

MELFA

Industrieroboter

Bedienungs- und
Programmieranleitung

Steuergeräte
CR1/CR2/CR2A/CR2B/CR3

Bedienungs- und Programmieranleitung
Steuergeräte CR1/CR2/CR2A/CR2B/CR3
Artikel-Nr.: 140015 F

Version			Änderungen / Ergänzungen / Korrekturen
A	08/2001	pdp-gb	—
B	09/2002	pdp-gb	Abschn. 3.2: JOG-Betriebsarten Kap. 6: Neue Befehle Kap. 7: Detaillierte Beschreibung der Roboterstatusvariablen Kap. 8: Detaillierte Beschreibung der Funktionen Kap. 9: Detaillierte Beschreibung der Parameter
C	01/2004	pdp-gb	Allgemein: Anpassung an die Software-Version J1 Kap. 6: Neue Befehle Kap. 7: Neue Roboterstatusvariablen Abschn. 9.19: Umschaltung zwischen RAM- und ROM-Modus Abschn. A.1.1: Überarbeitung der Fehlerbeschreibungen Neue Fehlercodes
D	07/2004	pdp-gb	Allgemein: Anpassung an die Software-Version J4 Kap. 6: Neue Befehle Kap. 7: Neue Roboterstatusvariablen Kap. 11: Neu Abschn. A.1.1: Überarbeitung der Fehlerbeschreibungen Neue Fehlercodes
E	02/2005	pdp-gb	Abschn. 4.4: Korrektur des Anweisungsbeispiels Abschn. 10.3: Darstellung des Anschlusses der NOT-HALT-Kreise
F	06/2006	pdp-gb	Allgemein: Anpassung an die Software-Version K4 Kap. 7: Neue Roboterstatusvariablen Kap. 8: Ergänzung der Funktionen durch die Funktion Warmlaufbetrieb und die Funktion zum Durchfahren singulärer Punkte Abschn. A.1.1: Überarbeitung der Fehlerbeschreibungen

Zu diesem Handbuch

Die in diesem Handbuch vorliegenden Texte, Abbildungen, Diagramme und Beispiele dienen ausschließlich der Erläuterung zur Installation, Bedienung und zum Betrieb der in diesem Handbuch beschriebenen Steuergeräte.

Sollten sich Fragen bezüglich Installation und Betrieb der in diesem Handbuch beschriebenen Geräte ergeben, zögern Sie nicht, Ihr zuständiges Verkaufsbüro oder einen Ihrer Vertriebspartner (siehe Umschlagseite) zu kontaktieren.

Aktuelle Informationen sowie Antworten auf häufig gestellte Fragen erhalten Sie über die Internet-Adresse www.mitsubishi-automation.de.

Die MITSUBISHI ELECTRIC EUROPE B.V. behält sich vor, jederzeit technische Änderungen dieses Handbuchs ohne besondere Hinweise vorzunehmen.

© 06/2006

Sicherheitshinweise

Zielgruppe

Dieses Handbuch richtet sich ausschließlich an anerkannt ausgebildete Elektrofachkräfte, die mit den Sicherheitsstandards der Automatisierungstechnik vertraut sind. Projektierung, Installation, Inbetriebnahme, Wartung und Prüfung der Roboter nebst Zubehör dürfen nur von einer anerkannt ausgebildeten Elektrofachkraft, die mit den Sicherheitsstandards der Automatisierungstechnik vertraut ist, durchgeführt werden.

Bestimmungsgemäßer Gebrauch

Die Steuergeräte CR1, CR2, CR2A, CR2B und CR3 sind nur für die Einsatzbereiche vorgesehen, die in diesem Handbuch beschrieben sind. Achten Sie auf die Einhaltung aller im Handbuch angegebenen Kenndaten.

Jede andere darüber hinausgehende Verwendung oder Benutzung gilt als nicht bestimmungsgemäß.

Sicherheitsrelevante Vorschriften

Bei der Projektierung, Installation, Inbetriebnahme, Wartung und Prüfung der Geräte müssen die für den spezifischen Einsatzfall gültigen Sicherheits- und Unfallverhütungsvorschriften beachtet werden.



ACHTUNG:

Im Lieferumfang des Roboters ist ein Sicherheitstechnisches Handbuch enthalten. Dieses Handbuch behandelt alle sicherheitsrelevanten Details zu Aufstellung, Inbetriebnahme und Wartung. Vor einer Aufstellung, Inbetriebnahme oder der Durchführung anderer Arbeiten mit oder am Roboter ist dieses Handbuch unbedingt durchzuarbeiten. Alle darin aufgeführten Angaben sind zwingend zu beachten! Sollte dieses Handbuch nicht im Lieferumfang enthalten sein, wenden Sie sich bitte umgehend an Ihren Mitsubishi-Vertriebspartner.

Darüber hinaus müssen folgende Vorschriften (ohne Anspruch auf Vollständigkeit) beachtet werden:

- VDE-Vorschriften
- Brandverhütungsvorschriften
- Unfallverhütungsvorschriften

Erläuterung zu den Gefahrenhinweisen

In diesem Handbuch befinden sich Hinweise, die wichtig für den sachgerechten sicheren Umgang mit dem Roboter sind.

Die einzelnen Hinweise haben folgende Bedeutung:



GEFAHR:

Bedeutet, dass eine Gefahr für das Leben und die Gesundheit des Anwenders, z. B. durch elektrische Spannung, besteht, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.



ACHTUNG:

Bedeutet eine Warnung vor möglichen Beschädigungen des Roboters, seiner Peripherie oder anderer Sachwerte, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

Symbolik des Handbuchs

Verwendung von Hinweisen

Hinweise auf wichtige Informationen sind besonders gekennzeichnet und werden folgenderweise dargestellt:

HINWEIS

| Hinweistext

Verwendung von Nummerierungen in Abbildungen

Nummerierungen in Abbildungen werden durch weiße Zahlen in schwarzem Kreis dargestellt und in einer anschließenden Tabelle unter der gleichen Zahl erläutert, z. B.:

① ② ③ ④

Verwendung von Handlungsanweisungen

Handlungsanweisungen sind Schrittfolgen bei der Inbetriebnahme, Bedienung, Wartung u. Ä., die genau in der aufgeführten Reihenfolge durchgeführt werden müssen.

Sie werden fortlaufend durchnummeriert (schwarze Zahlen in weißem Kreis):

- ① Text
- ② Text
- ③ Text

Verwendung von Fußnoten in Tabellen

Hinweise in Tabellen werden in Form von Fußnoten unterhalb der Tabelle (hochgestellt) erläutert. An der entsprechenden Stelle in der Tabelle steht ein Fußnotenzeichen (hochgestellt).

Liegen mehrere Fußnoten zu einer Tabelle vor, werden diese unterhalb der Tabelle fortlaufend nummeriert (schwarze Zahlen in weißem Kreis, hochgestellt):

- ① Text
- ② Text
- ③ Text

Inhaltsverzeichnis

1	Einführung	
1.1	Grundlegende Sicherheitshinweise	1-1
1.2	Die ersten Schritte	1-3
2	Funktionen	
2.1	Steuergerät	2-1
2.1.1	Bedien- und Signalelemente des Steuergerätes	2-1
2.1.2	LED-Anzeige	2-4
2.2	Teaching Box	2-5
2.2.1	Display-Anzeigen und Funktionen	2-5
2.2.2	Bedienelemente der Teaching Box	2-7
2.3	Betriebsrechte	2-10
2.4	Bewegungs- und Steuerfunktionen	2-12
3	Bedienung und Programmierung	
3.1	Bedienung der Teaching Box.	3-1
3.1.1	Menübaum	3-1
3.1.2	Menüpunkt auswählen	3-2
3.2	Roboter im JOG-Betrieb bewegen	3-4
3.2.1	JOG-Betriebsarten	3-4
3.2.2	JOG-Geschwindigkeit einstellen	3-6
3.2.3	Gelenk-JOG-Betrieb	3-6
3.2.4	Werkzeug-JOG-Betrieb	3-7
3.2.5	XYZ-JOG-Betrieb	3-7
3.2.6	3-Achsen-XYZ-JOG-Betrieb	3-8
3.2.7	Kreis-JOG-Betrieb	3-8
3.2.8	Kollisionsüberwachung im JOG-Betrieb	3-9

3.3	Werkzeugdaten umschalten	3-11
3.4	Handgreifer öffnen/schließen.	3-13
3.5	Handgreifer ausrichten	3-15
3.6	Programmierung	3-16
3.6.1	Roboterprogramm erstellen	3-16
3.6.2	Roboterprogramm editieren	3-19
3.6.3	Roboterprogramm testen	3-28
3.7	Servospannung ein-/ausschalten	3-31
3.8	Fehler zurücksetzen	3-34
3.9	Fehler temporär zurücksetzen.	3-35
3.10	Automatikbetrieb	3-36
3.10.1	Geschwindigkeit einstellen	3-36
3.10.2	Auswahl der Programmnummer	3-37
3.10.3	Starten des Automatikbetriebs	3-38
3.10.4	Stoppen des Automatikbetriebs	3-40
3.10.5	Fortsetzung des Automatikbetriebs aus dem Stoppzustand	3-40
3.10.6	Programm zurücksetzen	3-40
3.11	Programmverwaltungsfunktionen	3-42
3.11.1	Programmverzeichnis anzeigen	3-42
3.11.2	Programm schützen	3-43
3.11.3	Programm kopieren.	3-45
3.11.4	Programmnamen ändern	3-46
3.11.5	Programm löschen	3-47
3.12	Monitor-Funktionen	3-48
3.12.1	Monitor-Funktion für Eingangssignale	3-48
3.12.2	Monitor-Funktion für Ausgangssignale	3-49
3.12.3	Monitor-Funktion für Variable	3-50
3.12.4	Liste der aufgetretenen Fehlermeldungen	3-51
3.13	Zusatzfunktionen	3-52
3.13.1	Parameter einstellen	3-52
3.13.2	Alle gespeicherten Programme löschen	3-53
3.13.3	Gelenkbremsen lösen	3-54
3.13.4	Batteriezüher zurücksetzen	3-55
3.13.5	Batterie und Einschaltzeit anzeigen	3-56
3.13.6	Uhrzeit und Datum einstellen	3-57

4	MELFA-BASIC-IV-Programmierung	
4.1	Funktionsübersicht	4-1
4.2	Programmaufbau	4-2
4.2.1	Programmname	4-2
4.2.2	Anweisung	4-2
4.2.3	Variable	4-3
4.3	Steuerung der Roboterbewegung	4-5
4.3.1	Gelenk-Interpolation	4-5
4.3.2	Linear-Interpolation	4-7
4.3.3	Kreis-Interpolation	4-10
4.3.4	Kontinuierliche Bewegung	4-12
4.3.5	Beschleunigungs-/Bremszeit und Geschwindigkeit	4-14
4.3.6	Feinpositionierung	4-17
4.3.7	Verfahrweggenauigkeit	4-19
4.3.8	Hand- und Werkzeugsteuerung	4-21
4.4	Palettierung	4-23
4.5	Programmsteuerung	4-27
4.5.1	Verzweigungen und Wartezeit	4-27
4.5.2	Programmschleife	4-29
4.5.3	Interrupt	4-30
4.5.4	Unterprogramm	4-31
4.5.5	Timer	4-32
4.5.6	Stopp	4-33
4.6	Ein- und Ausgabe externer Signale	4-34
4.6.1	Eingangssignale	4-34
4.6.2	Ausgangssignale	4-35
4.7	Kommunikation	4-36
4.8	Ausdrücke und Operationen	4-38
4.8.1	Übersicht	4-38
4.8.2	Relative Konvertierung von Positionsdaten	4-40
4.9	Angehängte Anweisung	4-42
4.10	Multitask-Funktion	4-43
4.10.1	Beschreibung	4-43
4.10.2	Ausführung eines Multitasks	4-44
4.10.3	Betriebszustand eines Programmplatzes	4-45
4.10.4	Erstellung eines Multitask-Programms	4-48
4.10.5	Anwendung des Multitaskings	4-50
4.10.6	Beispiel zur Anwendung der Multitask-Funktion	4-51

5 MELFA-BASIC IV

5.1 Begriffserklärung5-1

5.1.1 Anweisung5-1

5.1.2 Angehängte Anweisung5-1

5.1.3 Zeilen5-2

5.1.4 Zeilennummern und Marken5-2

5.1.5 Zeichentypen5-3

5.1.6 Zeichen mit besonderer Bedeutung5-4

5.1.7 Datentypen5-5

5.1.8 Konstanten5-6

5.1.9 Variablen5-11

5.1.10 Externe Variablen5-15

5.1.11 Logische Werte5-23

5.1.12 Funktionen5-24

5.1.13 Operanden5-28

5.1.14 Rangfolge von Operationen5-30

5.1.15 Programmebenen5-30

5.1.16 Reservierte Wörter5-30

6 MELFA-BASIC-IV-Befehle

6.1 Allgemeine Hinweise6-1

6.1.1 Beschreibung des verwendeten Formats6-1

6.2 Übersicht der MELFA-BASIC-IV-Befehle6-2

6.2.1 Alphabetische Übersicht6-2

6.2.2 Anwendungsspezifische Übersicht6-5

6.3 Detaillierte Befehlsbeschreibung6-8

6.3.1 ACCEL (Accelerate)6-8

6.3.2 ACT (Act)6-10

6.3.3 BASE (Base)6-12

6.3.4 CALLP (Call P)6-14

6.3.5 CHRSRCH (Character search)6-16

6.3.6 CLOSE (Close)6-17

6.3.7 CLR (Clear)6-18

6.3.8 CMP JNT (Compliance Joint)6-20

6.3.9 CMP POS (Compliance Posture)6-22

6.3.10 CMP TOOL (Compliance Tool)6-25

6.3.11 CMP OFF (Compliance OFF)6-28

6.3.12 CMPG (Compliance Gain)6-29

6.3.13 CNT (Continuous)6-30

6.3.14	COLCHK (Col Check)	6-33
6.3.15	COLLVL (Col Level).	6-37
6.3.16	COM OFF (Communication OFF).	6-39
6.3.17	COM ON (Communication ON).	6-40
6.3.18	COM STOP (Communication STOP)	6-41
6.3.19	DEF ACT (Define act)	6-42
6.3.20	DEF ARCH (Define Arch)	6-45
6.3.21	DEF CHAR (Define Character)	6-47
6.3.22	DEF FN (Define function)	6-48
6.3.23	DEF INTE/FLOAT/DOUBLE (Define Integer/Float/Double)	6-50
6.3.24	DEF IO (Define IO)	6-52
6.3.25	DEF JNT (Define Joint)	6-54
6.3.26	DEF PLT (Define pallet)	6-55
6.3.27	DEF POS (Define Position)	6-58
6.3.28	DIM (Dim)	6-59
6.3.29	DLY (Delay)	6-61
6.3.30	ERROR (Error)	6-63
6.3.31	END (End)	6-64
6.3.32	FINE (Fine)	6-65
6.3.33	FOR-NEXT (For-Next)	6-67
6.3.34	FPRM (FPRM).	6-69
6.3.35	GETM (Get Mechanism)	6-70
6.3.36	GOSUB (Go Subroutine).	6-72
6.3.37	GOTO (Go To)	6-73
6.3.38	HLT (Halt)	6-74
6.3.39	HOPEN/HCLOSE (Hand Open/Hand Close)	6-75
6.3.40	IF ... THEN ... ELSE (If Then Else)	6-77
6.3.41	INPUT # (Input)	6-80
6.3.42	JOVRD (J Override)	6-81
6.3.43	JRC (Joint Roll Change)	6-82
6.3.44	LABEL (Label)	6-85
6.3.45	LOADSET (Load Set)	6-86
6.3.46	MOV (Move)	6-88
6.3.47	MVA (Move Arch)	6-90
6.3.48	MVC (Move C)	6-92
6.3.49	MVR (Move R)	6-94
6.3.50	MVR2 (Move R2)	6-96
6.3.51	MVR3 (Move R3)	6-98
6.3.52	MVS (Move S)	6-100
6.3.53	OADL (Optimum Acceleration/Deceleration)	6-103
6.3.54	ON COM GOSUB (ON Communication Go Subroutine)	6-105

6.3.55	ON GOSUB (ON GOSUB)	6-107
6.3.56	ON ... GOTO (On Go To)	6-109
6.3.57	OPEN (Open)	6-111
6.3.58	OVRD (Override)	6-113
6.3.59	PLT (Pallet)	6-115
6.3.60	PREC (Precision)	6-118
6.3.61	PRINT (Print)	6-119
6.3.62	PRIORITY (Priority)	6-121
6.3.63	RELM (Release Mechanism)	6-122
6.3.64	REM (Remarks)	6-123
6.3.65	RESET ERR (Reset Error)	6-124
6.3.66	RETURN (Return)	6-125
6.3.67	SELECT CASE	6-127
6.3.68	SERVO (Servo)	6-129
6.3.69	SKIP (Skip)	6-130
6.3.70	SPD (Speed)	6-131
6.3.71	TITLE (Title)	6-133
6.3.72	TOOL (Tool)	6-134
6.3.73	TORQ (Torque)	6-136
6.3.74	WAIT (Wait)	6-138
6.3.75	WHILE ~ WEND (While End)	6-139
6.3.76	WTH (With)	6-140
6.3.77	WTHIF (With If)	6-141
6.3.78	XCLR (X Clear)	6-142
6.3.79	XLOAD (X Load)	6-143
6.3.80	XRST (X Reset)	6-144
6.3.81	XRUN (X Run)	6-145
6.3.82	XSTP (X Stop)	6-147
6.3.83	SUBSTITUTE (Substitute)	6-148

7 Roboterstatusvariablen

7.1	Allgemeine Hinweise	7-1
7.1.1	Beschreibung des verwendeten Formats	7-1
7.2	Detaillierte Variablenbeschreibung	7-2
7.2.1	C_DATE	7-2
7.2.2	C_MAKER	7-3
7.2.3	C_MECHA	7-3
7.2.4	C_PRG	7-4
7.2.5	C_TIME	7-5
7.2.6	C_USER	7-5

7.2.7	J_COLMXL	7-6
7.2.8	J_CURR	7-8
7.2.9	J_ECURR	7-9
7.2.10	J_FBC/J_AMPFBC	7-10
7.2.11	J_ORIGIN	7-11
7.2.12	M_ACL/M_DACL/M_NACL/M_NDAACL/M_ACLSTS	7-12
7.2.13	M_BRKCQ	7-13
7.2.14	M_BTIME	7-14
7.2.15	M_CMPDST	7-15
7.2.16	M_CMPLMT	7-16
7.2.17	M_COLSTS	7-17
7.2.18	M_CSTP	7-18
7.2.19	M_CYS	7-19
7.2.20	M_DIN/M_DOUT	7-20
7.2.21	M_ERR/M_ERRLVL/M_ERNO	7-21
7.2.22	M_EXP	7-22
7.2.23	M_FBD	7-23
7.2.24	M_G	7-25
7.2.25	M_HNDCQ	7-25
7.2.26	M_IN/M_INB/M_INW	7-26
7.2.27	M_JOVRD/M_NJOVRD/M_OPOVRD/M_OVRD/M_NOVRD	7-27
7.2.28	M_LDFACT	7-28
7.2.29	M_LINE	7-30
7.2.30	M_MODE	7-31
7.2.31	M_ON/M_OFF	7-31
7.2.32	M_OPEN	7-32
7.2.33	M_OUT/M_OUTB/M_OUTW	7-33
7.2.34	M_PI	7-34
7.2.35	M_PSA	7-35
7.2.36	M_RATIO	7-36
7.2.37	M_RDST	7-37
7.2.38	M_RUN	7-38
7.2.39	M_SETADL	7-39
7.2.40	M_SKIPCQ	7-41
7.2.41	M_SPD/M_NSPD/M_RSPD	7-42
7.2.42	M_SVO	7-43
7.2.43	M_TIMER	7-44
7.2.44	M_TOOL	7-45
7.2.45	M_UAR	7-47
7.2.46	M_WAI	7-48

7.2.47	M_WUPOV	7-49
7.2.48	M_WUPRT	7-50
7.2.49	M_WUPST	7-51
7.2.50	P_BASE/P_NBASE	7-52
7.2.51	P_COLDIR	7-53
7.2.52	P_CURR	7-55
7.2.53	P_FBC	7-56
7.2.54	P_SAFE	7-57
7.2.55	P_TOOL/P_NTOOL	7-58
7.2.56	P_ZERO	7-59

8 Funktionen

8.1	Allgemeine Hinweise	8-1
8.1.1	Beschreibung des verwendeten Formats	8-1
8.2	Detaillierte Funktionsbeschreibung	8-2
8.2.1	ABS	8-2
8.2.2	ALIGN	8-3
8.2.3	ASC	8-4
8.2.4	ATN/ATN2	8-5
8.2.5	BIN\$	8-6
8.2.6	CALARC	8-7
8.2.7	CHR\$	8-9
8.2.8	CINT	8-9
8.2.9	CKSUM	8-10
8.2.10	COS	8-11
8.2.11	CVI	8-12
8.2.12	CVS	8-13
8.2.13	CVD	8-14
8.2.14	DEG	8-15
8.2.15	DIST	8-16
8.2.16	EXP	8-16
8.2.17	FIX	8-17
8.2.18	FRAM	8-18
8.2.19	HEX\$	8-20
8.2.20	INT	8-21
8.2.21	INV	8-22
8.2.22	JTOP	8-22
8.2.23	LEFT\$	8-23
8.2.24	LEN	8-24
8.2.25	LN	8-24

8.2.26	LOG	.8-25
8.2.27	MAX	.8-25
8.2.28	MID\$.8-26
8.2.29	MIN	.8-27
8.2.30	MIRROR\$.8-27
8.2.31	MKI\$.8-28
8.2.32	MKS\$.8-29
8.2.33	MKD\$.8-30
8.2.34	POSCQ	.8-30
8.2.35	POSMID	.8-31
8.2.36	PTOJ	.8-32
8.2.37	RAD	.8-33
8.2.38	RDFL1	.8-34
8.2.39	RDFL2	.8-35
8.2.40	RND	.8-36
8.2.41	RIGHT\$.8-37
8.2.42	SETFL1	.8-38
8.2.43	SETFL2	.8-40
8.2.44	SETJNT	.8-41
8.2.45	SETPOS	.8-43
8.2.46	SGN	.8-45
8.2.47	SIN	.8-45
8.2.48	SQR	.8-46
8.2.49	STRPOS	.8-46
8.2.50	STR\$.8-47
8.2.51	TAN	.8-47
8.2.52	VAL	.8-48
8.2.53	ZONE	.8-49
8.2.54	ZONE2	.8-51

9 Parameter

9.1	Allgemeines	.9-1
9.2	Bewegungsparameter	.9-2
9.3	Signalparameter	.9-15
9.4	Betriebsparameter	.9-17
9.5	Befehlsparameter	.9-21
9.6	Kommunikationsparameter	.9-27
9.7	Standard-Werkzeugkoordinaten	.9-30
9.7.1	Aufbau der Werkzeugdaten	.9-30

9.8	Standard-Basiskoordinaten	9-34
9.8.1	Aufbau der Basiskoordinatendaten	9-34
9.9	Benutzerdefinierter Bereich	9-36
9.10	Verfahrwegbegrenzungsebene	9-38
9.11	Automatische Rückkehr	9-39
9.12	Automatischer Programmstart nach dem Einschalten	9-42
9.13	Handgreifer	9-44
9.14	Handgreiferzustand nach Initialisierung	9-45
9.15	Ausgangsbitmuster	9-47
9.16	Kommunikationseinstellungen	9-49
9.16.1	Allgemeine Beschreibung	9-49
9.16.2	Datenübertragung über die RS232C-Schnittstelle	9-50
9.17	Hand- und Werkstückbedingung	9-53
9.17.1	Optimale Beschleunigung/Abbremsung	9-53
9.17.2	Handgreiferzustand	9-54
9.17.3	Definition der Koordinatensysteme für die Hand- und Werkstückbedingungen.	9-55
9.18	Fehlermeldung bei Erreichen des singulären Punktes	9-57
9.19	ROM- und Highspeed-RAM-Modus	9-58
9.19.1	Übersicht	9-58
9.19.2	Umschaltung zwischen ROM- und RAM-Modus	9-63
9.19.3	Umschaltung in den ROM-Modus	9-64
9.19.4	Anzeigen im ROM-Modus	9-66
9.19.5	Programmeditierung im ROM-Modus	9-66
9.19.6	Umschaltung in den RAM-Modus	9-67
9.19.7	Umschaltung in den Highspeed-RAM-Modus	9-69
9.20	Warmlaufbetrieb	9-70
9.20.1	Funktionsbeschreibung	9-70
9.20.2	Aktivierung des Warmlaufbetriebs	9-70
9.20.3	Aktivierter Warmlaufbetrieb	9-71
9.20.4	Parameter, spezielle Ein- und Ausgänge und Statusvariablen im Warmlaufbetrieb	9-73
9.20.5	Ausführung des Warmlaufbetriebs	9-74
9.20.6	Wenn der Warmlaufbetrieb freigegeben ist	9-75
9.20.7	Umschaltung zwischen Normal- und Warmlaufbetrieb	9-76
9.20.8	Alarmer im Warmlaufbetrieb	9-78

9.21	Durchfahren eines singulären Punktes	9-79
9.21.1	Positionen singulärer Punkte, die durchfahren werden können	9-79
9.21.2	Betrieb, mit aktivierter Funktion zum Durchfahren singulärer Punkte	9-80
9.21.3	Aktivierung der Funktion zum Durchfahren singulärer Punkte	9-82
9.21.4	Funktion zum Durchfahren singulärer Punkte im JOG-Betrieb	9-82
9.21.5	Funktion zum Durchfahren singulärer Punkte beim Anfahren definierter Positionen	9-83
9.21.6	Funktion zum Durchfahren singulärer Punkte im Automatikbetrieb	9-83
9.21.7	TYPE (Type)	9-83

10 Externe Ein-/Ausgänge

10.1	Einteilung	10-1
10.1.1	Allgemeine Übersicht der Ein- und Ausgänge	10-2
10.2	Parallele Ein-/Ausgangsschnittstelle	10-3
10.2.1	Ein-/Ausgangsbelegung der parallelen Ein-/Ausgangsschnittstelle	10-9
10.2.2	Programmsteuerung durch externe Signale	10-32
10.3	NOT-HALT-Eingang	10-36
10.3.1	Steuergerät CR1	10-36
10.3.2	Steuergerät CR2	10-37
10.3.3	Steuergeräte CR2A und CR2B	10-38
10.3.4	Steuergerät CR3	10-39
10.3.5	Verhalten des Roboters bei Betätigung des NOT-AUS-Schalters	10-40

11 Programmiertechniken

11.1	Programmiertechniken für Einsteiger	11-2
11.1.1	Aufbau leicht verständlicher Programme	11-2
11.1.2	Verwaltung von Programmversionen	11-8
11.1.3	Änderung der Betriebsgeschwindigkeit in einem Programm	11-9
11.1.4	Transportüberwachung des Werkstücks	11-10
11.1.5	Positioniergenauigkeit	11-12
11.1.6	Zeitverzögerte Signalprüfung	11-13
11.1.7	Synchronisierung durch externe Eingangssignale	11-15
11.1.8	Programmübergreifende Verwendung von Daten	11-17
11.1.9	Überwachung der Abweichung von Soll- und Istposition	11-19
11.1.10	Verkürzung der Zykluszeit	11-21

11.2	Programmiertechniken für fortgeschrittene Einsteiger	11-23
11.2.1	Schnelle Anpassung an unterschiedliche Werkstückgeometrien	11-23
11.2.2	Vielseitige Anwendung der Palettierungsfunktion	11-25
11.2.3	Schreiben eines Kommunikationsprogramms	11-27
11.2.4	Reduzierung von geteachten Positionen	11-30
11.2.5	Verwendung einer P-Variablen in einem Zähler	11-32
11.2.6	Sensorgesteuerte Übertragung von Positionsdaten	11-33
11.3	Programmiertechniken für Fortgeschrittene	11-35
11.3.1	Einsatz eines Roboters als einfache SPS	11-35
11.3.2	Implementierung einer Abbildungsfunktion	11-38
11.3.3	Ausgabe ausgeführter Zeilen	11-41
11.3.4	Status bei Auftreten eines Fehlers speichern	11-42

A Anhang

A.1	Fehlerdiagnose	A-1
A.1.1	Übersicht der Fehlercodes	A-2

1 Einführung

Die in diesem Handbuch vorliegenden Texte, Abbildungen, Diagramme und Beispiele gelten für folgende Software-Versionen:

- Steuergeräte CR1/CR2/CR2A/CR2B/CR3 ab Software-Version K4
- Teaching Box R28TB ab Software-Version B2

1.1 Grundlegende Sicherheitshinweise

Der MELFA-Roboter ist nach dem neuesten Stand der Technik gebaut und betriebssicher ausgeführt. Ungeachtet dessen können von dem Roboter Gefahren ausgehen, wenn er nicht von geschultem oder zumindest eingewiesenem Personal betrieben wird oder unsachgemäß bzw. zu nicht bestimmungsgemäßem Gebrauch eingesetzt wird.

Dies betrifft insbesondere:

- **Gefahren für Leib und Leben des Benutzers oder Dritter**
- **Beeinträchtigungen des Roboters, anderer Maschinen und weiterer Sachwerte des Anwenders**



ACHTUNG:

Jede Person, die im Betrieb des Anwenders mit der Aufstellung, Inbetriebnahme, Bedienung, Wartung und Reparatur des Roboters beauftragt ist, muss neben der zum Roboter gehörenden Technischen Dokumentation besonders das mitgelieferte

SICHERHEITSTECHNISCHE HANDBUCH

gelesen und verstanden haben.



ACHTUNG:

Achten Sie strikt auf die Einhaltung aller Sicherheitsrichtlinien. Im Rahmen dieser einführenden Sicherheitshinweise werden folgende weitere Instruktionen gegeben:

Der Roboter darf nur von ausgebildetem und autorisiertem Bedienungspersonal betrieben und bedient werden.

Die Zuständigkeiten für die unterschiedlichen Tätigkeiten im Rahmen des Betriebes des Roboters müssen klar festgelegt und eingehalten werden, damit unter dem Aspekt der Sicherheit keine unklaren Kompetenzen auftreten.

Bei allen Arbeiten, die die Aufstellung, die Inbetriebnahme, das Rüsten, den Betrieb, Änderungen der Einsatzbedingungen und Betriebsweisen, Wartung, Inspektion und Reparatur betreffen, sind die in der Betriebsanleitung angegebenen Ausschaltprozeduren zu beachten.

Die Lage der NOT-AUS-Taster muss bekannt sein und die NOT-AUS-Taster müssen jederzeit zugänglich sein.



Es ist jede Arbeitsweise zu unterlassen, die die Sicherheit an der Maschine beeinträchtigt.

Der Bediener hat dafür zu sorgen, dass keine Personen an dem Roboter arbeiten, die nicht dazu autorisiert sind (z. B. auch durch Betätigung von Einrichtungen gegen unbefugtes Benutzen).

Das verwendende Unternehmen hat dafür zu sorgen, dass der Roboter immer nur in einwandfreiem Zustand betrieben wird.

Der Verwenderbetrieb sollte das zuständige Bedienungspersonal besonders schulen und dazu verpflichten, alle Wartungs- und Inspektionsarbeiten ausschließlich bei abgeschaltetem Roboter und ausgeschalteter Peripherie durchzuführen.



GEFAHR:

Das Steuergerät darf ausschließlich über einen Leistungsschalter an die Netzspannung angeschlossen werden. Bei Nichtbeachtung besteht die Gefahr eines elektrischen Schlages.

Eine detaillierte Beschreibung des Netzanschlusses finden Sie im Technischen Handbuch des Roboters.

1.2 Die ersten Schritte

Nachfolgend erhalten Sie eine Darstellung der ersten Schritte mit Ihrem MELFA-Roboter:

- ① **Roboter und Steuergerät auspacken**
- ② **Sicherheitstechnisches Handbuch lesen**
Vor der ersten Inbetriebnahme des Robotersystems lesen Sie das Sicherheitstechnische Handbuch.
- ③ **Kabel anschließen**
Verbinden Sie alle Kabel, wie im Technischen Handbuch beschrieben, und schließen Sie die Teaching Box an.
- ④ **Netzspannungsversorgung einschalten**
Schalten Sie die Netzspannungsversorgung für das Steuergerät über den POWER-Schalter ein.
- ⑤ **Selbsttest des Steuergerätes**
Das Steuergerät startet einen Selbsttest mit einer Dauer von ca. 5 Sekunden.
Sollte nach dem Selbsttest eine Fehlermeldung erscheinen, versuchen Sie den Fehler mit Hilfe der Fehlerbeschreibung im Anhang dieses Handbuches zu beheben.
- ⑥ **Teaching Box einschalten (Dreistufenschalter drücken !!!)**
Drücken Sie den Dreistufenschalter auf der Rückseite der Teaching Box in die Mittelstellung und schalten Sie anschließend die Teaching Box ein (ENABLE/DISABLE-Schalter auf ENABLE).
 - System einstellen (DATA-Methode)
Zur Abgleichung des Systems muss die Grundposition (Nullpunkt) des Roboterarms eingestellt werden. Die genaue Vorgehensweise entnehmen Sie dem Technischen Handbuch.
Anschließend schalten Sie die Netzspannung des Steuergerätes kurzzeitig aus und wieder ein, um eine Übernahme der eingegebenen Werte zu gewährleisten.
- ⑦ **Teach-Modus auswählen**
Wählen Sie den Menüpunkt „1. TEACH“ aus, indem Sie lediglich die vorgegebene Auswahl mit der [INP/EXE]-Taste bestätigen.
Es wird der Teach-Modus für Positionsdaten aufgerufen.
- ⑧ **Programmnummer eingeben**
Sie werden jetzt nach der Programmnummer gefragt, unter der Sie die Positionsdaten definieren möchten.
Geben Sie z. B. „1“ für die Programmnummer 1 ein und betätigen Sie anschließend die [INP/EXE]-Taste. Dann halten Sie die Taste [POS/CHAR] gedrückt und betätigen einmal die Taste [ADD].
- ⑨ **Roboter bewegen**
Halten Sie den Dreistufenschalter in Mittelstellung und betätigen Sie die [STEP/MOVE]-Taste. Betätigen Sie die [STEP/MOVE]-Taste erneut und halten Sie diese gedrückt. Wenn Sie nun zusätzlich eine der mittleren Tasten für die Achsenbewegung betätigen, wird sich der Roboter in der entsprechenden Achse bewegen.

⑩ Position definieren

Halten Sie zum Definieren (Speichern) einer Position die [STEP/MOVE]-Taste gedrückt und betätigen Sie zweimal die [ADD]-Taste. Die momentane Position des Roboters wird unter der in der Anzeige der Teaching Box dargestellten Positionsnummer gespeichert.

Verfahren Sie den Roboter zu einer weiteren Position. Betätigen Sie einmal die [+ /FORWD]-Taste auf der Teaching Box, um eine um „1“ höhere Positionsnummer für das Speichern der neuen Position zu wählen.

Statt über die [+ /FORWD]-Taste können Sie auch die Positionsnummer direkt über die Tasten der Teaching Box eingeben. Halten Sie jetzt wie zuvor die [STEP/MOVE]-Taste gedrückt und betätigen Sie erneut zweimal die [ADD]-Taste. Die derzeitige Position wird unter der gewählten Positionsnummer gespeichert.

⑪ Definierte Positionen zum Testen anfahren

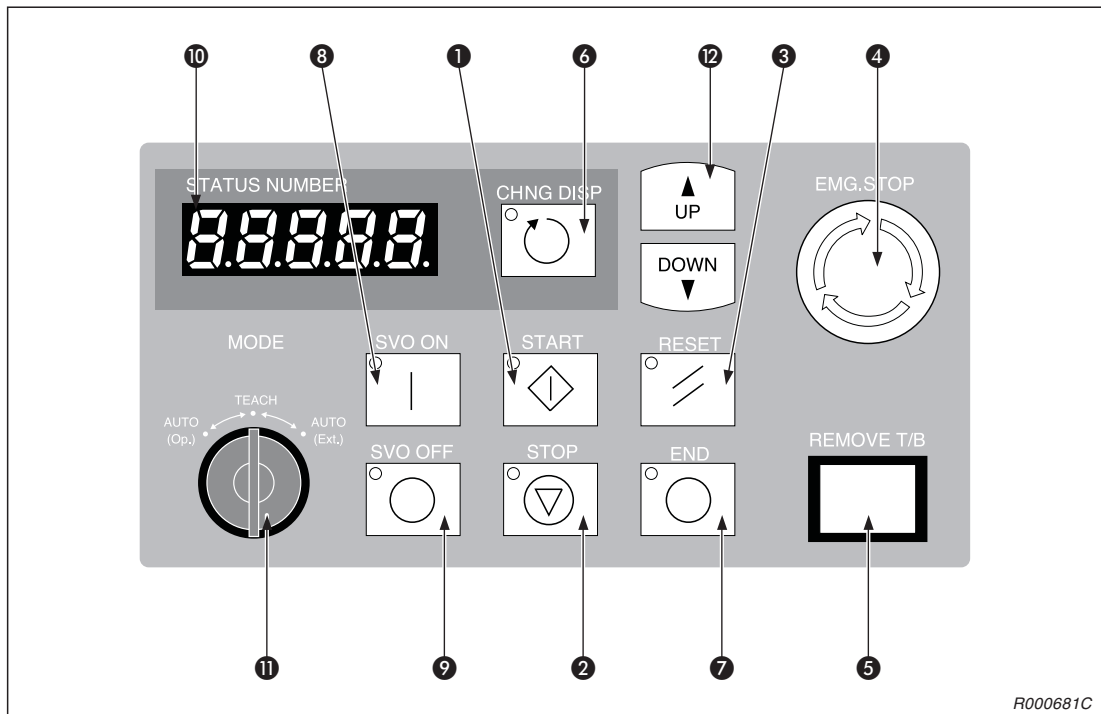
Geben Sie mit den Tasten der Teaching Box die Positionsnummer ein, die Sie zum Testen anfahren wollen oder betätigen Sie so oft die [+ /FORWD]-Taste bzw. [- /BACKWD]-Taste, bis die gewünschte Positionsnummer in der Anzeige der Teaching Box erscheint.

Wenn Sie die [STEP/MOVE]-Taste gedrückt halten und zusätzlich die [INP/EXE]-Taste betätigen, bewegt sich der Roboter zu der gewählten Position.

2 Funktionen

2.1 Steuergerät

2.1.1 Bedien- und Signalelemente des Steuergerätes



R000681C

Abb. 2-1: Frontansicht des Steuergerätes

Nr.	Bezeichnung	Funktion
1	START-Taster	Starten eines Programms und Betrieb des Roboters, kontinuierliche Abarbeitung des Programms Die grüne LED leuchtet während des Betriebs. Bei Ausführung eines Programms mit der Startbedingung „ALWAYS“ leuchtet die LED nicht.
2	STOP-Taster	Stoppen des Roboterprogramms Die Servoversorgungsspannung wird nicht abgeschaltet. Die rote LED leuchtet während eines Stopps (leuchtet, sobald ein Interrupt ausgeführt wird). Ein Programm mit der Startbedingung „ALWAYS“ wird nicht gestoppt.
3	RESET-Taster	Zurücksetzen eines haltenden Programms und Setzen auf den Anfang (nur bei Anzeige der Programmnummer), Quittierung eines Fehlercodes Die rote LED leuchtet bei anstehendem Fehler.
4	EMG.STOP-Schalter	Der Rastschalter dient dem NOT-HALT des Robotersystems. Wird der Schalter gedrückt, erfolgt die unmittelbare Abschaltung der Servoversorgungsspannung und der sich bewegende Roboter hält sofort an. Durch Rechtsdrehen wird der Schalter entriegelt und springt wieder heraus.

Tab. 2-1: Beschreibung der Bedien- und Signalelemente auf der Frontseite des Steuergerätes (1)

Nr.	Bezeichnung	Funktion	
5	REMOVE T/B-Tastschalter	Betätigen Sie den Schalter, wenn Sie die ausgeschaltete (disable) Teaching Box bei eingeschalteter Versorgungsspannung des Steuergerätes anschließen bzw. den Anschluss lösen möchten. Der Anschluss bzw. das Lösen des Anschlusses der Teaching Box muss innerhalb von 5 s nach Betätigung des Schalters vorgenommen werden. Ansonsten erfolgt eine Fehlermeldung.	
6	CHNG.DISP-Taster	Anzeigenwechsel auf dem Display des Steuergerätes in der Reihenfolge: Programmnummer → Zeilennummer → Übersteuerung Bei aufgetretenem Fehler erscheint: Programmnummer → Zeilennummer → Übersteuerung nur bei betätigtem Taster. Bei nicht betätigtem Taster erscheint die Fehlernummer.	
7	END-Taster	Stoppen des laufenden Programms in der letzten Zeile oder bei der END-Anweisung (zyklischer Betrieb) Die rote LED leuchtet bei zyklischem Betrieb. (Ein kontinuierlicher Betrieb wird unterbrochen.) Ab Software-Version J1 des Steuergerätes bewirkt ein erneutes Betätigen des [END]-Tasters die Rückkehr in den kontinuierlichen Betrieb.	
8	SVO.ON-Taster	Einschalten der Servoversorgungsspannung Die grüne LED leuchtet bei eingeschalteter Servoversorgungsspannung.	
9	SVO.OFF-Taster	Abschalten der Servoversorgungsspannung Die rote LED leuchtet bei ausgeschalteter Servoversorgungsspannung.	
10	STATUS.NUMBER-Anzeige	Anzeige von Alarm-, Fehlernummer, Programmnummer, Übersteuerungswert (%) usw. Buchstaben werden in vereinfachter Form dargestellt.	
11	MODE-Umschalter ^①	AUTO (Op.)	Ein Betrieb ist ausschließlich über das Steuergerät möglich. Der Betrieb über externe Signale oder die Teaching Box ist gesperrt.
		TEACH	Bei aktivierter Teaching Box ist ausschließlich ein Betrieb über die Teaching Box möglich. Der Betrieb über externe Signale oder das Steuergerät ist gesperrt.
		AUTO (Ext.)	Ein Betrieb ist ausschließlich über externe Signale möglich. Der Betrieb über die Teaching Box oder das Steuergerät ist gesperrt.
12	UP/DOWN-Taster	Scrollen der Anzeige (bei Programmnummern, Übersteuerungswerten und Fehlernummern)	

Tab. 2-1: Beschreibung der Bedien- und Signalelemente auf der Frontseite des Steuergerätes (2)



GEFAHR:

Beim Umschalten des [MODE]-Umschalters 11 am Steuergerät wird die Servoversorgungsspannung abgeschaltet. Achsen, die über keine Bremse verfügen, können aufgrund ihres Eigengewichtes unkontrolliert in den Endanschlag fallen. Es besteht Verletzungsgefahr. Damit die Servoversorgungsspannung eingeschaltet bleibt, gehen Sie beim Umschalten des [MODE]-Schalters wie auf der nächsten Seite beschrieben vor.

- ^① Soll die LED-Anzeige auf dem Steuergerät beim Umschalten des [MODE]-Schalters erhalten bleiben, ändern Sie Parameter OPDISP:
 0: Anzeige der Geschwindigkeitsübersteuerung (Grundeinstellung)
 1: aktuelle Anzeige beibehalten

Umschaltung von „TEACH“ auf „AUTO“

- ① Stellen Sie bei betätigtem Dreistufenschalter den [ENABLE/DISABLE]-Schalter der Teaching Box auf „DISABLE“.
- ② Stellen Sie bei betätigtem Dreistufenschalter den MODE-Schalter des Steuergerätes auf „AUTO“.
- ③ Geben Sie den Dreistufenschalter wieder frei.

Umschaltung von „AUTO“ auf „TEACH“

- ① Stellen Sie bei betätigtem Dreistufenschalter den [ENABLE/DISABLE]-Schalter der Teaching Box auf „DISABLE“.
- ② Stellen Sie bei betätigtem Dreistufenschalter den [MODE]-Schalter des Steuergerätes auf „TEACH“.
- ③ Stellen Sie bei betätigtem Dreistufenschalter den [ENABLE/DISABLE]-Schalter der Teaching Box auf „ENABLE“.

2.1.2 LED-Anzeige

Die Darstellung alphanumerischer Zeichen bei Programmnamen erfolgt auf der 7-Segment-LED-Anzeige des Steuergerätes in einer etwas vereinfachten Form.

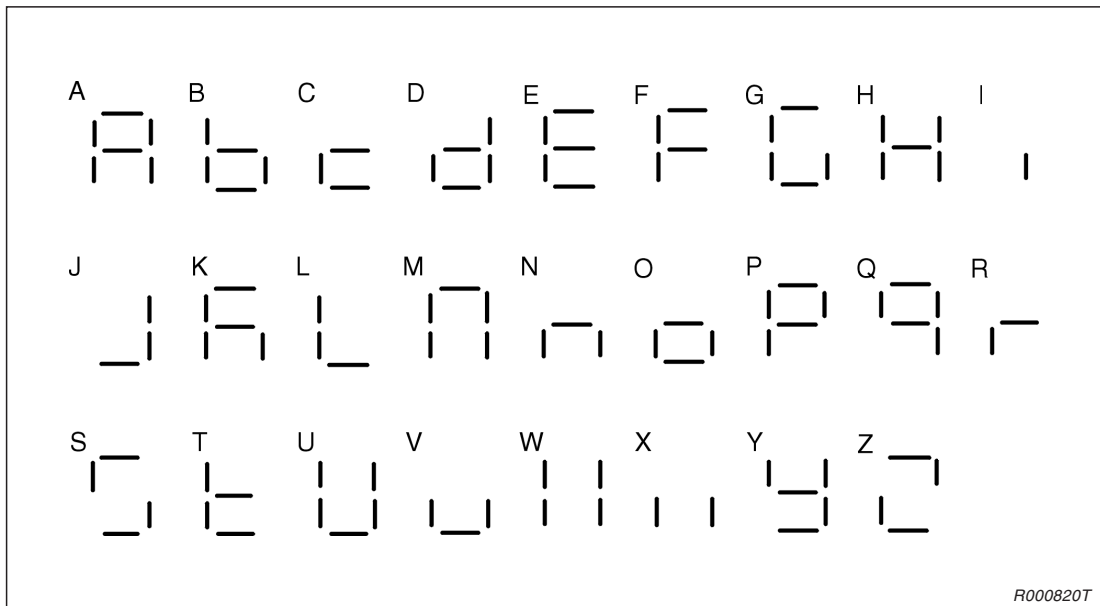


Abb. 2-2: Darstellung alphanumerischer Zeichen auf der STATUS-NUMBER-Anzeige

Der Buchstabe „P“ ist bei Programmnamen an der ersten Stelle fest vorgegeben. Es können maximal 4 weitere Zeichen angegeben werden. Achten Sie bei der Eingabe des Programmnamens darauf, dass die maximale Anzahl der Zeichen nicht überschritten wird. Es ist nicht möglich, Programmnamen mit mehr als 4 Zeichen über das Steuergerät aufzurufen. Ein Unterprogrammaufruf von Programmen mit längeren Programmnamen ist jedoch in der Roboter-Programmiersprache über den CALLP-Befehl möglich.

2.2 Teaching Box

2.2.1 Display-Anzeigen und Funktionen

Display	Funktion	Referenz
Eröffnungsbildschirm <pre>CRn-5xxVer.A3 RP-1AH Copyright (C) 1999 ANY KEY DOWN</pre>	Anzeige des Robotertyps und der Software-Version	Abschn. 3.1.1 „Bedienung der Teaching Box“
Hauptmenü <pre><MENU> 1 . TEACH 2 . RUN 3 . FILE 4 . MONI 5 . MAINT 6 . SET</pre>	Auswahl der folgenden Menüs	
Programmauswahl <pre><TEACH> (1) SELECT PROGRAM</pre>	Programmnummer auswählen oder ändern	Abschn. 3.6.1 „Roboterprogramm erstellen“
Programmstart <pre><RUN> 1 . SERVO 2 . CHECK</pre>	Servospannung EIN/AUS und Schrittbetrieb	Abschn. 3.7 „Servospannung ein-/ausschalten“
Dateifunktionen <pre><FILE> 1 . DIR 2 . COPY 3 . RENAME 4 . DELETE</pre>	Programmverzeichnis anzeigen	Abschn. 3.11.1 „Programmverzeichnis anzeigen“
	Programm schützen	Abschn. 3.11.2 „Programm schützen“
	Programm kopieren	Abschn. 3.11.3 „Programm kopieren“
	Programmnamen ändern	Abschn. 3.11.4 „Programmnamen ändern“
	Programm löschen	Abschn. 3.11.5 „Programm löschen“
Monitorfunktionen <pre><MONI> 1 . INPUT 2 . OUTPUT 3 . VAR 4 . ERROR 5 . REGISTER</pre>	Eingangssignale anzeigen	Abschn. 3.12.1 „Monitor-Funktion für Eingangssignale“
	Ausgangssignale anzeigen/einstellen	Abschn. 3.12.2 „Monitor-Funktion für Ausgangssignale“
	Variablen anzeigen	Abschn. 3.12.3 „Monitor-Funktion für Variable“
	Alarmliste anzeigen	Abschn. 3.12.4 „Liste der aufgetretenen Fehlermeldungen“
	Register anzeigen	Anwendung bei CC-Link-Optionen Benutzerhandbuch CC-Link-Schnittstelle

Tab. 2-2: Display-Anzeigen und Funktionen (1)

Display	Funktion	Referenz
Wartungsfunktionen <MAINT> 1 . PARAM 2 . INIT 3 . BRAKE 4 . ORIGIN 5 . POWER	Parameter anzeigen/einstellen	Abschn. 3.13.1 „Parameter einstellen“
	Speicher löschen	Abschn. 3.13.2 „Alle gespeicherten Programme löschen“
	Gelenkbremse lösen	Abschn. 3.13.3 „Gelenkbremsen lösen“
	Grundposition einstellen	Technisches Handbuch
	Batteriezüher zurücksetzen	Abschn. 3.10.4, 3.13.4 „Batteriezüher zurücksetzen“
	Batterie- und Einschaltzeit anzeigen	Abschn. 3.13.5 „Batterie und Einschaltzeit anzeigen“
Uhrzeit/Datum <SET> 1 . CLOCK	Uhrzeit und Datum anzeigen/einstellen	Abschn. 3.13.6 „Uhrzeit und Datum einstellen“

Tab. 2-2: Display-Anzeigen und Funktionen (2)

2.2.2 Bedienelemente der Teaching Box

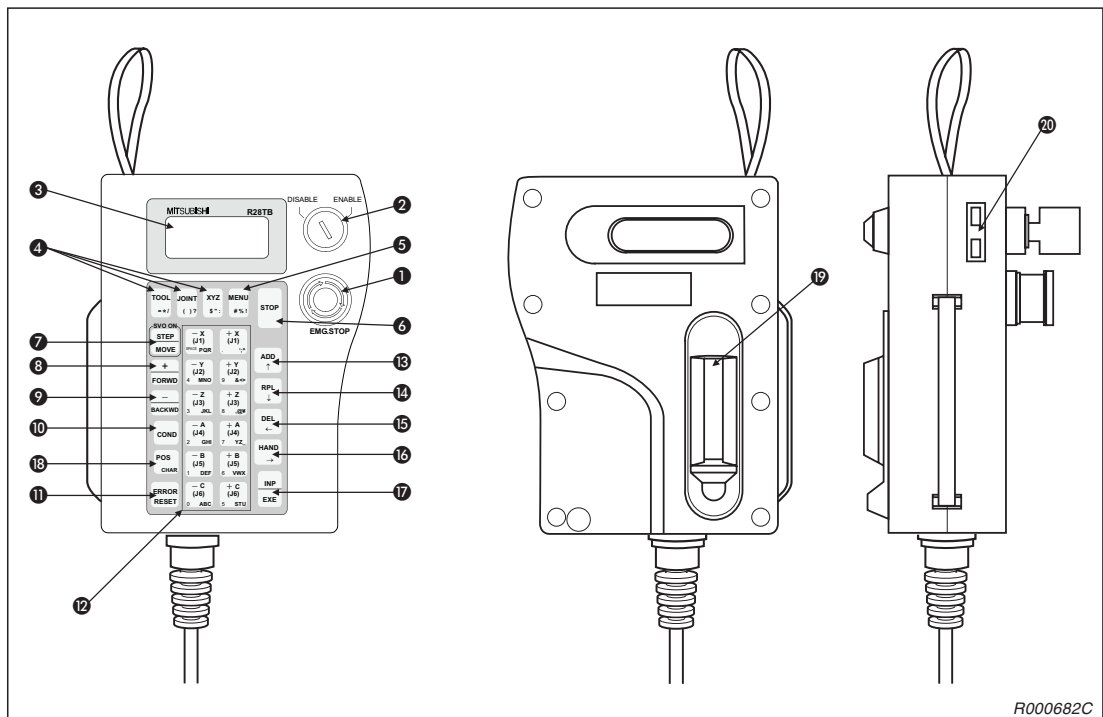













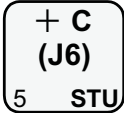






Abb. 2-3: Bedienelemente der Teaching Box

Nr.	Schalter / Taste	Beschreibung
1		Drucktaster mit Verriegelungsfunktion für NOT-HALT Nach Betätigung wird der Roboter unabhängig vom jeweiligen Betriebszustand sofort gestoppt. Durch Drehen der Drucktasterfläche wird der Taster entriegelt.
2		Freigabe der Steuerung über die Teaching Box Bringen Sie den Schalter in die Stellung „ENABLE“, um die Steuerung über die Teaching Box zu übernehmen. Wenn die Teaching Box aktiv ist, kann weder über das Bedienfeld des Steuergerätes noch von extern in die Steuerung eingegriffen werden. Die Freigabe des Betriebs kann auch im gesperrten Zustand in Abhängigkeit der Anzeige oder des Übersteuerungswertes umgeschaltet werden. Stellen Sie den Schalter nach der Editierung auf „DISABLE“, um das aktuelle Programm zu speichern.
3	LCD-Anzeige	Auf der LCD-Anzeige (4 Zeilen × 16 Zeichen) wird das aktuell ausgewählte Programm oder der Betriebszustand des Roboters angezeigt.

