227
Aktiviere	
Normalpositioniermodus	241
AND-LD	177
ASIN	160
ATAN	160
Automatikbetrieb	
Ablauf Störung	47
Automatikbetrieb mit STOP	46
Betrag	
einer Ganzzahl	164
einer Realzahl	164
BREAK	51
CANCEL	76
CANOpen-Bus	288
Applikationsbeispiele	300
Bearbeitung von Prozessdatenobjekten	296
Bearbeitung von Servicedatenobjekten	294
Kontrolle von Gerätefehlern	293
Netzwerk-Management-Befehle	289
CASE.CANCEL	81
CASE.JMP	60
CASE.JMP.N	60
CASE.RUN	77
CASE.SLEEP	79
CASE.STORE.N	216
CASE.SUB.N	68
CHN	189
CNTn	
=Ni	220
CNTn.i.Nm	221
CNTn.INIT.i	219
COPY	193
COS	160
COS / ACOS	161
DAi	206
DEC	71
Direkte Adressierung	153
N-Register	153
R-Register	153
Division	159
Encoderposition übernehmen	114

END	55
Endschalterüberwachung	
einschalten	112
ERROROFF	84
ERRORON	84, 85
F	99
Fahre Teilstrecke mit Start-Stop	104
FB	
= n	125
FB.i	
= n / FB.i	
=Rn	125
Fehlermanagement	
aktivieren	84
deaktivieren	84
ERROROFF	84
ERRORON	84
FRAC	164
G01	126
G01 A1	
=(POS) A2	
=(POS).....	128
G100	98
G100	98
G100.B.n	124
G101	137
G11	188
G123	101
G123Q	103
G123Q	103
G140	88
G141	89
G142	111
G143	112
G144	122
G145	123
G150	104
G16?	134
G170	207
G171	209
G172	210
G173f	211
G18	225
G180.	226
G181	228
G182	229
G183	232
G21	57
G210	236
G211	237
G212	239
G213	241
G22	67
G221	242
G222	244
G230	246
G231	248
G25.An	94
G26	106
G29	108
G2xx	235
G401.1.	254
G421.1.	251
G422.1.	252
G423.1.	253
G4xx	250

G500.	256
G501	258
G502	259
G503 / 504	260
G503/G504	260
G510	262
G510	262
G511	263
G511.	263
G512	264
G515	265
G520	266
G521	268
G531	270
G532	272
G533	273
G534	275
G540	277
G541	278
G542	279
G5xx	255
G600	282
G601	283
G602	284
G603	286
G604	287
G6xx	281
G701	289
G702	290
G703	290
G704	291
G705	291
G711	292
G721 / G722	293
G730	294
G731	295
G74x	298
G75x	299
G90	96
G90	96
G91	97
Ganzzahlregister	
Wertebereich	13
GET	199
GETI	200
GETM	148
Grundstellung	43
In.m	141
In.m	141
INC	70
Indirekte Adressierung	153
N-Register	153
R-Register	153
INT	163
INT	163
Interpolation	
Bedienung von Ausgängen vor dem Start	134
Bedienung von Ausgängen während der Interpolation	135
IPOEND	130
JMP	56
JMP-LINE.Ni	59
JMP-LINE-IPO.Ni	136
Lade - Register	55
LD, AND, OUT, NOT	171
LD, OR, OUT, NOT	172
LENGTH	196

Linearinterpolation	126
Liste der Systemmerker	15
Logische Verknüpfungen	
Befehl AND	168
Befehl AND-LD	169
Befehl LD	168
Befehl OR	168
Befehl OR-LD	169
Befehl OUT	170
komplexe logische -Verknüpfungen	179
mit Ausgängen	167
mit Eingängen	167
mit Merkern	167
Mn	
=m	144
Mn.M	146
Mn:=m	144
MODBUS	186
Motorstrom verändern	137
Ni	
=CNTn	220
=LINE	55
=PEAn / Ri	
=PEAn	113
=PROGSTAT	83
=XOR.Sn	197
Ni.Wert	182
Ni/Ri	
=MODBUS	186, 187
Nn	
=16#yyy	155
=2#xxxxxxx	155
=SNn	183
OFF	
An	90
On	
=m	143
ON.An	91
On.m	142
On:=m	143
OR-LD	178
PARAMETER.XX.Ai	117
POS	192
Position auf Null setzen	106
Positionsbedingter	
Unterprogrammaufruf	244
Positionsbedingter Sprung	242
Programm in Grundstellung	43
Programmierhinweise	
Abbruch	44
START nach STOP	44
STÖRUNG	44
Programmnamen	
Länge	48
Programmstatus	
Standardabfrage	83
Programmstruktur	48
Prüfe	
Zeichenübernahme im Hintergrund abgeschlossen	189
PUT	201
PUTI	202
RAn.m	
=Ri	120
RAn.m	
=Ri	32
RAn.m	

=Ri	121
Realzahlregister	
Wertebereich	13
Ri	
=PARAMETER.XX.Ai	119
=RA _n .m	121
R _n	
= / N _n	
= 154	
=AN / N _n	
=AN	110
=ENC _n / Ni	
=ENC _n	114
=FA _n	105
=FB	131
=FB _{AN}	133
=SR _n	184
=TCi.Parametername	224
R _n	
=FB _{AN}	133
R _n /N _n :=	34, 154
RUN	73
S0	
= CHN	191
=TIME	152
Schleppfehler	
LV-servoTEC	113
PA-CONTROL MP	113
SET / RES	173
SIN	160
SIN / ASIN	160
SLEEP	75
Sn	
= Si	190
=COPY	195
=Ni / S _n	
=Ri	198
Sprungmarke	
Länge	56
Sprungverteiler	60
SQRT	163
SSIn.i.R _m / SSIn.i.N _m	116
Standard Unterprogrammaufruf	63
START	52
START nach STOP	44
START.A _n	92
STOP	54
STOP.A _n	92
STORE f	213
STOREEND / STORESTART	215
STÖRUNG	44
String	
zusammenfügen	196
String zusammenfügen	196
SUB.name	63
SUB.N _n	66
SUB.S _x	65
Suche Position eines Zeichens	192
Systemfunktionen	43
Abbruch	44
Programm bei START	43
Programm bei START nach STOP	43
Programm bei STÖRUNG	43
Programm bei STOP	43
START nach STOP	44
STÖRUNG	44

T	149, 150
TAN	160
TAN / ATAN	162
TCn.Parametername	
=	223
Temperaturregelung	222
Parameter	222
TIME	
=	151
Übergabeparameter	257
Übersicht Befehlssatz	30
Übertrage Inhalt eines N-Registers	
in den lokalen Zeichenpuffer	38, 201
Übertrage Inhalt lokalen Zeichenpuffer S0	
in ein Ganzzahlregister	38, 199
UND, ODER, EXKLUSIV-ODER	181, 185
Unterprogrammaufruf	
Standard	62
Vergleiche	
Befehlsformen	165
komplexe Beispiele	166
Verschachtelungen	64
Wahl des Datenkanals	256
Wandle Zeichen	38, 193
XOR	176
Zeichenübernahme	
im Hintergrund	273, 275
in den lokalen Zeichenpuffer	37, 40, 272