Tab. 2-3: Bedienelemente der Teaching Box (1)

Nr.	Schalter / Taste	Beschreibung
4		Auswahl des Werkzeug-JOG-Betriebs
		Auswahl des Gelenk-JOG-Betriebs Rufen Sie den JOG-Betrieb einer Zusatzachse durch zweimalige Betätigung der Taste auf.
		Auswahl des XYZ-JOG-, 3-Achsen-XYZ-JOG-Betriebs oder Kreis-JOG-Betriebs Betätigen Sie die Taste, um den XYZ-JOG-Betrieb aufzurufen. Im Werkzeug- oder Gelenk-JOG-Betrieb wird durch zweimalige Betätigung der 3-Achsen-XYZ-JOG-Betrieb und durch dreimalige Betätigung der Kreis-JOG-Betrieb aufgerufen.
5		Rücksprung ins Hauptmenü Betätigen Sie die Taste nach der Editierung, um das Programm zu speichern.
6		Programmablauf und Roboterbewegung stoppen Die Taste hat die gleiche Funktion wie die STOP-Taste auf der Frontseite des Steuergerätes. Die Tastenfunktion ist unabhängig von der Stellung des [ENBL/DISABLE]-Schalters immer verfügbar.
7		Ausführen des JOG-Betriebs in Verbindung mit den JOG-Tasten 12, Ausführen von Anweisungsschritten in Verbindung mit der [INP/EXE]-Taste, Einschalten der Servoversorgungsspannung (bei gleichzeitiger Betätigung des Dreistufenschalters)
8		Ausführen von Vorwärtsschritten Anzeige der nächsten Programmzeile im Editiermodus, Zunahme der Übersteuerung in Verbindung mit der [STEP/MOVE]-Taste (auch bei ausgeschalteter Teaching Box möglich) Eingabe des Zeichen „+“ zur Programmerstellung
9		Ausführen von Rückwärtsschritten Anzeige der vorherigen Programmzeile im Editiermodus, Abnahme der Übersteuerung in Verbindung mit der [STEP/MOVE]-Taste (auch bei ausgeschalteter Teaching Box möglich) Eingabe des Zeichen „-“ zur Programmerstellung
10		Aufruf des Menüs zur Programmeditierung
11		Rücksetzen eines Alarms, Rücksetzen des Programms in Verbindung mit der [INP/EXE]-Taste

Tab. 2-3: Bedienelemente der Teaching Box (2)

Nr.	Schalter / Taste	Beschreibung
12	JOG-Tasten  bis 	Funktionstasten für JOG-Betrieb Im Gelenk-JOG-Betrieb können alle Gelenke einzeln bewegt werden. Im XYZ-JOG-Betrieb kann der Roboterarm an jeder der Koordinatenachsen entlang bewegt werden. Mit den Tasten erfolgt auch die Eingabe von Menüauswahlnummern oder Schrittnummern.
13		Zur Eingabe von Positionen oder Cursor nach oben bewegen (Teaching Box ab Version B1)
14		Zum Weiterblättern der Anzeige oder Cursor nach unten bewegen (Teaching Box ab Version B1)
15		Zum Löschen von Positionen oder Cursor nach links bewegen
16		Die [HAND]-Taste ermöglicht folgende Funktionen: <ul style="list-style-type: none"> ● in Verbindung mit der [+C/(J6)]- oder [-C/(J6)]-Taste das Öffnen und Schließen der ersten Greifhand, ● in Verbindung mit der [+B/(J5)]- oder [-B/(J5)]-Taste das Öffnen und Schließen der zweiten Greifhand, ● in Verbindung mit der [+A/(J4)]- oder [-A/(J4)]-Taste das Öffnen und Schließen der dritten Greifhand, ● in Verbindung mit der [+Z/(J3)]- oder [-Z/(J3)]-Taste das Öffnen und Schließen der vierten Greifhand oder ● Cursor nach rechts bewegen.
17		Zur Dateneingabe oder Schrittweitschaltung
18		Aufruf des Menüs zur Editierung von Positionsdaten und Wechsel zwischen Zahlen und Buchstaben beim Editieren von Positionsdaten usw.
19	Dreistufenschalter	Bei eingeschalteter Teaching Box wird der Servoantrieb bei nicht betätigtem oder durchgedrücktem Dreistufenschalter ausgeschaltet. Für ein Einschalten des Servoantriebes muss der Dreistufenschalter bis zur Mittelstellung betätigt sein. Ist der Servoantrieb während eines NOT-AUS oder einer Befehlsausführung ausgeschaltet, kann er durch den Dreistufenschalter nicht eingeschaltet werden.
20	LCD-Kontrasteinstellung	Zur Helligkeitseinstellung der LCD-Anzeige

Tab. 2-3: Bedienelemente der Teaching Box (3)

2.3 Betriebsrechte

Beim Anschluss mehrerer Geräte an das Steuergerät, z. B. Teaching Box und Personalcomputer, verfügt nur ein Gerät über die Betriebsrechte.

Zur Ausführung von Vorgängen, die den Roboter starten, z. B. ein Programmstart, benötigt ein Gerät die Betriebsrechte. Im Gegensatz dazu können alle Vorgänge, die den Roboter stoppen, z. B. ein Stopp-Befehl oder ein Ausschalten der Servoversorgung, aus Sicherheitsgründen auch ohne Betriebsrechte ausgeführt werden.

Schalter	T/B [ENABLE/DISABLE]	DISABLE			ENABLE		
	Steuergerät [MODE]	AUTO (Op.)	AUTO (Ext.)	TEACH	AUTO (Op.)	AUTO (Ext.)	TEACH
Betriebsrechte	T/B	—	—	—	— ^②	— ^②	●
	Steuergerät	●	—	—	— ^②	— ^②	—
	PC	—	● ^①	—	— ^②	— ^②	—
	Externes Signal	—	● ^①	—	— ^②	— ^②	—

Tab. 2-4: Einstellung der Betriebsrechte

- ① Erfolgt die Eingabe des Signals IOENA (Eingabe Betriebsrechte) über ein externes Gerät, besitzt das externe Signal die Betriebsrechte und die Betriebsrechte des PCs sind deaktiviert.
- ② Ist die Teaching Box auf „ENABLE“ gesetzt, erfolgt bei einer Einstellung der [MODE]-Taste auf die Stellung „AUTO“ die Fehlermeldung „5000“.

In folgender Tabelle sind die Vorgänge aufgeführt, die ein Betriebsrecht erfordern:

Vorgang	Betriebsrecht erforderlich	Funktion
Operation	●	Servo EIN
	—	Servo AUS
	●	Programmstart
	—	Programmstopp/Zyklusstopp
	●	Anwendungsinitialisierung (Programm zurücksetzen)
	—	Alarm zurücksetzen
	●	Geschwindigkeitsübersteuerung ändern (ist über die T/B immer möglich)
	—	Einlesen der Geschwindigkeitsübersteuerung
	●	Programmnummer ändern
	—	Programm-/Zeilennummer lesen
Ein-/Ausgangssignalfunktion	—	Eingangs-/Ausgangssignal lesen
	—	Ausgangssignal schreiben
	●	Spezielle Eingänge: Start, Reset, Servo EIN, Bremse EIN/AUS, manueller Moduswechsel, allgemeinen Ausgang zurücksetzen, Programmnummer festlegen, Zeilennummer festlegen, Geschwindigkeitsübersteuerung festlegen
	—	Spezielle Eingänge: Stopp, Servo AUS, kontinuierlicher Betrieb, Eingangssignal Betriebsrechte, Ausgabeanforderung Programmnummer, Ausgabeanforderung Zeilennummer, Ausgabeanforderung Geschwindigkeitsübersteuerung, Ausgabeanforderung Fehlernummer/numerische Eingabe
	—	Handsensor-/Handsteuersignal lesen
	●	Handsteuersignal schreiben
Programm- editierung ^①	—	Zeilennummer eingeben, lesen, aufrufen; Position hinzufügen, korrigieren, lesen; Variable schreiben, lesen
	●	Schrittweitschaltung, Ausführung
	—	Vorwärts-/Rückwärtsschritt
	●	Sprung, direkte Ausführung, JOG-Betrieb
Dateifunktion	—	Programmverzeichnis lesen, Programm schützen/kopieren/löschen/umbenennen/zurücksetzen
Wartungs- funktion	—	Parameter lesen, Uhrzeit einstellen/lesen, Betriebszeit lesen, Alarmliste lesen
	●	Grundposition einstellen, Parameter ändern

Tab. 2-5: Vorgänge und Betriebsrechte

^① Wird über ein Gerät eine Editierung online ausgeführt, ist keine Editierung über ein anderes Gerät möglich.

2.4 Bewegungs- und Steuerfunktionen

Das Steuergerät verfügt über folgende charakteristische Funktionen.

Funktion	Beschreibung	Referenz
Optimale Geschwindigkeit	Diese Funktion verhindert Fehler, die durch Geschwindigkeitsüberschreitungen hervorgerufen werden. Dazu werden bei einer Verfahrbewegung zwischen zwei Punkten Stellungen vermieden, die eine Geschwindigkeitsüberschreitung zur Folge hätten. Bei aktivierter Funktion ist die Geschwindigkeit an der Handspitze nicht konstant.	Abschn. 6.3.70 „Befehl SPD (Speed)“
Optimale Beschleunigung/Verzögerung	Die Funktion legt die optimale Beschleunigungs-/Verzögerungszeit beim Starten und Stoppen des Roboters in Abhängigkeit der Lasteinstellungen fest. Zu den Lasteinstellungen zählen Gewicht, Abmessungen und Schwerpunkt von Hand und Werkstück. Die Funktion bewirkt eine Verkürzung der Zykluszeit.	Abschn. 6.3.53 „Befehl OADL (Optimum Acceleration/Deceleration) Abschn. 6.3.45 „Befehl LOADSET (Load Set)“
XYZ-Weichheit	Die Funktion ermöglicht über die Daten des Servomotors eine nachgiebige Steuerung des Roboters. Die Funktion bewirkt ein sanftes Einsetzen von Werkstücken in Bohrungen o. Ä. Im kartesischen Koordinatensystem ist das Teach von Positionen auch bei aktivierter Achsenweichheit möglich. Der sinnvolle Einsatz dieser Funktion ist von den Werkstückbedingungen abhängig.	Abschn. 6.3.10 „Befehl CMP TOOL (Compliance Tool)“
Kollisionsüberwachung	Die Funktion bewirkt bei einem Zusammenstoß des Werkstücks oder des Roboterarms mit umliegenden Einrichtungen einen sofortigen Stopp des Roboters. Dadurch können entstehende Schäden begrenzt werden. Die Funktion wird ausschließlich von den Robotern RV-S und RH-S unterstützt. Ein Aktivierung der Funktion ist sowohl im Automatik- als auch im JOG-Betrieb möglich. Die Funktion kann nicht gemeinsam mit der Funktion zur Steuerung zusätzlicher Mechanismen verwendet werden.	Abschn. 6.3.14 „Befehl COLCHK (Col Check)“
Überwachung der Wartungsintervalle	In Abhängigkeit des Roboterbetriebes erfolgt die Überwachung wartungsrelevanter Daten. Überwacht werden z. B. die Roboterbatterien, Zahnriemen oder Schmierstoffe. Die Wartungsinformationen können über die Programmier-Software angezeigt werden. Die Funktion wird ausschließlich von den Robotern der RV-S- und RH-S-Serien unterstützt. Die Funktion kann nicht gemeinsam mit der Funktion zur Steuerung zusätzlicher Mechanismen verwendet werden.	Verwenden Sie die PC-Support-Software, um die Funktion zu nutzen. Die Funktion steht ab Version E1 der Programmier-Software zur Verfügung.
Wiederherstellung von Positionsdaten	Die Funktion zur Wiederherstellung von Positionsdaten berechnet Korrekturwerte für die Nullpunkt-, die Werkzeug- und die Basiskoordinaten. Bei einer Abweichung der Gelenkachsendaten, durch Austausch eines Motors, durch Verformung eines Handgreifers oder einer Abweichung der Basiskoordinaten kann durch die Angabe von maximal 10 Punkten eine Korrektur der Positionsabweichung erfolgen. Die Funktion kann mit der Programmier-Software genutzt werden. Die Funktion wird ausschließlich von den Robotern der RV-S- und RH-S-Serien unterstützt.	Verwenden Sie die PC-Support-Software, um die Funktion zu nutzen. Die Funktion steht ab Software-Version E1 des Steuergerätes zur Verfügung. Vertikal-Knickarmroboter: Die Funktion kann ab Software-Version E1 verwendet werden. RH-S-Serie: Die Funktion kann ab Software-Version F2 verwendet werden.
Kontinuierliche Bewegung	Die Funktion dient zur Steuerung einer Verfahrbewegung ohne Beschleunigung/Verzögerung zwischen verschiedenen Punkten. Sie bewirkt eine Verkürzung der Zykluszeit.	Abschn. 4.3.4 „Kontinuierliche Bewegung“ Abschn. 6.3.13 „Befehl CNT (Continuous)“

Tab. 2-6: Bewegungs- und Steuerfunktionen (1)

Funktion	Beschreibung	Referenz
Multitasking	Die Multitask-Funktion ermöglicht die parallele Ausführung mehrerer Programme zur Verkürzung der Taktzeiten. Der Roboter kann neben seiner Bewegung weitere Funktionen ausführen und mit der Peripherie kommunizieren, z. B. um Signale weiterzugeben. Mit Hilfe der Multitasking-Funktion kann über ein Programm eine Steuerung peripherer Einrichtungen ohne Einsatz einer SPS erfolgen.	Abschn. 4.10 „Multitasking-Funktion“ Abschn. 6.3.81 „Befehl XRUN (X Run)“
Ständige Programmausführung	Die Funktion ermöglicht die Ausführung eines Programmes, sobald die Versorgungsspannung eingeschaltet wird. Mit Hilfe der Funktion kann im Multitask-Betrieb über ein Programm eine SPS simuliert werden.	Abschn. 9.5 „Parameter SLT (ALWAYS)“
Programm fortsetzen	Für den Programmplatz 1 wird nach dem Ausschalten der Spannungsversorgung die aktuelle Position innerhalb der Anwendung gespeichert. Nach dem nächsten Einschalten der Spannungsversorgung startet die Anwendung von dieser gespeicherten Position.	Abschn. 9.5 „Parameter CTN“
Steuerung einer Zusatzachse	Die Funktion ermöglicht eine Steuerung von bis zu 2 Zusatzachsen. Da die Positionen der Achsen in den geteachten Roboterdaten gespeichert sind, kann eine vollständig synchrone Steuerung der Achsen erfolgen. Neben der Bewegung der Zusatzachse ist eine Bewegung über Kreis-Interpolation möglich. Die Funktion wird von den Steuergeräten der CR1- und CR2-Serie bei Einbau einer Schnittstellenkarte zur Steuerung von Zusatzachsen unterstützt.	Handbuch „Schnittstelle zur Steuerung einer Zusatzachse“
Steuerung zusätzlicher Mechanismen	Die Funktion ermöglicht neben den Standard-Robotern eine Steuerung von bis zu 2 zusätzlichen Mechanismen, die über Servomotoren angetrieben werden.	Handbuch „Schnittstelle zur Steuerung einer Zusatzachse“
Kommunikation	Zur Kommunikation mit externen Einheiten stehen folgende Möglichkeiten zur Verfügung: Steuerung des Steuergerätes und Programmsteuerung <ul style="list-style-type: none"> über Ein-/Ausgangssignale (Standardschnittstelle mit 32 Eingängen/32 Ausgängen bei den Steuergeräten CR2 und CR3 und 16 Eingängen/16 Ausgängen beim Steuergerät CR1) über CC-Link-Netzwerk (optional) Als Datenverbindung <ul style="list-style-type: none"> über RS232C-Schnittstelle (1 Standardschnittstelle und bis zu 4 Zusatzschnittstellen) über Ethernet Die Datenverbindung bezieht sich auf definierte Funktionen zum Austausch von Daten, z. B. Kompensationsdaten, mit externen Einheiten (z. B. optische Sensoren).	Abschn. 7.2.26 „Variablen M_IN, M_INB, M_INW“ Abschn. 7.2.33 „Variablen M_OUT, M_OUTB, M_OUTW“ Abschn. 9.16 „Kommunikationseinstellungen“
Interrupt-Überwachung	Die Funktion dient zur Überwachung von Signalen während der Programmabarbeitung. Bedingungsabhängig kann das Programm zur Ausführung einer Interrupt-Routine unterbrochen werden. Die Funktion wird z. B. verwendet, um zu überwachen, dass beim Transport von Werkstücken keine Werkstücke verloren gehen.	Abschn. 6.3.19 „Befehl DEF ACT (Define Act)“ Abschn. 6.3.2 „Befehl ACT (Act)“
Unterprogramm-aufruf	Die Funktion zum Aufruf eines Unterprogramms aus einem Hauptprogramm	Abschn. 6.3.4 „Befehl CALLP (Call P)“

Tab. 2-6: Bewegungs- und Steuerfunktionen (2)

Funktion	Beschreibung	Referenz
Palettierung	Die Funktion berechnet automatisch die Zwischenpositionen von in einer Palette angeordneten Werkstücken. Dadurch wird der Aufwand an zu teachenden Positionen minimiert. Die Funktion ermöglicht die Definition von Paletten im Spalten- und Zeilenformat sowie kreisförmigen Paletten.	Abschn. 4.4 „Palettierung“ Abschn. 6.3.26 „Befehl DEF PLT (Define Pallet)“ Abschn. 6.3.59 „Befehl PLT (Pallet)“
Benutzerdefinierter Bereich	Die Funktion ermöglicht eine Echtzeit-Überwachung des Roboters in einem über max. 8 Flächen frei definierbaren Bereich. Befindet sich die Handspitze des Roboters innerhalb dieses Bereichs, kann die Ausgabe eines Statussignals an eine externe Einheit erfolgen, die dann in einem Programm weiter verarbeitet wird oder eine Fehlermeldung zur Folge hat. Zusätzlich stehen 2 weitere ähnliche Funktionen für ein Roboterprogramm (ZONE und ZONE2) zur Verfügung.	Abschn. 9.9 „Benutzerdefinierter Bereich“ Abschn. 8.2.53 „Funktion ZONE“ Abschn. 8.2.54 „Funktion ZONE 2“
Verfahrweggrenzen für Gelenkbewegungen Verfahrweggrenzen für XYZ-Bewegungen Frei definierbare Begrenzungsfläche	Folgende Funktionen ermöglichen eine Begrenzung des Roboter-Arbeitsbereichs: Bei Gelenkbewegungen: Legt die Verfahrweggrenzen für jedes einzelne Gelenk fest. Bei XYZ-Bewegungen: Legt die Verfahrweggrenzen für das XYZ-Koordinatensystem fest. Frei definierbare Begrenzungsfläche: Der Arbeitsbereich eines Roboters kann über eine frei definierbare Fläche auf den Bereich vor oder hinter dieser Fläche begrenzt werden.	Kap. 9 „Parameter MEJAR“ und „Parameter MEPAR“ Abschn. 9.10 „Verfahrwegbegrenzungsebene“

Tab. 2-6: Bewegungs- und Steuerfunktionen (3)

3 Bedienung und Programmierung

3.1 Bedienung der Teaching Box

In diesem Abschnitt wird die Bedienung der Teaching Box und die Funktionen der einzelnen Menüs beschrieben.

3.1.1 Menübaum

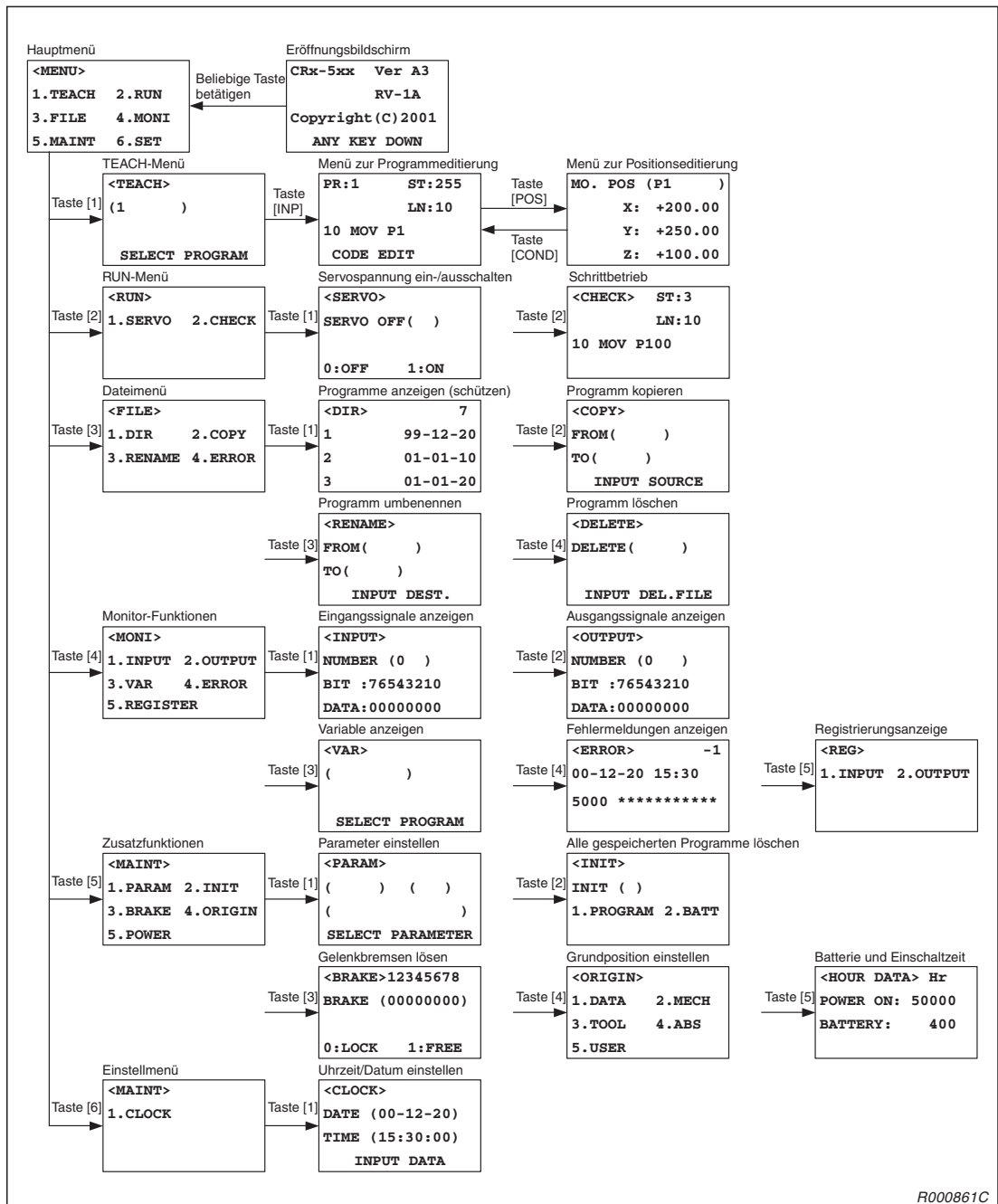


Abb. 3-1: Menübaum

3.1.2 Menüpunkt auswählen

Funktion


Zur Auswahl eines Menüpunkts stehen zwei Möglichkeiten zur Verfügung.

Ausführung

In Tab. 3-2 und Tab. 3-3 werden die beiden Möglichkeiten beispielhaft an der Auswahl des Menüpunkts „1. TEACH“ gezeigt.


Stellen Sie den [MODE]-Schalter des Steuergerätes auf die Stellung „TEACH“. Aktivieren Sie die Teaching Box, indem Sie den [ENABLE/DISABLE]-Schalter der Teaching Box auf „ENABLE“ stellen.

Nach dem Einschalten erscheint der Eröffnungsbildschirm.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre>CRn-5xx Ver.A3 RP-1AH COPYRIGHT(C)2001 ANY KEY DOWN</pre>		Betätigen Sie nach Erscheinen des Eröffnungsbildschirms die Taste [MENU], um das Hauptmenü aufzurufen.
②	<pre><MENU> 1 . TEACH 2 . RUN 3 . FILE 4 . MONI 5 . MAINT 6 . SET</pre>		Das Hauptmenü wird angezeigt.



Tab. 3-1: Aufruf des Hauptmenüs

1.) Menüauswahl über Eingabe der Nummer

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre><MENU> 1 . TEACH 2 . RUN 3 . FILE 4 . MONI 5 . MAINT 6 . SET</pre>		Das Menü „TEACH“ wird durch Eingabe der Ziffer „1“ ausgewählt.
②	<pre><TEACH> () SELECT PROGRAM</pre>		Das Menü „TEACH“ wird angezeigt.

Tab. 3-2: Beispiel zur Menüauswahl über Eingabe der Nummer

2.) Menüauswahl über Cursor

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre><MENU> 1 .TEACH 2 .RUN 3 .FILE 4 .MONI 5 .MAINT 6 .SET</pre>		Der Cursor wird über die Tasten [ADD ↑], [RPL ↓], [DEL ←] oder [HAND →] zum gewünschten Menüpunkt bewegt.
②	<pre><MENU> 1 .TEACH 2 .RUN 3 .FILE 4 .MONI 5 .MAINT 6 .SET</pre>		Die Auswahl wird bestätigt.
③	<pre><TEACH> () SELECT PROGRAM</pre>		Das Menü „TEACH“ wird angezeigt.

Tab. 3-3: Beispiel zur Menüauswahl über Cursor

HINWEISE

Solange der [MODE]-Schalter des Steuergerätes nicht auf „TEACH“ gestellt ist, sind über die ausgeschaltete Teaching Box nur bestimmte Funktionen (z. B. Anzeige der aktuellen Position im JOG-Betrieb, Änderung der Geschwindigkeitsübersteuerung, Anzeige der Ein- und Ausgangssignalzustände, Fehlerliste usw.) ausführbar.

Die Eingabe von Ziffern erfolgt über die Tasten mit einer Ziffer in der unteren linken Ecke. Die Eingabe eines Leerzeichens erfolgt über die [SPACE]-Taste.

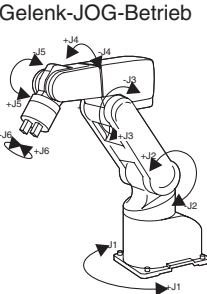
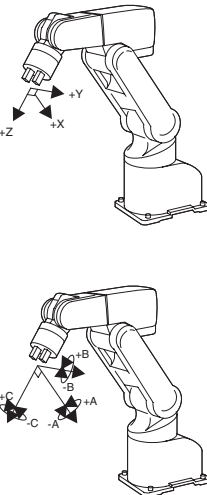
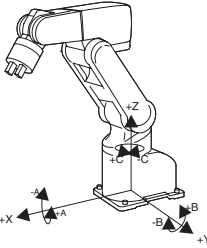
Das Löschen von Zeichen erfolgt über die gleichzeitige Betätigung der Tasten [CHAR] und [DEL ←]. Beim Löschen ist der Cursor rechts neben das zu löschende Zeichen zu positionieren. Zum Einfügen von Zeichen wird der Cursor über die [DEL ←]-Taste oder die [HAND →]-Taste an die Stelle bewegt, an der das Zeichen eingefügt werden soll. Anschließend kann die Eingabe des gewünschten Zeichens erfolgen.

3.2 Roboter im JOG-Betrieb bewegen

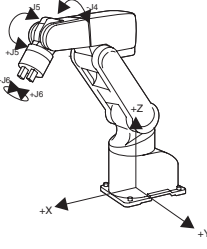
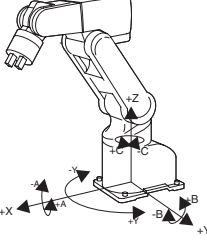
Im JOG-Betrieb kann der Roboter schrittweise manuell positioniert werden. In diesem Abschnitt wird der JOG-Betrieb anhand des Knickarmroboters RV-1A erläutert. Die Achsenkonfiguration ist abhängig vom verwendeten Robotertyp. Eine detaillierte Beschreibung zu den einzelnen Robotertypen finden Sie im Technischen Handbuch des Roboters.

3.2.1 JOG-Betriebsarten

Es werden fünf JOG-Betriebsarten unterschieden:

Betriebsart	Betrieb	Beschreibung
<p>Gelenk-JOG-Betrieb</p> 	<ul style="list-style-type: none"> ● Stellen Sie den [MODE]-Schalter der Teaching Box auf die Stellung „ENABLE“. ● Halten Sie den Dreistufenschalter in Mittelstellung. ● Betätigen Sie die [STEP/MOVE]-Taste. (Die Servoversorgungsspannung wird eingeschaltet.) ● Betätigen Sie die [JOINT]-Taste, um in den Gelenk-JOG-Betrieb zu wechseln. ● Betätigen Sie zur Bewegung der Gelenke die entsprechende Taste J1 bis J6. ● Das Menü zur Einstellung der Zusatzachsen rufen Sie durch zweimalige Betätigung der [JOINT]-Taste auf. 	<p>Im Gelenk-JOG-Betrieb können die Roboterachsen einzeln verfahren werden. Dabei ist eine unabhängige Einstellung der Achsen J1 bis J6 und der Zusatzachsen J7 und J8 möglich. Die Anzahl der Achsen hängt vom Robotertyp ab.</p> <p>Die Steuerung der Zusatzachsen J7 und J8 erfolgt über die Tasten [J1] und [J2].</p>
<p>Werkzeug-JOG-Betrieb</p> 	<p>Führen Sie die oben genannten ersten drei Punkte aus.</p> <ul style="list-style-type: none"> ● Betätigen Sie die [TOOL]-Taste, um in den Werkzeug-JOG-Betrieb zu wechseln. ● Betätigen Sie zur Bewegung der Achsen die entsprechende Taste X, Y, Z, A, B, C. 	<p>Im Werkzeug-JOG-Betrieb kann die Position der Handspitze entlang den Achsen im Werkzeug-Koordinatensystem bewegt werden.</p> <p>Die Handspitze wird linear bewegt. Die Stellung des Roboters kann über die Tasten A, B und C um die Achsen X, Y, und Z des Werkzeug-Koordinatensystems gedreht werden, ohne die Position der Handspitze zu verändern. Der Werkzeugmittelpunkt muss über den Parameter MEXTL festgelegt werden.</p> <p>Das Werkzeug-Koordinatensystem, in dem die Position der Handspitze festgelegt wird, ist vom Robotertyp abhängig. Beim Vertikal-Knickarmroboter ist die Richtung vom Handflansch zur Handspitze als +Z definiert.</p> <p>Beim SCARA-Roboter ist die Richtung von der Aufstellfläche nach oben als +Z definiert.</p>
<p>XYZ-JOG-Betrieb</p> 	<p>Führen Sie die oben genannten ersten drei Punkte aus.</p> <ul style="list-style-type: none"> ● Betätigen Sie die [XYZ]-Taste, um in den XYZ-JOG-Betrieb zu wechseln. 	<p>Im XYZ-JOG-Betrieb kann die Position der Handspitze entlang den Achsen im XYZ-Koordinatensystem bewegt werden.</p> <p>Die Stellung des Roboters kann über die Tasten A, B und C um die Achsen X, Y, und Z des XYZ-Koordinatensystems gedreht werden, ohne die Position der Handspitze zu verändern. Der Werkzeugmittelpunkt muss über den Parameter MEXTL festgelegt werden.</p>

Tab. 3-4: JOG-Betriebsarten (1)

Betriebsart	Betrieb	Beschreibung
<p>3-Achsen-XYZ-JOG-Betrieb</p> 	<p>Führen Sie die oben genannten ersten drei Punkte aus.</p> <ul style="list-style-type: none"> ● Betätigen Sie zweimal die [XYZ]-Taste, um in den 3-Achsen-XYZ-JOG-Betrieb zu wechseln. 	<p>Im 3-Achsen-XYZ-JOG-Betrieb kann die Position der Handspitze entlang den Achsen im XYZ-Koordinatensystem bewegt werden.</p> <p>Im Unterschied zum XYZ-JOG-Betrieb wird die Stellung des Roboters wie im Gelenk-JOG-Modus durch Drehung der Achsen J4, J5 und J6 verändert. Bei fest definierter Position der Handspitze wird die Stellung über die Achsen X, Y, Z, J4, J5 und J6 interpoliert, d. h. die Stellung ist nicht konstant.</p> <p>Der Werkzeugmittelpunkt muss über den Parameter MEXTL festgelegt werden.</p>
<p>Kreis-JOG-Betrieb</p> 	<p>Führen Sie die oben genannten ersten drei Punkte aus.</p> <ul style="list-style-type: none"> ● Betätigen Sie dreimal die [XYZ]-Taste, um in den Kreis-JOG-Betrieb zu wechseln. 	<p>Im Kreis-JOG-Betrieb kann die Position der Handspitze kreisförmig um den Nullpunkt bewegt werden.</p> <p>Eine Änderung der X-Achsen-Koordinate bewirkt vom Mittelpunkt des Roboters ausgehend eine radiale Bewegung der Handspitze. Eine Änderung der Y-Achsen-Koordinate bewirkt die gleiche Bewegung wie die Steuerung der J1-Achse im Gelenk-JOG-Betrieb. Eine Änderung der Z-Achsen-Koordinate bewirkt eine Bewegung der Hand in Z-Richtung wie beim XYZ-JOG-Betrieb.</p> <p>Bei einer Änderung der Koordinaten der A-, B- oder C-Achse erfolgt eine Drehung des Handgreifers wie im XYZ-JOG-Betrieb. Die Achsen sind bei Robotern vom Typ RH steuerbar.</p>

Tab. 3-4: JOG-Betriebsarten (2)

Nähert sich der Überwachungspunkt der Hand im Werkzeug-JOG-, XYZ-JOG- oder Kreis-JOG-Betrieb einem singulären Punkt, erscheint ein Warnsymbol auf der Tearing Box und es ertönt ein Warnton. Die Funktion kann über den Parameter MESNGLSW deaktiviert werden. Eine detaillierte Beschreibung der Parameter finden Sie im Kapitel 9. Eine Beschreibung der Funktion „Fehlermeldung bei Erreichen des singulären Punkts“ finden Sie in Abschn. 9.18.

3.2.2 JOG-Geschwindigkeit einstellen

Bei Betätigung der [STEP/MOVE]-Taste wird die aktuelle Position und die Geschwindigkeit in % angezeigt. Wenn Sie diesen Wert ändern wollen, halten Sie die [STEP/MOVE]-Taste gedrückt und betätigen Sie die [+]- oder die [-]-Taste. Folgende Geschwindigkeiten können eingestellt werden:

← [-]-Taste									[+]-Taste →
LOW	HIGH	3 %	5 %	10 %	30 %	50 %	70 %	100 %	

Die Einstellungen „LOW“ und „HIGH“ sind vordefinierte Werte. Bei diesen Einstellungen wird der Roboter bei jedem Tastendruck um einen bestimmten Verfahrensweg weiterbewegt. Die Größe des Verfahrenswegs hängt vom Robotertyp ab.

Schritte/Tastendruck	JOG-Betriebsart	
	Gelenk	Werkzeug / XYZ
LOW	0,01°	0,01 mm
HIGH	0,10°	0,10 mm

Tab. 3-5: Geschwindigkeitswerte für die einzelnen JOG-Betriebsarten

3.2.3 Gelenk-JOG-Betrieb

Im Gelenk-JOG-Betrieb kann jede Roboterachse in Winkelgraden einzeln verfahren werden.

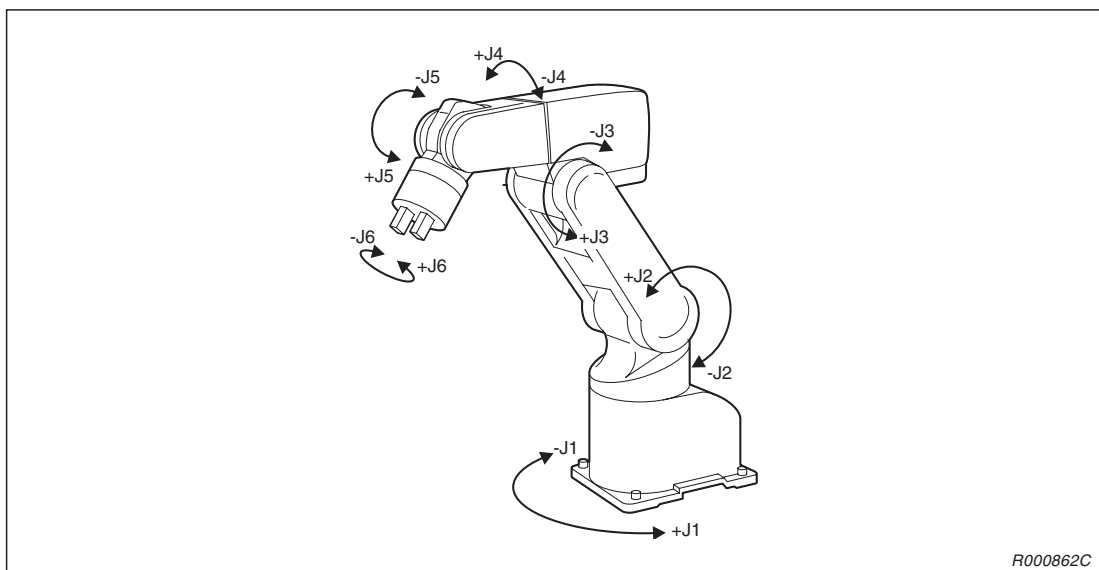


Abb. 3-2: Bewegungsrichtungen des Roboters im Gelenk-JOG-Betrieb

3.2.4 Werkzeug-JOG-Betrieb

Im Werkzeug-JOG-Betrieb kann die Position der Handspitze entlang den Achsen im Werkzeug-Koordinatensystem bewegt werden. Die Einstellung der Koordinaten X, Y und Z erfolgt in mm, die Einstellung der Orientierungsdaten A, B und C erfolgt in Grad.

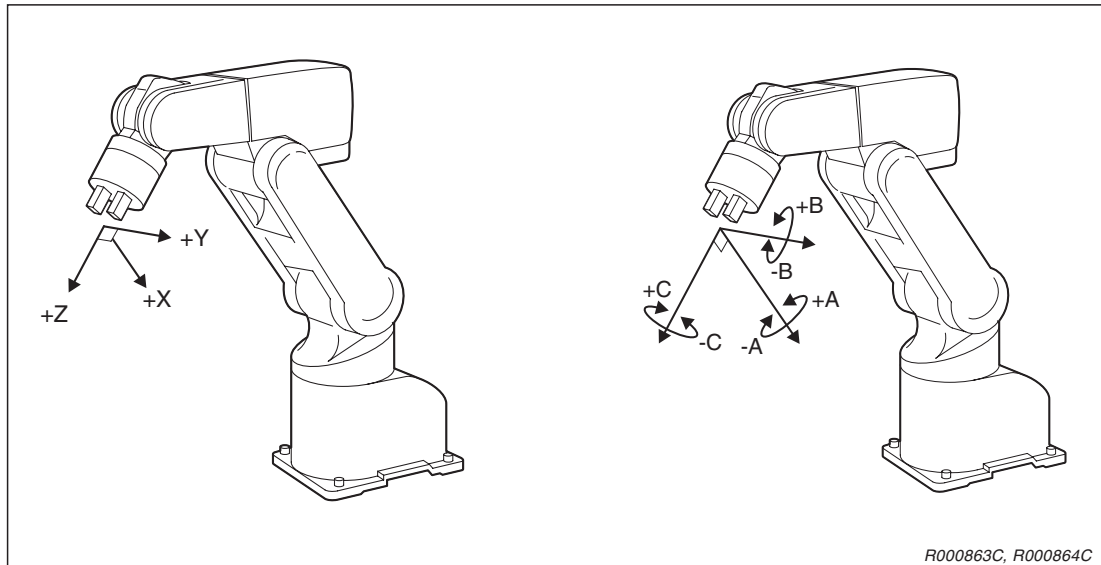


Abb. 3-3: Bewegungsrichtungen des Roboters im Werkzeug-JOG-Betrieb

3.2.5 XYZ-JOG-Betrieb

Im XYZ-JOG-Betrieb kann die Position der Handspitze entlang den Achsen im XYZ-Koordinatensystem bewegt werden. Die Einstellung der Koordinaten X, Y und Z erfolgt in mm, die Einstellung der Orientierungsdaten A, B und C erfolgt in Grad.

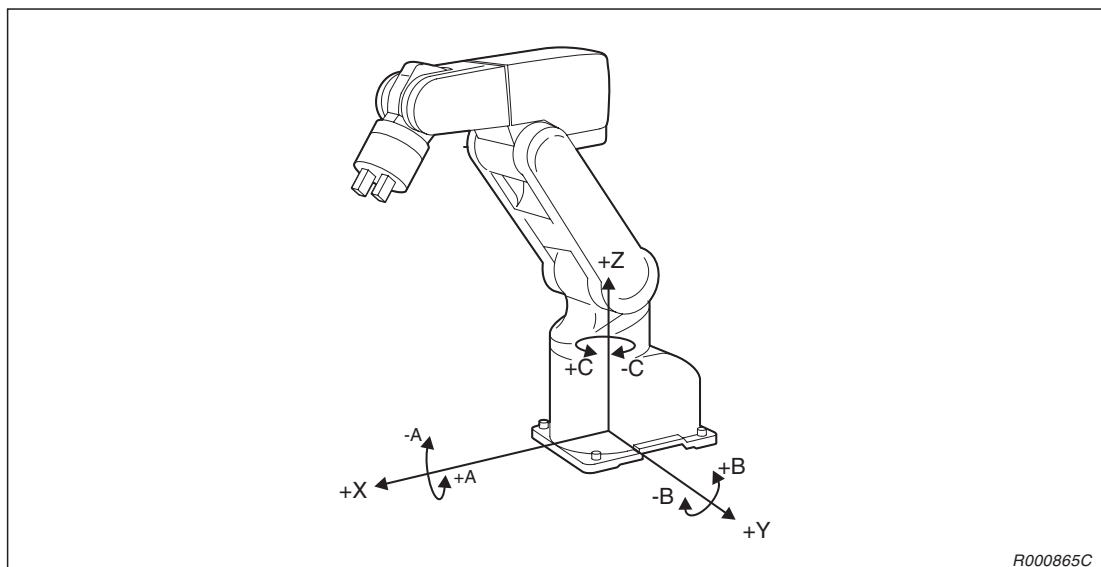


Abb. 3-4: Bewegungsrichtungen des Roboters im XYZ-JOG-Betrieb

3.2.6 3-Achsen-XYZ-JOG-Betrieb

Im 3-Achsen-XYZ-JOG-Betrieb erfolgt die Änderung der Koordinaten für die X-, Y- und Z-Achse wie im XYZ-JOG-Betrieb. Unabhängig davon erfolgt eine Änderung der Gelenkdaten wie im Gelenk-JOG-Betrieb, wobei die Position des Überwachungspunktes der Hand (X-, Y- und Z-Wert) durch Änderungen der Stellung aufrecht erhalten wird. Die Einstellung der Koordinaten X, Y und Z erfolgt in mm, die Einstellung der Gelenkdaten J4, J5 und J6 erfolgt in Grad.

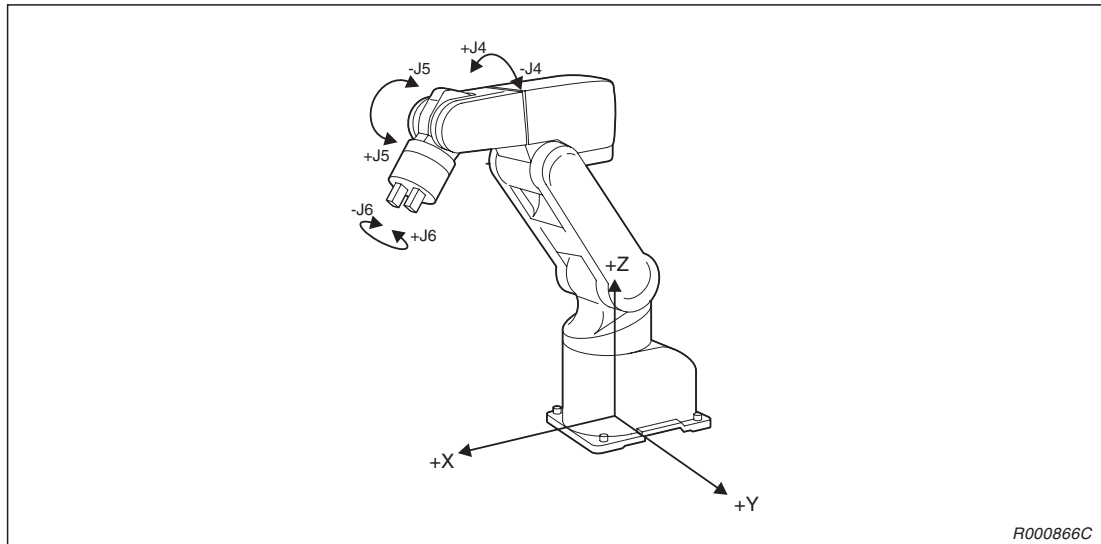


Abb. 3-5: Bewegungsrichtungen des Roboters im 3-Achsen-XYZ-JOG-Betrieb

3.2.7 Kreis-JOG-Betrieb

Eine Änderung der X-Achsen-Koordinate bewirkt vom Mittelpunkt des Roboters ausgehend eine radiale Bewegung der Handspitze. Eine Änderung der Y-Achsen-Koordinate resultiert in einer Drehung um die J1-Achse. Eine Änderung der Z-Achsen-Koordinate bewirkt eine Bewegung der Hand entlang der Z-Achse. Bei einer Änderung der Koordinaten der A-, B- oder C-Achse erfolgt eine Drehung des Handgreifers wie im XYZ-JOG-Betrieb. Die Einstellung der Koordinaten X und Z erfolgt in mm, die Einstellung der Daten Y, A, B und C erfolgt in Grad.

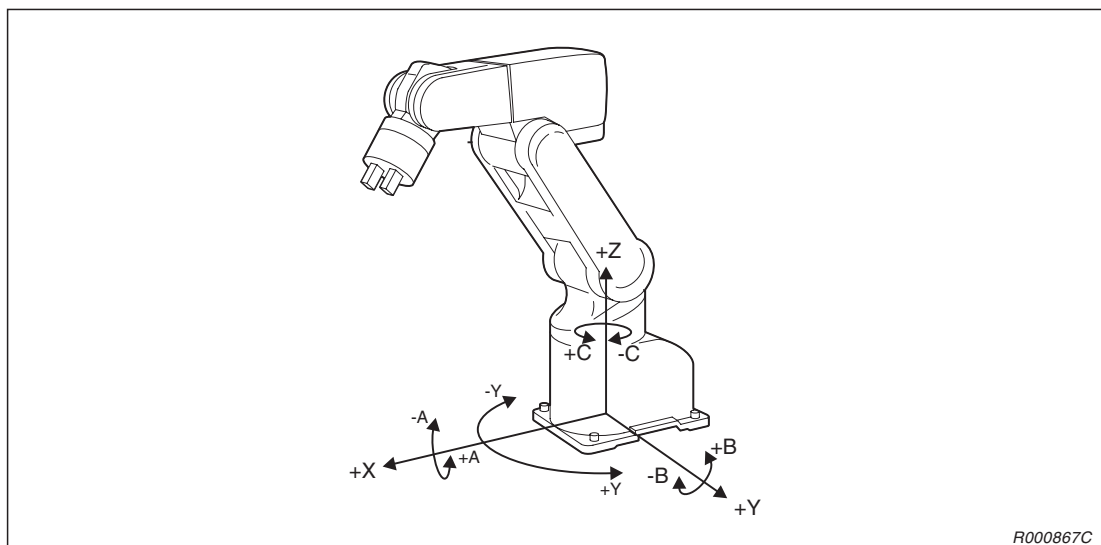


Abb. 3-6: Bewegungsrichtungen des Roboters im Kreis-JOG-Betrieb

3.2.8 Kollisionsüberwachung im JOG-Betrieb

Die Roboter RV-S und RH-S verfügen über eine Kollisionsüberwachung. Auch während des JOG-Betriebs ist eine Aktivierung dieser Funktion möglich. Standardmäßig ist sie jedoch deaktiviert. Erfasst das Steuergerät im JOG-Betrieb einen Zusammenstoß des Roboters mit einer umliegenden Einrichtung, erfolgt die Ausgabe der Fehlernummer 1010. Die erste Stelle gibt die Achsennummer wieder.

Parameter		Anzahl der Zeichen	Beschreibung	Werkseinstellung
Kollisionsüberwachung Dieser Parameter ist ab Software-Version J2 verfügbar	COL	Ganze Zahl 3	Freigabe der Kollisionsüberwachung und Aktivierung direkt nach Einschalten der Spannungsversorgung 1. Element: Kollisionsüberwachung 0 = deaktiviert 1 = aktiviert 2. Element: Aktivierung direkt nach Einschalten der Spannungsversorgung 0 = deaktiviert 1 = aktiviert 3. Element: Freigabe im JOG-Betrieb 0 = deaktiviert 1 = aktiviert 2 = NOERR-Modus Im NOERR-Modus erfolgt keine Fehlerausgabe, auch wenn die Kollisionsüberwachung anspricht. Die Servoversorgung wird jedoch abgeschaltet. Verwenden Sie diesen Modus, wenn kein störungsfreier Betrieb durch häufiges Ansprechen der Kollisionsüberwachung möglich ist.	Für die Robotermodelle RV-S: 0,0,1 RH-S: 1,0,1 für alle anderen Robotermodelle: 0,0,0
Ansprechschwelle im JOG-Betrieb	COLLVLJG	Ganze Zahl 8	Einstellung der Ansprechschwelle im JOG-Betrieb für jede Achse in % Zur Erhöhung der Empfindlichkeit ist der numerische Wert zu verringern. Spricht die Kollisionsüberwachung im JOG-Betrieb auch ohne einen Zusammenstoß an, erhöhen Sie den Wert. Einstellbereich: 1 bis 500 %	Der Standardwert ist 200,200,200,200, 200,200,200,200 und modellabhängig
Handbedingung	HNDDATO	Reelle Zahl 7	Einstellung der Handbedingungen beim Start (Festlegung im Werkzeugkoordinatensystem) Nach Einschalten der Spannungsversorgung werden im JOG-Betrieb die hier festgelegten Werte verwendet. Bei Aktivierung der Kollisionsüberwachung im JOG-Betrieb müssen diese Werte eingestellt werden, da die Überwachung sonst ungewollt ansprechen kann. (Gewicht, Größe X, Größe Y, Größe Z, Schwerpunkt X, Schwerpunkt Y, Schwerpunkt Z) Einheit: kg, mm	Nur für die Robotermodelle RV-S und RH-S. Der Wert ist vom jeweiligen Robotermodell abhängig. Als Last wird die maximale Last gesetzt.
Werkstückbedingung	WRKDAT0	Reelle Zahl 7	Einstellung der Werkstückbedingungen beim Start (Festlegung im Werkzeugkoordinatensystem) Nach Einschalten der Spannungsversorgung werden im JOG-Betrieb die hier festgelegten Werte verwendet. (Gewicht, Größe X, Größe Y, Größe Z, Schwerpunkt X, Schwerpunkt Y, Schwerpunkt Z) Einheit: kg, mm	Nur für die Robotermodelle RV-S und RH-S 0,0,0,0,0,0,0,0,0,0,0,0

Tab. 3-6: Parameter für die Kollisionsüberwachung

Einstellung der Kollisionsüberwachung

Standardmäßig ist die Ansprechschwelle der Kollisionsüberwachung hoch eingestellt. Zur Erhöhung der Empfindlichkeit ist die Ansprechschwelle über den Parameter COLLVLJG abzusenken. Achten Sie auch auf eine korrekte Einstellung der Parameter HNDDAT0 und WRKDAT0. Werden diese Werte nicht eingestellt, kann die Kollisionsüberwachung in Abhängigkeit der Roboterstellung ungewollt ansprechen.

**ACHTUNG:**

Die Kollisionsüberwachung schützt den Roboter, den Handgreifer, das Werkstück und andere Komponenten nicht vor Beschädigungen durch einen Zusammenstoß mit umliegenden Einrichtungen. Verfahren Sie den Roboter deshalb immer so, dass es zu keinen Kollisionen kommt.

Betrieb nach einem Zusammenstoß

Um ein Einschalten der Servoversorgungsspannung während eines Zusammenstoßes zu verhindern, erfolgt in diesem Fall nach dem Ansprechen der Kollisionsüberwachung beim Einschalten der Servoversorgungsspannung eine erneute Auslösung der Kollisionsüberwachung. Erfolgt auch weiterhin bei jedem Einschalten der Servoversorgungsspannung eine Fehlerausgabe, lösen Sie die Bremsen des Roboterarms (siehe auch Abschn. 3.13.3 „Gelenkbremsen lösen“) und schalten Sie dann die Servoversorgungsspannung erneut ein.

Weiterhin ist ein Einschalten der Servoversorgungsspannung auch nach dem temporären Zurücksetzen des Fehlers möglich (siehe auch Abschn. 3.9 „Fehler temporär zurücksetzen“).

HINWEIS

Ausschließlich die Roboter RV-S und RH-S verfügen über eine Kollisionsüberwachung. Andere Robotermodelle unterstützen diese Funktion nicht, auch wenn der Parameter einstellbar ist.

3.3 Werkzeugdaten umschalten

Ab der Software-Version J1 des Steuergerätes und Software-Version A2 der Teaching Box ist eine Umschaltung der Werkzeugdaten über die Teaching Box möglich. Geben Sie dazu die Werkzeugdaten in die Parameter MEXTL1 bis 4 ein und wählen Sie das Werkzeug über die in der folgenden Abbildung dargestellten Tasten.

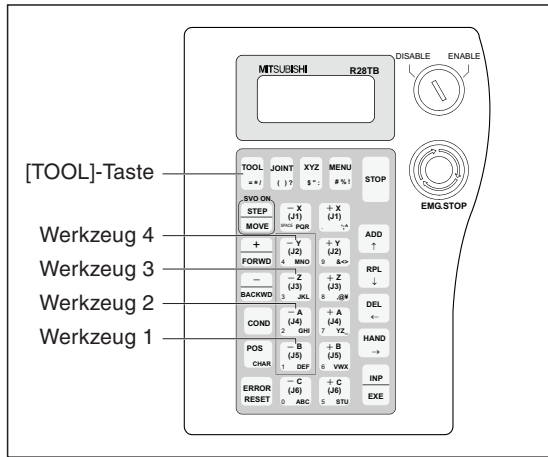


Abb. 3-7:
Anordnung der Tasten zum Umschalten der Werkzeugdaten

R001180C

Gehen Sie zum Umschalten der Werkzeugdaten wie folgt vor:

- ① Stellen Sie den [MODE]-Schalter des Steuergerätes auf „TEACH“.
- ② Stellen Sie den [ENABLE/DISABLE]-Schalter der Teaching Box auf „ENABLE“.
- ③ Halten Sie die [TOOL]-Taste gedrückt und betätigen Sie die entsprechende Taste, wie in folgender Tabelle gezeigt, um ein Werkzeug auszuwählen.

Nr.	Display-Darstellung	Tastenbetätigung	Beschreibung
①			Werkzeug 1 auswählen
②	<pre><TOOL SETTING> TOOL NUMBER: 0</pre>		Werkzeug 2 auswählen
③	<pre>PUSH 1 TO 4</pre>		Werkzeug 3 auswählen
④			Werkzeug 4 auswählen
⑥	<pre><JOINT> T2 100% J1 +34.50 J2 +20.00 J3 +80.00</pre>		In der JOG-Anzeige erscheint nach dem Zeichen „T“ die gewählte Werkzeugnummer.

Tab. 3-7: Zuordnung von Tasten und Werkzeugen

HINWEISE

Soll der Roboter bei einer Umschaltung der Werkzeugdaten (MEXTL1 bis 4) im Automatikbetrieb zu der eigentlich geteachten Position bewegt werden, schreiben Sie die entsprechende Werkzeugnummer in die Variable M_TOOL und fahren Sie die Position an, indem Sie die Werkzeugdaten umschalten. Beachten Sie, dass der Roboter unvorhersehbare Bewegungen ausführen kann, falls die Werkzeugdaten beim Teachen nicht mit denen im Automatikbetrieb übereinstimmen.

Wird der Roboter bei einer Umschaltung der Werkzeugdaten während der Ausführung des Programms im Schrittbetrieb bewegt, kann der Roboter unvorhersehbare Bewegungen ausführen, falls die geteachten Werkzeugdaten nicht mit denen der im Schrittbetrieb verwendeten Werkzeugnummer übereinstimmen.

Die aktuelle Werkzeugnummer kann über das Werkzeugmenü oder die Variable M_TOOL angezeigt werden.

Die aktuellen Werkzeugdaten sind im Parameter MEXTL gespeichert. Beachten Sie, dass der Wert bei Auswahl der Werkzeugdaten über die Parameter MEXTL1 bis 4 überschrieben wird. Um die Werkzeugnummer wieder auf „0“ zu setzen, führen Sie den Befehl TOOL aus.

Weitere Informationen zu den Werkzeugdaten finden Sie bei den Parametern MEXTL, MEXTL1, MEXTL2, MEXTL3 und MEXTL4, beim Befehl TOOL und beim Parameter M_TOOL.

3.4 Handgreifer öffnen/schließen

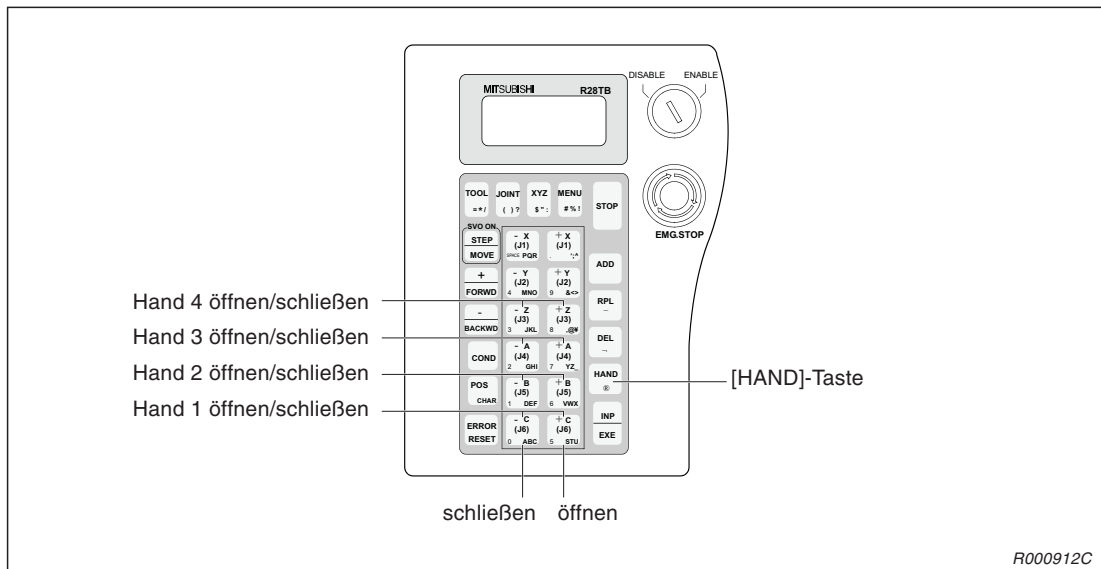














Abb. 3-8: Anordnung der Tasten zum Öffnen und Schließen des Handgreifers

Gehen Sie zum Öffnen bzw. Schließen eines Handgreifers wie folgt vor:

- ① Stellen Sie den [ENABLE/DISABLE]-Schalter der Teaching Box auf „ENABLE“.
- ② Halten Sie die [HAND]-Taste gedrückt und betätigen Sie die entsprechende Taste, wie in folgender Tabelle gezeigt, um den Handgreifer zu öffnen bzw. zu schließen.

Nr.	Auswahl der Hand	Tastenbetätigungen	Beschreibung
①	Hand 1		Greifer der Hand 1 wird geöffnet.
②			Greifer der Hand 1 wird geschlossen.
①	Hand 2		Greifer der Hand 2 wird geöffnet.
②			Greifer der Hand 2 wird geschlossen.

Tab. 3-8: Zuordnung von Tasten und Handgreifern (1)

Nr.	Auswahl der Hand	Tastenbetätigungen	Beschreibung
①	Hand 3	 →  	Greifer der Hand 3 wird geöffnet.
②		 →  	Greifer der Hand 3 wird geschlossen.
①	Hand 4	 →  	Greifer der Hand 4 wird geöffnet.
②		 →  	Greifer der Hand 4 wird geschlossen.

Tab. 3-8: Zuordnung von Tasten und Handgreifern (2)

Im Handbereich des Roboters können unterschiedliche Werkzeuge montiert werden. Erfolgt bei einer pneumatischen Greifhand die Ansteuerung über ein Zweifach-Ventil, dienen zwei Bits des Handsteuersignals der Überwachung des Handgreiferzustands. Weitere Informationen über die Handsteuersignale finden Sie in Abschn. 9.13 und Abschn. 9.14.

3.5 Handgreifer ausrichten

Ein Ausrichten des Handgreifers bewirkt eine Bewegung der Hand zu der Position, die den kleinstmöglichen Weg zur senkrechten oder waagerechten Stellung der Achsen A, B und C hat.

Sind die Werkzeugkoordinaten über den TOOL-Befehl oder über Parameter definiert, erfolgt die Ausrichtung der Hand in den festgelegten Koordinaten. Sind die Koordinaten nicht definiert, erfolgt die Ausrichtung der Hand im Mittelpunkt des Handflansches. Eine detaillierte Beschreibung der Werkzeugkoordinaten finden Sie in Abschn. 9.7.

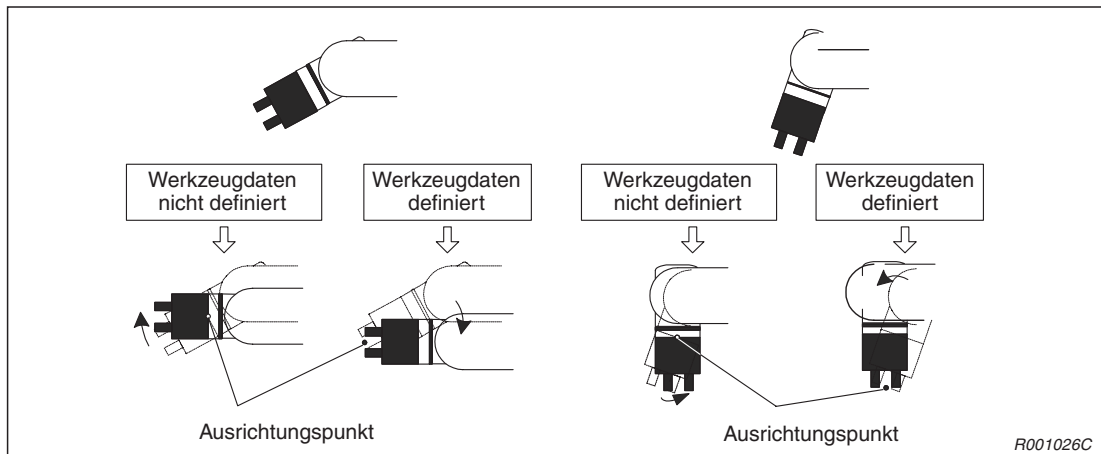


Abb. 3-9: Ausrichten des Handgreifers

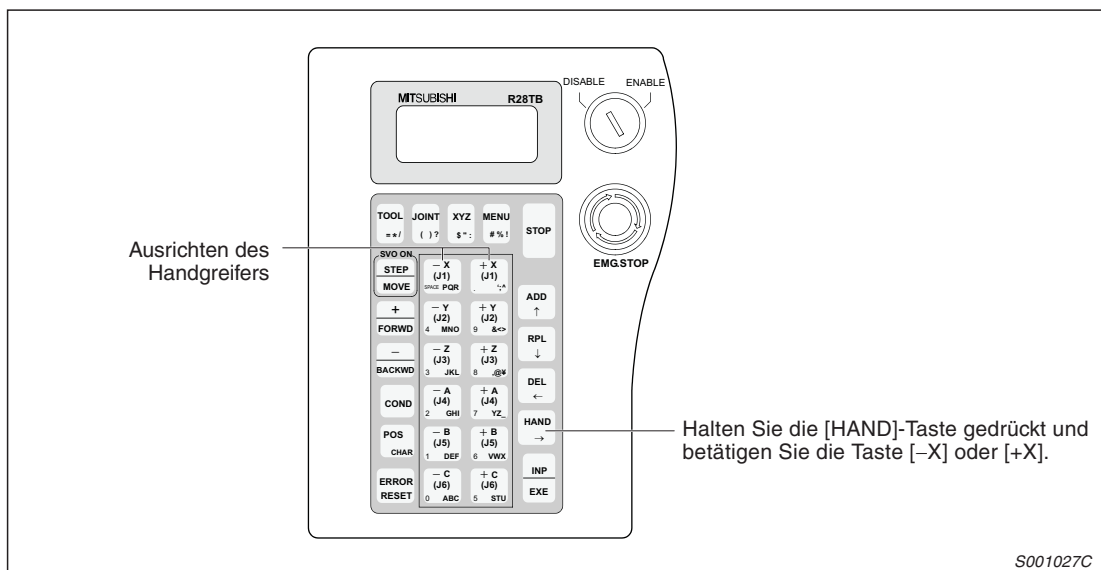


Abb. 3-10: Anordnung der Tasten zum Ausrichten des Handgreifers

Gehen Sie zum Ausrichten des Handgreifers wie folgt vor:

- ① Stellen Sie den [ENABLE/DISABLE]-Schalter der Teaching Box auf „ENABLE“.
- ② Halten Sie den Dreistufenschalter in Mittelstellung
- ③ Betätigen Sie die [STEP/MOVE]-Taste, um die Servoversorgungsspannung einzuschalten.
- ④ Halten Sie die [HAND]-Taste gedrückt und betätigen Sie die Taste [-X] oder [+X].

3.6 Programmierung

Die Programmiersprache MELFA-BASIC IV ermöglicht die Erstellung komplexer Programme mit umfangreichen Funktionen. In diesem Abschnitt wird die Vorgehensweise bei der Programmierung über die Teaching Box erläutert.

Eine detaillierte Beschreibung der MELFA-BASIC-IV-Befehle finden Sie in Abschn. 6.3.



Die folgende Tabelle zeigt die Möglichkeiten der Weiterverarbeitung der Daten einer Zeile:

Eingabeformat	Verarbeitung
Zeilennummer und Befehl (Beispiel: 10 MOV P1)	Eingabe wird als Zeile des Roboterprogramms verarbeitet.
Nur Zeilennummer (Beispiel: 10)	Löscht die angegebene Zeile aus dem Programm (durch Überschreiben)
Nur Befehl (Beispiel: MOV P1)	Führt den Befehl sofort aus (Direkt-Modus) Eingabe nur bei vorher geteachter Position möglich

Tab. 3-9: Weiterverarbeitung der übertragenen Daten einer Zeile

3.6.1 Roboterprogramm erstellen

Aufruf des Menüs zur Programmeditierung

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre><MENU> 1 .TEACH 2 .RUN 3 .FILE 4 .MONI 5 .MAINT 6 .SET</pre>		Das Menü „TEACH“ wird ausgewählt.
②	<pre><TEACH> (1) SELECT PROGRAM</pre>		Die Programmnummer „1“ wird ausgewählt.
③	<pre>PR : 1 ST : 1 LN : 0 --NO DATA--</pre>		Nach der Tastenbetätigung wird das Menü zur Programmeditierung aufgerufen.















Tab. 3-10: Aufruf des Menüs zur Programmeditierung

HINWEISE

Die Editierung eines ständig ausgeführten Programms (Startbedingung im Programmplatzparameter SLTn: ALWAYS) ist nur dann möglich, wenn zuerst die kontinuierliche Ausführung unterbrochen wird. Ändern Sie die Startbedingung im Programmplatzparameter von „ALWAYS“ auf „START“ und schalten Sie anschließend das Steuergerät aus und wieder ein, um die kontinuierliche Ausführung des Programms zu unterbrechen.

Zur Auswahl eines Programmes aus der Programmliste betätigen Sie die [INP/EXE]-Taste in einem leeren Feld des TEACH-Menüs. Die Programmliste wird angezeigt. Wählen Sie mit Hilfe des Cursors und anschließender Betätigung der [INP/EXE]-Taste das gewünschte Programm aus der Programmliste.

Erstellung eines neuen Programms

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre>PR:1 ST:1 LN:0 --NO DATA--</pre>	3 x 	Der Cursor wird zum Eingabefeld für die Befehle bewegt.
②	<pre>PR:1 ST:1 LN:0 CODE EDIT</pre>	 →  → 	Die Zeilennummer „10“ wird eingegeben.
③	<pre>PR:1 ST:1 LN:0 10 █ CODE EDIT</pre>	 → 	Das Zeichen „M“ wird eingegeben.
④	<pre>PR:1 ST:1 LN:0 10 M █ CODE EDIT</pre>	2 x  [POS/CHAR]-Taste 2-mal betätigen und gedrückt halten	Betätigen Sie die Taste [POS/CHAR] 2-mal und halten Sie sie gedrückt. Es werden die 4 Befehle angezeigt, die mit „M“ beginnen.
⑤	<pre>1.MOV 2.MVS 3.MVC 4.MVR 10 M █ CODE EDIT</pre>	 → 	Der Befehl „MOV“ wird über Kurzwahl (1) eingegeben.
⑥	<pre>1.MOV 2.MVS 3.MVC 4.MVR 10 MOV █ CODE EDIT</pre>	 → 	Das Zeichen „P“ wird eingegeben.
⑦	<pre>1.MOV 2.MVS 3.MVC 4.MVR 10 MOV P █ CODE EDIT</pre>		Die Ziffer „1“ wird eingegeben.
⑧	<pre>1.MOV 2.MVS 3.MVC 4.MVR 10 MOV P1 █ CODE EDIT</pre>	 	Jetzt ist die Programmzeile „10 MOV P1“ wirksam.
⑨	<pre>PR:1 ST:2 LN:0 CODE EDIT</pre>		Das Fenster zur Editierung des 2ten Programmschrittes wird angezeigt. Die Eingabe weiterer Programmschritte erfolgt in derselben Weise.

Tab. 3-11: Erstellung eines neuen Programms

Programm speichern

Neu erstellte oder überarbeitete Programme werden mit einer der folgenden Operationen gespeichert.

- Betätigen Sie die [MENU]-Taste. Das Hauptmenü wird angezeigt.
- Stellen Sie den [ENABLE/DISABLE]-Schalter der Teaching Box in die Position „DISABLE“.

HINWEISE

Wird bei angezeigtem Programmeingabe-Bildschirm die Spannungsversorgung abgeschaltet, werden die eingegebenen Programmabschnitte gelöscht.


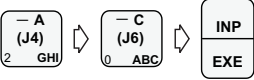





Speichern Sie aus Sicherheitsgründen Programme nicht nur im Steuergerät, sondern erstellen Sie auch Sicherheitskopien auf einem PC. Verwenden Sie dazu die optionale PC-Support-Software oder COSIROP.

Beschreibung

- Eingabe von Zeichen
Betätigen Sie die [POS/CHAR]-Taste und halten Sie diese gedrückt. Betätigen Sie dann zur Zeicheneingabe die zugehörige Taste. Jede Taste zur Zeicheneingabe ist dreifach belegt. Nach jeder Tastenbetätigung wird ein anderes Zeichen auf dem Display angezeigt. Lösen Sie die [POS/CHAR]-Taste, wenn das gewünschte Zeichen angezeigt wird.
- Eingabe von Befehlen
Befehle können zeichenweise (Beispiel „M“ → „O“ → „V“ für den MOV-Befehl) oder durch den Aufruf aus einer Liste eingegeben werden.
Durch Eingabe des Anfangsbuchstaben wird eine Liste der Befehle angezeigt, die mit dem gleichen Buchstaben beginnen. Dazu ist nach Eingabe des Anfangsbuchstaben 2-mal die [POS/CHAR]-Taste zu betätigen. Die Befehlsliste erscheint. Die Eingabe des Befehls erfolgt durch gleichzeitige Betätigung der zugeordneten Nummerntaste und der [POS/CHAR]-Taste. Wird der gewünschte Befehl nicht in der Liste angezeigt, ist die [POS/CHAR]-Taste erneut zu betätigen.
- Anzeige der vorhergehenden/nächsten Programmzeile
Die vorhergehende Programmzeile wird durch Betätigung der [-/BACKWD]-Taste, die nächste Programmzeile durch Betätigung der [+ /FORWD]-Taste aufgerufen.
- Anzeige einer bestimmten Programmzeile
Bewegen Sie den Cursor über die Taste [ADD ↑] im Menü zur Programmeditierung zum Eingabefeld für die Zeilennummer „LN:“. Geben Sie die Zeilennummer ein und betätigen Sie anschließend die [INP/EXE]-Taste. Die aufgerufene Zeilennummer erscheint.

3.6.2 Roboterprogramm editieren

Rufen Sie das Menü zur Programmeditierung auf (siehe Abschn. 3.6.1). Es wird automatisch das Programmverzeichnis angezeigt, wenn Sie keine Programmauswahl vornehmen.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre>PR:1 ST:1 LN:10 10 MOV P1 CODE EDIT</pre>		Der Cursor wird zum Eingabefeld für die Zeilennummer bewegt.
②	<pre>PR:1 ST:1 LN:(10) CODE EDIT</pre>		Die Zeilennummer „20“ wird eingegeben.
③	<pre>PR:1 ST:2 LN:(20) 20 MOV P2,-50 CODE EDIT</pre>		Der Cursor wird nach unten und rechts bewegt und eine Stelle hinter „V“ platziert.
④	<pre>PR:1 ST:2 LN:20 20 MOV P2,-50 CODE EDIT</pre>		Die Zeichen „OV“ werden gelöscht. Das Zeichen „M“ bleibt stehen.
⑤	<pre>PR:1 ST:2 LN:20 20 M P2,-50 CODE EDIT</pre>	 <p>[POS/CHAR]-Taste 2-mal betätigen und gedrückt halten</p>	Betätigen Sie die Taste [POS/CHAR] 2-mal und halten Sie sie gedrückt. Es werden die 4 Befehle angezeigt, die mit „M“ beginnen.
⑥	<pre>1.MOV 2.MVS 3.MVC 4.MVR 20 M P2,-50 CODE EDIT</pre>		Der Befehl „MVS“ wird eingegeben.
⑦	<pre>1.MOV 2.MVS 3.MVC 4.MVR 20 MVS P2,-50 CODE EDIT</pre>		Jetzt ist die Programmzeile „20 MVS P2, -50“ wirksam.
⑧	<pre>PR:1 ST:3 LN:30 30 MVS P3 CODE EDIT</pre>		Die nächste Programmzeile wird angezeigt.


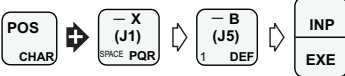



Tab. 3-12: Beispiel zum Editieren eines Programms

Beschreibung

- **Steuerung des Cursors**
Befehlszeilen können mit Hilfe des Cursors bearbeitet werden. Eine Steuerung des Cursors erfolgt dabei über die Tasten [ADD ↑], [RPL ↓], [DEL ←] oder [HAND →].
- **Anzeige einer bestimmten Programmzeile**
Bewegen Sie den Cursor im Menü zur Programmeditierung zum Eingabefeld für die Zeilennummer „LN:“. Geben Sie die Zeilennummer ein und betätigen Sie anschließend die [INP/EXE]-Taste. Die aufgerufene Zeilennummer erscheint. Mit den Tasten [+ /FORWD] (vorwärts) und [- /BACKWD] (rückwärts) können Sie im Programm „blättern“.
- **Vorsicht beim Editieren von Feldvariablen**
Die Anzahl der Elemente einer Feldvariablen, die bei der Deklaration festgelegt wurde (DIM), kann ab der Software-Version K3 geändert werden. Wird die Anzahl verkleinert, gehen die Daten der wegfallenden Elemente verloren. Die Dimensionen können nicht geändert werden. (Die Änderung einer eindimensionalen Variablen in eine zweidimensionale ist nicht möglich.)
- **Editieren eines Zeichens**
Betätigen Sie zum Löschen eines Zeichens die Taste [POS/CHAR] zusammen mit der Taste [DEL ←]. Wenn erst [DEL ←] und dann zusätzlich [POS/CHAR] gedrückt wird, wird ein anderes als das gewünschte Zeichen gelöscht! Daher: erst [POS/CHAR] und dann zusätzlich [DEL ←] drücken.
- **Prüfen des editierten Roboterprogramms**
Speichern Sie das Roboterprogramm nach erfolgter Editierung durch Betätigung der [MENU]-Taste. Prüfen Sie anschließend die ausgeführten Korrekturen im Schrittbetrieb.

Eingabe der aktuellen Positionsdaten

Im folgenden Beispiel wird die aktuelle Position als Position Nr. 1 definiert:

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre>PR:1 ST:13 LN:130 130 END CODE EDIT</pre>		Nach der Tastenbetätigung wird das Menü zur Editierung der Positionsdaten angezeigt.
②	<pre>MO.POS () X: +0.00 Y: +0.00 Z: +0.00</pre>		Die Positionsnummer „1“ wird eingegeben. Danach wird die Eingabe bestätigt und es erscheint die Anzeige der aktuell gespeicherten Position.
③	<pre>MO.POS (P1) X: +0.00 Y: +0.00 Z: +0.00</pre>		Es ertönt ein Summton und eine Bestätigungsabfrage wird angezeigt.
④	<pre>MO.POS (P1) X: +0.00 Y: +0.00 ADDITION?</pre>		Die Definition der neuen Position wird bestätigt. Nach der Anzeige von „EXECUTING“ und einem weiteren Summton ist die neue Position wirksam.
⑤	<pre>MO.POS (P1) X: +132.30 Y: +254.10 Z: +32.00</pre>		Die Koordinatenwerte der neuen Position sind nun registriert. Betätigen Sie die Taste [MENU], um die Eingabe zu speichern.

Tab. 3-13: Beispiel zur Eingabe der aktuellen Positionsdaten

HINWEIS

Ein Umschalten zwischen den Menüs zur Editierung von Befehlen und zur Editierung von Positionen erfolgt über die Betätigung der Taste [POS/CHAR]. Wird kein Cursor angezeigt, betätigen Sie die Taste [COND] und anschließend die Taste [POS/CHAR].

Position anfahren

Die Handspitze (TCP: **T**ool **C**enter **P**oint) des Roboters kann zu einer bereits definierten Position gefahren werden. Das Anfahren der Position erfolgt über den MOV- oder MVS-Bewegungsbefehl.

Schalten Sie die Servoversorgungsspannung ein, bevor Sie einen Bewegungsbefehl ausführen. Betätigen Sie den Dreistufenschalter, während Sie die Servospannung einschalten. Bei nicht betätigtem Dreistufenschalter lässt sich die Servospannung nicht einschalten.

Bezeichnung	Bewegung
MOV-Bewegungsbefehl	Die definierte Position wird mittels Gelenk-Interpolation angefahren. Der Bewegungsbefehl wird im Gelenk-JOG-Betrieb verwendet. Die Achsen werden wie beim MOV-Befehl verfahren.
MVS-Bewegungsbefehl	Die definierte Position wird mittels Linear-Interpolation angefahren. Es wird keine Verfahrbewegung ausgeführt, wenn der Stellungsmerker der aktuellen Position von der anzufahrenden Position abweicht. Der Bewegungsbefehl wird im XYZ-, 3-Achsen-XYZ-, Kreis- oder im Werkzeug-JOG-Betrieb verwendet. Die Achsen werden wie beim MVS-Befehl verfahren.

Tab. 3-14: Anfahren einer definierten Position

Das Anfahren einer Position über den MOV-Bewegungsbefehl ist nur im Gelenk-JOG-Modus möglich. Ändern Sie den JOG-Modus, falls nötig.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung								
①	<table border="1"> <tr> <td>X, Y, Z</td> <td>LOW</td> </tr> <tr> <td>X:</td> <td>+80.09</td> </tr> <tr> <td>Y:</td> <td>-21.78</td> </tr> <tr> <td>Z:</td> <td>+137.36</td> </tr> </table>	X, Y, Z	LOW	X:	+80.09	Y:	-21.78	Z:	+137.36		Es wird auf den Gelenk-JOG-Betrieb umgeschaltet.
X, Y, Z	LOW										
X:	+80.09										
Y:	-21.78										
Z:	+137.36										
②	<table border="1"> <tr> <td>JOINT</td> <td>LOW</td> </tr> <tr> <td>X:</td> <td>+34.50</td> </tr> <tr> <td>Y:</td> <td>+20.00</td> </tr> <tr> <td>Z:</td> <td>+80.00</td> </tr> </table>	JOINT	LOW	X:	+34.50	Y:	+20.00	Z:	+80.00		Das Menü für den Gelenk-JOG-Betrieb wird angezeigt.
JOINT	LOW										
X:	+34.50										
Y:	+20.00										
Z:	+80.00										
③	<table border="1"> <tr> <td>MO . POS (P2)</td> <td></td> </tr> <tr> <td>X:</td> <td>+132.30</td> </tr> <tr> <td>Y:</td> <td>+254.10</td> </tr> <tr> <td>Z:</td> <td>+32.00</td> </tr> </table>	MO . POS (P2)		X:	+132.30	Y:	+254.10	Z:	+32.00		Bei Tastenbetätigung wird die angezeigte Position mittels Gelenk-Interpolation angefahren. Der Roboter stoppt, sobald die [INP/EXE]-Taste losgelassen wird.
MO . POS (P2)											
X:	+132.30										
Y:	+254.10										
Z:	+32.00										

Tab. 3-15: Anfahren einer definierten Position über den MOV-Bewegungsbefehl


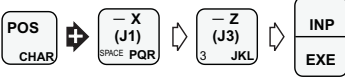
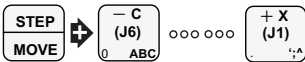
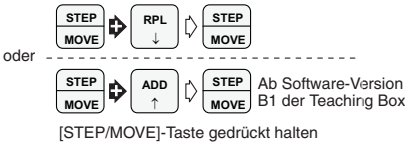

Das Anfahren einer Position über den MVS-Bewegungsbefehl ist nur im XYZ-, 3-Achsen-XYZ-, Kreis- oder im Werkzeug-JOG-Modus möglich. Ändern Sie den JOG-Modus, falls nötig.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung								
①	<table border="1"> <tr> <td>JOINT</td> <td>LOW</td> </tr> <tr> <td>X:</td> <td>+34.50</td> </tr> <tr> <td>Y:</td> <td>+20.00</td> </tr> <tr> <td>Z:</td> <td>+80.00</td> </tr> </table>	JOINT	LOW	X:	+34.50	Y:	+20.00	Z:	+80.00		Es wird auf den Werkzeug-JOG-Betrieb umgeschaltet.
JOINT	LOW										
X:	+34.50										
Y:	+20.00										
Z:	+80.00										
②	<table border="1"> <tr> <td>TOOL</td> <td>LOW</td> </tr> <tr> <td>X:</td> <td>+80.09</td> </tr> <tr> <td>Y:</td> <td>-21.78</td> </tr> <tr> <td>Z:</td> <td>+137.36</td> </tr> </table>	TOOL	LOW	X:	+80.09	Y:	-21.78	Z:	+137.36		Das Menü für den Werkzeug-JOG-Betrieb wird angezeigt.
TOOL	LOW										
X:	+80.09										
Y:	-21.78										
Z:	+137.36										
③	<table border="1"> <tr> <td>MS.POS (P2</td> <td>)</td> </tr> <tr> <td>X:</td> <td>+132.30</td> </tr> <tr> <td>Y:</td> <td>+254.10</td> </tr> <tr> <td>Z:</td> <td>+32.00</td> </tr> </table>	MS.POS (P2)	X:	+132.30	Y:	+254.10	Z:	+32.00		Bei Tastenbetätigung wird die angezeigte Position mittels Gelenk-Interpolation angefahren. Der Roboter stoppt, sobald die [INP/EXE]-Taste losgelassen wird. ①
MS.POS (P2)										
X:	+132.30										
Y:	+254.10										
Z:	+32.00										

Tab. 3-16: Anfahren einer definierten Position über den MVS-Bewegungsbefehl

① Ist die Ausführung einer linearen Verfahrbewegung von der aktuellen Position zur Zielposition nicht möglich, verbleibt der Roboter im Stillstand.

Positionsdaten ersetzen


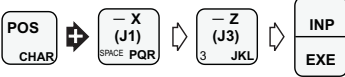

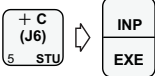
Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre>PR:1 ST:8 LN:80 80 MVS P3 CODE EDIT</pre>		Nach der Tastenbetätigung wird das Menü zur Editierung der Positionsdaten angezeigt.
②	<pre>MO.POS () X: +0.00 Y: +0.00 Z: +0.00</pre>		Die Positionsnummer „3“ wird eingegeben. Danach wird die Eingabe bestätigt und es erscheint die Anzeige der aktuell gespeicherten Position.
③	<pre>MO.POS (P3) X: +132.30 Y: +354.10 Z: +132.00</pre>	 <p>Dreistufenschalter betätigen</p>	Bewegen Sie den Roboter im JOG-Betrieb zu der Position, die angeglichen werden soll.
④	<pre>JOINT LOW X: +34.50 Y: +20.00 Z: +80.00</pre>		Die neue Position wird angefahren.
⑤	<pre>MO.POS (P3) X: +132.30 Y: +354.10 Z: +132.00</pre>		Es ertönt ein Summton und eine Bestätigungsabfrage wird angezeigt. Verwenden Sie ab Software-Version B1 der Teaching Box die untere Tastenkombination.
⑥	<pre>MO.POS (P3) X: +132.30 Y: -284.10 Z: +132.00</pre>		Das Ersetzen der Position wird bestätigt. Nach der Anzeige von „REPLACING“ und einem weiteren Summton ist die neue Position wirksam.
⑦	<pre>MO.POS (P3) X: +132.30 Y: -284.10 Z: +132.00</pre>		Das Ersetzen ist jetzt abgeschlossen.

Tab. 3-17: Beispiel zum Ersetzen von Positionsdaten

Beschreibung

- Aufruf einer Positionsvariablen
Geben Sie den Positionsvariablennamen, der aufgerufen werden soll, in die Klammern hinter dem Befehl MO.POS ein und betätigen Sie anschließend die [INP/EXE]-Taste. Mit den Tasten [+ /FORWD] (vorwärts) und [- /BACKWD] (rückwärts) können Sie durch die Variablenanzeige „blättern“.
- Prüfen des editierten Roboterprogramms
Speichern Sie das Roboterprogramm nach erfolgter Editierung durch Betätigung der [MENU]-Taste. Prüfen Sie anschließend die ausgeführten Korrekturen im Schrittbetrieb.
- Anzeige der Software-Version
Die Software-Version der Teaching Box erscheint nach Einschalten der Versorgungsspannung des Steuergerätes.





Positionsdaten ändern

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre>PR:1 ST:8 LN:80 80 MVS P3 CODE EDIT</pre>		Nach der Tastenbetätigung wird das Menü zur Editierung der Positionsdaten angezeigt.
②	<pre>MO.POS () X: +0.00 Y: +0.00 Z: +0.00</pre>		Die Positionsnummer „3“ wird eingegeben. Danach wird die Eingabe bestätigt und es erscheint die Anzeige der aktuell gespeicherten Position.
③	<pre>MO.POS (P3) X: +132.30 Y: +354.10 Z: +132.00</pre>		Der Cursor wird zum Y-Eingabefeld und zur „4“ bewegt.
④	<pre>MO.POS (P3) X: +132.30 Y: +354.10 Z: +132.00</pre>		Die Ziffer „4“ wird durch die Ziffer „5“ überschrieben.
⑤	<pre>MO.POS (P3) X: +132.30 Y: +355.10 Z: +132.00</pre>		Der neue Koordinatenwert wird angezeigt. Betätigen Sie die Taste [MENU], um die Eingabe zu speichern.

Tab. 3-18: Beispiel zum Ändern von Positionsdaten

Positionsdaten löschen





Es können nur Positionsdaten gelöscht werden, die nicht vom aktuell ausgeführten Programm verwendet werden. Bei einem Versuch, eine Position zu löschen, die im aktuellen Programm verwendet wird, erfolgt eine Fehlermeldung.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre>PR:1 ST:8 LN:80 80 MVS P3 CODE EDIT</pre>		Nach der Tastenbetätigung wird das Menü zur Editierung der Positionsdaten angezeigt.
②	<pre>MO.POS () X: +0.00 Y: +0.00 Z: +0.00</pre>		Die Positionsnummer „4“ wird eingegeben. Danach wird die Eingabe bestätigt und es erscheint die Anzeige der aktuell gespeicherten Position.
③	<pre>MO.POS (P4) X: +2.98 Y: +354.10 Z: +132.00</pre>		Es ertönt ein Summton und eine Bestätigungsabfrage wird angezeigt.
④	<pre>MO.POS (P4) X: +2.98 Y: +35.10 DELETE?</pre>		Die Koordinaten der Position 4 werden gelöscht. Es ertönt ein weiterer Summton.
⑤	<pre>MO.POS (P4) X:----- Y:----- Z:-----</pre>		Der Löschvorgang ist ausgeführt. Betätigen Sie die Taste [MENU], um den Vorgang zu speichern.

Tab. 3-19: Beispiel zum Löschen von Positionsdaten

Anzeige der Positonsdaten

Die Werte der Koordinaten für die X-, Y- und Z-Achse, der Orientierungsdaten A, B, und C, der Stellungsdaten und der Multirotationsdaten für jede Achse werden in den Positionsdaten abgespeichert. Die Anzeige der einzelnen Werte können Sie wie folgt aufrufen:

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre>PR:1 ST:13 LN:130 130 END CODE EDIT</pre>		Nach der Tastenbetätigung wird das Menü zur Editierung der Positionsdaten angezeigt.
②	<pre>MO.POS(P4) X: +2.98 Y: +354.10 Z: +132.00</pre>		Die Koordinatenwerte der X-, Y- und Z-Achse werden angezeigt. Nach Betätigung der Taste [RPL ↓] erscheinen die Orientierungsdaten.
③	<pre>MO.POS(P4) A: -180.000 B: 0.000 C: -180.000</pre>		Die Orientierungsdaten der A-, B- und C-Achse werden angezeigt. Nach Betätigung der Taste [RPL ↓] wird der Stellungsmerker 1 angezeigt.
④	<pre>MO.POS(P4) STRUC.FLAG(001) SET: 1:RAN FLAG: 0:LBF</pre>		Bei einer weiteren Betätigung der Taste [RPL ↓] wird wieder die erste Anzeige aufgerufen.
⑤	<pre>MO.POS(P4) X: +2.98 Y: +354.10 Z: +132.00</pre>		Die Koordinatenwerte der X-, Y- und Z-Achse werden angezeigt.

Tab. 3-20: Anzeige der Positionsdaten

Programm speichern

Neu erstellte oder überarbeitete Programme werden mit einer der folgenden Operationen gespeichert.

- Betätigen Sie die [MENU]-Taste. Das Hauptmenü wird angezeigt.
- Stellen Sie den [ENABLE/DISABLE]-Schalter der Teaching Box in die Position „DISABLE“.

HINWEIS

Wird bei angezeigtem Programmeingabe-Bildschirm die Spannungsversorgung abgeschaltet, werden die eingegebenen Programmabschnitte und somit auch die geteachten Positionen gelöscht.

3.6.3 Roboterprogramm testen

Nach der Programmerstellung sollten Sie das Programm „testen“. Mit Testen ist die Suche nach und die Beseitigung von Programmfehlern gemeint.

Das Testen des Programms erfolgt mit der Teaching Box.

ACHTUNG:
Testen Sie unbedingt jedes Roboterprogramm vor einem automatischen Betriebseinsatz!

Programm schrittweise ausführen (vorwärts)

Die Programmausführung erfolgt zeilenweise in Vorwärtsrichtung. Betätigen Sie nach Einschalten der Servoversorgungsspannung bei der Ausführung der folgenden Schritte den Dreistufenschalter auf der Rückseite der Teaching Box.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre>MO . POS (P1) X: +132.30 Y: +354.10 Z: +132.00</pre>		Nach der Tastenbetätigung wird das Menü zur Editierung der Befehle angezeigt.
②	<pre>PR:1 ST:1 LN:10 10 MOV P1 CODE EDIT</pre>	oder	Der Roboter startet die Verfahrbewegung. Wird die [INP/EXE]-Taste während der Bewegung losgelassen, stoppt der Roboter.
③	<pre>PR:1 ST:2 LN:20 20 MOV P2 CODE EDIT</pre>	oder	Nach Abarbeitung einer Zeile wird die nächste Zeile angezeigt. Die schrittweise Ausführung des Programms erfolgt in derselben Weise.
④	<pre>PR:1 ST:13 LN:130 130 END CODE EDIT</pre>		Ist die Roboterbewegung oder die Programmausführung fehlerhaft, gehen Sie wie unter „Programm schrittweise ausführen (rückwärts)“ vor.

Tab. 3-21: Schrittweise Ausführung eines Programms (vorwärts)







ACHTUNG:
Achten Sie bei der schrittweisen Ausführung des Programms genau auf die Bewegung des Roboters. Treten während der Roboterbewegung Unregelmäßigkeiten auf (z. B. Kollisionsgefahr mit umliegenden Einrichtungen usw.), lassen Sie die [INP/EXE] oder den Dreistufenschalter los oder drücken Sie den Dreistufenschalter ganz durch, um den Roboter zu stoppen.

Beschreibung

- **Schrittbetrieb**
 Im Schrittbetrieb wird das Roboterprogramm zeilenweise abgearbeitet. Die Verfahrensgeschwindigkeit ist niedrig und der Roboter stoppt nach Abarbeitung jeder Zeile, um eine Überprüfung der Programmfunktionen und der Verfahrensbewegung zu ermöglichen. Während des Schrittbetriebs leuchtet die grüne LED des START-Tasters am Steuergerät.
- **Stoppen des Roboters im Betrieb**
 - durch Betätigung des NOT-HALT-Schalters
 Die Servoversorgung wird abgeschaltet und der Roboter stoppt sofort. Setzen Sie zur Überprüfung des Betriebs den Fehler zurück, schalten Sie die Servoversorgung wieder ein und führen Sie das Programm im Schrittbetrieb aus.
 - durch Loslassen oder Durchdrücken des Dreistufenschalters
 Die Servoversorgung wird abgeschaltet und der Roboter stoppt sofort. Es erscheint die Fehlermeldung 2000. Setzen Sie zur Überprüfung des Betriebs den Fehler zurück, betätigen Sie den Dreistufenschalter bis zur Mittelstellung, schalten Sie die Servoversorgung über die Taste [SVO ON] wieder ein und führen Sie das Programm im Schrittbetrieb aus.
 - durch Loslassen der [EXE]-Taste
 Die Ausführung des Schrittes wird unterbrochen. Die Servoversorgung wird nicht abgeschaltet. Betätigen Sie zum Fortsetzen der Programmausführung die [EXE]-Taste.

Programm schrittweise ausführen (rückwärts)

Nach Abarbeitung einer Zeile wird die vorherige Zeile aufgerufen und ausgeführt. Die Funktion kann nur bei Interpolationsbefehlen verwendet werden. Die maximale Anzahl der Zeilen für die Programmausführung in Rückwärtsrichtung ist 4.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre>MO.POS (P1) X: +132.30 Y: +354.10 Z: +132.00</pre>		Nach der Tastenbetätigung wird das Menü zur Editierung der Befehle angezeigt.
②	<pre>PR:1 ST:1 LN:10 10 MOV P1 CODE EDIT</pre>	 oder  	Starten Sie die schrittweise Programmausführung in Vorwärtsrichtung wie zuvor beschrieben.
③	<pre>PR:1 ST:2 LN:20 20 MOV P2 CODE EDIT</pre>	 	Nach Abarbeitung einer Zeile wird die vorherige Zeile aufgerufen.
④	<pre>PR:1 ST:1 LN:10 10 MOV P1 CODE EDIT</pre>		Die vorherige Zeile wird angezeigt. Wird die [INP/EXE]-Taste während der Bewegung losgelassen, stoppt der Roboter.

Tab. 3-22: Schrittweise Ausführung eines Programms (rückwärts)

Programm eines anderen Programmplatzes schrittweise ausführen

Die schrittweise Ausführung eines Multitask-Programms kann im Menü „Schrittbetrieb“ und muss nicht im Menü zur Programmeditierung ausgeführt werden.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre><MENU> 1 .TEACH 2 .RUN 3 .FILE 4 .MONI 5 .MAINT 6 .SET</pre>		Nach der Tastenbetätigung wird das Menü zum Einschalten der Servo-Spannung angezeigt.
②	<pre><RUN> 1 .SERVO 2 .CHECK</pre>		Nach der Tastenbetätigung wird das Menü zur Ausführung des Schrittbetriebs angezeigt.
③	<pre><CHECK> ST:2 LN:100 100 M_OUT(10)=1</pre>		Wechseln Sie den Programmplatz und führen Sie den Schrittbetrieb wie im Menü zur Programmeditierung aus.

Tab. 3-23: Schrittweise Ausführung eines Multitask-Programms

HINWEIS

Setzen Sie die Programmplatznummer auf „0“, um den Schrittbetrieb für alle Programmplätze gleichzeitig auszuführen.

Sprung zu einer Programmzeile oder einem Programmschritt

Eine gewünschte Programmzeile oder ein Programmschritt kann aufgerufen werden.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre>PR:1 ST:2 LN:20 20 MOV P2 CODE EDIT</pre>		Bewegen Sie den Cursor über die Tasten [ADD ↑], [RPL ↓], [DEL ←] oder [HAND →] zur Eingabe des Programmschrittes oder der Zeilennummer.
②	<pre>PR:1 ST:13 LN:130 130 END CODE EDIT</pre>		Nach Eingabe der Zeilennummer oder des Programmschrittes wird die gewünschte Stelle aufgerufen.

Tab. 3-24: Sprung zu einer Programmzeile oder einem Programmschritt

HINWEIS

Nach Aufruf einer Zeilennummer oder eines Programmschrittes ist ein Schrittbetrieb möglich. Es erscheint eine Fehlermeldung, wenn eine Zeile zur Initialisierung von Variablen o. Ä. übersprungen wird.

3.7 Servospannung ein-/ausschalten

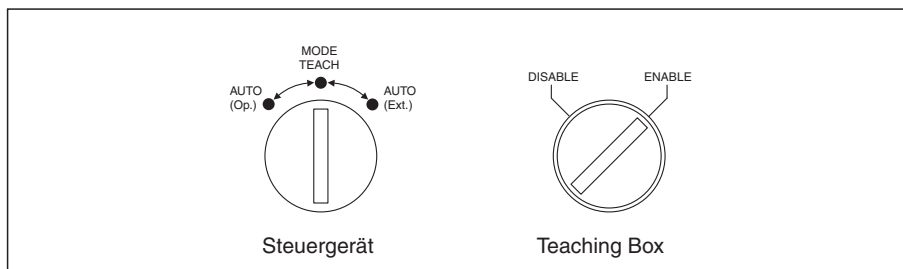
Aus Sicherheitsgründen kann die Servospannung im Teach-Modus nur bei betätigtem Dreistufenschalter eingeschaltet werden. Betätigen Sie den Dreistufenschalter bis zur Mittelstellung, bevor Sie die Servospannung einschalten.

HINWEIS

Die Bremsen werden automatisch aktiviert, wenn die Servospannung ausgeschaltet wird. Je nach Robotertyp verfügen nicht alle Achsen über Bremsen.

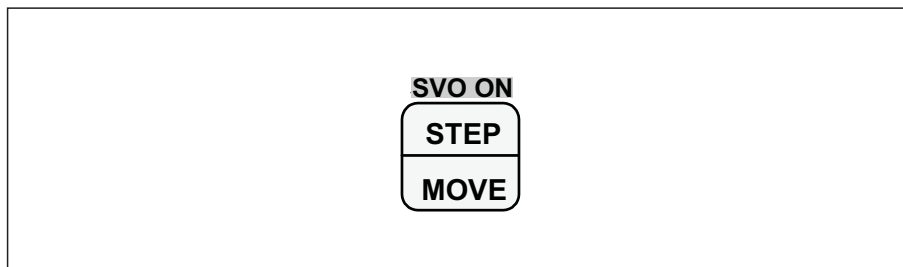
Servospannung über die [SVO ON]-Taste der Teaching Box einschalten

- ① Stellen Sie den [MODE]-Schalter des Steuergerätes auf „TEACH“ und den [ENABLE/DISABLE]-Schalter der Teaching Box auf „ENABLE“.



R000713C



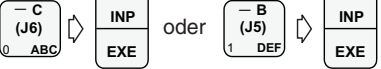
- ② Betätigen Sie die [SVO ON]-Taste ([STEP/MOVE]-Taste), um die Servospannung einzuschalten.



step_mov

Servospannung über die Teaching Box einschalten

Stellen Sie den [MODE]-Schalter des Steuergerätes auf „TEACH“ und den [ENABLE/DISABLE]-Schalter der Teaching Box auf „ENABLE“.

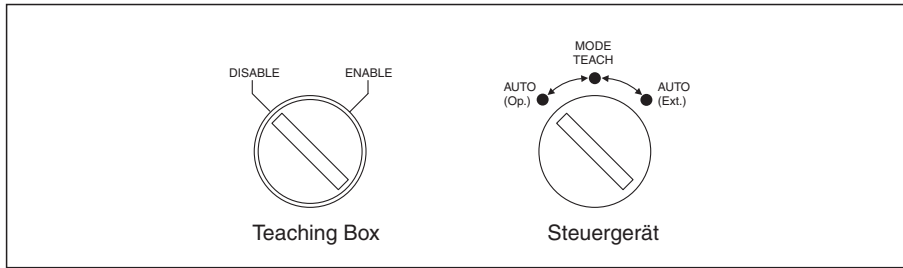
Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<div style="border: 1px solid black; padding: 5px;"> <MENU> 1 . TEACH 2 . RUN 3 . FILE 4 . MONI 5 . MAINT 6 . SET </div>		Das Menü RUN wird durch Eingabe der Ziffer „2“ aufgerufen.
②	<div style="border: 1px solid black; padding: 5px;"> <RUN> 1 . SERVO </div>		Der Menüpunkt SERVO wird ausgewählt.
③	<div style="border: 1px solid black; padding: 5px;"> <SERVO> SERVO ON (■) 0 : OFF 1 : ON </div>		Betätigen Sie den Dreistufenschalter und schalten Sie die Servospannung ein oder aus. (0: AUS, 1: EIN). ^①
④	<div style="border: 1px solid black; padding: 5px;"> <SERVO> SERVO ON (■) 0 : OFF 1 : ON </div>		Die Servospannung ist jetzt ein- oder ausgeschaltet.

Tab. 3-25: Servospannung ein-/ausschalten

① Die Servoversorgungsspannung kann auch über die Taste [STEP/MOVE] eingeschaltet werden.

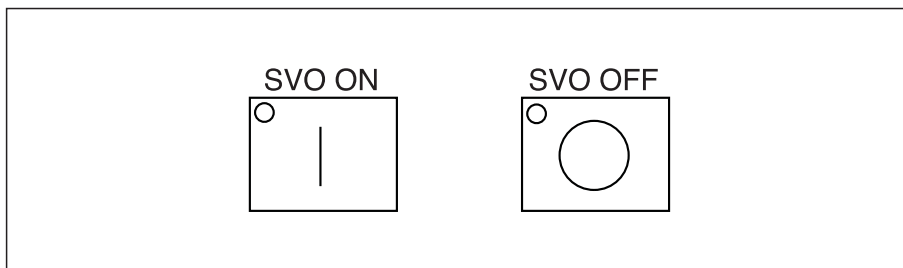
Servospannung über das Steuergerät einschalten

- ① Stellen Sie den [ENABLE/DISABLE]-Schalter der Teaching Box auf „DISABLE“ und den [MODE]-Schalter des Steuergerätes auf „AUTO (Op.)“.



R000707C

- ② Betätigen Sie die [SVO ON]-Taste, um die Servospannung einzuschalten. Die LED auf der [SVO ON]-Taste leuchtet. Betätigen Sie die [SVO OFF]-Taste, um die Servospannung auszuschalten. Die LED auf der [SVO OFF]-Taste leuchtet.

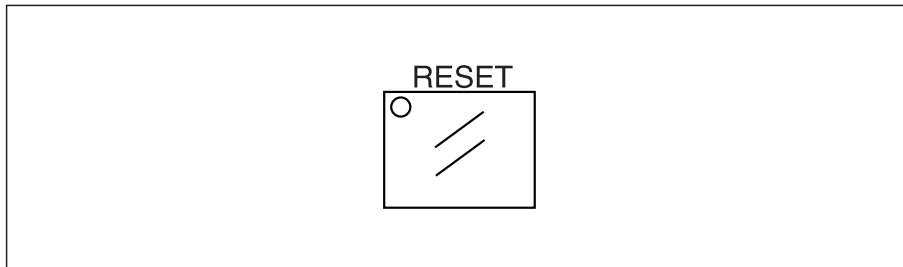


R000715C

3.8 Fehler zurücksetzen

Fehler über das Steuergerät zurücksetzen

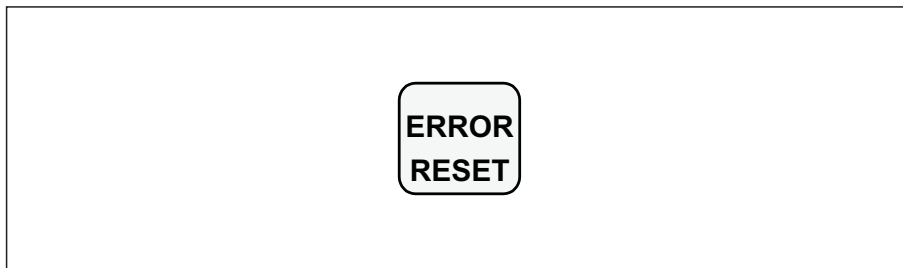
- ① Betätigen Sie die [RESET]-Taste, um den Fehler zurückzusetzen.



R000712C

Fehler über die Teaching Box zurücksetzen

- ① Betätigen Sie [ERROR RESET]-Taste, um den Fehler zurückzusetzen.



Error

HINWEIS

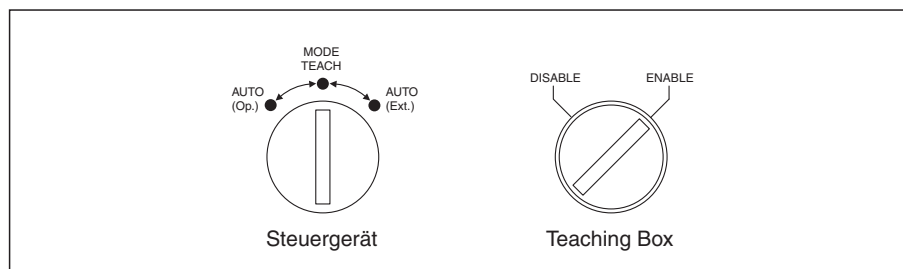
Fehler, wie z. B. H0070, lassen sich unabhängig vom Betriebsmodus des Steuergeräts bzw. der Teaching Box jederzeit zurücksetzen.

3.9 Fehler temporär zurücksetzen

In Abhängigkeit des Robotertyps können Fehler auftreten, die sich nicht zurücksetzen lassen. Das kann z. B. dann der Fall sein, wenn die Koordinaten einer Achse außerhalb des Bewegungsbereiches des Roboters liegen. Die Servoversorgungsspannung kann dann nicht mehr eingeschaltet werden, und es ist nicht möglich, den Roboter im JOG-Betrieb in den zulässigen Bereich zu bewegen. Hier muss der Fehler temporär zurückgesetzt werden. Anschließend kann der Roboter im JOG-Betrieb wieder in den zulässigen Bewegungsbereich verfahren werden.

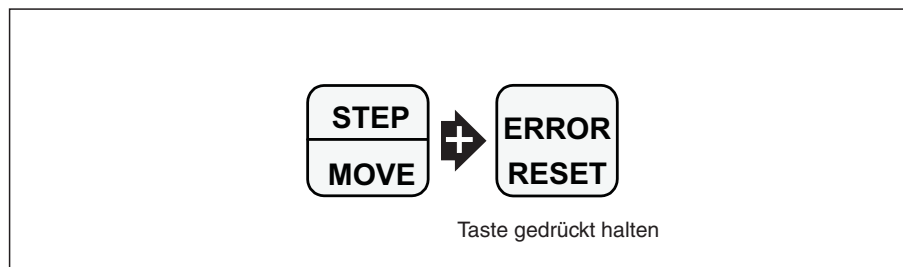
Führen Sie folgende Schritte aus, um einen Fehler temporär zurückzusetzen:

- ① Stellen Sie den [MODE]-Schalter des Steuergerätes auf „TEACH“ und den [ENABLE/DISABLE]-Schalter der Teaching Box auf „ENABLE“.



R000713C

- ② Betätigen Sie den Dreistufenschalter und drücken Sie anschließend die [ERROR RESET]-Taste, während Sie die [STEP/MOVE]-Taste gedrückt halten.



R000869C

HINWEIS

Die oben genannten Schritte dienen zum temporären Zurücksetzen eines Fehlers. Die [ERROR RESET]-Taste muss auch bei der Ausführung des JOG-Betriebes weiterhin gehalten werden. Wird die Taste losgelassen, tritt der Fehler erneut auf.

3.10 Automatikbetrieb

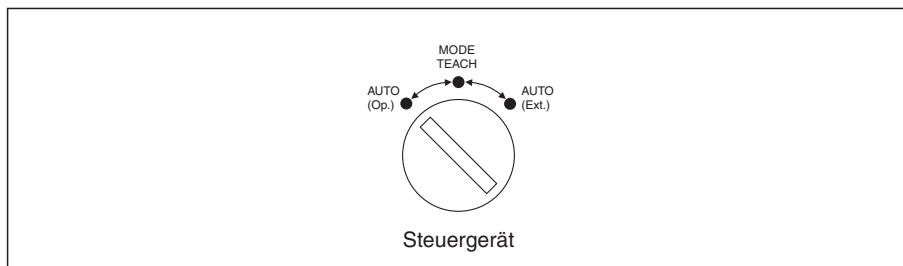
3.10.1 Geschwindigkeit einstellen

Die Geschwindigkeit kann über die Teaching Box oder das Steuergerät festgelegt werden. Die aktuelle Arbeitsgeschwindigkeit ergibt sich dabei aus:

$$\text{Aktuelle Arbeitsgeschwindigkeit} = \text{Einstellwert über Steuergerät (Teaching Box)} \times \text{Einstellwert im Programm}$$

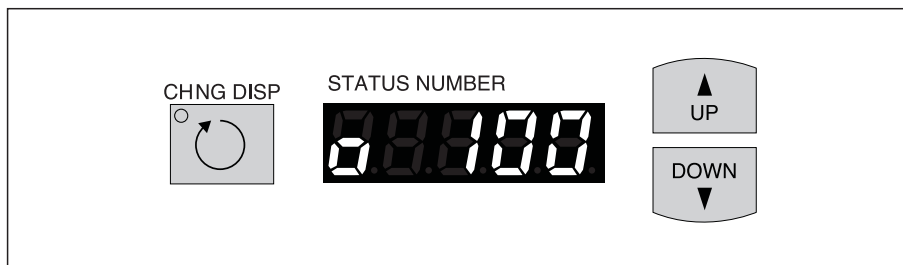
Einstellung über das Steuergerät

- ① Stellen Sie den [MODE]-Schalter des Steuergerätes auf „AUTO (Op.)“.



R000868C

- ② Betätigen Sie zweimal die Taste [CHNG DISP], um den Wert der Geschwindigkeitsübersteuerung „OVERRIDE“ anzuzeigen.
- ③ Stellen Sie die Geschwindigkeitsübersteuerung über die Tasten ▼ und ▲ ein. Der Wert kann in folgenden Schritten eingestellt werden: 10 → 20 → 30 → 40 → 50 → 60 → 70 → 80 → 90 → 100 %.



R000705C

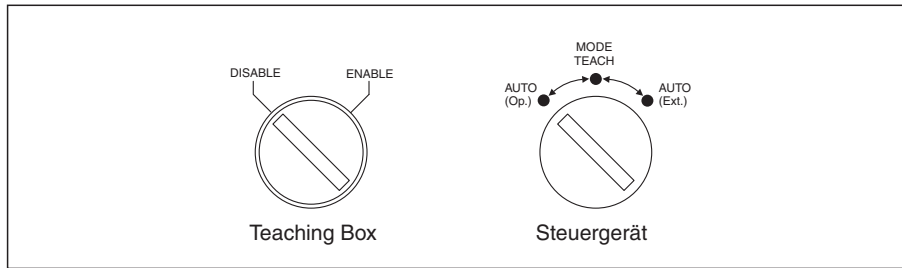
Einstellung über die Teaching Box

Die Einstellung der Geschwindigkeit über die Teaching Box erfolgt über die Tasten [STEP/MOVE] und [+ /FORWD] oder [STEP/MOVE] und [- /BACKWD]. Der Wert kann in folgenden Schritten eingestellt werden: LOW → HIGH → 3 → 5 → 10 → 30 → 50 → 70 → 100 %. Im Modus LOW und HIGH ist nur ein Schrittbetrieb möglich.

Die aktuell eingestellte Geschwindigkeit erscheint in der oberen rechten Ecke der Anzeige.

3.10.2 Auswahl der Programmnummer

- ① Stellen Sie den [ENABLE/DISABLE]-Schalter der Teaching Box auf „DISABLE“ und den [MODE]-Schalter des Steuergerätes auf „AUTO (Op.)“.



R000707C

- ② Betätigen Sie die [CHNG DISP]-Taste, um die Programmnummer anzuzeigen.
- ③ Wählen Sie die Programmnummer über die Tasten ▼ und ▲.



R000708C

HINWEIS

Besteht ein Programmnamen aus mehr als 5 Zeichen, kann er nicht angezeigt werden. Bei einem Programmaufruf über ein externes Gerät erscheint „P----“ auf der Anzeige.

Wird ein Programm während der Ausführung gestoppt, muss zunächst die Reset-Taste betätigt werden, bevor ein neues Programm ausgewählt werden kann (siehe auch Abschn. 3.10.6).

3.10.3 Starten des Automatikbetriebs



GEFAHR:

Stellen Sie vor einem Start des Automatikbetriebs sicher, dass die folgenden Bedingungen erfüllt sind:

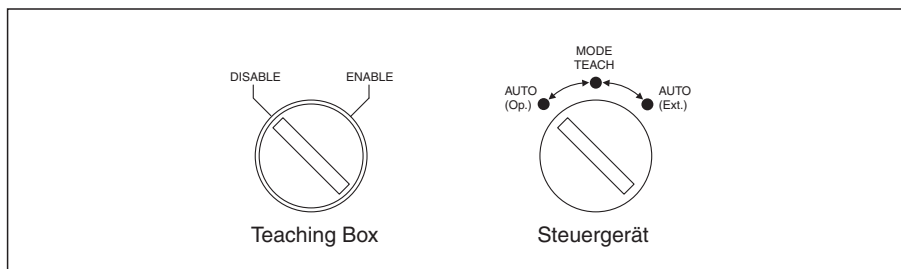
Es dürfen sich keine Personen im Umkreis des Roboters befinden.

Die Tür der Sicherheitsumzäunung muss geschlossen sein. Ein Öffnen der Tür muss zur sofortigen Unterbrechung des Roboterbetriebs führen.

Es dürfen sich keine Gegenstände innerhalb des Bewegungsbereiches des Roboters befinden, die zum Betrieb des Roboters nicht unbedingt notwendig sind (z. B. Werkzeuge usw.).

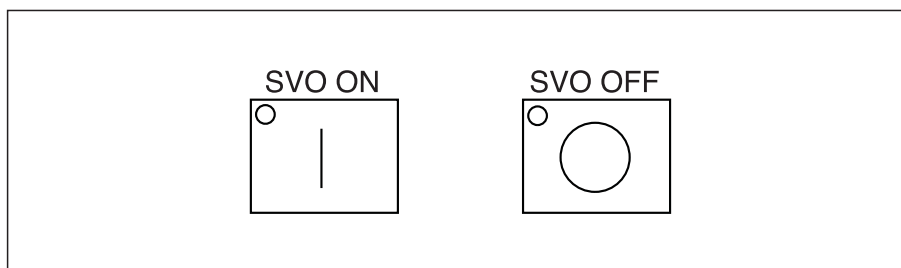
Überprüfen Sie die einwandfreie Funktion des Programms vor Ausführung des Automatikbetriebs im Schrittbetrieb.

- ① Stellen Sie den [ENABLE/DISABLE]-Schalter der Teaching Box auf „DISABLE“ und den [MODE]-Schalter des Steuergerätes auf „AUTO (Op.)“.



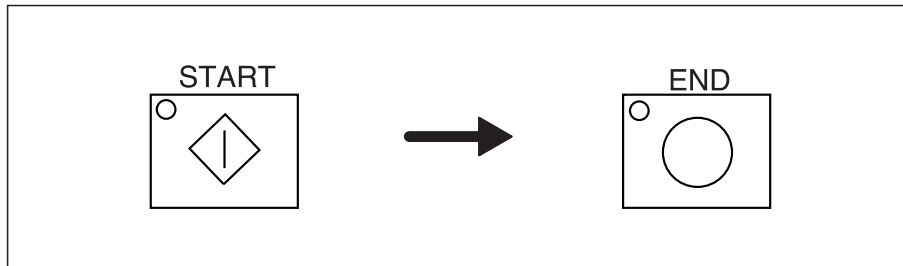
R000707C

- ② Betätigen Sie die [SVO ON]-Taste, um die Servospannung einzuschalten. Die LED auf der [SVO ON]-Taste leuchtet. Betätigen Sie die [SVO OFF]-Taste, um die Servospannung auszuschalten. Die LED auf der [SVO OFF]-Taste leuchtet.



R000715C

- ③ Betätigen Sie die [START]-Taste, nachdem die LED auf der [SVO ON]-Taste leuchtet, um den kontinuierlichen Automatikbetrieb zu starten. Bei Betätigung der [END]-Taste stoppt das Programm nach Ausführung eines Zyklus. Die LED blinkt während des Zyklusstopps. Ab Software-Version J1 bewirkt eine erneute Betätigung der [END]-Taste während eines Zyklusstopps die Rückkehr in den kontinuierlichen Betrieb.



R000709C

**GEFAHR:**

Prüfen Sie vor dem Starten des Automatikbetriebs, dass die korrekte Programmnummer ausgewählt wurde.

Unterbrechen Sie den Automatikbetrieb sofort durch Betätigung des NOT-HALT-Schalters, falls Ihnen Unregelmäßigkeiten auffallen.

Während des Automatikbetriebes darf die Spannungsversorgung des Steuergerätes nicht abgeschaltet werden, da das Programm durch einen Speicherfehler Schaden nehmen kann. Ist es erforderlich, den Roboter abrupt zu stoppen, betätigen Sie den NOT-HALT-Schalter.

3.10.4 Stoppen des Automatikbetriebs

Der Automatikbetrieb kann über das Steuergerät oder über die Teaching Box durch Betätigung der [STOP]-Taste gestoppt werden.

Bei Betätigung der [STOP]-Taste wird das Programm unterbrochen und der Roboter bis zum Stillstand abgebremst.

HINWEIS | Zur Ausführung der Stoppfunktion ist kein Betriebsrecht erforderlich.

3.10.5 Fortsetzung des Automatikbetriebs aus dem Stoppzustand

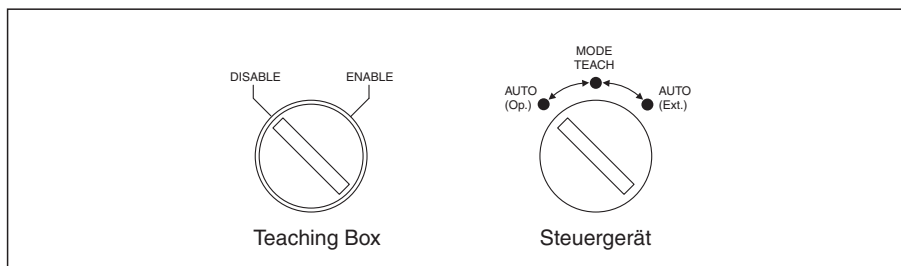
Zum Fortsetzen des Automatikbetriebs aus dem Stoppzustand führen Sie die Schritte aus, die im Absatz „Starten des Automatikbetriebs“ (siehe Seite 3-38) beschrieben werden.

3.10.6 Programm zurücksetzen

Der Stoppzustand des Programmes wird aufgehoben und die Programmsteuerung springt zum Programmmanfang.

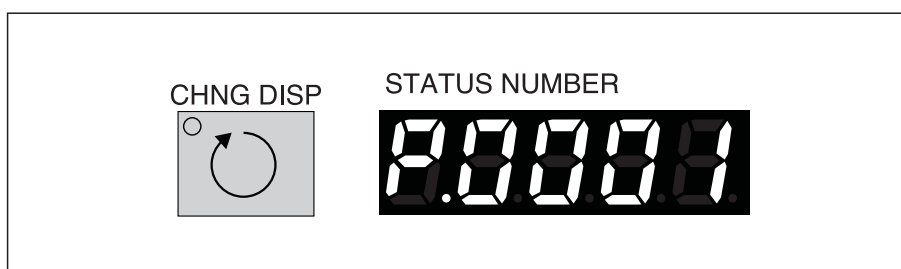
Rücksetzen über das Steuergerät

- ① Stellen Sie den [ENABLE/DISABLE]-Schalter der Teaching Box auf „DISABLE“ und den [MODE]-Schalter des Steuergerätes auf „AUTO (Op.)“.



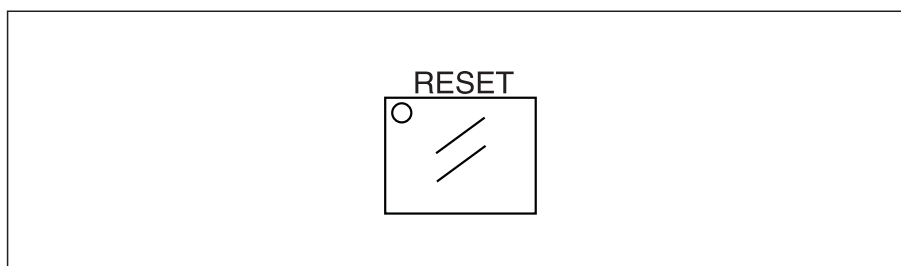
R000707C

- ② Betätigen Sie die [CHNG DISP]-Taste, um die Programmnummer anzuzeigen.



R000711C

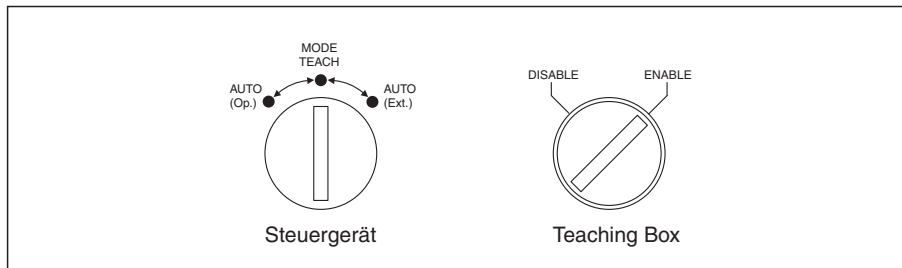
- ③ Betätigen Sie die [RESET]-Taste. Die STOP-LED erlischt.



R000712C

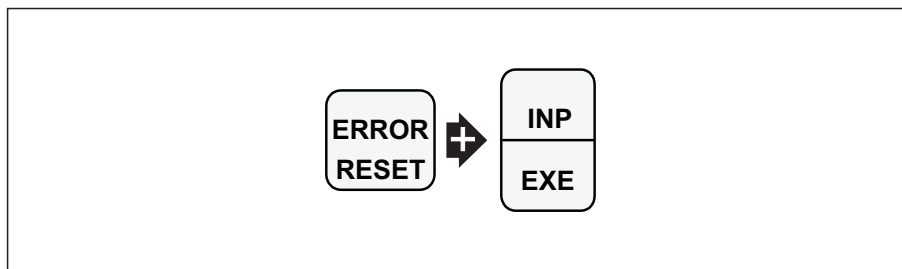
Rücksetzen über die Teaching Box

- ① Stellen Sie den [MODE]-Schalter des Steuergerätes auf „TEACH“ und den [ENABLE/DISABLE]-Schalter der Teaching Box auf „ENABLE“.



R000713C

- ② Betätigen Sie die [INP/EXE]-Taste, während Sie die [ERROR/RESET]-Taste gedrückt halten.



R000714C

HINWEISE









Während der Programmausführung kann ein Programm nicht zurückgesetzt werden. Das Programm muss zuerst gestoppt werden. Vor Ausführung des Rücksetzvorgangs über das Steuergerät muss die entsprechende Programmnummer im Display angezeigt werden.

Die LED der [STOP]-Taste erlischt, wenn das Programm zurückgesetzt ist.

3.11 Programmverwaltungsfunktionen

3.11.1 Programmverzeichnis anzeigen

Die Funktion ermöglicht die Anzeige der im Steuergerät gespeicherten Programme und deren Eigenschaften.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre><MENU> 1 .TEACH 2 .RUN 3 .FILE 4 .MONI 5 .MAINT 6 .SET</pre>		Das Menü FILE wird durch Eingabe der Ziffer „3“ aufgerufen.
②	<pre><FILE> 1 .DIR 2 .COPY 3 .RENAME 4 .DELETE</pre>		Der Menüpunkt DIR wird ausgewählt. Oben rechts erscheint die Anzahl der gespeicherten Programme.
③	<pre><DIR> 7 1 99-10-10 2 99-11-29 5 99-12-08</pre>	 	Mit den Cursortasten können die weiteren Programmnamen zur Anzeige gebracht werden.
④	<pre><DIR> 7 10 99-12-10 13 99-08-29 17 99-09-10</pre>		Es wird die Zeit der Programmerstellung angezeigt (Stunde, Minute, Sekunde). Über die Taste [DEL ←] gelangen Sie zur vorherigen Anzeige zurück.
⑤	<pre><DIR> 7 10 03:58:02 13 23:05:32 17 15:48:39</pre>		Es wird die Dateigröße in Bytes angezeigt. Oben rechts erscheint die Gesamtspeicherbelegung.
⑥	<pre><DIR> 25639 10 348 13 1978 17 3873</pre>		Es wird angezeigt, ob ein Programm geschützt ist.
⑦	<pre><DIR> PROTECT 10 OFF (0) 13 ON (1) 0:OFF 1:ON</pre>		Es wird angezeigt, ob eine Variable geschützt ist.
⑧	<pre><DIR> POS.PROTECT 10 ON (1) 13 ON (1) 0:OFF 1:ON</pre>		Eine detaillierte Beschreibung zum Schutz von Programmen finden Sie auf der nächsten Seite.

Tab. 3-26: Beispiel zum Anzeigen des Programmverzeichnisses


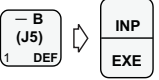
3.11.2 Programm schützen

Programmschutzfunktion

Die Funktion verhindert ein unbeabsichtigtes Löschen eines Programms und Programmänderungen.

Rufen Sie zuerst das Programmverzeichnis auf (siehe Seite 3-42).

Im folgenden Beispiel wird das Programm Nr. 2 geschützt:

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre><DIR> PROTECT 1 OFF (0) 2 OFF (0) 0:OFF 1:ON</pre>		Bewegen Sie den Cursor zu dem Programm, dessen Schreibschutz eingestellt werden soll.
②	<pre><DIR> PROTECT 1 OFF (0) 2 OFF (0) 0:OFF 1:ON</pre>		Der Programmschutz für Programm Nr. 2 wird eingeschaltet.
③	<pre><DIR> PROTECT 1 OFF (0) 2 ON (1) 0:OFF 1:ON</pre>		Programm Nr. 2 ist jetzt geschützt.

Tab. 3-27: Beispiel zum Schützen eines Programms

Beschreibung


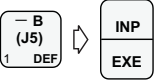
- Programme werden gegen die Operationen DELETE, RENAME und Programmänderungen geschützt.
- Beim Kopieren eines Programms wird der Schutzstatus nicht mitkopiert.
- Beim Löschen des Speichers über „INITIALIZATION“ (siehe Abschn. 3.13.2) wird der Schutzstatus ignoriert und das Programm gelöscht.

Variablenschutzfunktion

Die Funktion verhindert ein unbeabsichtigtes Löschen und Ändern von Variablen.

Rufen Sie zuerst das Programmverzeichnis auf (siehe Seite 3-42).

Im folgenden Beispiel werden die Variablen von Programm Nr. 2 geschützt:

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre><DIR> POS.PROTECT 1 OFF (0) 2 OFF (0) 0:OFF 1:ON</pre>		Bewegen Sie den Cursor zu dem Programm, dessen Variablen geschützt werden sollen.
②	<pre><DIR> POS.PROTECT 1 OFF (0) 2 OFF (0) 0:OFF 1:ON</pre>		Der Variablenschutz für Programm Nr. 2 wird eingeschaltet.
③	<pre><DIR> POS.PROTECT 1 OFF (0) 2 ON (1) 0:OFF 1:ON</pre>		Die Variablen in Programm Nr. 2 sind jetzt geschützt.

Tab. 3-28: Beispiel zum Schützen von Variablen





Beschreibung

- Variablen werden gegen das Überschreiben durch Positionsdaten, gegen Änderungen und gegen Substitutionen bei fehlerhafter Programmausführung geschützt.
- Beim Kopieren eines Programms wird der Schutzstatus nicht mitkopiert.
- Beim Löschen des Speichers über „INITIALIZATION“ (siehe Abschn. 3.13.2) wird der Schutzstatus ignoriert und die Variablen gelöscht.

3.11.3 Programm kopieren

Die Funktion dient zum Kopieren eines Roboterprogramms.

Im folgenden Beispiel wird das Programm Nr. 1 kopiert und unter dem neuen Programmnamen Nr. 5 nochmals abgespeichert:

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre><MENU> 1 .TEACH 2 .RUN 3 .FILE 4 .MONI 5 .MAINT 6 .SET</pre>		Das Menü FILE wird durch Eingabe der Ziffer „3“ aufgerufen.
②	<pre><FILE> 1 .DIR 2 .COPY 3 .RENAME 4 .DELETE</pre>		Der Menüpunkt COPY wird ausgewählt.
③	<pre><COPY> FROM () TO () INPUT SOURCE</pre>		Der Name (Nr. 1) des Programms, das kopiert werden soll, wird eingegeben.
④	<pre><COPY> FROM (1) TO (5) INPUT DEST.</pre>		Der neue Programmname (Nr. 5) wird eingegeben. Anschließend wird die Eingabe bestätigt und der Kopiervorgang wird ausgeführt.
⑤	<pre><FILE> 1 .DIR 2 .COPY 3 .RENAME 4 .DELETE</pre>		Nach Abschluss des Kopiervorgangs wird das Menü FILE angezeigt.

Tab. 3-29: Beispiel zum Kopieren eines Roboterprogramms





Beschreibung

- Es wird eine Fehlermeldung angezeigt, wenn derselbe Programmname zweimal angegeben wird, d. h. wenn Quelle und Ziel eines Kopiervorgangs identisch sind.
- Der Schutzstatus von Programmen und Variablen wird nicht mitkopiert.
- Es wird eine Fehlermeldung angezeigt, wenn das Zielprogramm belegt ist.

3.11.4 Programmnamen ändern

Die Funktion dient zum Ändern eines Programmnamens.

Im folgenden Beispiel wird der Programmname von „1“ auf „5“ geändert:

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre><MENU> 1 . TEACH 2 . RUN 3 . FILE 4 . MONI 5 . MAINT 6 . SET</pre>		Das Menü FILE wird durch Eingabe der Ziffer „3“ aufgerufen.
②	<pre><FILE> 1 . DIR 2 . COPY 3 . RENAME 4 . DELETE</pre>		Der Menüpunkt RENAME wird ausgewählt.
③	<pre><RENAME> FROM (1) TO () INPUT SOURCE</pre>		Der Name (Nr. 1) des Programms, das umbenannt werden soll, wird eingegeben.
④	<pre><RENAME> FROM (1) TO (5) INPUT DEST.</pre>		Der neue Programmname (Nr. 5) wird eingegeben. Anschließend wird die Eingabe bestätigt und der Änderungsvorgang wird ausgeführt.
⑤	<pre><FILE> 1 . DIR 2 . COPY 3 . RENAME 4 . DELETE</pre>		Nach Abschluss des Änderungsvorgangs wird das Menü FILE angezeigt.

Tab. 3-30: Beispiel zum Ändern eines Programmnamens



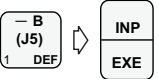
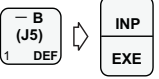
Beschreibung

- Wird derselbe Programmname zweimal angegeben, erfolgt eine Fehlermeldung.
- Der Programmname eines geschützten Programms oder eines Programms mit geschützten Variablen kann nicht geändert werden. Schalten Sie gegebenenfalls die Schutzfunktion aus.

3.11.5 Programm löschen

Die Funktion dient zum Löschen eines gespeicherten Programms.

Im folgenden Beispiel wird Programm Nr. 1 gelöscht:

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre><MENU> 1 . TEACH 2 . RUN 3 . FILE 4 . MONI 5 . MAINT 6 . SET</pre>		Das Menü FILE wird durch Eingabe der Ziffer „3“ aufgerufen.
②	<pre><FILE> 1 . DIR 2 . COPY 3 . RENAME 4 . DELETE</pre>		Der Menüpunkt DELETE wird ausgewählt.
③	<pre><DELETE> DELETE (1) INPUT DEL . FILE</pre>		Der Name (Nr. 1) des Programms, das gelöscht werden soll, wird eingegeben. Nach der Eingabe wird eine Bestätigungsabfrage angezeigt.
④	<pre><DELETE> DELETE 1 OK? (1) 1 : EXECUTE</pre>		Die Abfrage wird durch Eingabe der Ziffer „1“ bestätigt.
⑤	<pre><FILE> 1 . DIR 2 . COPY 3 . RENAME 4 . DELETE</pre>		Nach Abschluss des Änderungsvorgangs wird das Menü FILE angezeigt.

Tab. 3-31: Beispiel zum Löschen eines Programms

HINWEIS



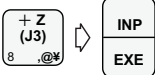
Ein geschütztes Programm oder eine Programm mit geschützten Variablen kann nicht gelöscht werden. Schalten Sie gegebenenfalls die Schutzfunktion aus.

3.12 Monitor-Funktionen

3.12.1 Monitor-Funktion für Eingangssignale

Die Funktion ermöglicht die Anzeige des Eingangssignalstatus.

Im folgenden Beispiel wird der Status der Eingangsbits 8 bis 15 angezeigt:

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre><MENU> 1 . TEACH 2 . RUN 3 . FILE 4 . MONI 5 . MAINT 6 . SET</pre>		Das Menü MONITOR wird durch Eingabe der Ziffer „4“ aufgerufen.
②	<pre><MONI> 1 . INPUT 2 . OUTPUT 3 . VAR 4 . ERROR</pre>		Der Menüpunkt INPUT wird ausgewählt.
③	<pre><INPUT> NUMBER (8) BIT: 76543210 DATA (00000000)</pre>		Die Ziffer „8“ wird eingegeben und anschließend wird die Eingabe bestätigt.
④	<pre><INPUT> NUMBER (8) BIT: 54321098 DATA (01001011)</pre>		Der Bitstatus der Eingangsbits 8 bis 15 wird angezeigt.

Tab. 3-32: Beispiel zum Überprüfen der Bitzustände der Eingangssignale



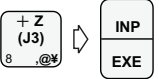

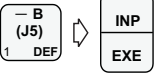
HINWEIS

Die Anzeige der Eingangsbitzustände kann auch ohne zugewiesene Betriebsrechte der Teaching Box erfolgen.

3.12.2 Monitor-Funktion für Ausgangssignale

Die Funktion ermöglicht die Anzeige und Einstellung der Ausgangssignalzustände.

Im folgenden Beispiel wird das 8. Ausgangsbit eingeschaltet:












Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre><MENU> 1 . TEACH 2 . RUN 3 . FILE 4 . MONI 5 . MAINT 6 . SET</pre>		Das Menü MONITOR wird durch Eingabe der Ziffer „4“ aufgerufen.
②	<pre><MONI> 1 . INPUT 2 . OUTPUT 3 . VAR 4 . ERROR</pre>		Der Menüpunkt OUTPUT wird ausgewählt.
③	<pre><OUTPUT> NUMBER (8) BIT: 76543210 DATA (00000000)</pre>		Die Ziffer „8“ wird eingegeben und anschließend wird die Eingabe bestätigt. Der Bitstatus der Ausgangsbits 8 bis 15 wird angezeigt.
④	<pre><OUTPUT> NUMBER (8) BIT: 54321098 DATA (01101000)</pre>		Bewegen Sie den Cursor zum 8. Bit.
⑤	<pre><OUTPUT> NUMBER (8) BIT: 54321098 DATA (01101001)</pre>		Der Signalzustand von Bit 8 wird auf „1“ gesetzt. Anschließend wird die Dateneingabe bestätigt.
⑥	<pre><OUTPUT> NUMBER (8) BIT: 54321098 DATA (01101001)</pre>		Der Signalzustand von Bit 8 ist auf „1“ gesetzt.

Tab. 3-33: Beispiel zum Einstellen der Bitzustände der Ausgangssignale

3.12.3 Monitor-Funktion für Variable

Die Funktion ermöglicht die Anzeige und Einstellung von Variablen.

Im folgenden Beispiel wird die numerische Variable M8 in Programm Nr. 1 von „2“ auf „5“ gesetzt:

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre><MENU> 1 . TEACH 2 . RUN 3 . FILE 4 . MONI 5 . MAINT 6 . SET</pre>		Das Menü MONITOR wird durch Eingabe der Ziffer „4“ aufgerufen.
②	<pre><MONI> 1 . INPUT 2 . OUTPUT 3 . VAR 4 . ERROR</pre>		Der Menüpunkt VAR wird ausgewählt.
③	<pre><VAR> (1) SELECT PROGRAM</pre>	 	Die Ziffer „1“ zur Auswahl des Programms Nr. 1 wird eingegeben.
④	<pre><VAR> V . NAME (M) DATA () SET V . NAME</pre>	   	Der Variablenname „M8“ wird eingegeben.
⑤	<pre><VAR> V . NAME (M8) DATA (+2) SET V . NAME</pre>	  	Der aktuelle Wert „+2“ wird durch den neuen Wert „+5“ überschrieben.
⑥	<pre><VAR> V . NAME (M8) DATA (+5) SET V . NAME</pre>		Der Wert der Variablen M8 ist jetzt „+5“.

Tab. 3-34: Beispiel zum Ändern von Variablenwerten




HINWEISE

Die Anzeige von Variablen kann auch ohne zugewiesene Betriebsrechte der Teaching Box erfolgen.

Roboterstatusvariablen können nicht direkt angezeigt werden. Schreiben Sie den Wert zuerst in eine Programmvariable und zeigen Sie die Programmvariable an.

3.12.4 Liste der aufgetretenen Fehlermeldungen

Die bisher aufgetretenen Fehlermeldungen werden in einer Liste angezeigt. Diese Funktion ist für eine Störungssuche sehr hilfreich.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre><MENU> 1 .TEACH 2 .RUN 3 .FILE 4 .MONI 5 .MAINT 6 .SET</pre>		Das Menü MONITOR wird durch Eingabe der Ziffer „4“ aufgerufen.
②	<pre><MONI> 1 .INPUT 2 .OUTPUT 3 .VAR 4 .ERROR 5 .REG</pre>		Der Menüpunkt ERROR wird ausgewählt.
③	<pre><ERROR>-1 99-08-10 10:20 2000 SERVO OFF</pre>		Mit den Cursortasten können die weiteren Fehlermeldungen zur Anzeige gebracht werden.
④	<pre><ERROR>-2 99-08-10 10:12 3110 ARGUMENT VALUE RANGE OVER</pre>		Anzeige des folgenden Fehlers

Tab. 3-35: Beispiel zum Anzeigen der Liste der aufgetretenen Fehlermeldungen

HINWEIS



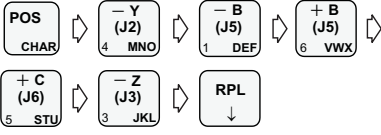


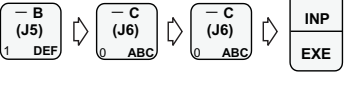
Die Anzeige von Fehlermeldungen kann auch ohne zugewiesene Betriebsrechte der Teaching Box erfolgen.

3.13 Zusatzfunktionen

3.13.1 Parameter einstellen

Die Funktionen der parallelen Ein-/Ausgabeschnittstelle, die Werkzeuglänge usw. sind über Parameter festgelegt. Diese Parameter können im Teach-Modus eingestellt werden.

Im folgenden Beispiel wird der Wert für die Z-Achse des Parameters „MEXTL (Werkzeugdaten)“ von 0 auf 100 mm gesetzt:

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre><MENU> 1 . TEACH 2 . RUN 3 . FILE 4 . MONI 5 . MAINT 6 . SET</pre>		Das Menü MAINTENANCE wird durch Eingabe der Ziffer „5“ aufgerufen.
②	<pre><MAINT> 1 . PARAM 2 . INIT 3 . BRAKE 4 . ORIGIN 5 . POWER</pre>		Das Menü PARAMETER wird ausgewählt.
③	<pre><PARAM> () () () SET PARAM.NAME</pre>		Die Zeichen „MEXTL“ werden eingegeben. Anschließend wird der Cursor zum Eingabefeld für die Achsennummer bewegt.
④	<pre><PARAM> (MEXTL) () () SET ELEMENT</pre>		Es wird die Achsennummer „3“ eingegeben.
⑤	<pre><PARAM> (MEXTL) (3) (+0.00) SET ELEMENT</pre>		Der aktuelle Wert „+0.00“ wird angezeigt. Anschließend wird der Cursor zum Feld für die Eingabe des neuen Wertes bewegt.
⑥	<pre><PARAM> (MEXTL) (3) (+0.00) SET ELEMENT</pre>		Der neue Wert „+100“ wird eingegeben.
⑦	<pre><PARAM> (MEXTL) (3) (+100.00) SET ELEMENT</pre>		Schalten Sie die Spannungsversorgung aus und wieder ein, damit der neue Wert wirksam werden kann.



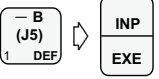
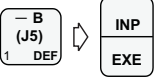
Tab. 3-36: Beispiel zum Einstellen eines Parameters

HINWEIS

Damit ein neuer Parameterwert wirksam wird, muss die Spannungsversorgung des Steuergerätes aus- und wieder eingeschaltet werden.

3.13.2 Alle gespeicherten Programme löschen

Diese Funktion löscht alle gespeicherten Programme mit einem Bedienschritt:

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre><MENU> 1 . TEACH 2 . RUN 3 . FILE 4 . MONI 5 . MAINT 6 . SET</pre>		Das Menü MAINTENANCE wird durch Eingabe der Ziffer „5“ aufgerufen.
②	<pre><MAINT> 1 . PARAM 2 . INIT 3 . BRAKE 4 . ORIGIN 5 . POWER</pre>		Das Menü INIT wird ausgewählt.
③	<pre><INIT> INIT (█) 1 . PROGRAM 2 . BATT.</pre>		Die Ziffer „1“ wird eingegeben. Anschließend wird die Eingabe bestätigt.
④	<pre><INIT> PROGRAM OK? (█) 1 : EXECUTE</pre>		Die Abfrage wird durch Eingabe der Ziffer „1“ bestätigt. Der Löschvorgang wird ausgeführt.
⑤	<pre><INIT> INIT (█) 1 . PROGRAM 2 . BATT.</pre>		Nach Ausführung des Löschvorgangs wird das Menü INIT angezeigt.

Tab. 3-37: Beispiel zum Löschen aller gespeicherten Programme

HINWEIS

Es werden auch geschützte Programme und Programme mit geschützten Variablen gelöscht.

3.13.3 Gelenkbremsen lösen

Die Bremsen für die Robotergelenke können bei ausgeschalteter Servospannung gelöst werden. Der Roboterarm kann dann direkt manuell bewegt werden.




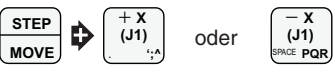
ACHTUNG:
Beachten Sie, dass der Roboterarm aufgrund des Eigengewichts bei gelösten Bremsen heruntersinken kann. Unterstützen Sie daher den Roboterarm vor dem Lösen der Bremsen.

Schalten Sie erst über die Teaching Box mit dem SERVO-Menü die Servos aus (siehe Abschn. 3.7). Anschließend bringen Sie den Dreistufenschalter in die Mittelstellung und folgen den Anweisungen aus Tab. 3-39.

Roboter	Achsen					
	1	2	3	4	5	6
RP-1AH/3AH/5AH	✓	✓	✓	✓	—	—
RH-5AH/10AH/15AH	—	—	✓	—	—	—
RH-6SH/12SH	—	—	✓	—	—	—
RV-1A/2AJ/2A/3AJ	✓	✓	✓	—	✓	—
RV-3SB/3SJB/6S/6SL/12S/12SL	✓	✓	✓	✓	✓	✓
RV-4A/3AL	✓	✓	✓	✓	✓	✓
RV-5AJ/4AJL	✓	✓	✓	—	✓	✓

Tab. 3-38: Gelenkbremsen der verschiedenen Robotertypen



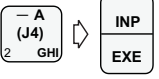
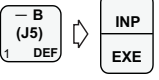
Im folgenden Beispiel wird beim Roboter RP-1AH die Gelenkbremse der Achse J1 gelöst:

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre><MENU> 1. TEACH 2. RUN 3. FILE 4. MONI 5. MAINT 6. SET</pre>		Das Menü MAINTENANCE wird durch Eingabe der Ziffer „5“ aufgerufen.
②	<pre><MAINT> 1. PARAM 2. INIT 3. BRAKE 4. ORIGIN 5. POWER</pre>		Das Menü BRAKE wird ausgewählt.
③	<pre><BRAKE>12345678 BRAKE (00000000) 0:LOCK 1:FREE</pre>		Es werden die Achsen auf „1“ gesetzt, deren Bremsen gelöst werden sollen.
④	<pre><BRAKE>12345678 BRAKE (10000000) 0:LOCK 1:FREE</pre>	 <p>Dreistufentaster betätigen</p>	Die Bremsen sind gelöst, solange die Tasten betätigt sind. Wird eine der Tasten losgelassen, sind alle Bremsen wieder aktiv.

Tab. 3-39: Beispiel zum Lösen der Gelenkbremsen

3.13.4 Batteriezähler zurücksetzen

Der Batteriezähler erfasst die Betriebszeit der Batterien im Roboterarm und im Steuergerät und dient als Referenz für die Warnmeldung zum Austausch der Batterien. Setzen Sie daher nach dem Austauschen der Batterien unbedingt den Batteriezähler zurück.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre><MENU> 1 . TEACH 2 . RUN 3 . FILE 4 . MONI 5 . MAINT 6 . SET</pre>		Das Menü MAINTENANCE wird durch Eingabe der Ziffer „5“ aufgerufen.
②	<pre><MAINT> 1 . PARAM 2 . INIT 3 . BRAKE 4 . ORIGIN 5 . POWER</pre>		Das Menü INIT wird ausgewählt.
③	<pre><INIT> INIT (█) 1 . PROGRAM 2 . BATT.</pre>		Die Ziffer „2“ wird eingegeben. Anschließend wird die Eingabe bestätigt.
④	<pre><INIT> BATT. OK? (█) 1 : EXECUTE</pre>		Die Abfrage wird durch Eingabe der Ziffer „1“ bestätigt. Der Rücksetzvorgang wird ausgeführt.
⑤	<pre><INIT> INIT (█) 1 . PROGRAM 2 . BATT.</pre>		Nach Ausführung des Rücksetzvorgangs wird das Menü INIT angezeigt.

Tab. 3-40: Beispiel zum Zurücksetzen des Batteriezählers



HINWEISE

Der Batteriezähler zählt rückwärts. Gezählt wird nur im ausgeschalteten Zustand.

Bei verbrauchter Batterie wird eine Warnmeldung ausgegeben. Die Gebrauchszeit der Batterie wird ab dem Zurücksetzen des Batteriezählers erfasst. Setzen Sie daher nach einem Austausch der Batterien den Batteriezähler zurück, um eine korrekte Erfassung der Gebrauchszeit zu gewährleisten.

3.13.5 Batterie und Einschaltzeit anzeigen

Die Betriebszeit des Steuergerätes und die verbleibende Lebensdauer der Batterie in Stunden werden angezeigt.



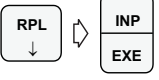
Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre><MENU> 1 . TEACH 2 . RUN 3 . FILE 4 . MONI 5 . MAINT 6 . SET</pre>		Das Menü MAINTENANCE wird durch Eingabe der Ziffer „5“ aufgerufen.
②	<pre><MAINT> 1 . PARAM 2 . INIT 3 . BRAKE 4 . ORIGIN 5 . POWER</pre>		Das Menü POWER wird ausgewählt.
③	<pre><HOUR DATA> Hr POWER ON: 1258 BATTERY: 4649</pre>		Es werden die Betriebszeit und die verbleibende Lebensdauer der Batterie angezeigt.

Tab. 3-41: Beispiel zum Anzeigen der Einschaltzeit und Restpufferung

3.13.6 Uhrzeit und Datum einstellen

Das Steuergerät ist mit einer internen Uhr für Uhrzeit- und Datumsfunktionen ausgerüstet. Diese Datumsfunktion nutzt das Steuergerät z. B. zum Eintragen des Erstellungsdatums in ein Programm.

Die interne Uhr sollten Sie nach der erstmaligen Inbetriebnahme des Roboters und danach in regelmäßigen Abständen kontrollieren. Nachfolgend wird das Einstellen von Uhrzeit und Datum beschrieben:

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre><MENU> 1 . TEACH 2 . RUN 3 . FILE 4 . MONI 5 . MAINT 6 . SET</pre>		Das Menü SET wird durch Eingabe der Ziffer „6“ aufgerufen.
②	<pre><SET> 1 . CLOCK</pre>		Im Menü SET wird der Menüpunkt CLOCK aufgerufen. Nach der Tastenbetätigung wird die aktuelle Uhrzeit und das aktuelle Datum angezeigt.
③	<pre><CLOCK> DATE (99-12-07) TIME (23:58:17) INPUT DATE</pre>		Stellen Sie das richtige Datum ein. Bestätigen Sie die Eingabe mit der [INP/EXE]-Taste. Wechseln Sie mit der Taste [RPL ↓] in die Zeile zur Einstellung der Uhrzeit.
③	<pre><CLOCK> DATE (99-10-07) TIME (23:58:17) INPUT DATE</pre>		Die Einstellung der Uhrzeit erfolgt in derselben Weise wie die Einstellung des Datums.

Tab. 3-42: Beispiel zum Einstellen von Uhrzeit/Datum

4 MELFA-BASIC-IV-Programmierung

In diesem Kapitel finden Sie eine Einführung in die Programmiersprache MELFA-BASIC IV. Eine detaillierte Beschreibung der einzelnen Befehle finden Sie in Kapitel 6 „MELFA-BASIC-IV-Befehle“.

4.1 Funktionsübersicht

Nr.	Zuordnung	Beschreibung	Befehl
1	Abschn. 4.3 „Steuerung der Roboterbewegung“	Abschn. 4.3.1 „Gelenk-Interpolation“	MOV
2		Abschn. 4.3.2 „Linear-Interpolation“	MVS
3		Abschn. 4.3.3 „Kreis-Interpolation“	MVR, MVR2, MVR3, MVC
4		Abschn. 4.3.4 „Kontinuierliche Bewegung“	CNT
5		Abschn. 4.3.5 „Beschleunigungs-/ Bremszeit und Geschwindigkeit“	ACCEL, OADL
6		Abschn. 4.3.6 „Feinpositionierung“	FINE, MOV, DLY
7		Abschn. 4.3.7 „Verfahrweggenauigkeit“	PREC
8		Abschn. 4.3.8 „Hand- und Werkzeug- steuerung“	HOPEN, HCLOSE, TOOL
9	Abschn. 4.4 „Palettierung“	—	DEF PLT, PLT
10	Abschn. 4.5 „Programmsteuerung“	Abschn. 4.5.1 „Verzweigungen und Wartezeit“	GOTO, IF THEN ELSE, WAIT usw.
11		Abschn. 4.5.2 „Programmschleife“	FOR NEXT, WHILE WEND
12		Abschn. 4.5.3 „Interrupt“	DEF ACT, ACT
13		Abschn. 4.5.4 „Unterprogramm“	GOSUB, CALLP, ON GSOU B usw.
14		Abschn. 4.5.5 „Timer“	DLY
15		Abschn. 4.5.6 „Stopp“	END (1 Zyklus Pause), HLT
16	Abschn. 4.6 „Ein- und Ausgabe externer Signale“	Abschn. 4.6.1 „Eingangssignale“	M_IN, M_INB, M_INW usw.
17		Ausgangssignale	M_OUT, M_OUTB, M_OUTW usw.
18	Abschn. 4.7 „Kommunikation“	—	OPEN, CLOSE, PRINT, INPUT usw.
19	Abschn. 4.8 „Ausdrücke und Operationen“	Abs. 4.8.1 „Übersicht der Operationen“	+; -, *, /, <>, <, > usw.
20		Abs. 4.8.2 „Relative Konvertierung (Multiplikation)“	P1 * P2
21		Abs. 4.8.2 „Relative Konvertierung (Addition)“	P1 + P2
22	Abschn. 4.9 „Angehängte Anweisung“	—	WTH, WTHIF

Tab. 4-1: Übersicht der MELFA-BASIC-IV-Funktionen

4.2 Programmaufbau

In diesem Abschnitt werden die wichtigsten Elemente zum Aufbau eines Programms erläutert. Eine detaillierte Erklärung der einzelnen Begriffe finden Sie im Kap. 5.

4.2.1 Programmname

Ein Programmname darf aus maximal 12 Zeichen bestehen. Auf der Anzeige des Steuergeräts können jedoch nur bis zu 4 Zeichen dargestellt werden. Es empfiehlt sich daher, bei der Vergabe von Programmnamen nur 4 Zeichen zu verwenden.

Verwendbare Zeichen für Programmnamen	
Buchstaben	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z ^①
Zahlen	0 1 2 3 4 5 6 7 8 9

Tab. 4-2: Für Programmnamen zugelassene Zeichen

- ① Verwenden Sie in Programmnamen nur Großbuchstaben. Die Verwendung von Kleinbuchstaben kann zu einer fehlerhaften Abarbeitung des Programmes führen.

HINWEIS

Es ist nicht möglich ein Programm mit einem Programmnamen, der aus mehr als 4 Zeichen besteht, über das Steuergerät auszuwählen. Soll ein auszuführendes Programm über ein externes Ausgangssignal ausgewählt werden, sind im Programmnamen nur Zahlen zu verwenden. Bei einem Programm, dass über den Befehl CALLP ausgeführt werden soll, darf der Programmname aus mehr als 4 Zeichen bestehen.

4.2.2 Anweisung

In diesem Abschnitt werden die Elemente zum Aufbau einer Anweisung erläutert.

10	MOV P1	WTH M_OUT(17) = 1
①	② ③	④

- ① **Zeilennummer**
Für die einwandfreie Funktion eines Programms müssen die Zeilennummern in aufsteigender Reihenfolge angeordnet sein. Das Programm wird in dieser Reihenfolge abgearbeitet.
- ② **Befehl**
Der Befehl legt die Aktion des Roboters fest.
- ③ **Befehlsparameter**
Der Befehlsparameter kann z. B. eine Variable oder ein Wert sein.
- ④ **Angehängte Anweisung**
Bei Interpolationsbefehlen ist es möglich, eine Verknüpfung an die Anweisung anzuhängen. Durch Anhängen einer Verknüpfung können bestimmte Befehle parallel zum Interpolationsbefehl ausgeführt werden.

4.2.3 Variable

Folgende Variablen können in einem Programm verwendet werden:

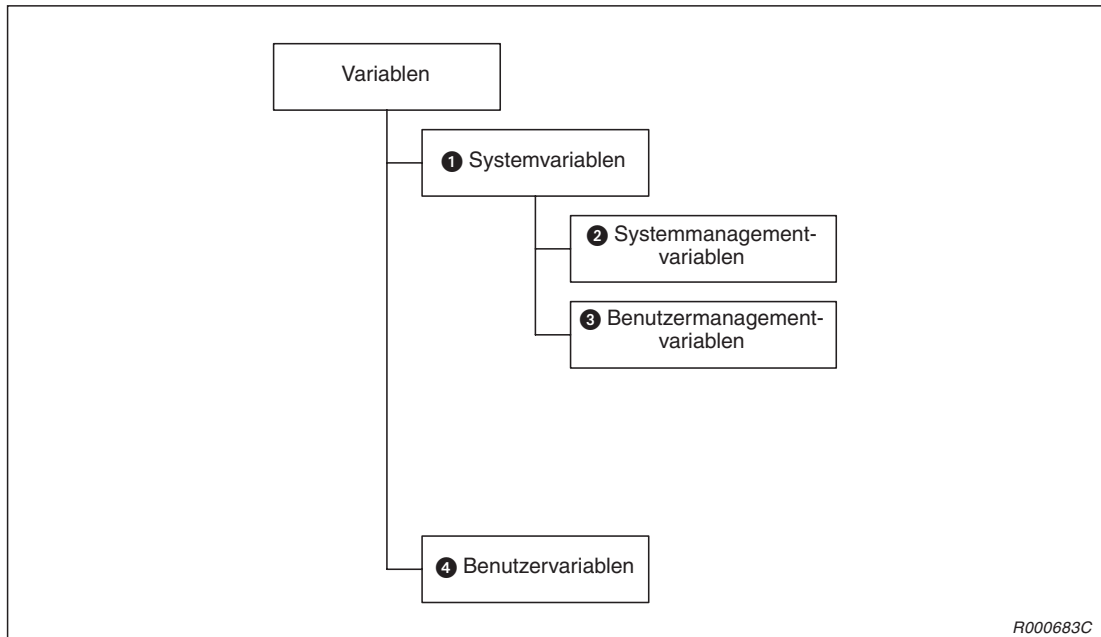


Abb. 4-1: Einteilung der Variablen

- ① Systemvariablen sind durch einen Variablennamen und einen gespeicherten Wert definiert.
- ② Systemmanagementvariablen können nur gelesen werden.
Beispiel: P_CURR
In dieser Variablen wird die aktuelle Position des Roboters ständig gespeichert.
- ③ Benutzermanagementvariablen können gelesen und geschrieben werden. Eingangssignale können nur gelesen werden.
Beispiel: M_OUT(17) = 1: Ausgangsbit 17 einschalten
M1 = M_IN(20): Schreibe den Wert des Eingangsbits 20 in die numerische Variable M1.
- ④ Benutzervariablen sind durch einen Variablennamen und den Verwendungszweck definiert.

Jede der oben aufgeführten Variablentypen ist in die folgenden Gruppen eingeteilt:

- **Positionsvariablen**
 Eine Positionsvariable enthält die kartesischen Koordinaten des Roboters. Der Variablenname beginnt mit „P“.
 Beispiel: `MOV P1` Der Roboter fährt die Position an, die in der Variablen P1 abgespeichert ist.

- **Gelenkvariablen**
 Eine Gelenkvariable enthält die Winkelwerte der Robotergelenke. Der Variablenname beginnt mit „J“.
 Beispiel: `MOV J1` Der Roboter fährt die Position an, die in der Variablen J1 abgespeichert ist.

- **Numerische Variablen**
 Eine numerische Variable enthält einen numerischen Wert (Integer, Reelle Zahl, usw.). Der Variablenname beginnt mit „M“.
 Beispiel: `M1 = 1` Der Wert „1“ wird in die Variable M1 geschrieben.

- **Zeichenkettenvariablen**
 Eine Zeichenkettenvariable enthält eine Zeichenkette. Dem Variablennamen folgt das Zeichen „\$“.
 Beispiel: `C1$ = "ERROR"` Die Zeichenkette „ERROR“ wird in die Variable C1\$ geschrieben.

4.3 Steuerung der Roboterbewegung

4.3.1 Gelenk-Interpolation

Die Handspitze wird mittels Gelenk-Interpolation zu einer festgelegten Position bewegt.

Erläuterung

Befehl	Beschreibung
MOV	Bewegt die Handspitze mittels Gelenk-Interpolation zu einer festgelegten Position Über eine TYPE-Anweisung kann der Interpolationstyp festgelegt werden. Die Verknüpfungen WTH oder WTHIF erlauben das Anhängen einer Anweisung.

Anweisungsbeispiele

MOV P1	Position P1 anfahren
MOV P1 + P2	Position anfahren, die sich aus der Addition der Koordinaten der Positionen P1 und P2 ergibt
MOV P1 * P2	Position anfahren, die sich aus der relativen Konvertierung von P1 zu P2 ergibt
MOV P1, -50	Position anfahren, die 50 mm in Werkzeuglängsrichtung von der Position P1 entfernt ist (siehe Achtungshinweis)
MOV P1 WTH M_OUT(17) = 1	Position P1 anfahren und Ausgangsbit 17 auf „1“ setzen
MOV P1 WTHIF M_IN(20) = 1, SKIP	Wird beim Anfahren der Position P1 das Eingangsbit 20 auf „1“ gesetzt, wird die Verfahrbewegung unterbrochen und das Programm bis zum nächsten Stopp fortgesetzt.
MOV P1 Type 1, 0	Position P1 indirekt (oder direkt) anfahren, wenn die Achsendrehungen größer als 180° sind (Grundeinstellung: indirekte Anfahrt)

Direkte und indirekte Anfahrt

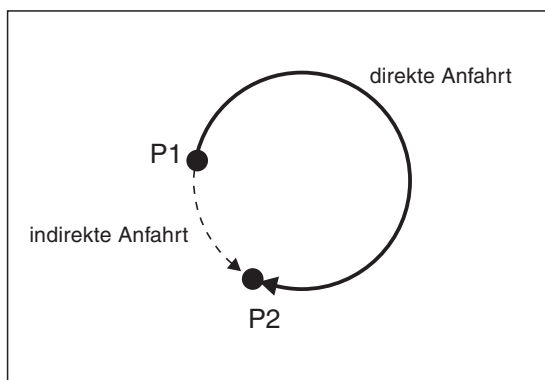


Abb. 4-2:

Direkte und indirekte Anfahrt einer Position

R000916C



ACHTUNG:

Die Richtung des Verfahrwegs im Werkzeugkoordinatensystem hängt vom Werkzeugkoordinatensystem des Roboters ab. Detaillierte Informationen zum Werkzeugkoordinatensystem finden Sie im Technischen Handbuch des Roboters.

Programmbeispiel

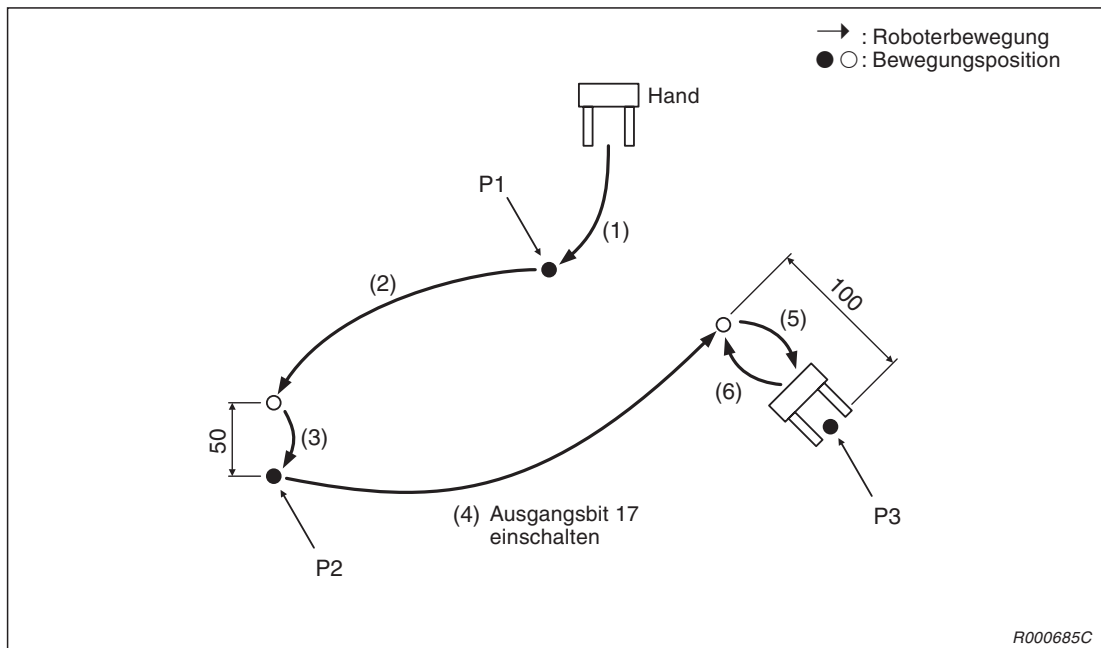


Abb. 4-3: Verlauf des Verfahrensweges bei Gelenk-Interpolation

10	MOV P1	Position P1 anfahren
20	MOV P2, -50	Position anfahren, die 50 mm in Werkzeuglängsrichtung von der Position P2 entfernt ist (siehe Achtungshinweis)
30	MOV P2	Position P2 anfahren
40	MOV P3, -100 WTH M_OUT(17) = 1	Position anfahren, die 100 mm in Werkzeuglängsrichtung von Position P3 entfernt ist und Ausgangsbit 17 auf „1“ setzen
50	MOV P3	Position P3 anfahren
60	MOV P3, -100	Position anfahren, die 100 mm in Werkzeuglängsrichtung von Position P3 entfernt ist
70	END	Programmende



ACHTUNG:

Die Richtung des Verfahrensweges im Werkzeugkoordinatensystem hängt vom Werkzeugkoordinatensystem des Roboters ab. Detaillierte Informationen zum Werkzeugkoordinatensystem finden Sie im Technischen Handbuch des Roboters.

Befehl steht in Beziehung zu folgenden Funktionen:

Festlegung der Verfahrensgeschwindigkeit	⇒	Abschn. 4.3.5
Festlegung der Beschleunigungs-/Bremszeit	⇒	Abschn. 4.3.5
Feinpositionierung	⇒	Abschn. 4.3.6
Kontinuierliche Bewegung	⇒	Abschn. 4.3.4
Linear-Interpolation	⇒	Abschn. 4.3.2
Kreis-Interpolation	⇒	Abschn. 4.3.3
Angehängte Anweisung	⇒	Abschn. 4.9

4.3.2 Linear-Interpolation

Die Handspitze wird mittels Linear-Interpolation zu einer festgelegten Position bewegt.

Erläuterung

Befehl	Beschreibung
MVS	Bewegt die Handspitze mittels Linear-Interpolation zu einer festgelegten Position Über eine TYPE-Anweisung kann der Interpolationstyp festgelegt werden. Die Verknüpfungen WTH oder WTHIF erlauben das Anhängen einer Anweisung.

Anweisungsbeispiele

MVS P1	Position P1 anfahren
MVS P1 + P2	Position anfahren, die sich aus der Addition der Koordinaten der Positionen P1 und P2 ergibt
MVS P1 * P2	Position anfahren, die sich aus der relativen Konvertierung von P1 zu P2 ergibt
MVS P1, -50	Position anfahren, die 50 mm in Werkzeuglängsrichtung von der Position P1 entfernt ist (siehe Achtungshinweis)
MVS, -50	Position anfahren, die 50 mm in Werkzeuglängsrichtung von der aktuellen Position entfernt ist (siehe Achtungshinweis)
MVS P1 WTH M_OUT(17) = 1	Position P1 anfahren und Ausgangsbit 17 auf „1“ setzen
MVS P1 WTHIF M_IN(20) = 1, SKIP	Wird beim Anfahren der Position P1 das Eingangsbit 20 auf „1“ gesetzt, wird die Verfahrbewegung unterbrochen und das Programm wird in der nächsten Zeile fortgesetzt.
MVS P1, TYPE 0, 0	Position P1 mittels Drehung anfahren
MVS P1, TYPE 0, 1	Position P1 mittels orthogonaler 3-Achsen-Interpolation anfahren



ACHTUNG:

Die Richtung des Verfahrwegs im Werkzeugkoordinatensystem hängt vom Werkzeugkoordinatensystem des Roboters ab. Detaillierte Informationen zum Werkzeugkoordinatensystem finden Sie im Technischen Handbuch des Roboters.

Befehl steht in Beziehung zu folgenden Funktionen:

Festlegung der Verfahrgeschwindigkeit	⇒	Abschn. 4.3.5
Festlegung der Beschleunigungs-/Bremszeit	⇒	Abschn. 4.3.5
Feinpositionierung	⇒	Abschn. 4.3.6
Kontinuierliche Bewegung	⇒	Abschn. 4.3.4
Gelenk-Interpolation	⇒	Abschn. 4.3.1
Kreis-Interpolation	⇒	Abschn. 4.3.3
Angehängte Anweisung	⇒	Abschn. 4.9

4.3.3 Kreis-Interpolation

Die Handspitze wird mittels 3D-Kreis-Interpolation entlang eines durch 3 Punkte festgelegten Kreises zu einer festgelegten Position bewegt. Entspricht die aktuelle Position nicht der Startposition, wird die Startposition mittels Linear-Interpolation angefahren.

Erläuterung

Befehl	Beschreibung
MVR	Bewegt die Handspitze mittels 3D-Kreis-Interpolation entlang eines durch die Startposition, Zwischenposition und Endposition festgelegten Kreisbogens Über eine TYPE-Anweisung kann der Interpolationstyp festgelegt werden. Die Verknüpfungen WTH oder WTHIF erlauben das Anhängen einer Anweisung.
MVR 2	Bewegt die Handspitze mittels 3D-Kreis-Interpolation von der Startposition zur Endposition Der Kreisbogen wird durch die Startposition, die Referenzposition und die Endposition festgelegt. Die Roboterbewegung geht dabei nicht durch den Referenzpunkt. Über eine TYPE-Anweisung kann der Interpolationstyp festgelegt werden. Die Verknüpfungen WTH oder WTHIF erlauben das Anhängen einer Anweisung.
MVR 3	Bewegt die Handspitze mittels 3D-Kreis-Interpolation von der Startposition zur Endposition Der Kreisbogen wird durch die Startposition, den Mittelpunkt und die Endposition festgelegt. Der Zentriwinkel zwischen Start- und Endposition liegt dabei zwischen 0° und 180°. Über eine TYPE-Anweisung kann der Interpolationstyp festgelegt werden. Die Verknüpfungen WTH oder WTHIF erlauben das Anhängen einer Anweisung.
MVC	Bewegt die Handspitze mittels 3D-Kreis-Interpolation entlang eines durch Startposition (Endposition), Zwischenposition 1, Zwischenposition 2 und Endposition festgelegten Kreisbogens Mit Hilfe der Verknüpfungen WTH oder WTHIF kann eine Anweisung angehängt werden.

Anweisungsbeispiele

MVR P1, P2, P3

Bewegung entlang des Kreisbogens
P1 → P2 → P3

MVR P1, P2, P3 WTH M_OUT(17) = 1

Bewegung entlang des Kreisbogens
P1 → P2 → P3 und Ausgangsbit 17
auf „1“ setzen

MVR P1, P2, P3 WTHIF M_IN(20) = 1, SKIP

Fährt entlang des Kreisbogens
P1 → P2 → P3 und unterbricht die
Bewegung, wenn Eingangsbit 20 auf
„1“ gesetzt wird
Die Programmsteuerung springt in
die nächste Zeile.

MVR P1, P2, P3 TYPE 0, 1

Bewegung entlang des Kreisbogens
P1 → P2 → P3

MVR2 P1, P3, P11

Bewegung entlang des Kreisbogens
von P1 nach P3, ohne die Referenz-
position P11 zu durchlaufen

MVR3 P1, P3, P10

Bewegung entlang des Kreisbogens
von P1 nach P3 in Richtung des
kleineren Zentriwinkels
P10 ist der Mittelpunkt.

MVC P1, P2, P3

Bewegung entlang des Kreisbogens
P1 → P2 → P3 → P1

Programmbeispiel

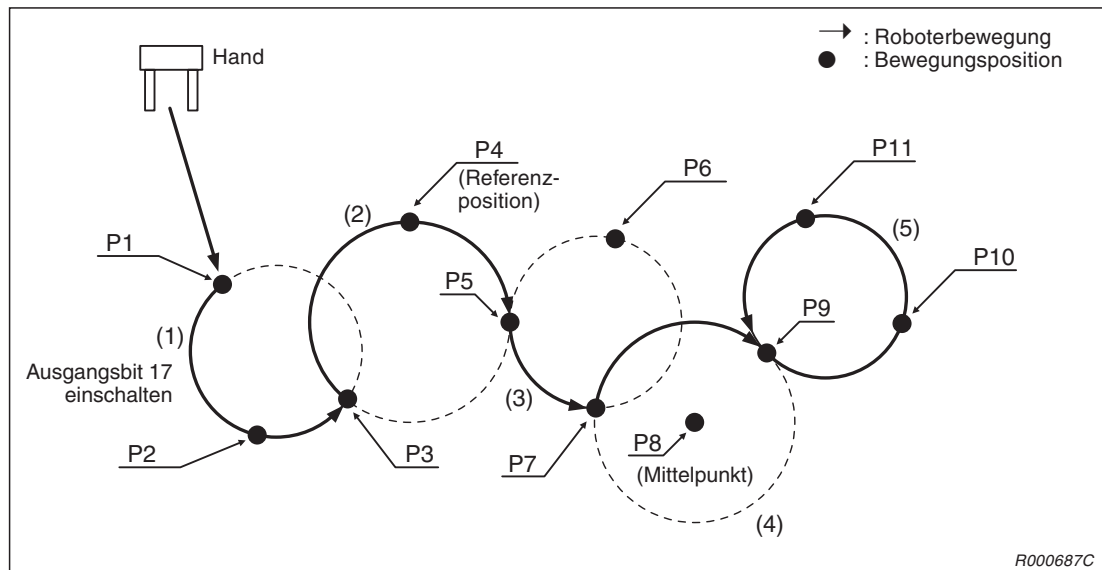


Abb. 4-5 Verlauf des Fahrweges bei Kreis-Interpolation

<p>10 MVR P1, P2, P3 WTH M_OUT(18) = 1</p> <p>20 MVR P3, P4, P5</p> <p>30 MVR2 P5, P7, P6</p> <p>40 MVR3 P7, P9, P8</p> <p>50 MVC P9, P10, P11</p> <p>60 END</p>	<p>Bewegung entlang des Kreisbogens P1 → P2 → P3 Die aktuelle Position entspricht nicht der Startposition. Der Roboter bewegt sich also zuerst mittels Linear-Interpolation zur Startposition (P1). Mit Beginn der Kreis-Interpolation wird das Ausgangsbit 18 auf „1“ gesetzt. Bewegung entlang des Kreisbogens P3 → P4 → P5 Bewegung entlang des Kreisbogens von P5 nach P7, ohne die Referenzposition P6 zu durchlaufen Bewegung entlang des Kreisbogens von P7 nach P9 in Richtung des kleineren Zentriwinkels P8 ist der Mittelpunkt. Bewegung entlang des Kreisbogens P9 → P10 → P11 → P9 Entspricht die aktuelle Position nicht der Startposition, wird die Startposition mittels Linear-Interpolation angefahren (1 Zyklus). Programmende</p>
------------------------------------------------------------------------------------------------------------------------------------------------------------------	-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Befehl steht in Beziehung zu folgenden Funktionen:

Festlegung der Fahrweggeschwindigkeit	⇒	Abschn. 4.3.5
Festlegung der Beschleunigungs-/Bremszeit	⇒	Abschn. 4.3.5
Feinpositionierung	⇒	Abschn. 4.3.6
Kontinuierliche Bewegung	⇒	Abschn. 4.3.4
Gelenk-Interpolation	⇒	Abschn. 4.3.1
Linear-Interpolation	⇒	Abschn. 4.3.2
Angehängte Anweisung	⇒	Abschn. 4.9

4.3.4 Kontinuierliche Bewegung

Bei freigegebener CNT-Einstellung fährt der Roboter die festgelegten Positionen ohne zu stoppen an. Der CNT-Befehl definiert den Start- bzw. Endpunkt der kontinuierlichen Bewegung. Die Geschwindigkeit kann während der kontinuierlichen Bewegung verändert werden.

Befehl	Beschreibung
CNT	Legt den Start- und Endpunkt für die kontinuierliche Bewegung fest

Anweisungsbeispiele

CNT 1	Freigeben der CNT-Einstellung
CNT 1, 100, 200	Freigeben der CNT-Einstellung Der Anfangspunkt- und Endpunkt- abstand der kontinuierlichen Bewegung beträgt 100 mm und der Endpunkt- abstand der kontinuierlichen Bewegung beträgt 200 mm.
CNT 0	Sperren der CNT-Einstellung

Programmbeispiel

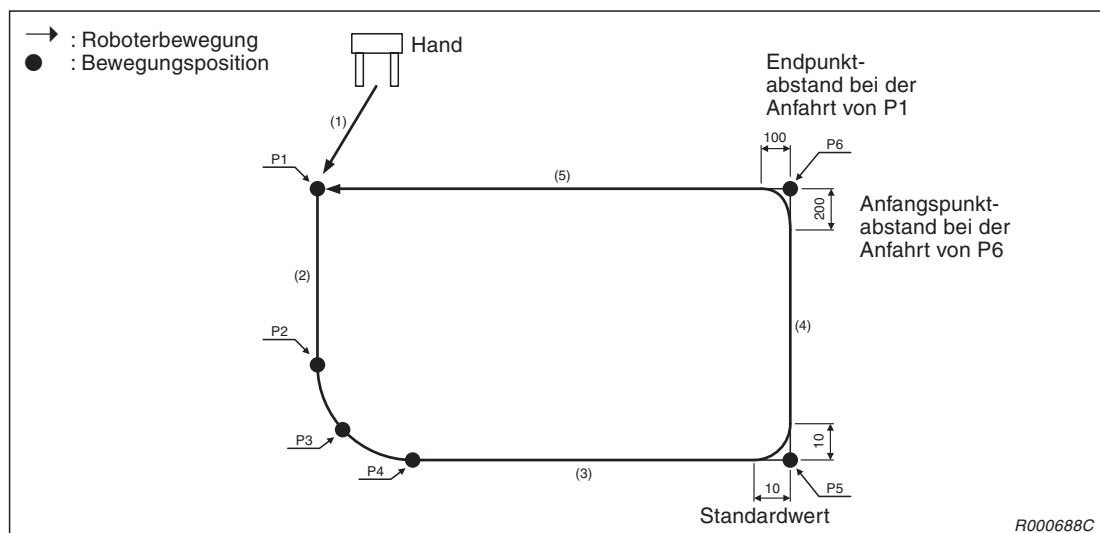


Abb. 4-6: Verlauf des Verfahrenswegs bei kontinuierlicher Bewegung

10	MOV P1	Position P1 mittels Gelenk-Interpolation anfahren
20	CNT 1	CNT-Einstellung freigeben CNT1 (10,10) ⇒ Standardwert 10, 10 (Alle folgenden Bewegungen sind kontinuierlich.)
30	MVR P2, P3, P4	Position P2 mittels Linear-Interpolation und Position P4 kontinuierlich mittels Kreis-Interpolation anfahren
40	MVS P5	Position P5 mittels Linear-Interpolation anfahren
50	CNT 1, 200, 100	Anfangspunkt- und Endpunkt- abstand der kontinuierlichen Bewegung auf 200 mm und Endpunkt- abstand der kontinuierlichen Bewegung auf 100 mm festlegen
60	MVS P6	Nach Erreichen von Position P5, Position P6 mittels Linear-Interpolation anfahren
70	MVS P1	Position P1 kontinuierlich mittels Linear-Interpolation anfahren
80	CNT 0	CNT-Einstellung sperren
90	END	Programmende

Befehl steht in Beziehung zu folgenden Funktionen:

Festlegung der Verfahrgeschwindigkeit	⇒	Abschn. 4.3.5
Festlegung der Beschleunigungs-/Bremszeit	⇒	Abschn. 4.3.5
Feinpositionierung	⇒	Abschn. 4.3.6
Gelenk-Interpolation	⇒	Abschn. 4.3.1
Linear-Interpolation	⇒	Abschn. 4.3.2
Kreis-Interpolation	⇒	Abschn. 4.3.3

4.3.5 Beschleunigungs-/Bremszeit und Geschwindigkeit

Die Beschleunigung/Abbremsung kann bezogen auf den Maximalwert eingestellt werden. Die Geschwindigkeit kann ebenfalls eingestellt werden.

Erläuterung

Befehl	Beschreibung
ACCEL	Die Beschleunigung/Abbremsung kann bezogen auf den Maximalwert (%) eingestellt werden.
OVRD	Die Geschwindigkeit für das gesamte Programm kann bezogen auf den Maximalwert (%) eingestellt werden.
JOVRD	Die Geschwindigkeit für die Gelenk-Interpolation kann bezogen auf den Maximalwert (%) eingestellt werden.
SPD	Legt die Geschwindigkeit (mm/s) für Linear- und Kreis-Interpolation fest
OADL	Freigabe der Einstellung für die optimalen Beschleunigung/Abbremsung

Anweisungsbeispiele

ACCEL	Setzt die Beschleunigung und die Abbremsung auf 100 %
ACCEL 60, 80	Setzt die Beschleunigung auf 60 % und die Abbremsung auf 80 % des Maximalwertes (Bei einer maximalen Beschleunigungs-/Bremszeit von 0,2 s ergibt sich eine Beschleunigungszeit von 0,33 s und eine Bremszeit von 0,25 s.)
OVRD 50	Legt die Geschwindigkeit für Gelenk-, Linear- und Kreis-Interpolation auf 50 % der maximalen Geschwindigkeit fest
JOVRD 70	Legt die Geschwindigkeit für Gelenk-Interpolation auf 70 % der maximalen Geschwindigkeit fest
SPD 30	Legt die Geschwindigkeit für Linear- und Kreis-Interpolation auf 30 mm/s fest
OADL ON	Gibt die optimale Beschleunigung/Abbremsung frei

Die aktuelle Arbeitsgeschwindigkeit ergibt sich:

Gelenk-Interpolation	=	Einstellung über T/B bzw. Steuergerät	×	Einstellwert des OVRD-Befehls	×	Einstellwert des JOVRD-Befehls
Linear-Interpolation	=	Einstellung über T/B bzw. Steuergerät	×	Einstellwert des OVRD-Befehls	×	Einstellwert des SPD-Befehls

Programmbeispiel

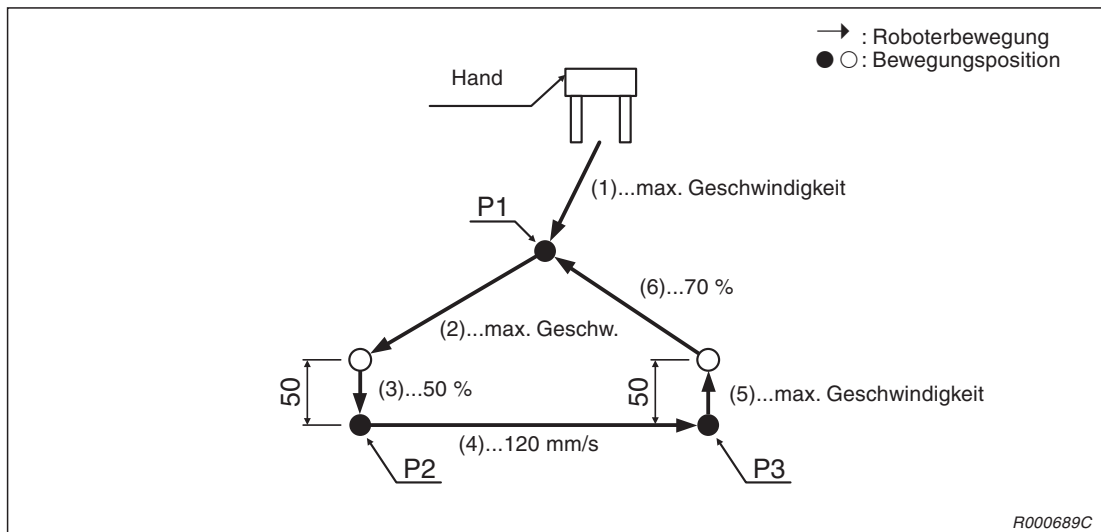


Abb. 4-7: Verfahrweg und Geschwindigkeiten

10	OVRD 100	Legt die Geschwindigkeit für das gesamte Programm auf den Maximalwert fest
20	MOV P1	(1) Position P1 mit Maximalgeschwindigkeit anfahren
30	MOV P2, -50	(2) Position mit Maximalgeschwindigkeit anfahren, die 50 mm in Werkzeuglängsrichtung von der Position P2 entfernt ist (siehe Achtungshinweis)
40	OVRD 50	Legt die Geschwindigkeit für das gesamte Programm auf den halben Maximalwert fest
50	MVS P2	(3) Position P2 mittels Linear-Interpolation und halber Maximalgeschwindigkeit anfahren
60	SPD 120	Legt die Endgeschwindigkeit auf 120 mm/s fest (Mit dem OVRD-Wert von 50 % ergibt sich eine aktuelle Geschwindigkeit von 60 mm/s.)
70	OVRD 100	Legt die Geschwindigkeit auf 100 % fest, so dass eine Endgeschwindigkeit von 120 mm/s erreicht wird
80	ACCEL 70, 70	Die Beschleunigung/Abbremsung wird auf 70 % des Maximalwerts gesetzt
90	MVS P3	(4) Position P3 mittels Linear-Interpolation und mit einer Endgeschwindigkeit von 120 mm/s anfahren
100	SPD M_NSPD	Setzt die Geschwindigkeit auf den Standardwert zurück
110	JOVRD 70	Legt die Geschwindigkeit für Gelenk-Interpolation auf 70 % fest
120	ACCEL	Legt die Beschleunigung/Abbremsung auf 100 % fest
130	MVS, -50	(5) Position mittels Linear-Interpolation und Standardgeschwindigkeit anfahren, die 50 mm in Werkzeuglängsrichtung von der aktuellen Position (P3) entfernt ist (siehe Achtungshinweis)
140	MOV P1	(6) Position P1 mit 70 % der Maximalgeschwindigkeit anfahren
140	END	Programmende

**ACHTUNG:**

Die Richtung des Verfahrwegs im Werkzeugkoordinatensystem hängt vom Werkzeugkoordinatensystem des Roboters ab. Detaillierte Informationen zum Werkzeugkoordinatensystem finden Sie im Technischen Handbuch des Roboters.

Befehl steht in Beziehung zu folgenden Funktionen:

Gelenk-Interpolation	⇒	Abschn. 4.3.1
Linear-Interpolation	⇒	Abschn. 4.3.2
Kreis-Interpolation	⇒	Abschn. 4.3.3
Kontinuierliche Bewegung	⇒	Abschn. 4.3.4

4.3.6 Feinpositionierung

Der Abschluss eines Positioniervorgangs wird durch eine Anzahl von Impulsen festgelegt. Die Einstellung ist bei Ausführung kontinuierlicher Bewegungen deaktiviert.

Erläuterung

Befehl	Beschreibung
FINE	Legt den Abschluss eines Positioniervorgangs durch eine Anzahl von Impulsen fest
MOV und DLY	Nach einem MOV-Befehl kann der Abschluss des Positioniervorgangs auch durch einen DLY-Befehl (Timer) erfolgen (sinnvoll bei Zahnriemenbetriebenen Robotern, z. B. RP-1AH, 3AH und 5AH).

Anweisungsbeispiele

- FINE 100 Legt die Anzahl der Impulse zur Feinpositionierung auf 100 fest
- MOV P1 Position P1 anfahren (Die Verfahrbewegung ist bei dem Befehlswert abgeschlossen.)
- DLY 0.1 Der Abschluss des Positioniervorgangs erfolgt über die Timer-Einstellung.

Programmbeispiel

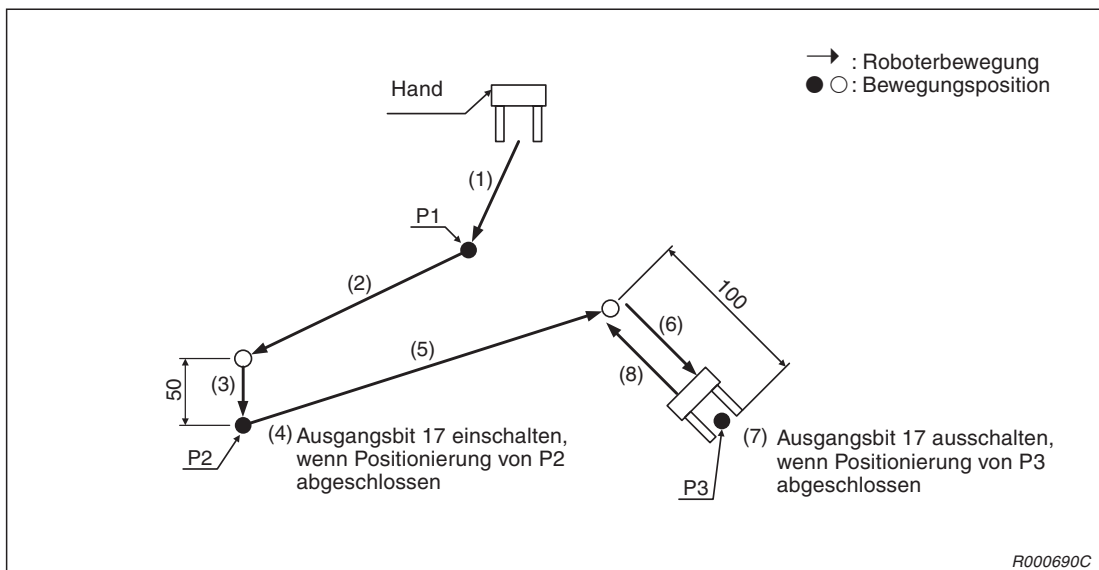


Abb. 4-8: Verfahrweg und Feinpositionierung

10	CNT 0	Die Feinpositionierung ist nur freigegeben, wenn die Einstellung für kontinuierliche Bewegungen deaktiviert ist.
20	MVS P1	(1) Position P1 mittels Linear-Interpolation anfahren
30	MVS P2, -50	(2) Position mittels Linear-Interpolation anfahren, die 50 mm in Werkzeuglängsrichtung von der Position P2 entfernt ist (siehe Achtungshinweis)
40	FINE 50	Legt die Anzahl der Impulse zur Feinpositionierung auf 50 fest
50	MVS P2	(3) Position P2 mittels Linear-Interpolation anfahren (Die Positionierung ist bei einer Impulszahl von kleiner gleich 50 abgeschlossen.)
60	M_OUT(17) = 1	(4) Ausgangsbit 17 wird auf „1“ gesetzt, wenn die Anzahl der Impulse 50 erreicht.
70	FINE 1000	Legt die Anzahl der Impulse zur Feinpositionierung auf 1000 fest
80	MVS P3, -100	(5) Position mittels Linear-Interpolation anfahren, die 100 mm in Werkzeuglängsrichtung von der Position P3 entfernt ist (siehe Achtungshinweis)
90	MVS P3	(6) Position P3 mittels Linear-Interpolation anfahren
100	DLY 0.1	Die Positionierung erfolgt über Timer.
110	M_OUT(17) = 0	(7) Ausgangsbit 17 wird auf „0“ gesetzt.
120	MVS, -100	(8) Position mittels Linear-Interpolation anfahren, die 100 mm in Werkzeuglängsrichtung von der aktuellen Position (P3) entfernt ist (siehe Achtungshinweis)
130	END	Programmende

**ACHTUNG:**

Die Richtung des Fahrwegs im Werkzeugkoordinatensystem hängt vom Werkzeugkoordinatensystem des Roboters ab. Detaillierte Informationen zum Werkzeugkoordinatensystem finden Sie im Technischen Handbuch des Roboters.

Befehl steht in Beziehung zu folgenden Funktionen:

Gelenk-Interpolation	⇒	Abschn. 4.3.1
Linear-Interpolation	⇒	Abschn. 4.3.2
Kreis-Interpolation	⇒	Abschn. 4.3.3
Kontinuierliche Bewegung	⇒	Abschn. 4.3.4

4.3.7 Verfahrweggenauigkeit

Die Verfahrwegtreue bei der Ausführung von Bewegungsbefehlen kann erhöht werden. Die Funktion ist nur für bestimmte Robotermodelle verfügbar. Zur Zeit kann der Befehl mit folgenden 5- oder 6-achsigen Vertikal-Knickarmrobotern verwendet werden: RV-1A/2AJ, RV-2A/3AJ, RV-4A/5AJ/3AL/4AJL, RV-3SB/3SJB und RV-6S/6SL/12S/12SL.

Befehl	Beschreibung
PREC	Legt die Präzision des Verfahrwegs fest

Anweisungsbeispiele

- PREC ON Hohe Verfahrweggenauigkeit aktiviert
- PREC OFF Hohe Verfahrweggenauigkeit deaktiviert

Programmbeispiel

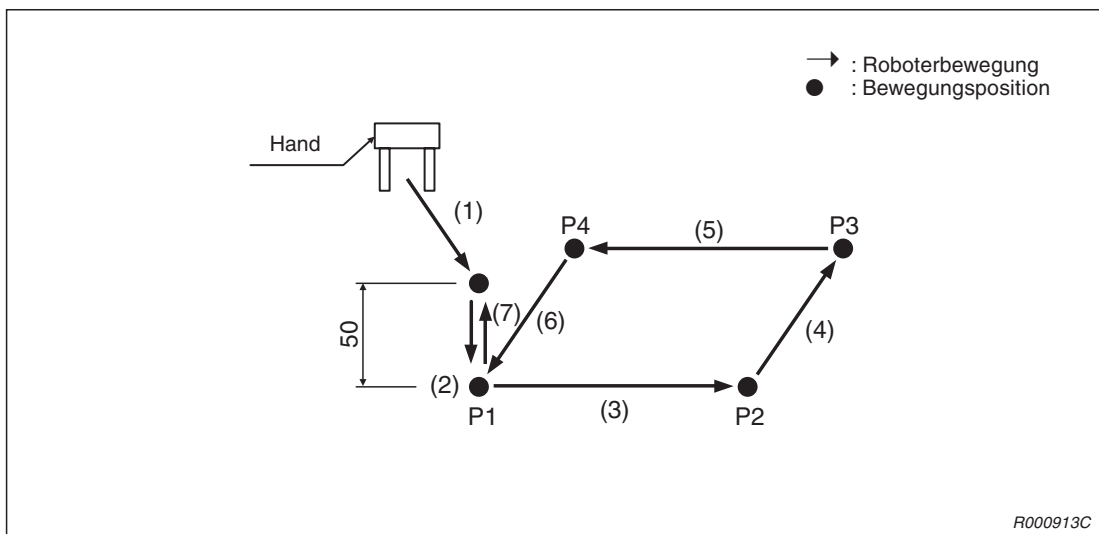


Abb. 4-9: Verfahrweggenauigkeit

HINWEIS

Die Ausführung eines Bewegungsbefehls mit hoher Verfahrweggenauigkeit (PREC ON) erhöht die Verfahrwegtreue an der Handspitze des Roboters. Dadurch nehmen jedoch die Beschleunigungs- und Bremszeiten und somit auch die Zykluszeiten zu. Wird eine weitere Erhöhung der Verfahrweggenauigkeit gewünscht, sollte auf die Verwendung des CNT-Befehls zur Ausführung kontinuierlicher Roboterbewegungen verzichtet werden.

10	MOV P1, -50	(1) Position mittels Gelenk-Interpolation anfahren, die 50 mm in Werkzeuglängsrichtung von der Position P1 entfernt ist (siehe Achtungshinweis)
20	OVRD 50	Legt die Geschwindigkeit für das gesamte Programm auf den halben Maximalwert fest
30	MVS P1	(2) Position P1 mittels Linear-Interpolation anfahren
40	PREC ON	Aktiviert die hohe Verfahrweggenauigkeit
50	MVS P2	(3) Verfahrweg zwischen Position P1 und Position P2 mit hoher Genauigkeit zurücklegen
60	MVS P3	(4) Verfahrweg zwischen Position P2 und Position P3 mit hoher Genauigkeit zurücklegen
70	MVS P4	(5) Verfahrweg zwischen Position P3 und Position P4 mit hoher Genauigkeit zurücklegen
80	MVS P1	(6) Verfahrweg zwischen Position P4 und Position P1 mit hoher Genauigkeit zurücklegen
90	PREC OFF	Deaktiviert die hohe Verfahrweggenauigkeit
100	MVS P1, -50	(7) Position mittels Linear-Interpolation anfahren, die 50 mm in Werkzeuglängsrichtung von der Position P3 entfernt ist (siehe Achtungshinweis)
110	END	Programmende

**ACHTUNG:**

Die Richtung des Verfahrwegs im Werkzeugkoordinatensystem hängt vom Werkzeugkoordinatensystem des Roboters ab. Detaillierte Informationen zum Werkzeugkoordinatensystem finden Sie im Technischen Handbuch des Roboters.

4.3.8 Hand- und Werkzeugsteuerung

Der Handgreiferzustand (offen/geschlossen) und die Werkzeugdaten können festgelegt werden.

Erläuterung

Befehl	Beschreibung
HOPEN	Die festgelegte Hand wird geöffnet.
HCLOSE	Die festgelegte Hand wird geschlossen.
TOOL	Die Werkzeugdaten und der Überwachungspunkt können eingestellt werden.

Anweisungsbeispiele

- HOPEN 1 Öffnet Hand 1
- HOPEN 2 Öffnet Hand 2
- HCLOSE 1 Schließt Hand 1
- HCLOSE 2 Schließt Hand 2
- TOOL (0, 0, 95, 0 , 0, 0) Als Überwachungspunkt wird ein Punkt definiert, der 95 mm in Werkzeuglängsrichtung vom Handflansch entfernt ist.

Programmbeispiel

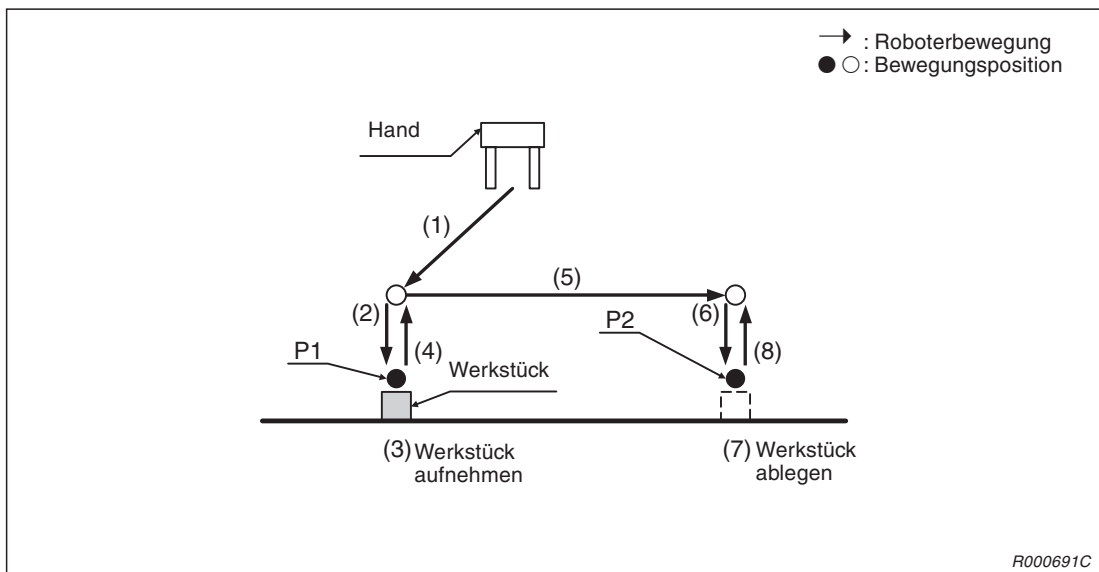


Abb. 4-10: Verfahrbewegung und Handsteuerung

10	TOOL (0, 0, 95, 0, 0, 0)(0, 0)	Legt die Werkzeuglänge auf 95 mm fest
20	MOV P1, -50	(1) Position mittels Gelenk-Interpolation anfahren, die 50 mm in Werkzeuglängsrichtung von der Position P1 entfernt ist (siehe Achtungshinweis)
30	OVRD 50	Legt die Geschwindigkeit auf den halben Maximalwert fest
40	MVS P1	(2) Position P1 mittels Linear-Interpolation anfahren (Anfahren der Position zur Werkstückaufnahme)
50	HCLOSE 1	(3) Schließt Hand 1 (Werkstück aufnehmen)
60	DLY 0.5	Wartezeit von 0,5 s
70	OVRD 100	Legt die Geschwindigkeit auf den Maximalwert fest
80	MVS, -50	(4) Position mittels Linear-Interpolation anfahren, die 50 mm in Werkzeuglängsrichtung von der aktuellen Position (P1) entfernt ist (Anheben des Werkstücks) (siehe Achtungshinweis)
90	MOV P2, -50	(5) Position mittels Gelenk-Interpolation anfahren, die 50 mm in Werkzeuglängsrichtung von der Position P2 entfernt ist (siehe Achtungshinweis)
100	OVRD 50	Legt die Geschwindigkeit auf den halben Maximalwert fest
110	MVS P2	(6) Position P2 mittels Linear-Interpolation anfahren (Anfahren der Position zur Werkstückablage)
120	HOPEN 1	Öffnet Hand 1 (Werkstück ablegen)
130	DLY 0.5	Wartezeit von 0,5 s
140	OVRD 100	(7) Legt die Geschwindigkeit auf den Maximalwert fest
150	MVS, -50	(8) Position mittels Linear-Interpolation anfahren, die 50 mm in Werkzeuglängsrichtung von der aktuellen Position (P2) entfernt ist (Entfernen vom Werkstück) (siehe Achtungshinweis)
160	END	Programmende

**ACHTUNG:**

Die Richtung des Fahrwegs im Werkzeugkoordinatensystem hängt vom Werkzeugkoordinatensystem des Roboters ab. Detaillierte Informationen zum Werkzeugkoordinatensystem finden Sie im Technischen Handbuch des Roboters.

Befehl steht in Beziehung zu folgenden Funktionen:

Angehängte Anweisung ⇒ Abschn. 6.3.77

4.4 Palettierung

Mit Hilfe der Palettierungsfunktion können Werkstücke geordnet abgelegt oder geordnete Werkstücke aufgenommen werden. Dabei reicht ein Teachen der Position des Referenz-Werkstücks aus. Alle anderen Positionen werden daraus berechnet.

Erläuterung

Befehl	Beschreibung
DEF PLT	Definiert eine Palette
PLT	Berechnet die Koordinaten eines Gitterpunktes der festgelegten Palette und weist die berechneten Koordinaten der festgelegten Position zu

Anweisungsbeispiele

DEF PLT 1, P1, P2, P3, P4, 4, 3, 1

Definiert Palette Nummer 1 mit Bezugsposition = P1, Endpunkt A = P2, Endpunkt B = P3, Paletteneckpunkt, der gegenüber der Bezugsposition liegt = P4, Anzahl der Gitterpunkte: 12 (Anzahl der Gitterpunkte zwischen Bezugsposition und Endpunkt A = 4, Anzahl der Gitterpunkte zwischen Bezugsposition und Endpunkt B = 3) und einer Bewegungsrichtung = 1

DEF PLT 2, P1, P2, P3, , 8, 5, 2

Definiert Palette Nummer 2 mit Bezugsposition = P1, Endpunkt A = P2, Endpunkt B = P3, Anzahl der Gitterpunkte: 40 (Anzahl der Gitterpunkte zwischen Bezugsposition und Endpunkt A = 5, Anzahl der Gitterpunkte zwischen Bezugsposition und Endpunkt B = 8) und einer Bewegungsrichtung = 2

DEF PLT 3, P1, P2, P3, , 8, 5, 3

Definiert kreisförmige Palette Nummer 3 mit 5 Positionen auf einem Kreisbogen über Startposition = P1, Zwischenposition = P2 und Endposition = P3 (insgesamt 3 Punkte)

(PLT 1, 5)

Berechnet die 5te Position der Palette Nummer 1

(PLT 1, M1)

Berechnet die in der numerischen Variablen M1 festgelegte Position der Palette Nummer 1

Palettendefinition und Bewegungsrichtung

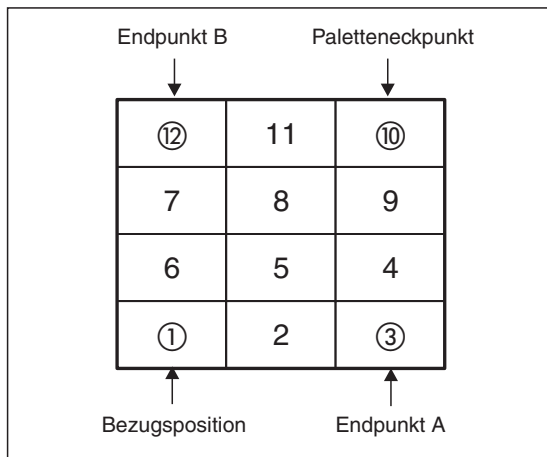


Abb. 4-11:
Palettendefinition mit
Bewegungsrichtung = 1 (zickzack)

R000693C

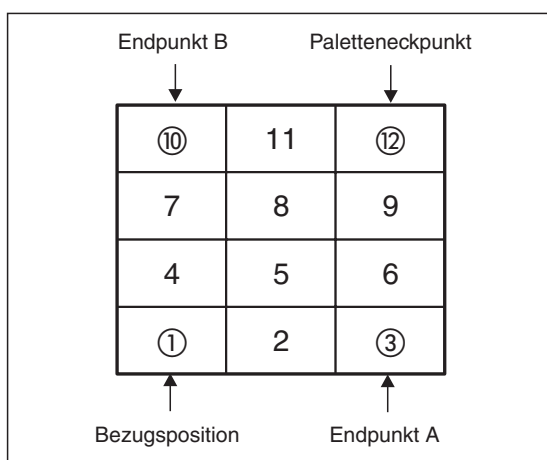


Abb. 4-12:
Palettendefinition mit
Bewegungsrichtung = 2
(Richtung beibehalten)

R000694C

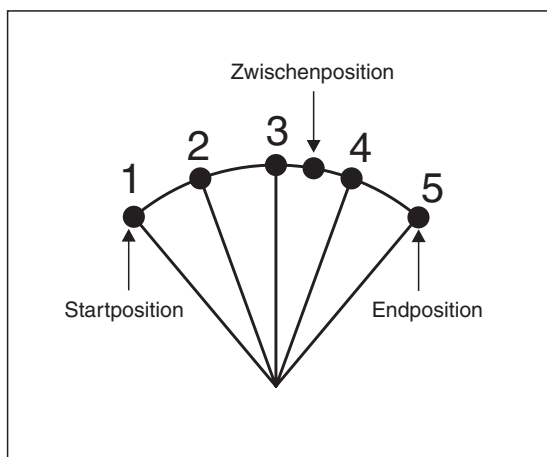


Abb. 4-13:
Palettendefinition mit
Bewegungsrichtung = 3
(kreisförmig)

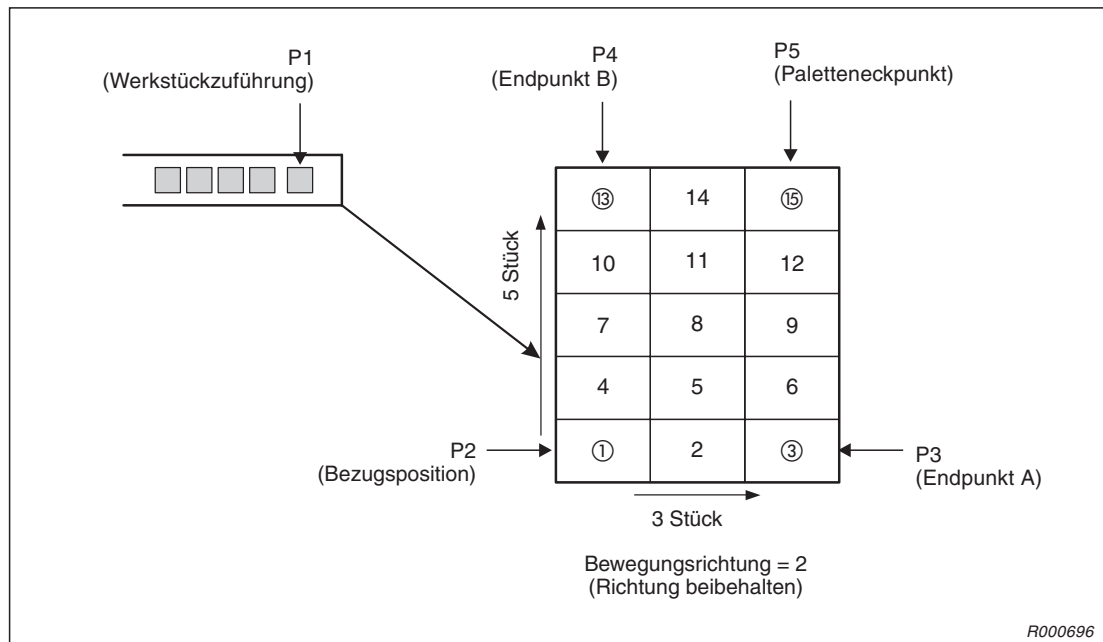
R000695C

HINWEIS

Die Vorzeichen der Stellungsdaten (A, B und C) müssen an den vier Punkten der Palette (Bezugsposition, Endpunkt A, Endpunkt B, Paletteneckpunkt) übereinstimmen. Bei einem Knickarm-Roboter mit nach unten gerichtetem Handflansch können die Achsen A, B und C (insbesondere A und C) einen Drehwinkel von 180° erreichen. In diesem Fall kann das Vorzeichen positiv oder negativ sein. Da jede Palettenposition aus den Positionsdaten der Anfangsposition und der Endpositionen berechnet wird, rotiert die Hand bei abweichenden Vorzeichen.

Ein und dieselbe Position kann über einen Drehwinkel von +180° oder -180° erreicht werden. Verwenden Sie zur Kennzeichnung unterschiedlicher Vorzeichen konsequent „+“ und „-“. Bei einem Drehwinkel von genau 180° kann das Vorzeichen entfallen.

Programmbeispiel



R000696

Abb. 4-14: Palettierung

<pre> 10 DEF PLT 1, P2, P3, P4, P5, 3, 5, 2 20 M1 = 1 30 *LOOP 40 MOV P1, -50 50 OVRD 50 60 MVS P1 70 HCLOSE 1 80 DLY 0.5 90 OVRD 100 100 MVS, -50 110 P10 = (PLT 1, M1) </pre>	<p>Definiert Palette Nummer 1 mit Bezugsposition = P2, Endpunkt A = P3, Endpunkt B = P4, Paletteneckpunkt, der gegenüber der Bezugsposition liegt = P5, Anzahl der Gitterpunkte: 15 (Anzahl der Gitterpunkte zwischen Bezugsposition und Endpunkt A = 3, Anzahl der Gitterpunkte zwischen Bezugsposition und Endpunkt B = 5) und einer Bewegungsrichtung = 2</p> <p>Setzt M1 auf „1“ (M1 dient als Zähler)</p> <p>Definiert Marke LOOP</p> <p>Position mittels Gelenk-Interpolation anfahren, die 50 mm in Werkzeuglängsrichtung von der Position 1 entfernt ist (siehe Achtungshinweis)</p> <p>Legt die Geschwindigkeit auf den halben Maximalwert fest</p> <p>Position 1 mittels Linear-Interpolation anfahren (anfahren der Position zur Werkstückaufnahme)</p> <p>Schließt Hand 1 (Werkstück aufnehmen)</p> <p>Wartezeit von 0,5 s</p> <p>Legt die Geschwindigkeit auf den Maximalwert fest</p> <p>Position mittels Linear-Interpolation anfahren, die 50 mm in Werkzeuglängsrichtung von der aktuellen Position (P1) entfernt ist (Anheben des Werkstücks) (siehe Achtungshinweis)</p> <p>Berechnet die in der numerischen Variablen M1 festgelegte Position der Palette Nummer 1 und schreibt den Wert in P10</p>
-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

120 MOV P10, -50	Position mittels Gelenk-Interpolation anfahren, die 50 mm in Werkzeuglängsrichtung von der Position P10 entfernt ist (siehe Achtungshinweis)
130 OVRD 50	Legt die Geschwindigkeit auf den halben Maximalwert fest
140 MVS P10	Position P10 mittels Linear-Interpolation anfahren (Anfahren der Position zur Werkstückablage)
150 HOPEN 1	Öffnet Hand 1 (Werkstück ablegen)
160 DLY 0.5	Wartezeit von 0,5 s
170 OVRD 100	Legt die Geschwindigkeit auf den Maximalwert fest
180 MVS, -50	Position mittels Linear-Interpolation anfahren, die 50 mm in Werkzeuglängsrichtung von der aktuellen Position (P10) entfernt ist (Entfernen vom Werkstück) (siehe Achtungshinweis)
190 M1 = M1 + 1	Numerische Variable M1 um 1 erhöhen (Palettenzähler erhöhen)
200 IF M1 <= 15 THEN *LOOP	Ist der Wert der numerischen Variablen M1 kleiner als 15, springe zur Marke LOOP, sonst gehe in die nächste Zeile.
210 END	Programmende

**ACHTUNG:**

Die Richtung des Fahrwegs im Werkzeugkoordinatensystem hängt vom Werkzeugkoordinatensystem des Roboters ab. Detaillierte Informationen zum Werkzeugkoordinatensystem finden Sie im Technischen Handbuch des Roboters.

Befehl steht in Beziehung zu folgenden Funktionen:

Ausdrücke und Operationen	⇒	Abschn. 4.8
Verzweigung und Wartezeit	⇒	Abschn. 4.5.1

4.5 Programmsteuerung

Der Programmfluss kann über Verzweigungen, Interrupts, Unterprogrammaufrufe, Stoppbefehle usw. gesteuert werden.

4.5.1 Verzweigungen und Wartezeit

Ein Sprung in eine bestimmte Programmzeile kann durch eine unbedingte oder durch eine bedingte Verzweigung erfolgen.

Erläuterung

Befehl	Beschreibung
GOTO	Bewirkt einen unbedingten Sprung in eine festgelegte Zeile
ON GOTO	Bewirkt einen Sprung in Abhängigkeit vom Wert einer Variablen Die Reihenfolge der Sprungziele entspricht der der Integer-Zahlenreihe (0, 1, 2, 3, 4 ...).
IF THEN ELSE (Anweisungen in einer Zeile)	Bewirkt einen Sprung in Abhängigkeit vom Wert einer Variablen und den festgelegten Bedingungen für diesen Wert Die Bedingungen für die Werte können frei gewählt werden. Es darf nur eine Verzweigungsart pro Anweisung verwendet werden. Ist die Bedingung erfüllt, wird die THEN-Anweisung ausgeführt. Ist die Bedingung nicht erfüllt, wird die ELSE-Anweisung ausgeführt.
IF THEN ELSE END IF (Anweisungen in mehreren Zeilen)	Bewirkt die Abarbeitung einer oder mehrerer Zeilen in Abhängigkeit vom Wert einer Variablen und den festgelegten Bedingungen für diesen Wert Die Bedingungen für die Werte können frei gewählt werden. Es darf nur eine Verzweigungsart pro Anweisung verwendet werden. Ist die Bedingung erfüllt, werden die Zeilen zwischen der THEN- und der ELSE-Anweisung ausgeführt. Ist die Bedingung nicht erfüllt, werden die Zeilen zwischen der ELSE- und der END IF-Anweisung ausgeführt. Diese Funktion ist bei Steuergeräten ab der Software-Version G1 verfügbar.
SELECT CASE END SELECT	Bewirkt einen Sprung in Abhängigkeit vom Wert einer Variablen und den festgelegten Bedingungen für diesen Wert Die Bedingungen für die Werte können frei gewählt werden. Es dürfen mehrere Verzweigungsarten pro Anweisung verwendet werden.
BREAK	Bewirkt einen Sprung in die Zeile nach der END SELECT-Anweisung
WAIT	Bewirkt eine Wartezeit, bis eine Variable den festgelegten Wert erreicht hat

Anweisungsbeispiele

GOTO 200	Unbedingter Sprung in Zeile 200
GOTO *FIN	Unbedingter Sprung zur Marke *FIN
ON M1 GOTO 100, 200, 300	Sprung in Zeile 100, falls der Wert der Variablen M1 = 1 ist, Sprung in Zeile 200, falls M1 = 2 und Sprung in Zeile 300, falls M1 = 3 Entspricht M1 keinem dieser Werte, wird der nächste Programmschritt ausgeführt.
IF M1 = 1 THEN 100	Sprung in Zeile 100, falls M1 = 1, sonst wird das Programm in der nächsten Zeile fortgesetzt
IF M1 = 1 THEN 100 ELSE 200	Sprung in Zeile 100, falls M1 = 1, sonst Zeile 200
IF M1 = 1 THEN M2 = 1 M3 = 2 ELSE M2 = -1 M3 = -2 ENDIF	Ist M1 = 1, werden die Anweisungen M2 = 1 und M3 = 2 ausgeführt. Ist M1 ≠ 1, werden die Anweisungen M2 = -1 und M3 = -2 ausgeführt.
SELECT M1 CASE 10 : BREAK CASE IS 11 : BREAK CASE IS < 5 : BREAK CASE 6 TO 9 : BREAK DEFAULT : END SELECT	Sprung zur CASE-Anweisung in Abhängigkeit von M1 Ist M1 = 10, wird das Programm nur zwischen CASE 10 und CASE IS 11 ausgeführt. Sprung in die Zeile nach der END SELECT-Anweisung Ist M1 = 11, wird das Programm nur zwischen den CASE IS 11 und CASE IS < 5 ausgeführt. Sprung in die Zeile nach der END SELECT-Anweisung Ist M1 < 5, wird das Programm nur zwischen CASE IS < 5 und CASE 6 TO 9 ausgeführt. Sprung in die Zeile nach der END SELECT-Anweisung Ist 6 < M1 < 9, wird das Programm nur zwischen CASE 6 TO 9 und DEFAULT ausgeführt. Sprung in die Zeile nach der END SELECT-Anweisung Entspricht M1 keinem der Werte, wird das Programm nur zwischen DEFAULT und END SELECT ausgeführt.
WAIT M_IN(1) = 1	Wartezeit, bis Eingangsbit 1 eingeschaltet wird

Befehl steht in Beziehung zu folgenden Funktionen:

Programmschleife	⇒	Abschn. 4.5.2
Interrupt	⇒	Abschn. 4.5.3
Unterprogramm	⇒	Abschn. 4.5.4
Eingangssignale	⇒	Abschn. 4.6.1

4.5.2 Programmschleife

Bestimmte Programmteile können in Abhängigkeit einer Bedingung wiederholt werden.

Erläuterung

Befehl	Beschreibung
FOR NEXT	Bewirkt eine Wiederholung des Programmteils zwischen der FOR- und NEXT-Anweisung, bis die Abbruchbedingung erfüllt ist
WHILE WEND	Bewirkt eine Wiederholung des Programmteils zwischen der WHILE- und WEND-Anweisung, solange die Ausführungsbedingung erfüllt ist

Anweisungsbeispiele

FOR M1 = 1 TO 10 : : NEXT	10-malige Wiederholung des Programmteils zwischen der FOR- und NEXT-Anweisung Der Startwert der Variablen M1 ist 1. Er wird bei jeder Wiederholung um 1 erhöht.
FOR M1 = 0 TO 10 STEP 2 : : NEXT	5-malige Wiederholung des Programmteils zwischen der FOR- und NEXT-Anweisung Der Startwert der Variablen M1 ist 0. Er wird bei jeder Wiederholung um 2 erhöht.
WHILE (M1 >= 1) AND (M1 <= 10) : : WEND	Wiederholung des Programmteils zwischen der WHILE- und WEND-Anweisung, solange der Wert der numerischen Variablen M1 größer als 1 und kleiner als 10 ist

Befehl steht in Beziehung zu folgenden Funktionen:

Verzweigung	⇒	Abschn. 4.5.1
Interrupt	⇒	Abschn. 4.5.3
Eingangssignale	⇒	Abschn. 4.6.1

4.5.3 Interrupt

Die Ausführung eines Programms kann mittels eines Interrupts unterbrochen und verzweigt werden.

Erläuterung

Befehl	Beschreibung
DEF ACT	Festlegung des Status und der Ausführung des Interrupts
ACT	Freigeben oder Sperren eines Interrupts
RETURN	Bewirkt den Rücksprung aus einer Interrupt-Routine in die Zeile, in der der Interrupt aufgerufen wurde

Anweisungsbeispiele

DEF ACT 1, M_IN(10) = 1 GOSUB 100	Definiert einen Unterprogrammssprung zu Zeile 100 mit der Priorität 1, falls das Eingangssignalbit 10 auf „1“ gesetzt wird und der Roboter bis zum Stillstand abgebremst ist. Die Bremszeit hängt von den über die ACCEL- und OVRD-Anweisungen festgelegten Werten ab.
DEF ACT 2, M_IN(11) = 1 GOSUB 200, L	Definiert einen Unterprogrammssprung zu Zeile 200 mit der Priorität 2, falls das Eingangssignalbit 11 auf „1“ gesetzt wird und die Ausführung der aktuellen Anweisung beendet ist
DEF ACT 3, M_IN(12) = 1 GOSUB 300, S	Definiert einen Unterprogrammssprung zu Zeile 300 mit der Priorität 3, falls das Eingangssignalbit 12 auf „1“ gesetzt wird und der Roboter in der kürzestmöglichen Zeit mit dem kürzesten Bremsweg bis zum Stillstand abgebremst ist.
ACT 1 = 1	Interrupt 1 freigeben
ACT 2 = 0	Interrupt 2 sperren
RETURN 0	Rücksprung in die Zeile, in der der Interrupt aufgerufen wurde
RETURN 1	Rücksprung in die Zeile, die der Zeile mit dem Interrupt-Aufruf folgt

Befehl steht in Beziehung zu folgenden Funktionen:

Verzweigung	⇒	Abschn. 4.5.1
Unterprogramm	⇒	Abschn. 4.5.4
Kommunikation	⇒	Abschn. 4.7

4.5.4 Unterprogramm

Mit Hilfe von Unterprogrammen und Routinen kann die Anzahl der Schritte im Hauptprogramm reduziert werden. Ein hierarchischer Aufbau und eine bessere Verständlichkeit des Programms sind somit möglich.

Erläuterung

Befehl	Beschreibung
GOSUB	Bewirkt einen Sprung zu einem Unterprogramm, das durch eine festgelegte Zeilennummer oder eine Marke definiert ist
ON GOSUB	Bewirkt einen Sprung zu einem Unterprogramm in Abhängigkeit vom Wert einer Variablen. Die Reihenfolge der Sprungziele entspricht der Integer-Zahlenreihe (0, 1, 2, 3, 4 ...).
RETURN	Bewirkt den Rücksprung aus einer Interrupt-Routine in die Zeile, die der Zeile folgt, aus der der Unterprogrammaufruf mit dem GOSUB-Befehl erfolgte
CALLP	Bewirkt den Aufruf eines Programms. Wird im aufgerufenen Programm die END-Anweisung ausgeführt, erfolgt der Rücksprung in die Zeile des aufrufenden Programms, die der Zeile folgt, aus der der Programmaufruf mit dem CALLP-Befehl erfolgte. Beim Programmaufruf können Daten übergeben werden.
FPRM	Legt die Daten fest, die beim Aufruf eines Programms mit dem CALLP-Befehl übergeben werden

Anweisungsbeispiele

GOSUB 100	Springt zum Unterprogramm in Zeile 100
GOSUB *GET	Springt zum Unterprogramm mit der Marke GET
ON M1 GOSUB 100, 200, 300	Springt zum Unterprogramm in Zeile 100, falls M1 = 1 ist, springt zum Unterprogramm in Zeile 200, falls M1 = 2 ist und springt zum Unterprogramm in Zeile 300, falls M1 = 3 ist
	Entspricht M1 keinem der Werte, wird der nächste Programmschritt ausgeführt.
RETURN	Rücksprung in die Zeile, die dem Unterprogrammaufruf mit dem GOSUB-Befehl folgt
CALLP "10"	Aufruf des Programms Nummer 10
CALLP "20", M1, P1	Aufruf des Programms Nummer 20 und Übergabe der numerischen Variablen M1 und der Positionsvariablen P1
FPRM M10, P10	Festlegung der numerischen Variablen M10 und der Positionsvariablen P10, die bei Aufruf des Unterprogramms mit CALLP übernommen werden

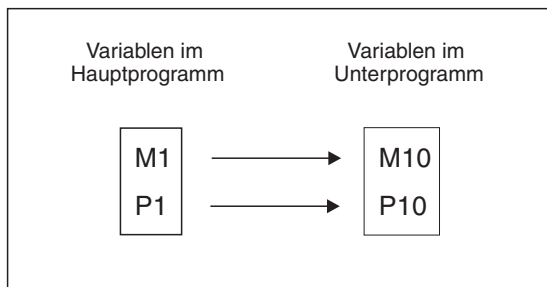


Abb. 4-15:
Übergabe der numerischen Variablen und der Positionsvariablen

Befehl steht in Beziehung zu folgenden Funktionen:

Interrupt	⇒	Abschn. 4.5.3
Kommunikation	⇒	Abschn. 4.7
Verzweigung	⇒	Abschn. 4.5.1

4.5.6 Stopp

Durch den HLT-Befehl wird der Programmablauf unterbrochen und der Roboter bis zum Stillstand abgebremst.

Erläuterung

Befehl	Beschreibung
HLT	Bewirkt eine Unterbrechung des Programmablaufs und ein Stoppen des Roboters Bei einem erneuten Programmstart wird das unterbrochene Programm von der nächsten Zeile an ausgeführt.
END	Definiert das Ende eines Programmzyklus Im kontinuierlichen Betrieb erfolgt nach Abarbeitung der END-Anweisung eine erneute Ausführung des Programms ab der Startzeile. Im zyklischen Betrieb endet das Programm bei einem Zyklusstopp nach Abarbeitung der END-Anweisung.

Anweisungsbeispiele

HLT	Programmablauf unterbrechen und Roboterbewegung stoppen
IF M_IN(20) = 1 THEN HLT	Programmablauf unterbrechen und Roboterbewegung stoppen, wenn Eingangsbit 20 gleich 1 ist
MOV P1 WTHIF M_IN(20) = 1, HLT	Programmablauf unterbrechen und Roboterbewegung stoppen, wenn bei Anfahrt der Position P1 das Eingangsbit 20 gleich 1 ist Programmablauf wird auch mitten in der Ausführung unterbrochen
END	Beendet das Programm bei Ausführung des END-Befehls bei zyklischer Ausführung oder bei Betätigung der [END]-Taste

Befehl steht in Beziehung zu folgenden Funktionen:

Angehängte Anweisung ⇒ Abschn. 6.3.77

4.6 Ein- und Ausgabe externer Signale

4.6.1 Eingangssignale

Die Steuergeräte können Signale von externen Geräten, z. B. einer SPS, empfangen und verarbeiten. Ein Einlesen der Eingangssignale erfolgt über die Roboterstatusvariablen M_IN() usw. (siehe auch Abschn. 5.1.10).

Erläuterung

Befehl	Beschreibung
WAIT	Bewirkt eine Programmunterbrechung, bis die festgelegte Eingangsbedingung erfüllt ist

Systemvariablen

M_IN, M_INB, M_INW und M_DIN

Anweisungsbeispiele

WAIT M_IN(1) = 1

M1 = M_INB(20)

M1 = M_INW(5)

Wartet, bis das Eingangsbit 1 eingeschaltet wird

Schreibt die Eingangssignalbits 20 bis 27 als 8-Bit-Wort in die numerische Variable M1

Schreibt die Eingangssignalbits 5 bis 20 als 16-Bit-Wort in die numerische Variable M1

Befehl steht in Beziehung zu folgenden Funktionen:

Ausgangssignale ⇒ Abschn. 4.6.2
 Verzweigung ⇒ Abschn. 4.5.1
 Interrupt ⇒ Abschn. 4.5.3

4.6.2 Ausgangssignale

Die Steuergeräte können Signale an externe Geräte, z. B. eine SPS, ausgeben. Eine Ausgabe der Ausgangssignale erfolgt über die Roboterstatusvariablen M_OUT() usw. (siehe auch Abschn. 5.1.10).

Erläuterung

Befehl	Beschreibung
CLR	Bewirkt ein Zurücksetzen der allgemeinen Ausgangssignale auf das über die Parameter ORST0 bis ORST224 vorgegebene Bitmuster

Systemvariablen

M_OUT, M_OUTB, M_OUTW und M_DOUT

Anweisungsbeispiele

CLR 1	Zurücksetzen der Ausgänge auf das vorgegebene Bitmuster
M_OUT(1) = 1	Ausgangsbit 1 einschalten
M_OUTB(8) = 0	8 Bits, von Ausgangsbit 8 bis 15, ausschalten
M_OUTW(20) = 0	16 Bits, von Ausgangsbit 20 bis 35, ausschalten
M_OUT(1) = 1 DLY 0.5	Ausgangsbit 1 für 0,5 s einschalten (Impulsausgang)
M_OUTB(10) = &H0F	4 Bits, von Ausgangsbit 10 bis 13 einschalten und 4 Bits von Ausgangsbit 14 bis 17 ausschalten

Befehl steht in Beziehung zu folgenden Funktionen:

Eingangssignale	⇒	Abschn. 4.6.1
Timer	⇒	Abschn. 4.5.5

4.7 Kommunikation

Die Kommunikationsfunktionen ermöglichen einen Datenaustausch zwischen dem Steuergerät und externen Geräten (z. B. Personalcomputer).

Erläuterung

Befehl	Beschreibung
OPEN	Öffnet eine Kommunikationsleitung
CLOSE	Schließt eine geöffnete Kommunikationsleitung
PRINT #	Ausgabe von Daten im ASCII-Format Nach jeder PRINT-Anweisung wird ein „Carriage Return“ ausgeführt.
INPUT #	Eingabe von Daten im ASCII-Format Nach jeder INPUT-Anweisung wird ein „Carriage Return“ ausgeführt.
ON COM GOSUB	Legt den Sprung in ein Unterprogramm fest, wenn ein Interrupt von einer Kommunikationsleitung anliegt Ein Interrupt erfolgt durch Eingabe von Daten über ein externes Gerät.
COM ON	Gibt die Interrupts von Kommunikationsleitungen frei
COM OFF	Sperrt die Interrupts von Kommunikationsleitungen Ein generierter Interrupt bleibt wirkungslos.
COM STOP	Der Interrupt-Prozess von Kommunikationsleitungen wird unterbrochen. Ein generierter Interrupt wird gespeichert und nach Freigabe der Kommunikationsleitung abgearbeitet.

Anweisungsbeispiele

OPEN "COM1:" AS#1	Öffnet die RS232C-Schnittstelle als Datei Nr. 1
CLOSE #1	Schließt die Datei Nr. 1
CLOSE	Schließt alle geöffneten Dateien
PRINT #1, "TEST"	Zeichenkette „TEST“ an Schnittstelle ausgeben
PRINT #2, "M ="; M1	Zeichenkette „M =“ an Schnittstelle und M1 an Datei Nr. 2 ausgeben
	Ausgabe: „M1 = 1“ + CR (wenn M1 = 1)
PRINT #3, P1	Positionsvariable P1 an Datei Nr. 3 ausgeben
	Ausgabe: „(123.7, 238.9, 33.1, 19.3, 0, 0)(1, 0)“ + CR (wenn X = 123.7, Y = 238.9, Z = 33.1, A = 19.3, B = 0, C = 0, FL1 = 1, FL2 = 0)
PRINT #1, M5, P5	Numerische Variable M5 und Positionsvariable P5 an Datei Nr. 1 ausgeben
	M5 und P5 werden durch ein Komma getrennt (hexadezimal, 2C).
	Ausgabe: „8, (123.7, 238.9, 33.1, 19.3, 0, 0)(1, 0)“ + CR (wenn M5 = 8, P5: X = 123.7, Y = 238.9, Z = 33.1, A = 19.3, B = 0, C = 0, FL1 = 1, FL2 = 0)
INPUT #1, M3	Wandelt die Eingangsdaten in einen Wert um und schreibt diesen in die numerische Variable M3
	Eingabe: „8“ + CR (bei Eingabe von 8)
INPUT #1, P10	Wandelt die Eingangsdaten in Werte um und schreibt diese in die Positionsvariable P10
	Eingabe: „(123.7, 238.9, 33.1, 19.3, 0, 0)(1, 0)“ + CR (wenn P5: X = 123.7, Y = 238.9, Z = 33.1, A = 19.3, B = 0, C = 0, FL1 = 1, FL2 = 0)

INPUT #1, M8, P6	Schreibt die ersten eingegeben Daten in die numerische Variable M8 Die folgenden Daten werden in die Positionsvariable P6 geschrieben. M8 und P6 werden durch ein Komma getrennt (hexadezimal, 2C). Eingabe: „7, (123.7, 238.9, 33.1, 19.3, 0, 0)(1, 0)“ + CR (wenn M8 = 7, P6: X = 123.7, Y = 238.9, Z = 33.1, A = 19.3, B = 0, C = 0, FL1 = 1, FL2 = 0)
ON COM(1) GOSUB 300	Springt zum Unterprogramm in Zeile 300, falls über COM1 eine Dateneingabe erfolgt
ON COM(2) GOSUB *RECV	Springt zur Marke *RECV, falls über COM2 eine Dateneingabe erfolgt
COM(1) ON	Freigabe des Interrupts von COM1
COM(2) OFF	Sperren des Interrupts von COM2
COM(1) STOP	Stoppt den Interrupt von COM1

Befehl steht in Beziehung zu folgenden Funktionen:

Unterprogramm	⇒	Abschn. 4.5.4
Interrupt	⇒	Abschn. 4.5.3

4.8 Ausdrücke und Operationen

4.8.1 Übersicht

Folgende Tabelle zeigt die in MELFA-BASIC IV möglichen Operationen, deren Verwendung und Anwendungsbeispiele:

Operation	Operator	Bedeutung	Beispiel	
Substitution	=	Der rechte Operand wird in den linken geschrieben.	P1 = P2 P5 = P_CURR P10.Z = 100.0 M1 = 1 STS\$ = "OK"	Schreibe P2 in P1. Schreibe die aktuelle Position in P5. Setze die Z-Koordinate von P10 auf 100.0. Schreibe den Wert 1 in die numerische Variable M1. Schreibe die Zeichenkette „OK“ in die Zeichenkettenvariable STS.
Operationen mit numerischen Daten	+	Addition	P10 = P1 + P2 MOV P8 + P9 M1 = M1 + 1 STS\$ = "ERR" + "001"	Schreibe das Ergebnis der Addition von P1 und P5 in P10. Fahre die Position an, die sich aus der Summe von P8 und P9 ergibt. Addiere 1 zu der numerischen Variablen M1. Addiere die Zeichenketten „ERR“ und „001“ und schreibe das Ergebnis in die Zeichenkettenvariable STS.
	-	Subtraktion	P10 = P1 - P2 MOV P8 - P9 M1 = M1 - 1	Schreibe das Ergebnis der Subtraktion von P1 minus P2 in P10. Fahre die Position an, die sich aus der Differenz von P8 und P9 ergibt. Subtrahiere 1 von der numerischen Variablen M1.
	*	Multiplikation	P1 = P10 * P3 M1 = M1 * 5	Schreibe das Ergebnis der relativen Konvertierung von P10 und P3 in P1. Multipliziere die numerische Variable M1 mit 5.
	/	Division	P1 = P10 / P3 M1 = M1 / 2	Schreibe das Ergebnis der umgekehrten relativen Konvertierung von P10 und P3 in P1. Dividiere die numerische Variable M1 durch 2.
	^	Exponential-Funktion	M1 = M1 ^ 2	Quadriere die numerische Variable M1.
	\	Integer-Division	M1 = M1 \ 3	Dividiere die numerische Variable durch 3 und runde das Ergebnis ab.
	MOD	Modulo-Arithmetik	M1 = M1 MOD 3	Dividiere die numerische Variable M1 durch 3 und schreibe den Rest in M1.
	-	Vorzeichenumkehr	P1 = - P1 M1 = - M1	Kehre das Vorzeichen jeder Koordinate der Positionsvariablen P1 um. Kehre das Vorzeichen der numerischen Variablen M1 um.

Tab. 4-3: Ausdrücke und Operationen (1)

Operation	Operator	Bedeutung	Beispiel	
Vergleichs- operation	=	Gleich	IF M1 = 1 THEN 200 IF STS\$ = "OK" THEN 100	Springe zu Zeile 200, falls die numerische Variable M1 gleich 1 ist. Springe zu Zeile 100, falls die Zeichenkettenvariable STS\$ „OK“ ist.
	<> oder ><	Ungleich	IF M1 <> 2 THEN 300 IF STS\$ <> "OK" THEN 900	Springe zu Zeile 300, falls die numerische Variable M1 ungleich 2 ist. Springe zu Zeile 900, falls die Zeichenkettenvariable STS\$ ungleich „OK“ ist.
	<	Kleiner als	IF M1 < 10 THEN 300 IF STS\$ < 3 THEN 100	Springe zu Zeile 300, falls die numerische Variable M1 kleiner als 10 ist. Springe zu Zeile 100, falls die Anzahl der Zeichen der Zeichenkette STS\$ kleiner als 3 ist.
	>	Größer als	IF M1 > 9 THEN 200 IF STS\$ > 2 THEN 300	Springe zu Zeile 200, falls die numerische Variable M1 größer als 9 ist. Springe zu Zeile 300, falls die Anzahl der Zeichen der Zeichenkette STS\$ größer als 2 ist.
	=< oder <=	Kleiner oder gleich	IF M1 <= 10 THEN 200 IF STS\$ <= 5 THEN 300	Springe zu Zeile 200, falls die numerische Variable M1 kleiner oder gleich 10 ist. Springe zu Zeile 300, falls die Anzahl der Zeichen der Zeichenkette STS\$ kleiner oder gleich 5 ist.
	=> oder >=	Größer oder gleich	IF M1 => 11 THEN 200 IF STS\$ >= 6 THEN 300	Springe zu Zeile 200, falls die numerische Variable M1 größer oder gleich 11 ist. Springe zu Zeile 300, falls die Anzahl der Zeichen der Zeichenkette STS\$ größer oder gleich 6 ist.
Logische Operation	AND	Logisches UND	M1 = M_INB(1) AND &H0F	Konvertiere die Eingangsbits 1 bis 4 und schreibe das Ergebnis in die numerische Variable M1. (Eingangsbits 5 bis 8 bleiben AUS.)
	OR	Logisches ODER	M_OUTB(20) = M1 OR &H80	Ausgabe der numerischen Variablen M1 an Ausgangsbits 20 bis 27 (Ausgangsbit 27 ist dabei immer EIN)
	NOT	Negation	M1 = NOT M_INW(1)	Negiere die Eingangsbits 1 bis 16 und schreibe den Wert in die numerische Variable M1.
	XOR	Exklusives ODER	M2 = M1 XOR M_INW(1)	Schreibe das Ergebnis der exklusiven ODER-Verknüpfung von M1 und den Eingangsbits 1 bis 16 in die numerische Variable M2.
	<<	Logische Linksverschiebung	M1 = M1 << 2	Verschiebe die numerische Variable M1 2 Bits nach links.
	>>	Logische Rechtsverschiebung	M1 = M1 >> 1	Verschiebe die numerische Variable M1 1 Bit nach rechts.

Tab. 4-3: Ausdrücke und Operationen (2)

4.8.2 Relative Konvertierung von Positionsdaten

Numerische Variable können mit Hilfe der 4 Grundrechenarten verarbeitet werden. Rechenoperationen mit Positionsvariablen beinhalten darüber hinaus auch eine Konvertierung der Koordinaten. Dies soll an Hand einfacher Beispiele erläutert werden.

Relative Konvertierung (Multiplikation)

Beispiel ▾

```

10 P2 = (10,5,0,0,0,0)(0,0)   Definiert Position 2
20 P100 = P1 * P2             Multiplikation von P1 und P2
30 MOV P1                     Position P1 mittels Gelenk-Interpolation anfahren
40 MVS P100                   Position P100 mittels Linear-Interpolation anfahren

```

mit P1 = (200,150,100,0,0,45)(4,0)

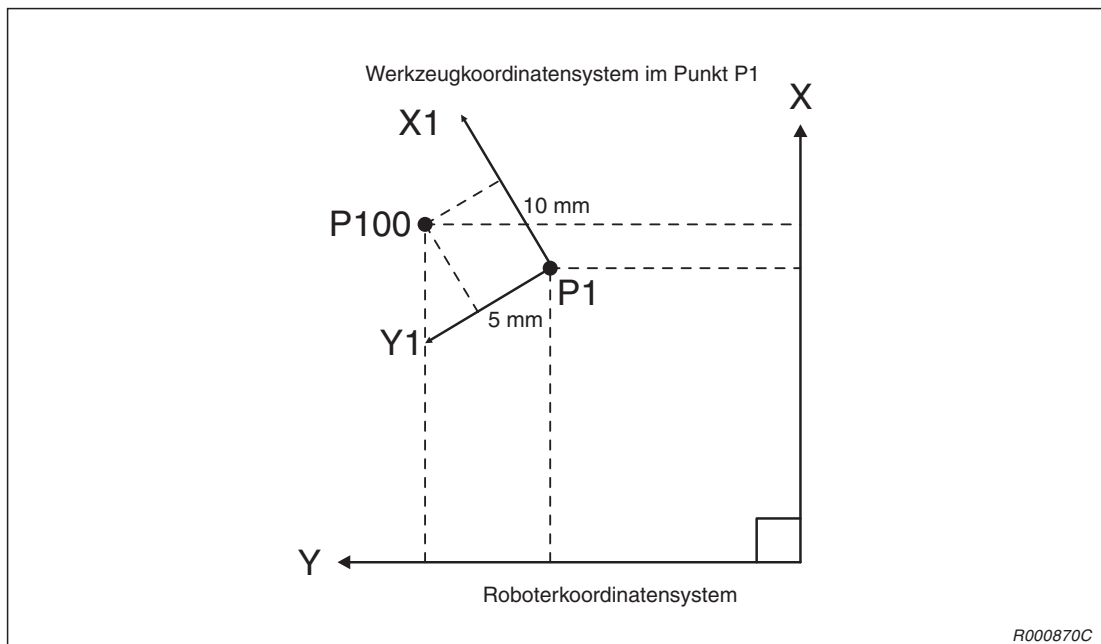


Abb. 4-16: Multiplikation von Positionsvariablen

Im oben dargestellten Beispiel erfolgt von der geteachten Position P1 aus eine relative Bewegung der Handspitze im Werkzeugkoordinatensystem. Die X- und Y-Koordinaten der Position P2 entsprechen dabei den im Werkzeugkoordinatensystem zurückgelegten Strecken. Die relative Konvertierung erfolgt durch Multiplikation der Positionsvariablen. Dabei muss beachtet werden, dass ein Vertauschen von Multiplikant und Multiplikator zu einem anderen Ergebnis führt. Die Variable, die die Größe der zurückzulegenden Strecken der relativen Bewegung festlegt (P2), muss als 2ter Faktor eingegeben werden.

Sind die Stellungsdaten der Position P2 (A, B und C) gleich 0, bleibt die Stellung der Position P1 erhalten. Sind die Daten ungleich 0, ergibt sich die neue Stellung bezogen auf die Stellung der Position P1 durch eine relative Drehung der Hand um die Z-, Y- und X-Achse (in der Reihenfolge C, B und A) .

Die Multiplikation von Positionsdaten entspricht einer Addition, die Division einer Subtraktion im Werkzeugkoordinatensystem.

△

Relative Konvertierung (Addition)**Beispiel** ▾

```

10 P2 = (5,10,0,0,0,0)(0,0)   Definiert Positon 2
20 P100 = P1 + P2             Addition von P1 und P2
30 MOV P1                     Position P1 mittels Gelenk-Interpolation anfahren
40 MVS P100                   Position P100 mittels Linear-Interpolation anfahren

```

mit P1 = (200,150,100,0,0,45)(4,0)

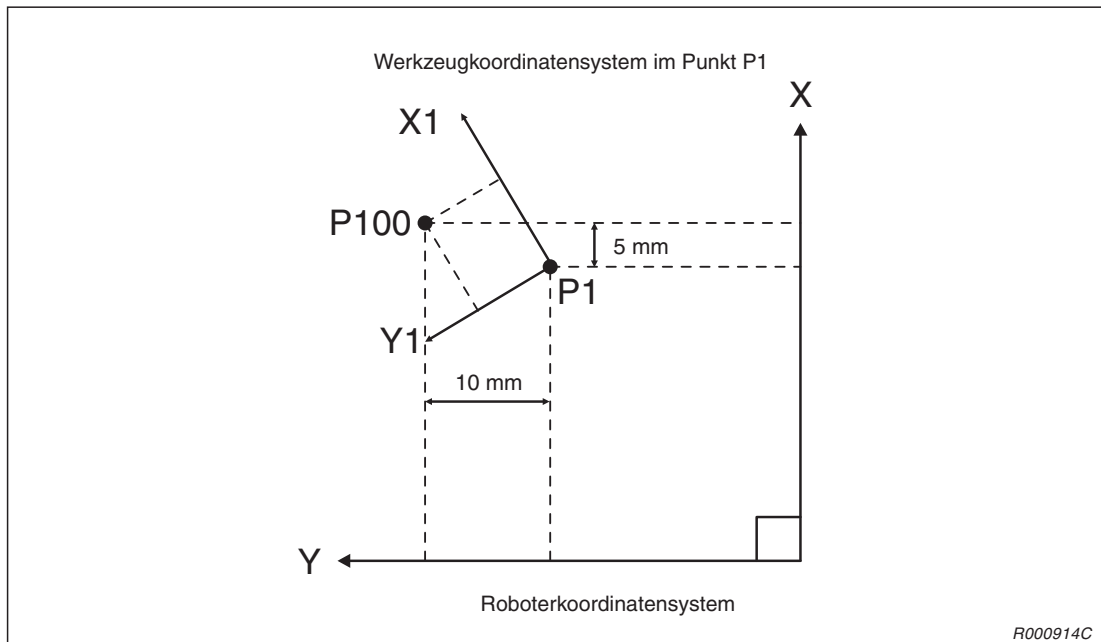


Abb. 4-17: Addition von Positionsvariablen

Im oben dargestellten Beispiel erfolgt von der geteachten Position P1 aus eine relative Bewegung der Handspitze im Roboterkoordinatensystem. Die X- und Y-Koordinaten der Position P2 entsprechen dabei den im Roboterkoordinatensystem zurückgelegten Strecken. Die relative Konvertierung erfolgt durch Addition der Positionsvariablen.

Durch einen Wert für die C-Achse der Position P2 kann der Wert der C-Achse von Position P100 geändert werden. Das Ergebnis entspricht der Summe der Werte für die C-Achsen von Position P1 und Position P2.

△

HINWEIS

In oben gezeigten Beispielen wird die relative Konvertierung aus Gründen der Anschaulichkeit im zweidimensionalen Raum erläutert. Beim Roboter findet der Vorgang im dreidimensionalen Raum statt. Zusätzlich hängt die Lage des Werkzeugkoordinatensystems dabei von der Stellung des Roboters ab.

4.9 Angehängte Anweisung

Bei Interpolationsbefehlen ist es möglich, eine Verknüpfung an die Anweisung anzuhängen. Durch Anhängen einer Verknüpfung können bestimmte Befehle parallel zum Interpolationsbefehl ausgeführt werden.

Erläuterung

Befehl	Beschreibung
WTH	Während einer Interpolationsbewegung wird eine unbedingte, zusätzliche Anweisung ausgeführt.
WTHIF	Während einer Interpolationsbewegung wird eine bedingte, zusätzliche Anweisung ausgeführt.

Anweisungsbeispiele

MOV P1 WTH M_OUT(20) = 1	Position P1 anfahren und Ausgangsbit 20 auf „1“ setzen
MOV P1 WTHIF M_IN(20) = 1, HLT	Stoppt, falls während der Anfahrt von P1 Eingangsbit 20 auf „1“ gesetzt wird
MOV P1 WTHIF M_IN(19) = 1, SKIP	Stoppt, falls während der Anfahrt von P1 Eingangsbit 19 auf „1“ gesetzt wird und springt in die nächste Zeile

Befehl steht in Beziehung zu folgenden Funktionen:

Gelenk-Interpolation	⇒	Abschn. 4.3.1
Linear-Interpolation	⇒	Abschn. 4.3.2
Kreis-Interpolation	⇒	Abschn. 4.3.3
Stopp	⇒	Abschn. 4.5.6

4.10 Multitask-Funktion

4.10.1 Beschreibung

Die Multitask-Funktion ermöglicht die parallele Ausführung mehrerer Programme zur Verkürzung der Taktzeiten. Der Roboter kann neben seiner Bewegung weitere Funktionen ausführen und mit der Peripherie kommunizieren, z. B. um Signale weiterzugeben.

Beim Multitasking wird jedem Programm ein Programmplatz (Slot/Task) zugeordnet. Die Definition der Programmplätze erfolgt über die Zuweisung von Programmnamen, Format, Startbedingungen und Priorität eines Programms in den Programmplatzparametern SLT1 bis SLT32.

Die Ausführung des Multitaskings kann über das Steuergerät, einen speziellen Eingang oder über einen auf das Multitasking bezogenen Befehl erfolgen. Insgesamt ist die parallele Ausführung von 32 Programmen möglich. (In der Werkseinstellung ist die Anzahl der Programmplätze auf 8 gesetzt.)

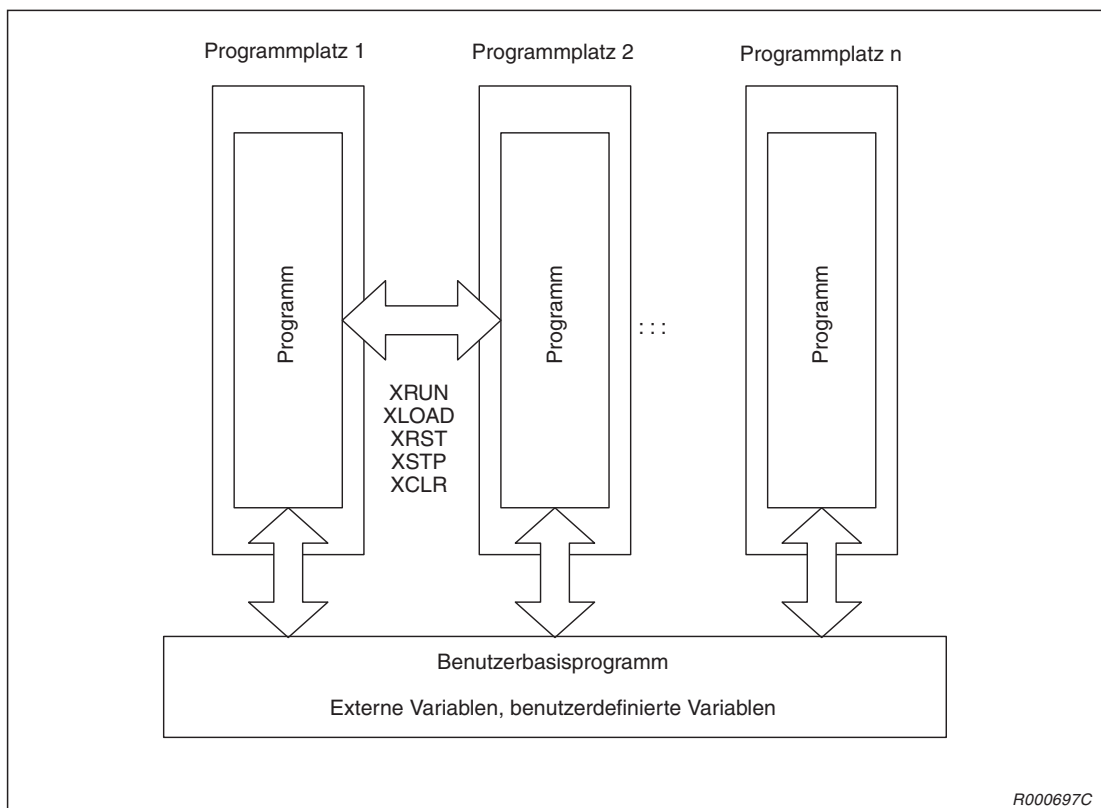


Abb. 4-18: Multitasking

HINWEIS

Bei Ausführung eines Programmes wird das Programm einem Programmplatz (Slot/Task) zugeordnet und gestartet. Bei Aufruf eines Programmes über das Bedienfeld des Steuergerätes wird das Programm vom Steuergerät automatisch dem Programmplatz 1 zugeordnet.

4.10.2 Ausführung eines Multitasks

Zur Ausführung eines Multitasks stehen drei Möglichkeiten zur Verfügung:

- **Ausführung aus einem Programm**
Bei dieser Methode wird die parallele Ausführung von Programmen aus einer nicht festgelegten Position heraus über einen MELFA-BASIC-IV-Befehl gestartet. Dabei können die Programme, die parallel ausgeführt oder die Programme, die bei einer parallelen Verarbeitung gestoppt werden sollen, gewählt werden.
Diese Methode ist sinnvoll, wenn die Auswahl der Programme aus dem Programmfluss heraus erfolgen soll. Bei dieser Methode werden die Befehle XLOAD, XRUN, XSTP und XRST verwendet. Eine detaillierte Beschreibung der Befehle finden Sie in Abschn. 6.3.
- **Ausführung über das Steuergerät oder ein externes Ein-/Ausgangssignal**
Bei dieser Methode wird die parallele Ausführung von Programmen im Startbetrieb, als durchgehender paralleler Betrieb oder als paralleler Betrieb bei Auftreten einer Fehlermeldung in Abhängigkeit der Programmplatzparameter gestartet. Die Programmplatzparameter (SLT□) müssen vor der Ausführung gesetzt werden.
Diese Methode ist vom Programmfluss unabhängig und für eine gleichzeitige Ausführung mit voreingestelltem Format oder eine sequentielle Ausführung sinnvoll.
- **Automatische Ausführung bei Einschalten der Versorgungsspannung**
Bei dieser Methode wird die Ausführung direkt nach dem Booten des Steuergerätes gestartet. Ist im Programmplatzparameter die Startbedingung „ALWAYS“ eingestellt, startet das Programm nach dem Booten automatisch im kontinuierlichen Betrieb. Dadurch werden Probleme beim Starten von Programmen zur Überwachung von Ein- und Ausgangssignalen über Programmplätze durch die SPS verhindert.
Zusätzlich kann die Ausführung eines Programms aus einem anderen Programm zur kontinuierlichen Steuerung von Roboterbewegungen heraus erfolgen. Setzen Sie zur Freigabe von X□□-Befehlen, wie XRUN oder XLOAD, des SERVO-Befehls oder des RESET-Befehls den Parameter ALWENA auf „7“.

4.10.3 Betriebszustand eines Programmplatzes

Der Betriebszustand eines Programmplatzes ist von den ausgeführten Operationen und Befehlen abhängig. Jeder Zustand kann über eine Roboterstatusvariable oder ein externes Ausgangssignal angezeigt werden.

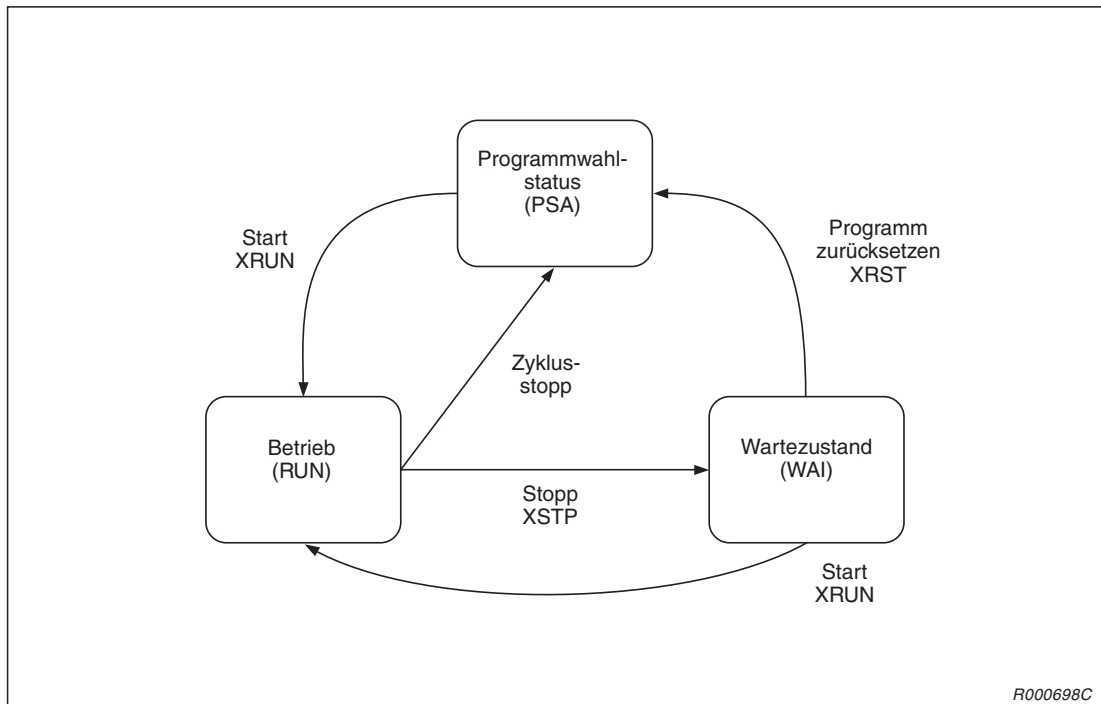


Abb. 4-19: Betriebszustand eines Programmplatzes

Programmplatzparameter

Mit Hilfe der Programmplatzparameter SLT1 bis SLT32 können der Programmname, das Format, die Startbedingung und die Priorität für 32 Programmplätze festgelegt werden. (In der Werkseinstellung ist die Anzahl der Programmplätze auf 8 gesetzt.) Eine detaillierte Beschreibung der Parameter finden Sie in Kap. 9.

Die Festlegung erfolgt in der Reihenfolge:

Parametername = 1. Programmname, 2. Ausführungsformat, 3. Startbedingung, 4. Priorität

Programmplatzparameter	Werkseinstellung	Einstellung	Funktion
Programmname	SLT1: über Steuergerät gewähltes Programm SLT2 bis 32: über Parameter festgelegtes Programm	Einstellung eines registrierten Programms	Über den Parameter kann ein registriertes Programm zur Ausführung im Multitasking festgelegt werden. Ist die Ausführung der Programme von unterschiedlichen Bedingungen abhängig, kann eine Steuerung über die Befehle XLOAD und XRUN aus einem anderen Programm heraus erfolgen. Das auf dem Steuergerät angewählte Programm ist automatisch SLT1 zugeordnet.
Ausführungsformat	REP	REP: kontinuierlicher Betrieb	Das Programm springt nach der letzten Zeile oder der END-Anweisung an den Anfang zurück und wird erneut ausgeführt.
		CYC: zyklischer Betrieb	Bei Ausführung der END-Anweisung wird das Programm beendet und die Auswahl des Programmes zurückgesetzt. Soll das Programm weiterhin angewählt bleiben, muss der Parameter SLOTON entsprechend eingestellt werden. Detaillierte Hinweise zur Einstellung des Parameters finden Sie in Kap. 9.
Startbedingung	START	START: Startanforderung Die Programmausführung wird über die START-Taste des Steuergerätes oder ein externes Ein-/Ausgangssignal gestartet.	Die Einstellung „START“ ist die übliche Einstellung zum Starten einer Programmausführung. ^①
		ALWAYS: Ständig Die Programmausführung wird automatisch nach dem Booten des Steuergerätes gestartet.	Die Einstellung „ALWAYS“ wird zur kontinuierlichen Ausführung von Programmen verwendet. Bei einem kontinuierlichen Betrieb über die Einstellung „ALWAYS“ können keine Befehle, die die Roboterbewegung steuern, z. B. der MOV-Befehl, ausgeführt werden. Ein Programm, das mit der Einstellung „ALWAYS“ ausgeführt wird, kann über den Befehl XSTP gestoppt werden. Ein Stoppen des Programms über das Bedienfeld des Steuergerätes, ein externes Eingangssignal oder über den NOT-HALT-Schalter ist nicht möglich. Die Ausführungsformate (REP/CYC) werden ignoriert.
		ERROR: Fehler Die Programmausführung wird bei Auftreten eines Fehlers gestartet.	Die Einstellung „ERROR“ wird gewählt, wenn die Programmausführung nach Auftreten eines Fehler erfolgen soll. Es können keine Befehle, die die Roboterbewegung steuern, z. B. der MOV-Befehl, ausgeführt werden. Das Ausführungsformat (REP/CYC) wird unabhängig von der Einstellung auf einen Zyklus (CYC) gesetzt.

Tab. 4-4: Programmplatzparameter (1)

Programmplatzparameter	Werks-einstellung	Einstellung	Funktion
Priorität (Anzahl der auszuführenden Zeilen)	1	1 bis 31: Die Einstellung legt die Anzahl der auszuführenden Zeilen für einen Durchgang fest.	Je größer der Einstellwert, desto mehr Zeilen des Programms werden bei einem Durchgang ausgeführt. Ist die Priorität z. B. für SLT1 auf „10“, für SLT 2 auf „5“ und für SLT 3 auf „1“ gesetzt, werden nach Abarbeitung von 10 Zeilen des Programms in SLT1 5 Zeilen des Programms in SLT2 und 1 Zeile des Programms in SLT3 ausgeführt. Nach einem Durchgang beginnt dieser Zyklus von vorne.

Tab. 4-4: Programmplatzparameter (2)

- ① Beim Start über das Steuergerät oder über das spezielle Eingangssignal START werden die Programme aller Programmplätze gestartet, für die ein Starten über Startanforderung definiert wurde.
Ein unabhängiger Programmstart erfolgt über die Startsignale S1START bis S32START. In diesem Fall ist die Zeilennummer dem Ein-/Ausgangsparameter zugewiesen. Eine detaillierte Beschreibung der speziellen Parameter für die Ein-/Ausgänge finden Sie in Abschn. 10.2.1.

Beispiel ▾

Dieses Beispiel zeigt die Parametereinstellungen zur Festlegung der folgenden Bedingungen für Programmplatz 2:

Programmname: 5
 Ausführungsformat: kontinuierlicher Betrieb
 Startbedingung : ständig
 Priorität: 10

Einstellung: SLT 2 = 5, REP, ALWAYS, 10



4.10.4 Erstellung eines Multitask-Programms

Anzahl der Programmplätze und Verarbeitungszeit

Bei der Ausführung eines Multitasks scheinen alle gestarteteten Programme gleichzeitig ausgeführt zu werden. In Wirklichkeit erfolgt jedoch nur jeweils die Verarbeitung einer einzelnen Zeile und die Programmsteuerung springt von einem Programm in das nächste. Die Anzahl der in einem Programm nacheinander abzuarbeitenden Zeilen kann über die Programmplatzparameter SLT festgelegt werden (siehe auch Kap. 9). Je mehr Programme also ausgeführt werden, desto größer wird die Verarbeitungszeit. Die Programmzahl beim Multitasking ist daher aus Gründen der Effizienz auf ein Minimum zu begrenzen. Programme für andere Anwendungen, z. B. zur Ausführung von Roboterbewegungsbefehlen (MOV oder MVS) können jedoch zu jeder Zeit ausgeführt werden.

Anzahl der Programme, die parallel ausgeführt werden sollen

Die Einstellung der Anzahl der Programme, die parallel ausgeführt werden sollen, erfolgt über den Parameter TASKMAX (Werkseinstellung: 8). Sollen mehr als 8 Programme parallel ausgeführt werden, muss der Parameterwert geändert werden.

Datenaustausch zwischen Programmen über externe Variablen

Über externen Variablen wie M_00 oder P_00 und benutzerdefinierte Variablen kann ein Datenaustausch zwischen den im Multitasking betriebenen Programmen stattfinden. Eine detaillierte Beschreibung der externen Variablen finden Sie in Abschn. 5.1.10.

Beispiel ▾

Dieses Beispiel zeigt eine Steuerung des EIN/AUS-Zustandes des Eingangsbits 8 über Programmplatz 2. Der EIN-Zustand wird über die externe Variable M_00 an Programmplatz 1 übertragen.

Programmplatz 1 (Slot 1)

10	M_00 = 0	Setzt die Variable M_00 auf „0“
20	IF M_00 = 0 THEN 20	Wartestatus, bis M_00 ungleich „0“
30	M_00 = 0	Setzt die Variable M_00 auf „0“
40	MOV P1	Fortsetzung des normalen Betriebs
50	MOV P2	
	:	
100	GOTO 20	Wiederhole ab Zeile 20

Programmplatz 2 (Slot 2)

10	IF M_IN(8) <> 1 THEN 30	Sprung in Zeile 30, falls Eingangsbit 8 nicht EIN ist
30	M_00 = 1	Setzt die Variable M_00 auf „1“
40	MOV P1	Fortsetzung des normalen Betriebs
	:	

△

Überwachung des Programmstatus über Roboterstatusvariablen

Über die Roboterstatusvariablen (M_RUN, M_WAI und M_ERR) kann der Programmstatus einer im Multitasking betriebenen Anwendung von jedem Programmplatz aus überwacht werden.

Beispiel ▾

M1 = M_RUN(2) Schreibt Programmstatus des Programmplatzes 2 in M1

Eine detaillierte Beschreibung der Roboterstatusvariablen finden Sie im Abschn. 5.1.10.



Externe Signalein- und -ausgänge

Die externen Signalein- und -ausgänge können von jedem Programmplatz verwendet werden.

Programmplatz 1

Das Hauptsteuerprogramm, in dem die Bewegungsbefehle des Roboters (MOV-Befehle usw.) festgelegt sind, wird in der Regel Programmplatz 1 zugeordnet. Eine andere Zuordnung muss über die Befehle GETM und RELM erfolgen. Eine detaillierte Beschreibung der Befehle finden Sie in Abschn. 6.3.

Initialisierung kontinuierlich ausgeführter Programme

Programme, deren Startbedingung über die Programmplatzparameter auf „ALWAYS“ gesetzt sind, werden kontinuierlich ausgeführt, auch wenn das Ausführungsformat auf „CYC“ eingestellt ist. Die Programme sollten so aufgebaut sein, dass die Initialisierung einmal unter Verwendung einer Sprunganweisung (z. B. GOTO) im Programm erfolgt.

HINWEIS

In der Grundeinstellung wird der Mechanismus 1 (Roboter bei Standardsystem) dem Programmplatz 1 zugeordnet. Bewegungsbefehle können somit ohne Zuordnung über den GETM-Befehl im Programmplatz 1 definiert werden. Sollen Bewegungsbefehle in einer Anwendung ausgeführt werden, so ist zuerst die Zuordnung von Programmplatz und Mechanismus über den Befehl RELM aufzuheben und anschließend die neu gewünschte Zuordnung über den Befehl GETM zu definieren.

4.10.5 Anwendung des Multitaskings

Multitasking starten

Bei Ausführung des Startvorgangs über das Bedienfeld der Steuereinheit oder den speziellen Eingang START starten alle Anwendungen gleichzeitig, für die in den Programmplatzparametern die Startbedingung „Startanforderung“ festgelegt wurde. Ein separates Starten der Programme ist über die Starteingänge S1START bis S32START möglich. In diesem Fall ist die Zeilennummer dem Ein-/Ausgangsparameter zugewiesen. Eine detaillierte Beschreibung der speziellen Parameter für die Ein-/Ausgänge finden Sie in Abschn. 10.2.1.

Anzeige des Betriebszustandes

Die LEDs der Taster START und STOP auf dem Bedienfeld der Steuereinheit und die speziellen Ein-/Ausgangssignale START und STOP zeigen den Betriebszustand aller Programme an, deren Startbedingung in den Programmplatzparametern SLT□ auf „START“ gesetzt ist. Bei Ausführung eines Programmes leuchtet die LED des START-Tasters und das Ausgangssignal START wird eingeschaltet. Bei einem Programmstopp leuchtet die LED des STOP-Tasters und das Ausgangssignal STOP wird eingeschaltet.

Die speziellen Ausgangssignale S1START bis S32START und S1STOP bis S32STOP zeigen den Betriebszustand jedes einzelnen Programmplatzes an. Zur unabhängigen Anzeige der Betriebszustände muss die entsprechende Zeilennummer dem Ein-/Ausgangsparameter zugewiesen werden. Eine detaillierte Beschreibung der speziellen Parameter für die Ein-/Ausgänge finden Sie in Abschn. 10.2.1.

Der Betriebszustand eines Programms mit der Startbedingung auf „ALWAYS“ oder „ERROR“ wird nicht über die LEDs der Taster START und STOP angezeigt. Der Betriebszustand kontinuierlich ausgeführter Programme kann über die optionale Programmier-Software des Roboters angezeigt werden.

4.10.6 Beispiel zur Anwendung der Multitask-Funktion

Detaillierte Beschreibung des Arbeitsablaufs

Der Arbeitsablauf wird in zwei Programme aufgeteilt:

- Programm mit Bewegungsbefehlen
Dem Programm, das die Bewegungsbefehle enthält, ist der Programmplatz 1 zugewiesen.
- Programm zum Einlesen von Positionsdaten
Dem Programm zum Einlesen der Positionsdaten ist der Programmplatz 2 zugewiesen. Wird während der Roboterbewegung ein Startsignal an einen Sensor ausgegeben, erfolgt über das Programm zum Einlesen von Positionsdaten eine Datenabfrage des Personalcomputers. Der Personalcomputer überträgt die Positionsdaten über Programmplatz 2 zum Roboter.

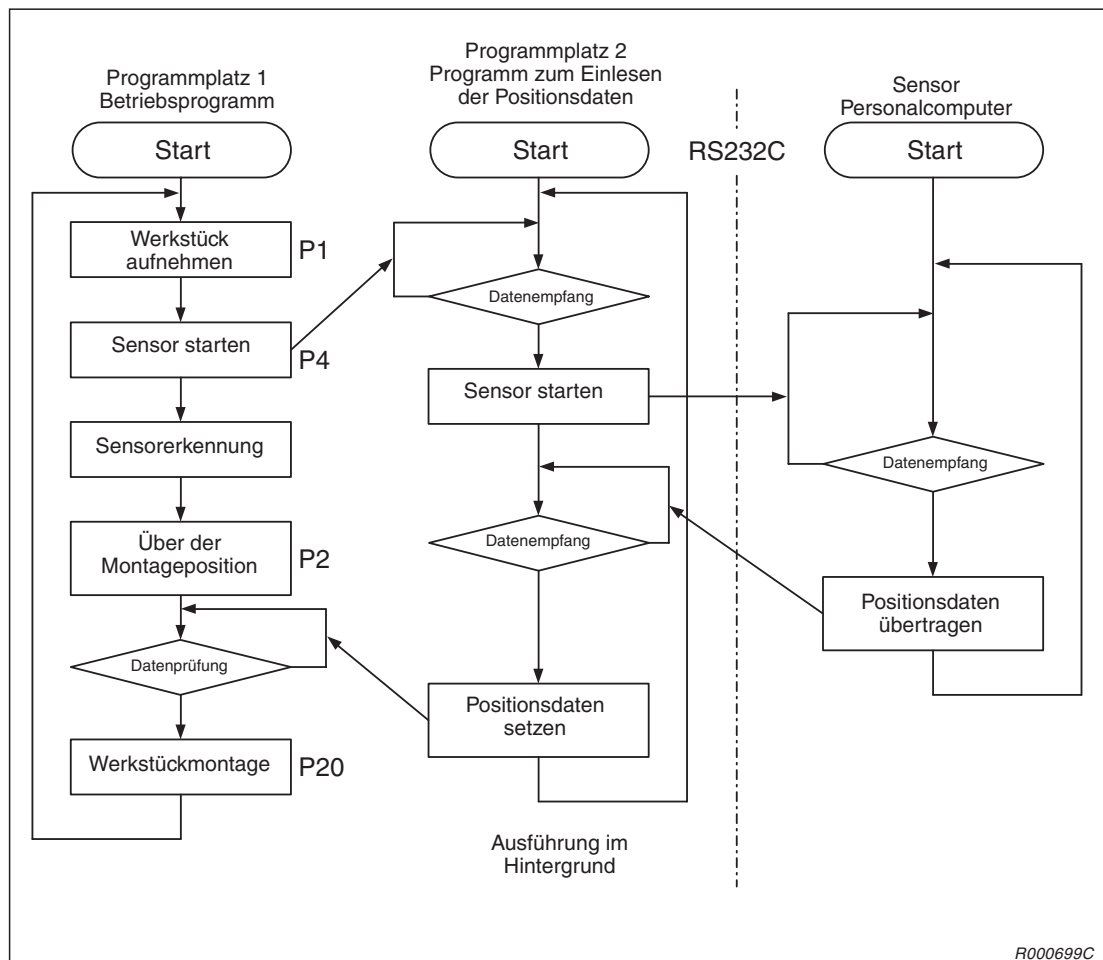


Abb. 4-20: Flussdiagramm

Positionen

P1: Aufnahme des Werkstücks (Wartezeit Vakuumgreifer DLY 0.05)

P2: Ablage des Werkstücks (Wartezeit DLY 0.05)

P3: Position vor der Überwachung (Position ohne Stopp durchlaufen CNT)

P4: Position nach der Überwachung (Position ohne Stopp durchlaufen CNT)

P_01: Kompensationsdaten der Überwachung

P20: Relative Konvertierung der Kompensationsdaten der Überwachung P_01 und P2

Bewegungsablauf

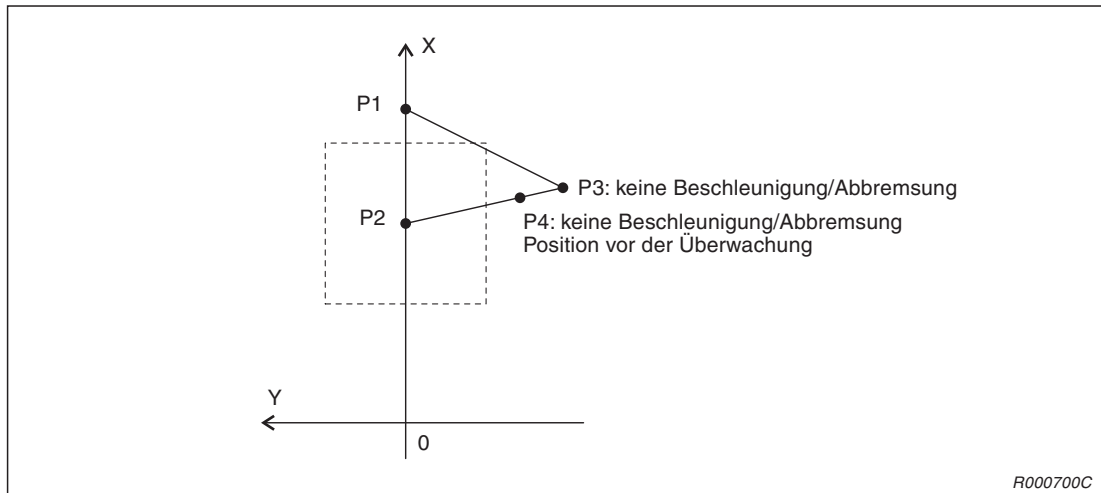


Abb. 4-21: Verfahrbewegung und Positionen

Programm mit Bewegungsbefehlen (Programmplatz 1)

100 CNT 1	'Überschleiffunktion freigeben
110 MOV P2, 10	'Position 10 mm über P2 anfahren
120 MOV P1, 10	'Position 10 mm über P1 anfahren
130 MOV P1	'Aufnahmeposition P1 anfahren
135 M_OUT(10) = 0	'Werkstück aufnehmen
140 DLY 0.05	'Wartezeit 0,05 s
150 MOV P1, 10	'Position 10 mm über P1 anfahren
160 MOV P3	'Position P3 anfahren
165 SPD 500	'Geschwindigkeit auf 500 mm/s setzen
170 MVS P4	'Mit Überfahren von P4 Überwachung starten
180 M_02# = 0	'Hintergrundprozess zum Einlesen der Daten mit Sperrvariablen (M_01 = 1/M_02 = 0) starten
190 M_01# = 1	'Einlesen der Daten im Hintergrund starten
200 MVS P2, 10	'Position 10 mm über P2 anfahren
210 IF M_02# = 0 THEN GOTO 210	'Warten, bis Sperrvariable M_02 gleich 1
220 P20 = P2 * P_01	'Multipliziere P_01 mit P2
230 MOV P20, 10	'Position 10 mm über P20 anfahren
240 MOV P20	'Ablageposition P20 anfahren
245 M_OUT(10) = 1	'Werkstück ablegen
250 DLY 0.05	'Wartezeit 0,05 s
260 MOV P20, 10	'Position 10 mm über P20 anfahren
270 CNT 0	'Überschleiffunktion sperren
280 END	'Zyklusende

Programm zum Einlesen von Positionsdaten (Programmplatz 2)

100 IF M_01# = 0 THEN GOTO 100	'Warten, bis Sperrvariable M_01 gleich 1
105 OPEN "COM1:" AS #1	'Öffnet die RS232C-Kommunikationsschnittstelle
110 DLY M_03#	'Hypothetische Wartezeit (0,05 s)
115 PRINT #1, "SENS"	'Zeichenkette „SENS“ über RS232C-Schnittstelle ausgeben (Anzeige)
117 INPUT #1, M1, M2, M3	'Wartet auf Einlesen der Kompensationsdaten (relative Daten)
120 P_01.X = M1	'Überschreiben der ΔX -Koordinate
130 P_01.Y = M2	'Überschreiben der ΔY -Koordinate
140 P_01.Z = 0.0	'
150 P_01.A = 0.0	'
160 P_01.B = 0.0	'
170 P_01.C = RAD(M3)	'Überschreiben der ΔC -Koordinate
175 CLOSE	'Kommunikationsschnittstelle schließen
180 M_01# = 0	'Setzen der Sperrvariablen M_01 = 0
190 M_02# = 1	'Setzen der Sperrvariablen M_02 = 1
200 END	'Programm beenden

Einstellung der Programmplatzparameter

SLT 1 = 1, REP, START, 1
 SLT 2 = 2, REP, START, 2

Zur Aktivierung der Programmplatzparameter muss die Versorgungsspannung aus und wieder eingeschaltet werden.

Start

Der Start von Programm 1 und 2 erfolgt über das Bedienfeld des Steuergerätes.

5 MELFA-BASIC IV

Die nachfolgenden Abschnitte enthalten eine Auflistung aller in MELFA-BASIC IV verwendeten Datentypen und deren Anwendungsmöglichkeiten.

5.1 Begriffserklärung

5.1.1 Anweisung

Eine Anweisung ist die kleinste Einheit eines Programms. Sie besteht aus einem Befehl und einem Befehlsparameter.

Beispiel ▾

MOV P1 = Anweisung

MOV = Befehl

P1 = Befehlsparameter



5.1.2 Angehängte Anweisung

Bei Interpolationsbefehlen ist es möglich, eine Verknüpfung an die Anweisung anzuhängen. Durch Anhängen einer Verknüpfung können bestimmte Befehle parallel zum Interpolationsbefehl ausgeführt werden. Es darf pro Zeile nur eine Verknüpfung angehängt werden.

Beispiel ▾

Folgender Befehl bewirkt, dass die Position P1 mittels Gelenk-Interpolation angefahren und gleichzeitig das Ausgangsbit 17 auf „1“ gesetzt wird:

MOV P1 WTH M_OUT(17) = 1



Mit Hilfe der Befehle WTH und WTHIF können Verknüpfungen an eine Anweisung angehängt werden.

5.1.3 Zeilen

Eine Zeile besteht aus einer Zeilennummer und einer Anweisung oder zwei Anweisungen, wenn zusätzlich eine Konjunktion angehängt ist.

Die Zeilenlänge darf maximal 127 Zeichen betragen. Das Zeilenendzeichen wird dabei nicht mitgezählt.

HINWEIS

In der MELFA-BASIC-Programmiersprache ist es nicht erlaubt, mehrere durch Semikolons getrennte Anweisungen in eine Zeile zu setzen, wie es bei vielen BASIC-Dialekten möglich ist.

5.1.4 Zeilennummern und Marken

Zeilennummern

Für die einwandfreie Funktion eines Programmes müssen die Zeilennummern in aufsteigender Reihenfolge angeordnet sein. Beim Abspeichern wird das Programm in dieser Reihenfolge im Speicher abgelegt. Der Wertebereich für die Zeilennummern beträgt 1 bis 32 767.

Eine Ausnahme bildet die direkte Befehlsausführung:

HINWEIS

Bei fehlender Zeilennummer wird eine Anweisung direkt nach Betätigung der Eingabetaste ausgeführt. Die Anweisung wird dabei nicht gespeichert.

Marken

Eine Marke ist ein benutzerdefiniertes Wort, das ein Sprungziel festlegt. Erzeugt wird eine Marke durch Eingabe des Asterisk-Zeichens (*) hinter der Zeilennummer und einer alphanumerischen Zeichenkette. Hierbei wird auch zwischen Groß- und Kleinbuchstaben unterschieden. Beginnt die Zeichenkette mit dem Zeichen „L“, kann als nächstes Zeichen der Unterstrich (_) verwendet werden. Das erste Zeichen muss ein Buchstabe sein. Es werden nur die ersten 8 Zeichen ausgewertet.

Beispiel ▾

```
120 *ABLAGE;  
170 *LAGE_1
```

**HINWEIS**

Für Markennamen dürfen keine reservierten Wörter (z. B. DLY, HOPEN usw.), keine Namen, die mit einem Symbol oder einer Zahl beginnen und keine Namen, die schon für eine Variable oder eine Funktion vergeben wurden, benutzt werden.

5.1.5 Zeichentypen

Die in MELFA-BASIC IV verwendbaren Zeichentypen sind in Tab. 5-1 aufgeführt. Detaillierte Beschreibungen zu Programmnamen finden Sie in Abschn. 4.2.1, zu Variablennamen in Abschn. 5.1.9 und zu Markennamen in Abschn. 5.1.4.

Kategorie	Verwendbare Zeichen in MELFA-BASIC IV	Programmnamen	Variablennamen	Markennamen
Buchstaben	A B C D E F G H I J K L M N O P Q R S T U V W X Y Z	✓	✓	✓
	a b c d e f g h i j k l m n o p q r s t u v w x y z	—	①	①
Zahlen	0 1 2 3 4 5 6 7 8 9	✓	②	✓
Symbole	” ’ # & () * + - . , / : ; = < > ? @ ` [\] ^ { } ~	—	—	—
	! # \$ %	—	—	—
	_ (Unterstrich)	—	③	④
Leerstellen	Leerzeichen	—	—	—

Tab. 5-1: In MELFA-BASIC IV verwendbare Zeichen

✓ : verwendbar

— : nicht verwendbar

- ① Kleinbuchstaben in Variablen- und Markennamen werden zu Großbuchstaben konvertiert.
- ② Ein Variablenname muss mit einem Buchstaben beginnen. Alle weiteren Zeichen können auch Zahlen sein.
- ③ Der Unterstrich kann in Variablennamen als zweites und für die folgenden Zeichen verwendet werden. Alle Variablen mit einem Unterstrich als zweitem Zeichen sind externe Variablen.
- ④ Soll der Unterstrich in einem Markennamen verwendet werden, muss der Namen mit einem Asterisk (*) gefolgt von dem Buchstaben „L“ beginnen.

HINWEIS

Kleinbuchstaben, die in Kommentaren oder in Zeichenketten verwendet werden, werden auch als Kleinbuchstaben abgespeichert. In allen anderen Fällen werden sie zu Großbuchstaben konvertiert, sobald das Programm gelesen wird.

5.1.6 Zeichen mit besonderer Bedeutung

Unterstrich (_)

Der Unterstrich wird bei Variablennamen als zweites Zeichen verwendet, wenn diese als programmexterne Variablen benutzt werden.

Beispiele ▾

P_CURR
M_01
M_ABC



Apostroph (')

Der Apostroph wird vor einen Kommentar gesetzt. Es hat die gleiche Funktion wie der REM-Befehl (Kennzeichnung eines Kommentars).

Beispiele ▾

100 MOV P1 'GET	GET wird als Kommentar deklariert
150 'GET PARTS	Entspricht der Zeile: 150 REM GET PARTS



Asterisk (*)

Das Asterisk-Zeichen wird vor alle Sprungmarken gesetzt. Die Sprungmarke darf maximal aus 8 Zeichen bestehen.

Beispiel ▾

200 *LADEN



Komma (,)

Das Komma dient bei Angabe mehrerer Parameter oder Suffixe zur Trennung.

Beispiel ▾

P1 = (100, 150, ...)



Punkt (.)

Der Punkt dient als Dezimalpunkt und zur Unterteilung der einzelnen Komponenten bei mehrteiligen Daten wie Positions- und Gelenkvariablen.

Beispiel ▾

M1 = P2.X Schreibt die X-Koordinate der Positionsvariablen P2 in die Variable M1



Leerzeichen

In Zeichenketten und Kommentaren wird das Leerzeichen wie jedes andere Zeichen interpretiert. Zwischen einzelnen Daten, nach Zeilennummern und Anweisungen dient es zur Trennung.

Im Eingabeformat bei der detaillierten Befehlsbeschreibung (siehe Abschn. 6.3) wird ein notwendiges Leerzeichen durch das Zeichen „□“ dargestellt.

5.1.7 Datentypen

Datentypen umfassen Werte, Positionsdaten, Gelenkdaten und Zeichen. Bei Zahlen unterscheidet man zwischen reellen und ganzen Zahlen.

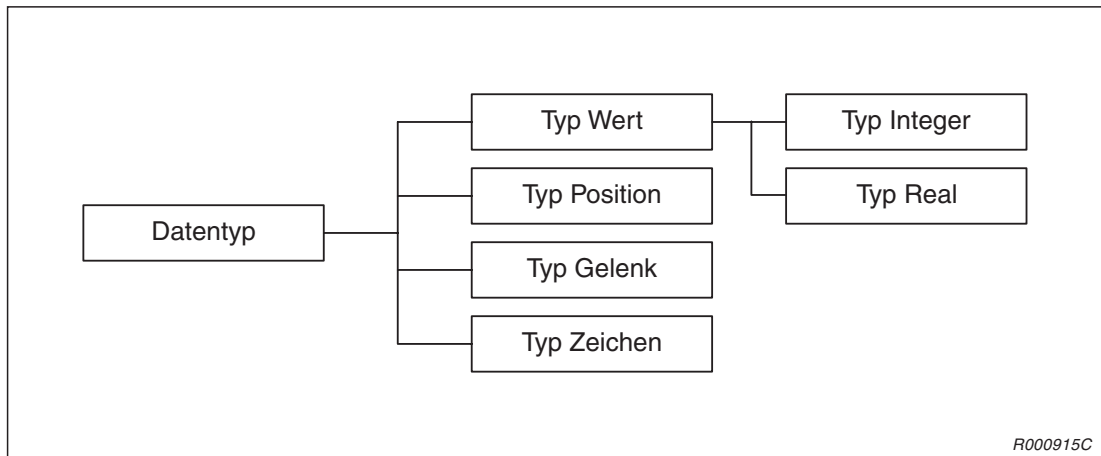


Abb. 5-1: Datentypen

5.1.8 Konstanten

Man unterscheidet fünf Arten von Konstanten:

- Numerische Konstanten
- Alphanumerische Konstanten
- Positionskonstanten
- Gelenkkonstanten
- Winkelkonstanten

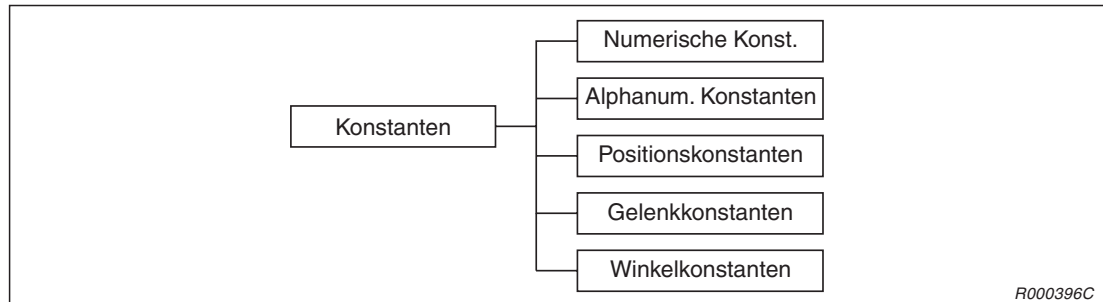


Abb. 5-2: Konstanten

Numerische Konstanten

Numerische Konstanten sind wie folgt aufgebaut:

- Dezimalzahlen

Beispiele ▾ 1, 1.7, -10,5, +1.2E+5 (Exponentialdarstellung)

Der Wertebereich für Zahlen ist $-1.7976931348623157E+308$ bis $1.7976931348623157E+308$.

△

- Hexadezimale Zahlen

Beispiele ▾ &H132; &HC011; &H1AC4

Der Wertebereich hexadezimaler Zahlen ist &H0000 bis &HFFFF.

△

- Binäre Zahlen

Beispiele ▾ &B010011; &B1101

Der Wertebereich binärer Zahlen ist &B0000000000000000 bis &B1111111111111111.

△

Der Typ einer Konstanten wird durch ein Symbol am Ende der Konstanten definiert:

Beispiele ▾ 10% (Typ Integer)

1.0005! (Typ mit einfacher Genauigkeit)

10.000000003# (reelle Zahl mit doppelter Genauigkeit)

△

Zeichenkettenkonstanten

Zeichenketten werden in Anführungszeichen dargestellt ("").

Beispiele ▾

"ABCDEFGHJKLMN"; "123"



HINWEIS

Eine Zeichenkette kann aus bis zu 127 Zeichen bestehen. Dies umfasst auch die Zeilennummer und die Anführungszeichen. Soll die Zeichenkette selbst ein Anführungszeichen enthalten, muss das Zeichen zweimal hintereinander eingegeben werden: Für die Zeichenkette AB"CD, muss "AB""CD" eingegeben werden.

Positionskonstanten

Folgende Abbildung zeigt die Syntax der Positionskonstanten:

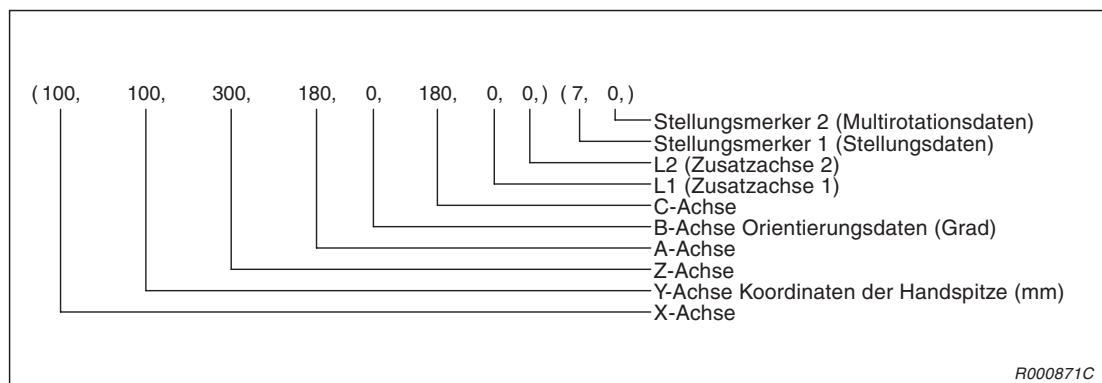


Abb. 5-3: Positionskonstanten

Beispiele ▾

P1 = (300,100,400,180,0,180,0,0)(7,0)

P2 = (0,0,-5,0,0,0)(0,0) [Beispiel ohne Angabe der Zusatzachsensdaten]

P3 = (100,200,300,0,0,90)(4,0) [Beispiel für 4-achsigen SCARA-Roboter]



Die Koordinaten, Stellungsdaten und die zusätzlichen Achsendaten haben folgende Struktur und Bedeutung:

Struktur: X, Y, Z, A, B, C, L1, L2

- X, Y, Z sind die Daten der Koordinaten. Sie geben die Position der Handspitze (TCP = Tool Center Point) des Roboters im kartesischen Koordinatensystem wieder. Sie werden in mm angegeben.
- A, B, C sind Orientierungsdaten der Roboterhand. Sie geben die Orientierung der Hand im Raum wieder. Ihre Einheit ist Grad. Die Teaching Box, die PC-Support-Software oder COSIROP zeigen die Einheit Grad an. Programminterne Substitutionen und Berechnungen werden jedoch in Radiant ausgeführt.
- L1, L2 sind Zusatzachsendaten. Sie geben die Koordinaten für die zusätzlichen Achsen 1 und 2 an. Ihre Einheit ist mm oder Grad.

Bedeutung der Stellungsdaten (fehlen die Stellungsdaten, werden die Standardeinstellungen (7,0) verwendet (modellabhängig)):

Struktur: FL1, FL2

- FL1: Stellungsmerker

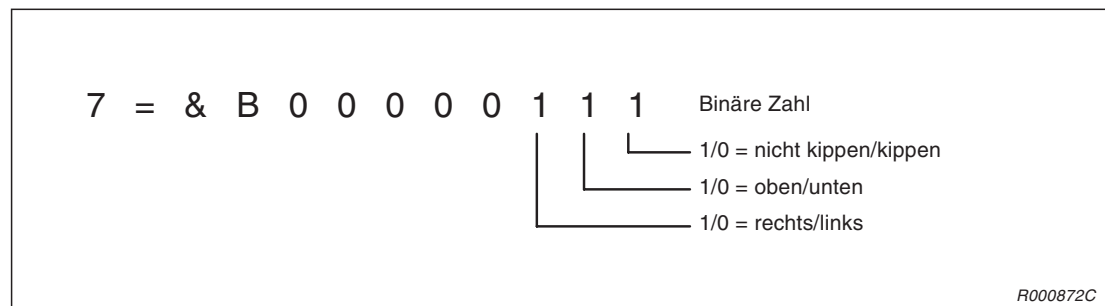


Abb. 5-4: Bedeutung der Stellungsmerker

Beispiele ▾

P1 = (X,Y,Z,A,B,C)(FL1,FL2)

P1 = (X,Y,Z,A,B,C,L1,L2)(FL1,FL2)



- FL2: Multirotationsinformation Standardeinstellung = 0 (Der Einstellbereich ist 0 bis +429496725. Informationen für 8 Achsen werden mit 4 Bits für 1 Achse dargestellt. Für den PC existieren 2 Anzeigemöglichkeiten: Anzahl der Rotationen (-8 bis 7) für jede Achse in dezimaler oder in hexadezimaler Form.

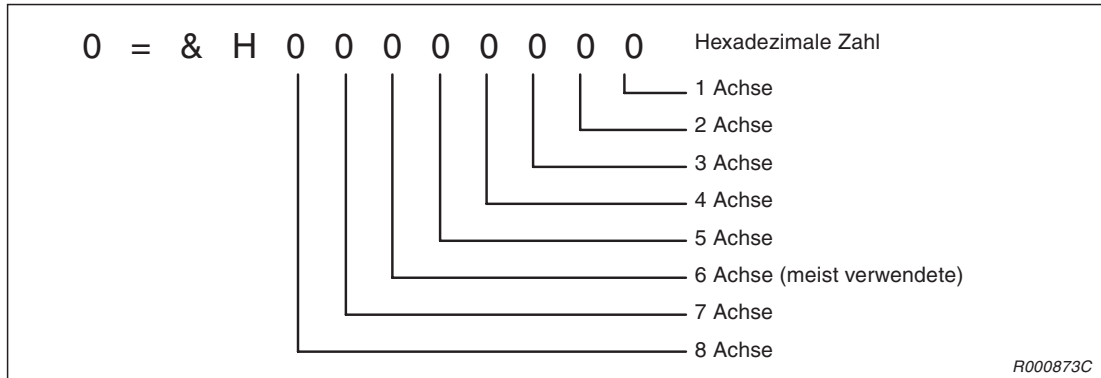


Abb. 5-5: Multirotationsdaten in hexadezimaler Form

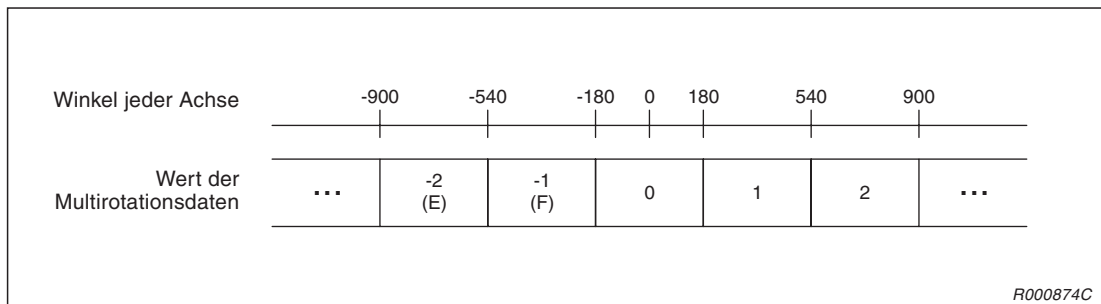


Abb. 5-6: Wert der Multirotationsdaten

HINWEISE

Es ist nicht erforderlich, die Koordinaten und Stellungsdaten für sämtliche 8 Achsen anzugeben. Bei unvollständigen Angaben werden die folgenden Achsendaten als undefiniert verarbeitet. Geben Sie die Daten für einen 4-achsigen Roboter (Achsenkonfiguration: X, Y, Z, C) wie folgt an: (X, Y, Z, , , C) oder (X, Y, Z, 0, 0, C).

Eine Positionskonstante darf keine Variable als Element enthalten.

Gelenkkonstanten

Folgende Abbildung zeigt die Syntax der Gelenkkonstanten:

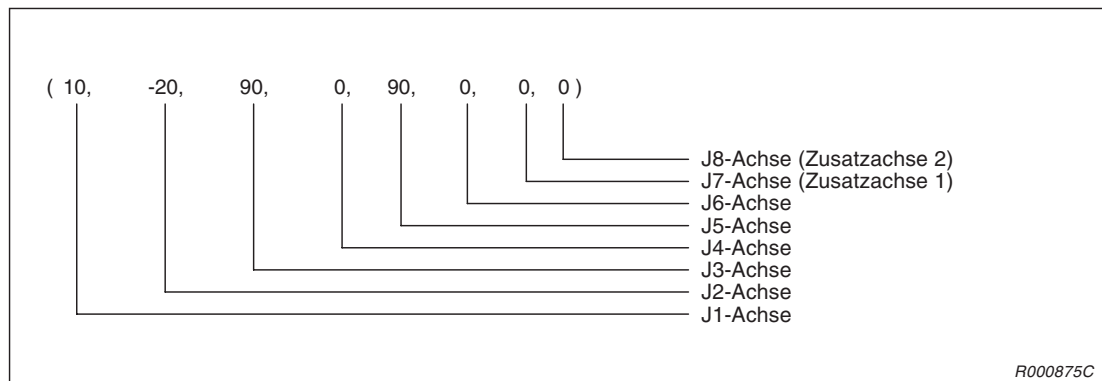


Abb. 5-7: Gelenkkonstanten

Beispiele ▾

6-achsiger Roboter	J1 = (0,10,80,10,90,0)
6-achsiger Roboter mit Zusatzachse	J1 = (0,10,80,10,90,0,10,10)
5-achsiger Roboter	J1 = (0,10,80,0,90,0)
5-achsiger Roboter mit Zusatzachse	J1 = (0,10,80,0,90,0,10,10)
4-achsiger Roboter	J1 = (0,20,90,0)
4-achsiger Roboter mit Zusatzachse	J1 = (10,20,90,0, , , 10,10)



Struktur und Bedeutung der Achsendaten:

Struktur: J1, J2, J3, J4, J5, J6, J7, J8

- J1 bis J6: Achsendaten (in mm oder Grad)
- J7 und J8: Daten der optionalen Zusatzachsen (je nach Einstellung des Parameters AXUN in mm oder Grad)
Bei SCARA-Robotern mit direkt angetriebener J3-Achse ist die Einheit mm.

HINWEIS

| Eine Gelenkkonstante darf keine Variable als Element enthalten.

Winkelbetrag

Die Angabe des Winkelbetrages erfolgt in Grad (nicht in Radiant). Bei einer Schreibweise 100DEG wird der Wert als Winkel interpretiert und kann so in trigonometrischen Funktionen verarbeitet werden.

Beispiel ▾

Der Sinus eines 90°-Winkels wird folgendermaßen dargestellt: SIN(90DEG).



5.1.9 Variablen

Ein Variablenname darf aus bis zu 8 Zeichen bestehen. Die folgenden Variablenwerte können eingesetzt werden. Sie unterscheiden sich anhand der Daten, die in ihnen abgelegt werden.

- Numerische Variablen, Zeichenkettenvariablen, Positionsvariablen, Gelenkvariablen, Ein- und Ausgabevariablen
- Numerische Variablen lassen sich weiterhin in ganze Zahlen (Integer), reelle Zahlen (Real) mit einfacher Genauigkeit und reelle Zahlen mit doppelter Genauigkeit unterteilen.

Variablen können nach ihrem Verwendungsbereich in lokale und globale Variablen eingeteilt werden.

- Lokale Variablen werden innerhalb eines Programms verwendet.
- Roboterstatusvariablen, externe Variablen und benutzerdefinierte externe Variablen sind globale Variablen und können auch programmübergreifend verwendet werden (Benutzerdefinierte externe Variablen sind durch einen Unterstrich (_) als zweites Zeichen gekennzeichnet.)

Folgende Abbildung zeigt die verschiedenen Variablentypen:

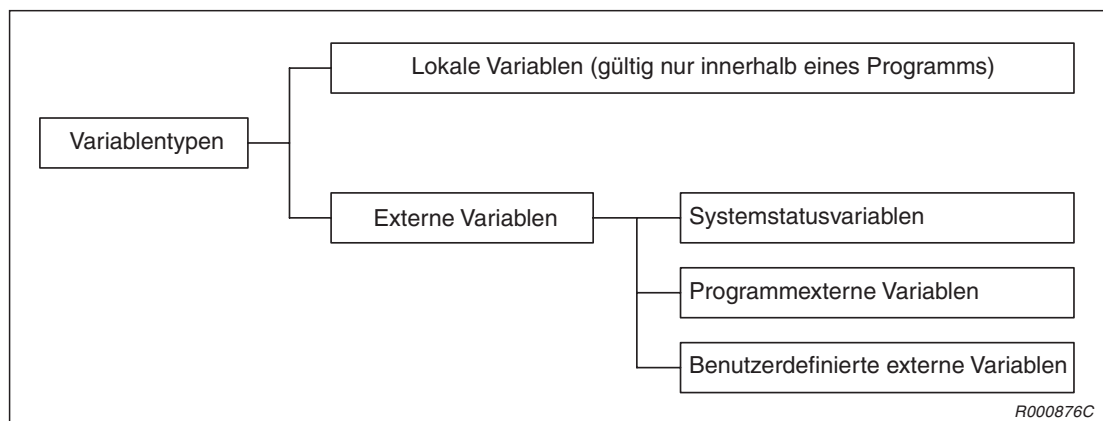


Abb. 5-8: Variablentypen

Folgende Abbildung zeigt die Syntax von Variablen:

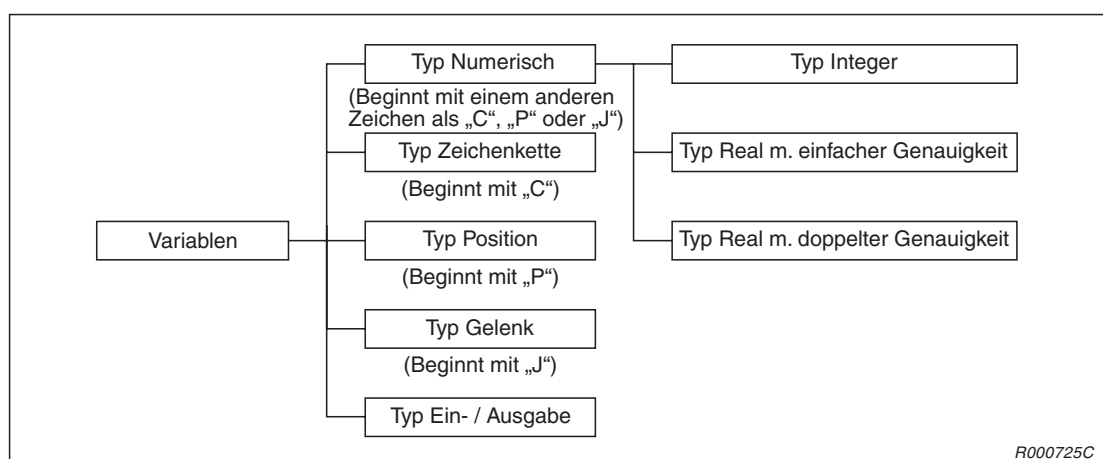


Abb. 5-9: Variablen

HINWEIS

Variablen werden beim Generieren und beim Laden oder Zurücksetzen eines Programms nicht gelöscht.

Numerische Variablen

Variablen, deren Namen mit einem anderen Zeichen als mit „P“, „J“ oder „C“ beginnen, sind numerische Variablen. In MELFA-BASIC IV wird eine numerische Variable in Anlehnung an den Anfangsbuchstaben des Wortes „Mathematik“ oft durch ein „M“ als erstes Zeichen im Variablennamen gekennzeichnet.

Beispiele ▽

M1 = 100
 M2! = -1.73E+10
 M3# = 0.123
 ABC = 1



Es besteht die Möglichkeit, den Typ einer numerischen Variablen durch ein Zeichen am Ende des Variablennamens zu kennzeichnen. Fehlt das Suffix in der Variablendeklaration, so wird die Variable als reelle Zahl mit einfacher Genauigkeit interpretiert.

Suffixe numerischer Variablen	Variablentyp
%	Integer
!	Real mit einfacher Genauigkeit
#	Real mit doppelter Genauigkeit

Tab. 5-2: Typen von numerischen Variablen

HINWEISE

Ein einmal registrierter Variablentyp kann nur vom Typ Integer in den Typ Real mit einfacher Genauigkeit umgewandelt werden. Es ist z. B. nicht möglich, eine Variable vom Typ Integer in den Typ Real mit doppelter Genauigkeit oder eine Variable vom Typ Real mit einfacher Genauigkeit in den Typ Real mit doppelter Genauigkeit umzuwandeln.

Es ist nicht möglich, zu einer bereits deklarierten Variablen ein Suffix hinzuzufügen. Das Suffix muss bei der Variablendeklaration während der Programmerstellung festgelegt werden.

Werden Zahlen mit einfacher Genauigkeit und Zahlen mit doppelter Genauigkeit gemeinsam verarbeitet oder ersetzt, tritt ein Fehler auf.

Typ	Bereich
Integer	-32 768 bis 32 767
Real mit einfacher Genauigkeit	-3,40282347E+38 bis 3,40282347E+38
Real mit doppelter Genauigkeit	-1,7976931348623157E+308 bis 1,7976931348623157E+308

Tab. 5-3: Wertebereich

Zeichenkettenvariablen

Der Name einer Zeichenkettenvariablen sollte mit „C“ beginnen und mit „\$“ enden:

Beispiele ▾

```
C1$ = "ABC"
CS$ = C1$
DEF CHAR MOJI
MOJI = "MOJIMOJI"
```

△

Positionsvariablen

Der Name einer Positionsvariablen sollte mit „P“ beginnen. Es können jedoch auch andere Buchstaben verwendet werden. Die Deklaration einer Positionsvariablen erfolgt über die Anweisung DEF POS. Es ist möglich, auf einzelne Komponenten einer Positionsvariablen zuzugreifen. In diesem Fall muss nach dem Variablennamen ein Punkt „.“ gesetzt und der Name der Komponente (z. B. „X“) angegeben werden:

```
P1.X, P1.Y, P1.Z, P1.A, P1.B, P1.C, P1.L1, P1.L2
```

Die Einheit der Winkelkomponenten (z. B. P1.A) ist RAD. Verwenden Sie zur Umwandlung der Winkelkomponente in Grad die Funktion DEG.

Beispiele ▾

```
P1 = PORG
DIM P3(10)
M1 = P1.X           (Einheit: mm)
M2 = DEG(P1.A)     (Einheit: Grad)
DEG POS L10
MOV L10
```

△

Gelenkvariablen

Der Name einer Gelenkvariablen sollte mit „J“ beginnen. Es können jedoch auch andere Buchstaben verwendet werden. Die Deklaration einer Gelenkvariablen erfolgt über die Anweisung DEF JNT. Es ist möglich, auf einzelne Komponenten einer Gelenkvariablen zuzugreifen. In diesem Fall muss nach dem Variablennamen ein Punkt „.“ gesetzt und der Name der Komponente (z. B. „J1“) angegeben werden:

```
JDATA.J1, JDATA.J2, JDATA.J3, JDATA.J4, JDATA.J5, JDATA.J6, JDATA.J7, JDATA.J8
```

Beispiele ▾

```
JSTART = (0,0,90,0,90,0,0,0)
JDATA = JSTART
DIM J3(10)
M1 = J1.J1         (Einheit: Radiant)
M2 = DEG(J1.J2)   (Einheit: Grad)
DEG JNT K10
MOV K10
```

△

E/A-Variablen

Folgende Ein- und Ausgangsvariablen können verwendet werden. Sie sind als Roboterstatusvariablen vordefiniert.

E/A-Variable	Beschreibung
M_IN	Für Lese-Zugriff auf Eingangssignalbits
M_INB	Für Lese-Zugriff auf Eingangssignalbytes (8-Bit-Signal)
M_INW	Für Lese-Zugriff auf Eingangssignalbytes (16-Bit-Signal)
M_OUT	Für Schreib-/Lese-Zugriff auf Ausgangssignalbits
M_OUTB	Für Schreib-/Lese-Zugriff auf Ausgangssignalbytes (8-Bit-Signal)
M_OUTW	Für Schreib-/Lese-Zugriff auf Ausgangssignalbytes (16-Bit-Signal)
M_DIN	Für Lese-Zugriff auf Eingangsregister bei CC-Link-Verbindung
M_DOUT	Für Schreib-/Lese-Zugriff auf Ausgangsregister bei CC-Link-Verbindung

Tab. 5-4: E/A-Variablen

Detaillierte Hinweise zu den E/A-Variablen finden Sie im Abschn. 7.2.26 „M_IN/M_INB/M_INW“, im Abschn. 7.2.33 M_OUT/M_OUTB/M_OUTW und im Abschn. 7.2.20 „M_DIN/M_DOUT“.

Feldvariablen

Numerische Variablen, Zeichenkettenvariablen, Positionsvariablen und Gelenkvariablen können in Feldvariablen verwendet werden. Die Festlegung der Feldvariablenelemente erfolgt über die Indizierung der jeweiligen Variablen. Deklarieren Sie die Feldvariablen, bevor Sie sie verwenden (siehe auch Abschn. 6.3.28 „DIM-Befehl“). Eine Feldvariable darf aus maximal 3 Dimensionen bestehen.

Beispiele ▾

DIM M1(10) Typ Real mit einfacher Genauigkeit
 DIM M2%(10) Typ Integer
 DIM M3!(10) Typ Real mit einfacher Genauigkeit
 DIM M4#(10) Typ Real mit doppelter Genauigkeit
 DIM P1(20)
 DIM J1(5)
 DIM ABC (10,10,10)

△

Die Indizierung einer Variablen beginnt mit 1. Nur bei den speziellen Ein-/Ausgangssignalvariablen (M_IN, M_OUT etc.) beginnt die Indizierung bei 0. Stellen Sie vor der Deklaration von Feldvariablen sicher, dass genügend freier Speicherplatz vorhanden ist.

5.1.10 Externe Variablen

Externe Variablen sind durch einen Unterstrich () an der zweiten Stellen des Bezeichners (Variablenname) gekennzeichnet. Benutzerdefinierte externe Variablen müssen im Benutzerbasisprogramm registriert sein. Die Werte können programmübergreifend verwendet werden. Externe Variablen ermöglichen somit den Austausch von Daten zwischen Programmen.

Man unterscheidet vier Typen von externen Variablen:

- numerische Variablen
- Positionsvariablen
- Gelenkvariablen
- Zeichenkettenvariablen

Folgende externe Variablen stehen zur Verfügung:

Externe Variablen	Beschreibung	Beispiel
Programmexterne Variablen	Typ externer Variablen	P_01, M_01, P_100(1) usw.
Benutzerdefinierte externe Variablen	Der Name kann vom Benutzer frei definiert werden. Die Variablendeklaration erfolgt über DEF POS, DEF JNT, DEF CHAR oder DEF INTE/FLOAT/DOUBLE im Benutzerbasisprogramm	P_GENTEN, M_MACHI
Roboterstatusvariablen	Roboterstatusvariablen sind Systemvariablen mit festgelegten Funktionen.	M_IN, M_OUT, P_CURR, M_PI usw.

Tab. 5-5: Externe Variablen

Programmexterne Variablen

In folgender Tabelle sind die verfügbaren programmexternen Variablen aufgeführt. Die Variablenamen sind festgelegt, die Anwendung kann vom Anwender definiert werden.

Datentyp	Variablenname	Anzahl
Position	P_00 bis P_19 P_20 bis P_39 ^①	20 20
Positions-Feldvariable (Anzahl der Elemente: 10)	P_100() bis P_104() P_105() bis P_109() ^①	5 5
Gelenk	J_00 bis J_19 J_20 bis J_39 ^①	20 20
Gelenk-Feldvariable (Anzahl der Elemente: 10)	J_100() bis J_104() J_105() bis J_109() ^①	5 5
Numerisch	M_00 bis M_19 M_20 bis M_39 ^①	20 20
Numerische Feldvariable (Anzahl der Elemente: 10)	M_100() bis M_104() M_105() bis M_109() ^①	5 5
Zeichenketten	C_00 bis C_19 C_20 bis C_39 ^①	20 20
Zeichenketten-Feldvariablen (Anzahl der Elemente: 10)	C_100() bis C_104() C_105() bis C_109() ^①	5 5

Tab. 5-6: Programmexterne Variablen

^① Ab Software-Version J1 ist der Bereich der programmexternen Variablen erweitert worden. Zur Nutzung des erweiterten Bereichs ist Parameter PRGGBL von „0 (Standard/Grundeinstellung)“ auf „1 (erweiterter Bereich)“ zu ändern und die Spannungsversorgung aus- und wieder einzuschalten. Wurde für eine programmexterne und eine benutzerdefinierte externe Variable zweimal derselbe Name vergeben, erfolgt beim Einschalten der Spannungsversorgung eine Fehlermeldung und eine Erweiterung des Bereichs ist nicht möglich. Ändern Sie in diesem Fall den Namen der benutzerdefinierten Variablen.

Benutzerdefinierte externe Variablen

Reicht die oben aufgelistete Anzahl an programmexternen Variablen nicht aus oder sollen spezielle Namen vergeben werden, können Sie im Benutzerbasisprogramm externe Variablen definieren.

Bevor Sie benutzerdefinierte externe Variablen verwenden können, müssen Sie:

- Ein Benutzerbasisprogramm erstellen
- Das Programm im Parameter „PRGUSR“ registrieren und die Versorgungsspannung aus- und einschalten.
- Ein Programm zur Verwendung der benutzerdefinierten externen Variablen erstellen.

Eine über den Befehl DEF deklarierte Variable, mit einem Unterstrich (_) an der zweiten Stelle im Variablennamen, wird als externe Variable verarbeitet. Dazu muss die Variable im Basisprogramm deklariert sein. Das Benutzerbasisprogramm muss nicht ausgeführt werden. Es reicht aus, nur die Zeilen mit den Variablendeklarationen zu erstellen.

Sollen im Benutzerbasisprogramm Feldvariablen erstellt und als externe Variablen eingesetzt werden, so ist eine zweite Deklaration über die DIM-Anweisung in dem Programm erforderlich, in dem sie verwendet werden. Einzelne Variablen erfordern keine erneute Deklaration.

Folgendes Beispiel zeigt die Verwendung benutzerdefinierter externer Variablen.

Beispiel ▾

Hauptprogramm (Programm 1)

```

10 DIM P_100(10)           'Zweite Deklaration der externen Variablen
20 DIM M_200(10)           'Zweite Deklaration der externen Variablen
30 MOV P_100(1)
40 IF M_200(1) = 1 THEN HLT

```

Benutzerbasisprogramm (Programm UBP)

```

10 DEF POS P_900, P_901, P_903 'Zweite Deklaration der externen Variablen
20 DIM P_100(10)               'Die Variable muss in dem Programm, in dem sie
                               verwendet wird, noch einmal deklariert werden.

30 DEF INTE M_100
40 DIM M_200(10)               'Die Variable muss in dem Programm, in dem sie
                               verwendet wird, noch einmal deklariert werden.

```

Der Parameter „PRGUSR“ muss auf „UBP“ gesetzt werden.

△

Erstellung eines Benutzerbasisprogramms

Bei Verwendung von benutzerdefinierten Variablen wird ein Benutzerbasisprogramm benötigt. Dieses Programm wird nicht ausgeführt. Das Benutzerbasisprogramm enthält die Deklaration der benutzerdefinierten Variablen und wird über den Parameter PRGUSR definiert. Nach Einstellung des Parameters muss die Versorgungsspannung aus- und wieder eingeschaltet werden.

Die Registrierung des Benutzerbasisprogramms erfolgt über die PC-Support-Software oder COSIROP. Bei Verwendung der Software müssen lediglich die Anweisungen und anschließend die Positionsdaten festgelegt werden.

Ein Benutzerbasisprogramm kann über die Teaching Box oder einen Personalcomputer erstellt werden. Gehen Sie bei Verwendung der PC-Support-Software wie folgt vor:

- ① Speichern Sie ein als Benutzerbasisprogramm erstelltes Programm auf Ihrem Personalcomputer.
- ② Starten Sie den Programm-Manager über den Programm-Editor.
- ③ Legen Sie im Programm-Manager das in ① erstellte Programm als Quelle und den Roboter als Ziel fest. Führen Sie einen Kopiervorgang aus. Achten Sie darauf, dass das Kontrollkästchen „Position Variables“ deaktiviert und das Kontrollkästchen „Instructions“ aktiviert ist.
- ④ Wiederholen Sie die unter ③ aufgeführten Anweisungen nach Abschluss des Kopiervorgangs. Führen Sie den Kopiervorgang jedoch mit aktiviertem Kontrollkästchen „Position Variables“ und deaktiviertem Kontrollkästchen „Instructions“ aus.

Roboterstatusvariablen

Roboterstatusvariablen werden verwendet, um einen schnellen Zugriff auf den Roboterzustand zu ermöglichen oder um ihn zu ändern. Die Variablennamen und die Funktion der Roboterstatusvariablen sind vordefiniert. Eine detaillierte Funktionsbeschreibung der Variablen finden Sie in Kap. 7.

Nr.	Variablenname	Feld- element ^①	Inhalt	Zugriff	Datentyp, Einheit	Seite
1	P_CURR	Mechanismusnummer (1-3)	Augenblicksposition (XYZ)	Lesen	Position	7-55
2	J_CURR	Mechanismusnummer (1-3)	Augenblicksposition (Gelenk)	Lesen	Gelenk	7-8
3	J_ECURRE	Mechanismusnummer (1-3)	Aktuelle Encoderposition	Lesen	Gelenk	7-9
4	J_FBC	Mechanismusnummer (1-3)	Gelenkposition abgeleitet aus der Servorückmeldung	Lesen	Gelenk	7-10
5	J_AMPFBC	Mechanismusnummer (1-3)	Aktueller Wert der Servorückmeldung	Lesen	Gelenk	7-10
6	P_FBC	Mechanismusnummer (1-3)	XYZ-Position abgeleitet aus der Servorückmeldung	Lesen	Position	7-56
7	M_FBD	Mechanismusnummer (1-3)	Differenz zwischen der durch den Befehlswert vorgegebenen Sollposition und der durch die Encoderimpulse gemeldeten Istposition	Lesen	Position	7-23
8	M_CMPDST	Mechanismusnummer (1-3)	Abweichung zwischen Befehlswert und aktueller Position	Lesen	Real mit einfacher Genauigkeit, mm	7-15
9	M_CMPLMT	Mechanismusnummer (1-3)	Zurücksetzen des Fehlerstatus durch Aufruf eines Interruptprozesses, wenn nach Aktivierung der Achsenweichheit ein Grenzwert überschritten wird	Lesen	Integer	7-16
10	P_TOOL	Mechanismusnummer (1-3)	Zuletzt festgelegte Werkzeugkonvertierungsdaten	Lesen	Position	7-58
11	P_BASE	Mechanismusnummer (1-3)	Zuletzt festgelegte Basiskonvertierungsdaten	Lesen	Position	7-52
12	P_NTOOL	Mechanismusnummer (1-3)	Standardwert der Werkzeugkonvertierungsdaten	Lesen	Position	7-58
13	P_NBASE	Mechanismusnummer (1-3)	Standardwert der Basiskonvertierungsdaten	Lesen	Position	7-52
14	M_TOOL	Mechanismusnummer (1-3)	Zuletzt festgelegte Werkzeugnummer	Lesen/ schreiben	Integer	7-45
15	J_COLMXL	Mechanismusnummer (1-3)	Maximale Drehmomentabweichung bei aktivierter Kollisionsüberwachung	Lesen/ schreiben	Gelenk	7-6
16	M_COLSTS	Mechanismusnummer (1-3)	Status der Kollisionsüberwachung	Lesen/ schreiben	Integer	7-17
17	P_COLDIR	Mechanismusnummer (1-3)	Bewegungsrichtung vor einem Zusammenstoß	Lesen	Position	7-53
18	M_OPOVRD	—	Geschwindigkeitsübersteuerung über das Bedienfeld des Steuergerätes (0-100 %)	Lesen	Integer, %	7-27
19	M_OVRD	Programmplatznummer (1-32)	Zuletzt festgelegter Übersteuerungswert/gültig für das gesamte Programm (0-100 %)	Lesen	Integer, %	7-27

Tab. 5-7: Roboterstatusvariablen (1)

Nr.	Variablenname	Feld- element ^①	Inhalt	Zugriff	Datentyp, Einheit	Seite
20	M_JOVRD	Programm- platznummer (1–32)	Zuletzt festgelegter Über- steuerungswert/gültig nur bei Gelenk-Interpolation (0–100 %)	Lesen	Integer, %	7-27
21	M_NOVRD	Programm- platznummer (1–32)	Systemstandardwert (M_OVRD Standardwert)	Lesen	Real mit einfacher Genauigkeit, %	7-27
22	M_NJOVRD	Programm- platznummer (1–32)	Systemstandardwert (M_JOVRD Standardwert)	Lesen	Real mit einfacher Genauigkeit, %	7-27
23	M_WUPOV	Mechanismus- nummer (1–3)	Übersteuerung im Warmlauf- betrieb (50–100 %)	Lesen	Real mit einfacher Genauigkeit, %	7-49
24	M_WUPRT	Mechanismus- nummer (1–3)	Restzeit einer Achse im Warm- laufbetrieb (s)	Lesen	Real mit einfacher Genauigkeit, s	7-50
25	M_WUPST	Mechanismus- nummer (1–3)	Zeit bis zur Wiederholung des Warmlaufbetriebs (s)	Lesen	Real mit einfacher Genauigkeit, s	7-51
26	M_RATIO	Programm- platznummer (1–32)	Position bezogen auf die Zielposition	Lesen	Integer, %	7-36
27	M_RDST	Programm- platznummer (1–32)	Restverfahrweg zur Zielposition der Komponenten X, Y und Z	Lesen	Real mit einfacher Genauigkeit, mm	7-37
28	M_SPD	Programm- platznummer (1–32)	Zuletzt festgelegter Geschwin- digkeitswert (gültig für Linear- und Kreis-Interpolation)	Lesen	Real mit einfacher Genauigkeit, mm/s	7-42
29	M_NSPD	Programm- platznummer (1–32)	Systemstandardwert (M_SPD Standardwert)	Lesen	Real mit einfacher Genauigkeit, mm/s	7-42
30	M_RSPD	Programm- platznummer (1–32)	Aktuelle Geschwindigkeit (gültig für Linear- und Kreis- Interpolation)	Lesen	Real mit einfacher Genauigkeit, mm/s	7-42
31	M_ACL	Programm- platznummer (1–32)	Zuletzt festgelegte Beschleuni- gungszeit	Lesen	Real mit einfacher Genauigkeit, %	7-12
32	M_DACL	Programm- platznummer (1–32)	Zuletzt festgelegte Abbremszeit	Lesen	Real mit einfacher Genauigkeit, %	7-12
33	M_NACL	Programm- platznummer (1–32)	Systemstandardwert (M_ACL Standardwert)	Lesen	Real mit einfacher Genauigkeit, %	7-12
34	M_NDAACL	Programm- platznummer (1–32)	Systemstandardwert (M_DACL Standardwert)	Lesen	Real mit einfacher Genauigkeit, %	7-12
35	M_ACLSTS	Programm- platznummer (1–32)	Aktueller Status der Beschleuni- gung-/Abbremsung 0 = gestoppt, 1 = beschleunigt, 2 = konstante Geschwindigkeit, 3 = bremst	Lesen	Integer	7-12
36	M_SETADL	Achsen- nummer (1–8)	Verhältnis der Beschleunigungs/ Abbremszeiten Ab Software-Version J1	Lesen/ schreiben	Real mit einfacher Genauigkeit, %	7-39
37	M_LDFACT	Achsen- nummer (1–8)	Lastverhältnis der einzelnen Servomotorachsen Ab Software-Version J1	Lesen	Real mit einfacher Genauigkeit, %	7-28
38	M_RUN	Programm- platznummer (1–32)	Programmstatus 1 = Betrieb, 0 = kein Betrieb	Lesen	Integer	7-38

Tab. 5-7: *Roboterstatusvariablen (2)*

Nr.	Variablenname	Feld- element ^①	Inhalt	Zugriff	Datentyp, Einheit	Seite
39	M_WAI	Programm- platznummer (1–32)	Wartestatus 1 = Pause, 0 = keine Pause	Lesen	Integer	7-48
40	M_PSA	Programm- platznummer (1–32)	Status der Programm- wählbarkeit 1 = Auswahl freigegeben 0 = Auswahl gesperrt (Pause)	Lesen	Integer	7-35
41	M_CYS	Programm- platznummer (1–32)	Zyklusbetrieb aktiv 1 = Zyklusbetrieb; 0 = kein Zyklusbetrieb	Lesen	Integer	7-19
42	M_CSTP	—	Zyklusstopp 1 = Betrieb, 0 = kein Betrieb	Lesen	Integer	7-17
43	C_PRG	Programm- platznummer (1–32)	Programmname für Ausführung	Lesen	Zeichenkette	7-4
44	M_LINE	Programm- platznummer (1–32)	Aktuell ausgeführte Zeilennummer	Lesen	Integer	7-30
45	M_SKIPCQ	Programm- platznummer (1–32)	Wird nach Ausführung eines SKIP-Befehls auf „1“, sonst auf „0“ gesetzt	Lesen	Integer	7-41
46	M_BRKCQ	—	Ergebnis der BREAK-Befehls- ausführung (1: BREAK, 0: kein BREAK)	Lesen	Integer	7-13
47	M_ERR	Programm- platznummer (1–32)	Fehlermeldung 1 = Fehler, 0 = kein Fehler	Lesen	Integer	7-21
48	M_ERRLVL	—	Schwelle für Lesefehler: Warnung/niedrig/hoch1/hoch2 = 1/2/3/4	Lesen	Integer	7-21
49	M_ERRNO	—	Fehlernummer lesen	Lesen	Integer	7-21
50	M_SVO	Mechanismus- nummer (1–3)	Servospannung EIN 1 = Servo EIN, 0 = Servo AUS	Lesen	Integer	7-43
51	M_UAR	Mechanismus- nummer (1–3)	Bitdaten 1 = innerhalb des benutzer- definierten Bereiches, 0 = außerhalb des benutzer- definierten Bereiches; Bit 0: Bereich 1 ... Bit 7: Bereich 2	Lesen	Integer	7-47
52	M_IN	Eingangs- nummer (0–32767)	Verwenden Sie die Variable zur Eingabe externer Bitsignale Allgemeine Bit-Schnittstelle: Bit-Eingang: 0 = AUS, 1 = EIN Für CC-Link stehen ab 6000 Signalnummern zur Verfügung.	Lesen	Integer	7-26
53	M_INB	Eingangs- nummer (0–32767)	Verwenden Sie die Variable zur Eingabe externer Bytesignale (8 Bit) Allgemeine Bit-Schnittstelle: Byte-Eingang Für CC-Link stehen ab 6000 Signalnummern zur Verfügung.	Lesen	Integer	7-26

Tab. 5-7: *Roboterstatusvariablen (3)*

Nr.	Variablenname	Feld-element ^①	Inhalt	Zugriff	Datentyp, Einheit	Seite
54	M_INW	Eingangsnummer (0–32767)	Verwenden Sie die Variable zur Eingabe externer Bytesignale (16 Bit) Allgemeine Bit-Schnittstelle: Wort-Eingang Für CC-Link stehen ab 6000 Signalnummern zur Verfügung.	Lesen	Integer	7-26
55	M_OUT	Ausgangsnummer (0–32767)	Verwenden Sie die Variable zur Ausgabe externer Bitsignale Allgemeine Bit-Schnittstelle: Bit-Ausgang: 0 = AUS, 1 = EIN Für CC-Link stehen ab 6000 Signalnummern zur Verfügung.	Lesen/ Schreiben	Integer	7-33
56	M_OUTB	Ausgangsnummer (0–32767)	Verwenden Sie die Variable zur Ausgabe externer Bytesignale (8 Bit) Allgemeine Bit-Schnittstelle: Byte-Ausgang Für CC-Link stehen ab 6000 Signalnummern zur Verfügung.	Lesen/ Schreiben	Integer	7-33
57	M_OUTW	Ausgangsnummer (0–32767)	Verwenden Sie die Variable zur Ausgabe externer Bytesignale (16 Bit) Allgemeine Bit-Schnittstelle: Wort-Ausgang Für CC-Link stehen ab 6000 Signalnummern zur Verfügung.	Lesen/ Schreiben	Integer	7-33
58	M_DIN	Eingangsnummer (ab 6000)	Dezentrales CC-Link-Register: Eingangsregister	Lesen	Integer	7-20
59	M_DOUT	Ausgangsnummer (ab 6000)	Dezentrales CC-Link-Register: Ausgangsregister	Lesen/ Schreiben	Integer	7-20
60	M_HNDCQ	Eingangsnummer (1–8)	Eingang Handgreiferzustand	Lesen	Integer	7-25
61	P_SAFE	Mechanismusnummer (1–3)	Position des Rückzugspunktes	Lesen	Position	7-57
62	J_ORIGIN	Mechanismusnummer (1–3)	Gelenkkordinaten des Referenzpunktes	Lesen	Gelenk	7-11
63	M_OPEN	Dateinummer (1–8)	Prüft, ob eine Datei oder Kommunikationsleitung geöffnet ist	Lesen	Integer	7-32
64	C_MECHA	Programmplatznummer (1–32)	Name des Mechanismus	Lesen	Zeichenkette	7-3
65	C_MAKER	—	Herstellerinformation (max. 64 Zeichen)	Lesen	Zeichenkette	7-3
66	C_USER	—	Inhalt des Parameters „USERMSG“ (max. 64 Zeichen)	Lesen	Zeichenkette	7-5
67	C_DATE	—	Aktuelles Datum Jahr/Monat/Datum	Lesen	Zeichenkette	7-2
68	C_TIME	—	Aktuelle Zeit Stunde/Minute/Sekunde	Lesen	Zeichenkette	7-5
69	M_BTIME	—	Restzeit der Batterie	Lesen	Integer, Stunden	7-14

Tab. 5-7: Roboterstatusvariablen (4)

Nr.	Variablenname	Feld-element ^①	Inhalt	Zugriff	Datentyp, Einheit	Seite
70	M_TIMER	Timer-nummer (1–8)	Zeitdauer ab Bezugszeit	Lesen	Real mit einfacher Genauigkeit, ms	7-44
71	P_ZERO	—	Variable, deren Komponenten (X, Y, Z, A, B, C, FL1, FL2) alle auf „0“ gesetzt sind.	Lesen	Position	7-59
72	M_PI	—	Kreiszahl (3.1415...)	Lesen	Real mit doppelter Genauigkeit	7-34
73	M_EXP	—	Basis des natürlichen Logarithmus (2.71828...)	Lesen	Real mit doppelter Genauigkeit	7-22
74	M_G	—	Erdbeschleunigung (9.80665)	Lesen	Real mit doppelter Genauigkeit	7-25
75	M_ON	—	Eine „1“ wird gesetzt.	Lesen	Integer	7-31
76	M_OFF	—	Eine „0“ wird gesetzt.	Lesen	Integer	7-31
77	M_MODE	—	Betriebsmoduseinstellung des Drehschalters am Steuergerät: Teach/AUTO (Op.)/AUTO (Ext.) = 1/2/3	Lesen	Integer	7-31

Tab. 5-7: *Roboterstatusvariablen (5)*

- ^① Mechanismusnummer: Festlegung der Mechanismusnummer (1–3) im Multitask-Betrieb
 Programmplatznummer: Festlegung des Programmplatzes (1–32) im Multitask-Betrieb
 Eingangsnummer: Bit-Nummer des Eingangssignals (0–32767)
 Ausgangsnummer: Bit-Nummer des Ausgangssignals (0–32767)

5.1.11 Logische Werte

Logische Werte geben die Resultate von Vergleichsoperationen oder Ein- und Ausgangszuständen wieder. Ein Ergebnis ungleich 0 entspricht dem Wert „wahr“ und ein Ergebnis gleich 0 entspricht dem Wert „unwahr“. Ergebnisse werden im Integer-Format angezeigt. Bei Substitutionen wird für den Wert „wahr“ eine „1“ gesetzt. Folgende Tabelle zeigt die logischen Werte und deren Bedeutung:

Durch eine „1“ dargestellte Zustände	Durch eine „0“ dargestellte Zustände
Ergebnis einer Vergleichsoperation (falls wahr)	Ergebnis einer Vergleichsoperation (falls unwahr)
Ergebnis einer logischen Operation (falls wahr)	Ergebnis einer logischen Operation (falls unwahr)
Schalter EIN	Schalter AUS
Ein-/Ausgangsignal EIN	Ein-/Ausgangsignal AUS
Handgreifer offen (Stromfluss durch die Hand)	Handgreifer geschlossen (kein Stromfluss durch die Hand)
Einstellungen für Freigabe oder Gültigkeit (wie zum Beispiel bei Interrupts)	Einstellungen für Sperren oder Ungültigkeit (wie zum Beispiel bei Interrupts)

Tab. 5-8: Logische Werte und deren Bedeutung

5.1.12 Funktionen

Mit dem Argument einer Funktion wird eine durch die Funktion festgelegte Rechenoperation durchgeführt. Das Ergebnis kann ein numerischer Typ oder eine Zeichenkette sein.

Benutzerdefinierte Funktionen

Benutzerdefinierte Funktionen werden mit dem Befehl DEF FN erstellt.

Beispiel ▾

DEF FNMADD(MA, MB) = MA + MB



Funktionen beginnen mit den Zeichen „FN“. Das dritte Zeichen dient zur Beschreibung des Datentyps (Zeichenkette: C, Numerischer Wert: M, Position: P, Gelenk: J). Es können bis zu 8 Zeichen verwendet werden.

Fest definierte Funktionen

Folgende Tabelle zeigt die fest definierten Funktionen:

Funktionsart	Funktionsname (Format)	Bedeutung	Seite	Ergebnis
Numerische Funktionen	ABS (<Numerischer Ausdruck>)	Bildet den Betrag	8-2	Numerischer Wert
	CINT (<Numerischer Ausdruck>)	Rundet den dezimalen Wert zu einer Integer-Zahl	8-9	
	DEG (<Numerischer Ausdruck: radian>)	Wandelt die Einheit des Winkels von Radiant (rad) in Grad (deg) um	8-15	
	EXP (<Numerischer Ausdruck>)	Berechnet den Wert der Exponentialfunktion	8-16	
	FIX (<Numerischer Ausdruck>)	Erzeugt einen Integer-Anteil	8-17	
	INT (<Numerischer Ausdruck>)	Erzeugt die größtmögliche Integer-Zahl, die kleiner als der Wert des numerischen Ausdrucks ist	8-21	
	LEN (<Ausdruck für eine Zeichenkette>)	Berechnet die Länge der Zeichenkette	8-24	
	LN (<Numerischer Ausdruck>)	Berechnet den natürlichen Logarithmus	8-24	
	LOG (<Numerischer Ausdruck>)	Berechnet den dekadischen Logarithmus	8-25	
	MAX (<Numerischer Ausdruck>...)	Berechnet den Maximalwert der numerischen Ausdrücke	8-25	
	MIN (<Numerischer Ausdruck>...)	Berechnet den Minimalwert der numerischen Ausdrücke	8-27	
	RAD (<Numerischer Ausdruck: deg>)	Wandelt die Einheit des Winkels von Grad (deg) in Radiant (rad) um	8-33	
	SGN (<Numerischer Ausdruck>)	Prüft das Vorzeichen des numerischen Ausdrucks	8-45	
	SQR (<Numerischer Ausdruck>)	Berechnet die Quadratwurzel	8-46	
	STRPOS (<Ausdruck für eine Zeichenkette>, <Ausdruck für eine Zeichenkette>)	Gibt die Position der zweiten Zeichenkette innerhalb der ersten Zeichenkette an	8-46	
RND (<Numerischer Ausdruck>)	Ermittelt eine Zufallszahl	8-36		

Tab. 5-9: Fest definierte Funktionen (1)

Funktionsart	Funktionsname (Format)	Bedeutung	Seite	Ergebnis
Numerische Funktionen	ASC (<Typ Zeichenkette>)	Erzeugt den ASCII-Code für das erste Zeichen in der Zeichenkette	8-4	Numerischer Wert
	CVI (<Typ Zeichenkette>)	Wandelt eine 2-Byte-Zeichenkette in einen Integer-Wert um	8-12	
	CVS (<Typ Zeichenkette>)	Wandelt eine 4-Byte-Zeichenkette in einen Real-Wert mit einfacher Genauigkeit um	8-13	
	CVD (<Typ Zeichenkette>)	Wandelt eine 8-Byte-Zeichenkette in einen Real-Wert mit doppelter Genauigkeit um	8-14	
	VAL (<Typ Zeichenkette>)	Wandelt eine Zeichenkette in einen numerischen Wert um	8-48	
Trigonometrische Funktionen	ATN (<Numerischer Ausdruck>)	Berechnet den Arcus Tangens (Einheit: rad) Definitionsbereich: numerischer Wert, $-\pi/2$ bis $+\pi/2$	8-5	Numerischer Wert
	ATN2 (<Numerischer Ausdruck>, <Numerischer Ausdruck>)	Berechnet den Arcus Tangens (Einheit: rad) ($\theta = \text{ATN2}(\Delta y, \Delta x)$) Definitionsbereich: numerischer Wert von Δy und Δx ungleich 0, $-\pi$ bis $+\pi$	8-5	
	COS (<Numerischer Ausdruck>)	Berechnet den Kosinus (Einheit: rad) Definitionsbereich: numerischer Wert, -1 bis $+1$	8-11	
	SIN (<Numerischer Ausdruck>)	Berechnet den Sinus (Einheit: rad) Definitionsbereich: numerischer Wert, -1 bis $+1$	8-45	
	TAN (<Numerischer Ausdruck>)	Berechnet den Tangens (Einheit: rad) Definitionsbereich: numerischer Wertebereich	8-47	
Zeichenkettenfunktionen	BIN\$ (<Numerischer Ausdruck>)	Wandelt den Wert des numerischen Ausdrucks in eine binäre Zeichenkette um	8-6	Zeichenkette
	CHR\$ (<Numerischer Ausdruck>)	Erzeugt ein Zeichen, das dem Wert des numerischen Ausdrucks entspricht	8-9	
	HEX\$ (<Numerischer Ausdruck>)	Wandelt den Wert des numerischen Ausdrucks in eine hexadezimale Zeichenkette um	8-20	
	LEFT\$ (<Zeichenkette>, <Numerischer Ausdruck>)	Erzeugt einen Teil der Zeichenkette Die Länge der erzeugten Zeichenkette, beginnend mit dem linken Zeichen, ist im zweiten Argument festgelegt.	8-23	
	MID\$ (<Zeichenkette>, <Numerischer Ausdruck>, <Numerischer Ausdruck>)	Erzeugt einen Teil der Zeichenkette Die Länge der erzeugten Zeichenkette ist im dritten, die Position von links im zweiten Argument festgelegt.	8-26	
	MIRROR\$ (<Typ Zeichenkette>)	Spiegelung der binären Bits der Zeichenkette	8-27	
	MKI\$ (<Numerischer Ausdruck>)	Wandelt den Wert des numerischen Ausdrucks in eine 2-Byte-Zeichenkette um	8-28	
	MKS\$ (<Numerischer Ausdruck>)	Wandelt den Wert des numerischen Ausdrucks in eine 4-Byte-Zeichenkette um	8-29	
	MKD\$ (<Numerischer Ausdruck>)	Wandelt den Wert des numerischen Ausdrucks in eine 8-Byte-Zeichenkette um	8-30	
	RIGHT\$ (<Zeichenkette>, <Numerischer Ausdruck>)	Erzeugt einen Teil der Zeichenkette Die Länge der erzeugten Zeichenkette, beginnend mit dem rechten Zeichen, ist im zweiten Argument festgelegt.	8-37	
STR\$ (<Numerischer Ausdruck>)	Wandelt den Wert des numerischen Ausdrucks in eine dezimale Zeichenkette um	8-47		
	CKSUM (<Zeichenkette>, <Numerischer Ausdruck>, <Numerischer Ausdruck>)	Bildet die Prüfsumme der Zeichenkette Schreibt den Wert des niederwertigen Bytes aus der Summe der im zweiten und dritten Argument festgelegten Zeichenkette in die Zeichenkette des ersten Arguments	8-10	Numerischer Wert

Tab. 5-9: Fest definierte Funktionen (2)

Funktionsart	Funktionsname (Format)	Bedeutung	Seite	Ergebnis
Positionsvariablen	DIST (<Position>, <Position>)	Berechnet den Abstand zwischen zwei Punkten	8-16	Position
	FRAM (<Position 1>, <Position 2>, <Position 3>)	Berechnet das Koordinatensystem über 3 Punkte Position 1 entspricht dem Flächenursprung, Position 2 dem Punkt in der Fläche der X- und Position 3 dem Punkt in der Fläche der Y-Achse. Der Flächenursprungspunkt und die Stellung sind durch die X-, Y- und Z-Koordinaten der 3 Positionen beschrieben und können über den Rücksetzwert (Position) zurückgesetzt werden. Die Ausführung erfolgt ohne Berücksichtigung der Mechanismusstruktur über 6 Achsen und 3 Dimensionen. Die Funktion kann nicht bei 5-achsigen Robotern verwendet werden, da die Orientierungsdaten A, B und C eine andere Bedeutung haben.	8-18	
	RDFL1 (<Position>, <Numerischer Wert>)	Überträgt den Stellungsmerker der festgelegten Position als Zeichenkette Argument <numerischer Wert>: 0 = R/L, 1 = A/B, 2 = F/N	8-34	Zeichenkette
	SETFL1 (<Position>, <Zeichen>)	Änderung des Stellungsmerkers der festgelegten Position Die zu ändernden Daten werden über Zeichen definiert (R/L/A/B/F/N).	8-38	
	RDFL2 (<Position>, <Numerischer Wert>)	Überträgt die Multirotationsdaten der festgelegten Position als numerischen Wert (-2 bis 1) Das Argument <numerischer Ausdruck> überträgt die Achsennummer (1 bis 8).	8-35	Numerischer Wert
	SETFL2 (<Position>, <Numerischer Wert>, <Numerischer Wert>)	Änderung der Multirotationsdaten der festgelegten Position als numerischer Wert (-2 bis 1) Die linke Seite des Ausdrucks entspricht der Achsennummer, die geändert werden soll, die rechte Seite entspricht dem Wert.	8-40	
	ALIGN (<Position>)	Setzt den Wert der Position mit dem kleinstmöglichen senkrechten oder waagerechten Abstand zur Stellung (A, B, C) der Position 1 Die Funktion kann nicht bei 5-achsigen Robotern verwendet werden, da die Orientierungsdaten A, B und C eine andere Bedeutung haben.	8-3	
	INV (<Position>)	Invertiert die Positions-Matrix	8-22	Position
	PTOJ (<Position>)	Konvertiert die Positions- in Gelenkdaten	8-32	Gelenk
	JTOP (<Position>)	Konvertiert die Gelenk- in Positionsdaten	8-22	Position
	ZONE (<Position 1>, <Position 2>, <Position 3>)	Prüft, ob die Position 1 innerhalb des durch die Positionen 2 und 3 definierten Quaders liegt (außerhalb = 0, innerhalb = 1) Für Komponenten, die nicht geprüft werden sollen oder fehlen, müssen die entsprechenden Positionskoordinaten auf folgende Werte gesetzt werden: Ist die Einheit Grad, muss Position 2 auf -360° und Position 3 auf 360° gesetzt werden. Ist die Einheit mm, muss Position 2 auf -10000 und Position 3 auf 10000 gesetzt werden.	8-49	Numerischer Wert

Tab. 5-9: Fest definierte Funktionen (3)

Funktionsart	Funktionsname (Format)	Bedeutung	Seite	Ergebnis
Positionsvariablen	ZONE2 (<Position 1>, <Position 2>, <Position 3>, <Numerischer Wert 1>, <Numerischer Wert 2>, <Numerischer Wert 3>, <Position 4>)	Prüft, ob die Position 1 innerhalb des durch die Positionen 2 und 3 definierten Zylinders liegt (außerhalb = 0, innerhalb = 1) Es werden nur die Koordinaten X, Y und Z geprüft; die Orientierungsdaten A, B und C werden ignoriert.	8-51	Numerischer Wert
	POSCQ (<Position>)	Prüft, ob die Position innerhalb des gültigen Bewegungsbereiches liegt	8-30	Numerischer Wert
	POSMID (<Position 1>, <Position 2>, <Numerischer Wert 1>, <Numerischer Wert 2>)	Berechnet die mittlere Position zwischen <Position 1> und <Position 2>	8-31	Position
	CALARC (<Position 1>, <Position 2>, <Position 3>, <Numerischer Wert 1>, <Numerischer Wert 2>, <Numerischer Wert 3>, <Position 4>)	Enthält die Daten des über die <Position 1>, <Position 2> und <Position 3> definierten Kreises	8-7	Numerischer Wert
	SETJNT (<J1-Achse>, <J2-Achse>, <J3-Achse>, <J4-Achse>, <J5-Achse>, <J6-Achse>, <J7-Achse>, <J8-Achse>)	Ändert die Werte einer Gelenkvariablen	8-41	Gelenk
	SETPOS (<X-Achse>, <Y-Achse>, <Z-Achse>, <A-Achse>, <B-Achse>, <C-Achse>, <L1-Achse>, <L2-Achse>)	Ändert die Werte einer Positionsvariablen	8-43	Position

Tab. 5-9: Fest definierte Funktionen (3)

5.1.13 Operanden

Numerische Variablen müssen in MELFA-BASIC IV nicht als Typ Integer oder Real deklariert werden. In Abhängigkeit von der ausgeführten Operation werden die Daten automatisch konvertiert. Dabei kann das Ergebnis in Abhängigkeit von der Reihenfolge der Datentypen unterschiedlich sein. Folgende Tabelle zeigt einige Beispiele:

Linkes Argument	Operation	Rechtes Argument	Ergebnis
15 (Typ Numerisch)	AND	256 (Typ Numerisch)	15 (Typ Numerisch)
P1 (Typ Position)	*	M1 (Typ Numerisch)	P2 (Typ Position)
M1 (Typ Numerisch)	*	P1 (Typ Position)	FEHLER

Tab. 5-10: Operationsergebnisse in Abhängigkeit der Datenreihenfolge

Konvertierung der Datentypen in Abhängigkeit der Operation

Folgende Tabelle zeigt die Konvertierung von Datentypen in Abhängigkeit von der ausgeführten Operation. Bei der Angabe von logischen Operationen ist die logische Negation ausgenommen.

Typ linkes Argument	Operation	Typ rechtes Argument					
		Zeichenkette	Numerischer Wert		Position	Gelenk	
			Integer	Real			
Zeichenkette	Substitution	Zeichenkette	—	—	—	—	
	Addition	Zeichenkette	—	—	—	—	
	Vergleich	Integer	—	—	—	—	
Numerischer Wert	Integer	Addition	—	Integer	Real	—	—
		Subtraktion	—	Integer	Real	—	—
		Multiplikation	—	Integer	Real	—	—
		Division	—	Integer	Real	—	—
		Integer-Division	—	Integer	Integer	—	—
		Modulo	—	Integer	Integer	—	—
		Exponential	—	Integer	Integer	—	—
		Substitution	—	Integer	Integer	—	—
		Vergleich	—	Integer	Integer	—	—
	Logisch	—	Integer	Integer	—	—	
	Real	Addition	—	Real	Real	—	—
		Subtraktion	—	Real	Real	—	—
		Multiplikation	—	Real	Real	—	—
		Division	—	Real	Real	—	—
		Integer-Division	—	Integer	Integer	—	—
		Modulo	—	Integer	Integer	—	—
		Exponential	—	Integer	Real	—	—
		Substitution	—	Integer	Real	—	—
Vergleich		—	Integer	Integer	—	—	
Logisch	—	Integer	Integer	—	—		

Tab. 5-11: Konvertierung der Datentypen (1)

Typ linkes Argument	Operation	Typ rechtes Argument				
		Zeichenkette	Numerischer Wert		Position	Gelenk
			Integer	Real		
Position	Addition	—	—	—	Position	—
	Subtraktion	—	—	—	Position	—
	Multiplikation	—	Position	Position	Position	—
	Division	—	Position	Position	Position	—
	Integer-Division	—	—	—	—	—
	Modulo	—	—	—	—	—
	Exponential	—	—	—	—	—
	Substitution	—	—	—	Position	—
	Vergleich	—	—	—	—	—
	Logisch	—	—	—	—	—
Gelenk	Addition	—	—	—	—	Gelenk
	Subtraktion	—	—	—	—	Gelenk
	Multiplikation	—	Gelenk	Gelenk	—	—
	Division	—	Gelenk	Gelenk	—	Gelenk
	Integer-Division	—	—	—	—	—
	Modulo	—	—	—	—	—
	Exponential	—	—	—	—	—
	Substitution	—	—	—	—	Gelenk
	Vergleich	—	—	—	—	—
Logisch	—	—	—	—	—	
Nur linkes Argument	Vorzeichenumkehr	—	Integer	Integer	Position	Gelenk
	Negation NOT	—	Integer	Integer	—	—

Tab. 5-11: Konvertierung der Datentypen (2)


HINWEISE

| Eine Division durch „0“ ist nicht möglich.

| Bei der Ausführung exponentieller, modulo und logischer Operationen werden reelle Zahlen vor der Verarbeitung in ganzzahlige Werte umgewandelt und abgerundet.

5.1.14 Rangfolge von Operationen

Werden in einem Ausdruck mehrere Operationen ausgeführt, gilt die in folgender Tabelle dargestellte Rangfolge:

Operation (Operator)	Typ der Operation	Priorität
Operation in Klammern ()	—	Hoch  Niedrig
Funktion	Funktion	
Exponent (^)	Operation mit numerischen Daten	
Operation mit einem Argument (+, -)	Operation mit numerischen Daten	
* /	Operation mit numerischen Daten	
\	Operation mit numerischen Daten	
MOD	Operation mit numerischen Daten	
+ -	Operation mit numerischen Daten	
<< >>	Logische Operation	
Vergleichsoperation (=, <>, ><, <, <=, =<, >, >=, ==)	Vergleichsoperation	
NOT	Logische Operation	
AND	Logische Operation	
OR	Logische Operation	
XOR	Logische Operation	

Tab. 5-12: Rangfolge von Operationen

5.1.15 Programmebenen

Beim Entwurf eines Programms muss die Anzahl der Ebenen und die Struktur festgelegt werden. Werden die in folgender Tabelle aufgeführten Befehle verwendet, erweitert sich die Programmstruktur um eine Ebene. Für jeden Befehl gibt es eine maximale Anzahl der Ebenen. Wird diese Anzahl überschritten, erfolgt eine Fehlermeldung.

Anzahl der Ebenen	Verfügbare Befehle
16 Ebenen	Wiederholschleifen (FOR ~ NEXT, WHILE ~ WEND)
8 Ebenen	Funktionsaufruf (CALLP)
800 Ebenen	Unterprogrammaufruf (GOSUB) ^①

Tab. 5-13: Programmebenen

^① Durch die Verwendung der Anweisungen FOR-NEXT, WHILE-WEND und CALLP wird die Programmstruktur flacher.

5.1.16 Reservierte Wörter

Reservierte Wörter haben im System eine bestimmte, festliegende Bedeutung. Sie dürfen zum Beispiel nicht als Programmname etc. vergeben werden. Zu den reservierten Wörtern zählen z. B. Anweisungen, Funktionen und Systemstatusvariablen.

6 MELFA-BASIC-IV-Befehle

6.1 Allgemeine Hinweise

In den nachfolgenden Abschnitten finden Sie eine Auflistung aller MELFA-BASIC-IV-Befehle und deren Anwendungsmöglichkeiten.

6.1.1 Beschreibung des verwendeten Formats

Funktion

Hier finden Sie eine Funktionsbeschreibung des Befehls.

Eingabeformat

Hier finden Sie das genaue Format zur Eingabe des Befehls. Befehlsparameter werden in spitzen Klammern „<>“ angegeben. Die eckige Klammer „[]“ kennzeichnet die wahlfreien Befehlsparameter. Die notwendige Eingabe eines Leerzeichens wird durch „□“ dargestellt.

Programmbeispiel

Hier finden Sie die Verwendung des Befehls in einem Beispielprogramm.

Erläuterung

Hier finden Sie eine detaillierte Beschreibung, Besonderheiten usw. des Befehls.

6.2 Übersicht der MELFA-BASIC-IV-Befehle

6.2.1 Alphabetische Übersicht

Befehl		Funktion	Seite
ACCEL	(Accelerate)	Beschleunigung und Verzögerung einstellen	6-8
ACT	(Act)	Interrupt freigeben/sperrern	6-10
BASE	(Base)	Basis	6-12
CALLP	(Call P)	Programm aufrufen	6-14
CHRSRCH	(Character Search)	Zeichenkette in einer Zeichenketten-Feldvariablen suchen	6-16
CLOSE	(Close)	Datei oder Kommunikationsleitung schließen	6-17
CLR	(Clear)	Löschen	6-18
CMP JNT	(Compliance Joint)	Achsenweichheit im Gelenkkordinatensystem aktivieren	6-20
CMP POS	(Compliance Posture)	Achsenweichheit im XYZ-kordinatensystem aktivieren	6-22
CMP TOOL	(Compliance Tool)	Achsenweichheit im Werkzeugkordinatensystem aktivieren	6-25
CMP OFF	(Compliance OFF)	Achsenweichheit deaktivieren	6-28
CMPG	(Compliance Gain)	Achsenweichheit einstellen	6-29
CNT	(Continuous)	Roboterbewegung überschleifen	6-30
COLCHK	(Col Check)	Kollisionsüberwachung aktivieren	6-33
COLLVL	(Col Level)	Empfindlichkeit der Kollisionsüberwachung einstellen	6-37
COM OFF	(Communication OFF)	Kommunikations-Interrupt sperren	6-39
COM ON	(Communication ON)	Kommunikations-Interrupt freigeben	6-40
COM STOP	(Communication Stop)	Kommunikations-Interrupt stoppen	6-41
DEF ACT	(Define Act)	Interrupt-Prozess definieren	6-42
DEF ARCH	(Define Arch)	Bogen definieren	6-45
DEF CHAR	(Define Character)	Zeichenkettenvariable definieren	6-47
DEF FN	(Define function)	Funktion definieren	6-48
DEF INTE/ FLOAT/ DOUBLE	(Define Integer/ Float/Double)	Numerische Variable definieren	6-50
DEF IO	(Define IO)	Ein-/Ausgangsvariable definieren	6-52
DEF JNT	(Define Joint)	Gelenkvariable definieren	6-54
DEF PLT	(Define pallet)	Palette definieren	6-55
DEF POS	(Define Position)	Positionsvariable definieren	6-58
DIM	(Dim)	Dimension einer Feldvariablen definieren	6-59
DLY	(Delay)	Verzögerung einstellen	6-61
ERROR	(Error)	Fehler generieren	6-63
END	(End)	Programmende	6-64
FINE	(Fine)	Feinpositionierung	6-65
FOR-NEXT	(For-next)	Programmschleife	6-67
FPRM	(FPRM)	Parameter definieren	6-69

Tab. 6-1: Übersicht der Befehle (1)

Befehl		Funktion	Seite
GETM	(Get Mechanism)	Mechanismus definieren	6-70
GOSUB	(Go Subroutine)	Sprung zu einem Unterprogramm	6-72
GOTO	(Go To)	Sprung zu einer Programmzeile oder Marke	6-73
HLT	(Halt)	Programmablauf stoppen	6-74
HOPEN/ HCLOSE	(Hand open/Hand close)	Handgreiferzustand festlegen	6-74
IF THEN ELSE	(If Then Else)	WENN ... DANN ... SONST-Schleife	6-77
INPUT #	(Input)	Daten einlesen	6-80
JOVRD	(J override)	Übersteuerung Gelenk-Interpolation	6-81
JRC	(Joint Roll Change)	Gelenkposition verändern	6-82
Label	(Label)	Sprungmarke	6-85
LOADSET	(Load set)	Hand- und Werkstückbedingung einstellen	6-86
MOV	(Move)	Bewegung mit Gelenk-Interpolation	6-88
MVA	(Move Arch)	Bewegung mit Bogen-Interpolation	6-90
MVC	(Move C)	Kreis-Interpolation	6-92
MVR	(Move R)	Kreis-Interpolation	6-94
MVR2	(Move R2)	Kreis-Interpolation	6-96
MVR3	(Move R3)	Kreis-Interpolation	6-98
MVS	(Move S)	Bewegung mit Linear-Interpolation	6-100
OADL	(Optimum Acceleration/ Deceleration)	Optimale Beschleunigung/Abbremsung	6-103
ON COM GOSUB	(ON Communication Go Subroutine)	Sprung zu einem Unterprogramm	6-105
ON-GOSUB	(ON GOSUB)	Sprung zu einem Unterprogramm	6-107
ON GOTO	(On go to)	Programmverzweigung	6-109
OPEN	(Open)	Datei oder Kommunikationsleitung öffnen	6-111
OVRD	(Override)	Übersteuerung	6-113
PLT	(Pallet)	Koordinaten für Palette berechnen	6-115
PREC	(Precision)	Verfahrweggenauigkeit erhöhen	6-118
PRINT #	(Print)	Daten übertragen	6-119
PRIORITY	(Priority)	Priorität festlegen	6-121
RELM	(Release mechanism)	Mechanismuszuordnung aufheben	6-122
REM	(Remarks)	Kommentar	6-123
RESET ERR	(Reset Error)	Fehler zurücksetzen	6-124
RETURN	(Return)	Rücksprung zum Hauptprogramm Zeile hinter GOSUB	6-125
SELECT CASE	(Select case)	Prozess ausführen	6-127
SERVO	(Servo)	Servo ein-/ausschalten	6-129
SKIP	(Skip)	Sprung in die nächste Zeile	6-130
SPD	(Speed)	Geschwindigkeit festlegen	6-131
TITLE	(Title)	Programmtitle festlegen	6-133
TOOL	(Tool)	Werkzeug-Konvertierungsdaten	6-134

Tab. 6-1: Übersicht der Befehle (2)

Befehl		Funktion	Seite
TORQ	(Torque)	Drehmomentgrenze definieren	6-136
WAIT	(Wait)	Wartestatus definieren	6-138
WHILE~ WEND	While End	Programmschleife	6-139
WTH	(With)	Anweisung hinzufügen	6-140
WTHIF	(With If)	Anweisung hinzufügen, wenn ...	6-141
XCLR	(X Clear)	Programmplatzauswahl zurücksetzen	6-142
XLOAD	(X Load)	Programm laden	6-143
XRST	(X Reset)	Programm zurücksetzen	6-144
XRUN	(X Run)	Programm starten	6-145
XSTP	(X Stop)	Programm stoppen	6-147
Substitute	(Substitute)	Daten ersetzen	6-148

Tab. 6-1: Übersicht der Befehle (3)

6.2.2 Anwendungsspezifische Übersicht

In folgender Tabelle sind die MELFA-BASIC-IV-Befehle in anwendungsspezifische Gruppen zusammengefasst. Die Reihenfolge wird durch die Verwendungshäufigkeit der Programmierbefehle bestimmt.

Anwendung	Befehl		Beschreibung	Seite
Befehle zur Bewegungssteuerung	MOV	(Move)	Gelenk-Interpolation	6-88
	MVS	(Move S)	Linear-Interpolation	6-100
	MVR	(Move R)	Kreis-Interpolation	6-94
	MVR2	(Move R2)	Kreis-Interpolation 2	6-96
	MVR3	(Move R3)	Kreis-Interpolation 3	6-98
	MVC	(Move C)	Kreis-Interpolation	6-92
	MVA	(Move Arch)	Bogen-Interpolation	6-90
	OVRD	(Override)	Übersteuerung	6-113
	SPD	(Speed)	Geschwindigkeit festlegen	6-131
	JOVRD	(J Override)	Übersteuerung Gelenk-Interpolation	6-81
	CNT	(Continuous)	Roboterbewegung überschleifen	6-30
	ACCEL	(Accelerate)	Beschleunigung und Verzögerung einstellen	6-8
	CMP JNT	(Compliance Joint)	Achsenweichheit im Gelenkkoordinatensystem einstellen	6-20
	CMP POS	(Compliance Posture)	Achsenweichheit im XYZ-Koordinatensystem einstellen	6-22
	CMP TOOL	(Compliance Tool)	Achsenweichheit im Werkzeugkoordinatensystem einstellen	6-25
	CMP OFF	(Compliance OFF)	Achsenweichheit deaktivieren	6-28
	CMPG	(Compliance Gain)	Verstärkung der Achsenweichheit	6-29
	OADL	(Optimal Acceleration)	Optimale Beschleunigung/Abbremsung	6-103
	LOADSET	(Load Set)	Hand- und Werkstückbedingung einstellen	6-86
	PREC	(Precision)	Verfahrweggenauigkeit	6-118
	TORQ	(Torque)	Drehmomentgrenze definieren	6-136
	JRC	(Join Roll Change)	Freigabe der Multirotation der Handgelenkachse	6-82
	FINE	(Fine)	Feinpositionierung	6-65
	SERVO	(Servo)	Servo ein-/ausschalten	6-129
WTH	(With)	Anweisung hinzufügen	6-140	
WTHIF	(With If)	Anweisung hinzufügen, wenn ...	6-141	
Befehle zur Programmsteuerung	REM	(Remarks)	Kommentar schreiben	6-123
	IF THEN ELSE ENDIF	(If Then Else)	Bedingte Verzweigung	6-77
	SELECT CASE	(Select Case)	Prozess ausführen	6-127
	GOTO	(Go To)	Sprung zu einer Programmzeile oder Marke	6-73
	GOSUB (RETURN)	(Go Subroutine)	Sprung zu einem Unterprogramm (Rücksprung)	6-72
	RESET ERR	(Reset Error)	Fehler zurücksetzen (darf nicht bei der Initialisierung verwendet werden)	6-124
	CALLP	(Call P)	Programm aufrufen	6-14
	FPRM	(FPRM)	Parameter definieren	6-69
	DLY	(Delay)	Verzögerung einstellen	6-61
HLT	(Halt)	Programmablauf stoppen	6-74	

Tab. 6-2: Einteilung der Befehle in anwendungsspezifische Gruppen (1)

Anwendung	Befehl		Beschreibung	Seite
Befehle zur Programmsteuerung	END	(End)	Programmende	6-64
	ON-GOSUB	(On Gosub)	Sprung zu einem Unterprogramm	6-107
	ON GOTO	(On Goto)	Programmverzweigung	6-109
	FOR-NEXT	(For-Next)	Programmschleife	6-67
	WHILE-WEND	(While End)	Programmschleife	6-139
	OPEN	(Open)	Datei oder Kommunikationsleitung öffnen	6-111
	PRINT #	(Print)	Daten übertragen	6-119
	INPUT #	(Input)	Daten einlesen	6-80
	CLOSE	(Close)	Datei oder Kommunikationsleitung schließen	6-17
	COLCHK	(Col Check)	Kollisionsüberwachung aktivieren	6-33
	ON COM GOSUB	(On Communication Go Subroutine)	Sprung zu einem Unterprogramm	6-105
	COM ON	(Communication ON)	Kommunikations-Interrupt freigeben	6-40
	COM OFF	(Communication OFF)	Kommunikations-Interrupt sperren	6-39
	COM STOP	(Communication STOP)	Kommunikations-Interrupt stoppen	6-41
	HOPEN/HCLOSE	(Hand Open/ Hand Close)	Handgreiferzustand festlegen	6-74
	ERROR	(Error)	Fehler generieren	6-63
	SKIP	(Skip)	Sprung in die nächste Zeile	6-130
	WAIT	(Wait)	Wartestatus	6-138
CLR	(Clear)	Löschen	6-18	
Definitions-befehle	DIM	(Dim)	Dimension einer Feldvariablen definieren	6-59
	DEF PLT	(Define Pallet)	Palette definieren	6-55
	PLT	(Pallet)	Koordinaten für Palette berechnen	6-115
	DEF ACT	(Define Act)	Interrupt-Prozess definieren	6-42
	ACT	(Act)	Interrupt freigeben/sperren	6-10
	DEF JNT	(Define Joint)	Gelenkvariable definieren	6-54
	DEF POS	(Define Position)	Positionsvariable definieren	6-58
	DEF INTE/FLOAT/DOUBLE	(Define Integer/Float/ Double)	Numerische Variable definieren	6-50
	DEF CHAR	(Define Character)	Zeichenkettenvariable definieren	6-47
	DEF IO	(Define IO)	Ein-/Ausgangsvariable definieren	6-52
	DEF FN	(Define Function)	Funktion definieren	6-48
	TOOL	(Tool)	Werkzeug-Konvertierungsdaten	6-134
	BASE	(Base)	Basis-Konvertierungsdaten	6-12

Tab. 6-2: Einteilung der Befehle in anwendungsspezifische Gruppen (2)

Anwendung	Befehl		Beschreibung	Seite
Multitask-Befehle	XLOAD	(X Load)	Programm laden	6-143
	XRUN	(X Run)	Programm starten	6-145
	XSTP	(X Stop)	Programm stoppen	6-147
	XRST	(X Reset)	Programm zurücksetzen	6-144
	XCLR	(X Clear)	Programmplatzauswahl zurücksetzen	6-142
	GETM	(Get Mechanism)	Mechanismus definieren	6-70
	RELM	(Release Mechanism)	Mechanismuszuordnung aufheben	6-122
	PRIORITY	(Priority)	Programmplatzpriorität ändern	6-121
	RESET ERR	(Reset Error)	Fehler zurücksetzen (darf nicht bei der Initialisierung verwendet werden)	6-124
Andere Befehle	CHRSRCH	(Character Search)	Zeichenkette in einer Zeichenketten-Feldvariablen suchen	6-16
	GET POS	(Get Position)	Reserviert	—

Tab. 6-2: Einteilung der Befehle in anwendungsspezifische Gruppen (3)

6.3 Detaillierte Befehlsbeschreibung

In diesem Abschnitt finden Sie eine detaillierte Beschreibung sowie Programmbeispiele zur Anwendung der Befehle.

6.3.1 ACCEL (Accelerate)

Funktion: Beschleunigung und Abbremsung einstellen

Legt den Wert für die Beschleunigung und Abbremsung in Prozent fest. Der Wert ist auch während der optimalen Beschleunigung/Abbremsung wirksam.

Ist die optimale Beschleunigung/Abbremsung über die Anweisung OADL aktiviert, ergibt sich die Beschleunigungs-/Bremszeit unter Berücksichtigung des Wertes, der in der Variablen M_SETADL festgelegt ist.

Eingabeformat

```
ACCEL □ [<Beschleunigung>] [, <Abbremsung>]
```

Ab Software-Version G2:

```
ACCEL □ [<Beschleunigung>] [, <Abbremsung>],
        [<Beschleunigung bei Aufwärtsbewegung>],
        [<Abbremsung bei Aufwärtsbewegung>],
        [<Beschleunigung bei Abwärtsbewegung>],
        [<Abbremsung bei Abwärtsbewegung>]
```

<Beschleunigung/Abbremsung>

Legt den Prozentwert der Beschleunigung/Abbremsung in einem Einstellbereich von 1 bis 100 % vom Stillstand bis zur maximalen Geschwindigkeit fest
Die Werte können als Konstante oder Variable angegeben werden. Erfolgt keine Angabe, werden die Werte auf 100 [%] gesetzt.
100 % entsprechen dabei der maximalen Beschleunigung bzw. Abbremsung.

<Beschleunigung/Abbremsung bei Aufwärtsbewegung>

Legt die Beschleunigung/Abbremsung bei einer Aufwärtsbewegung mit Bogen-Interpolation (MVA-Befehl) fest
Die Werte können als Konstante oder Variable angegeben werden. Erfolgt keine Angabe, werden die Werte auf 100 [%] gesetzt.

<Beschleunigung/Abbremsung bei Abwärtsbewegung>

Legt die Beschleunigung/Abbremsung bei einer Abwärtsbewegung mit Bogen-Interpolation (MVA-Befehl) fest
Die Werte können als Konstante oder Variable angegeben werden. Erfolgt keine Angabe, werden die Werte auf 100 [%] gesetzt.

Programmbeispiel

10	ACCEL 50,100	Beschleunigung/Abbremsung für große Last einstellen (Bei einem Grundwert der Beschleunigungs-/Abbremszeit von 0,2 s ist eine Beschleunigungszeit von 0,4 s und eine Abbremszeit von 0,2 s wirksam.)
20	MOV P1	Position P1 mittels Gelenk-Interpolation anfahren
30	ACCEL 100,100	Beschleunigung/Abbremsung für Standardlast einstellen
40	MOV P2	Position P2 mittels Gelenk-Interpolation anfahren
50	DEF ARCH 1,10,10,25,25,1,0,0	Bogen definieren
60	ACCEL 100,100,20,20,20,20	Beschleunigung/Abbremsung für die Bogen-Interpolation für Auf- und Abwärtsbewegung auf 20 % reduzieren
70	MVA P3,1	Position P3 über Bogen 1 mittels Bogen-Interpolation anfahren

Erläuterung

- Die maximale Beschleunigung/Abbremsung ist vom verwendeten Robotertyp abhängig. Stellen Sie die Beschleunigung/Abbremsung als Prozentwert des Maximalwertes ein. Als Standardwerte sind die Werte 100, 100 eingestellt.
- Die Beschleunigungs- bzw. Abbremszeit berechnet sich aus:

$$\text{Zeit} = \frac{100 \%}{\text{Beschleunigung [\%]}} \times 0,2 \text{ s}$$
- Einstellungen größer 100 werden bei einigen Robotermodellen automatisch auf 100 gesetzt. Eine Einstellung größer 100 kann die Lebensdauer des Roboters verkürzen und die Wahrscheinlichkeit von Fehlern aufgrund von Geschwindigkeits- oder Lastüberschreitungen steigt. Vermeiden Sie daher Einstellungen größer 100.
- Die über diesen Befehl eingestellte Beschleunigung/Abbremsung wird beim Zurücksetzen des Programms und bei Ausführung der END-Anweisung auf die Standardwerte zurückgesetzt.
- Die Verfahrkurve für eine kontinuierliche, gleichmäßige Bewegung (CNT freigegeben) kann von der Verfahrkurve mit Beschleunigung abweichen. Die Größe der Abweichung ist abhängig vom Wert der eingestellten Beschleunigungszeit. Für eine gleichmäßige Bewegung mit einer konstanten Geschwindigkeit sollten die Beschleunigungs- und die Abbremszeit gleich sein. In der Grundeinstellung ist die CNT-Einstellung gesperrt.
- Die Einstellung des ACCEL-Befehls ist auch bei aktivierter optimaler Beschleunigung/Abbremsung (OADL ON) wirksam.

Steht in Beziehung zu folgenden Befehlen:

OADL, LOADSET

Steht in Beziehung zu folgenden Systemvariablen:

M_ACL/M_DACL/M_NACL/M_NDAACL/M_ACLSTS, M_SETADL

Steht in Beziehung zu folgenden Parametern:

JADL

6.3.2 ACT (Act)

Funktion: Interrupt freigeben/sperrern

Über diesen Befehl kann die Ausführung von Interrupt-Prozessen während des Betriebes freigegeben oder gesperrt werden.

Eingabeformat

ACT □ <Priorität> = <1/0>

<Priorität>	Gibt den Interrupt frei oder sperrt ihn $1 \leq \text{Priorität} \leq 8$ Legt die mit der Anweisung DEF ACT definierte Priorität des Interrupts fest Hinter dem ACT-Befehl muss ein Leerzeichen stehen. Die Schreibweise ACT1 wird als Anweisung zur Deklaration einer Variablen gewertet.
<1/0>	1 = freigeben, 0 = sperren

Programmbeispiel

Beim Einschalten des Eingangssignal 1s während der Verfahrbewegung von P1 nach P2, wird eine Warteschleife durchlaufen, bis das Signal wieder auf „0“ gesetzt wird.

10 DEF ACT 1,M_IN(1) = 1 GOSUB *INTR	Weist den Eingang 1 dem Interrupt 1 zu
20 MOV P1	Position P1 mittels Gelenk-Interpolation anfahren
30 ACT 1 = 1	Interrupt 1 freigeben
40 MOV P2	Position P2 mittels Gelenk-Interpolation anfahren
50 ACT 1 = 0	Interrupt 1 sperren
:	
100 *INTR	Ändert sich das Eingangssignal 1 auf EIN (1) während der Roboter sich von P1 nach P2 bewegt, wird die Warteschleife durchlaufen.
110 IF M_IN(1) = 1 GOTO 110	Durchläuft die Warteschleife bis M_IN(1) auf „0“ gesetzt wird.
120 RETURN 0	

Beim Einschalten des Eingangssignals 1 während der Verfahrbewegung von P1 nach P2, wird der Betrieb unterbrochen und das Ausgangssignal 10 ausgegeben.

10	DEF ACT 1,M_IN(1) = 1 GOSUB *INTR	Weist den Eingang 1 dem Interrupt 1 zu
20	MOV P1	Position P1 mittels Gelenk-Interpolation anfahren
30	ACT 1 = 1	Interrupt 1 freigeben
40	MOV P2	Position P2 mittels Gelenk-Interpolation anfahren
	:	
100	*INTR	Ändert sich das Eingangssignal 1 auf EIN (1) während der Roboter sich von P1 nach P2 bewegt, wird der Betrieb unterbrochen.
110	ACT 1 = 0	Interrupt 1 sperren
120	M_OUT(10) = 1	Ausgabe des Ausgangssignals 10
130	RETURN 1	Springt eine Zeile hinter die Zeile, in der der Interrupt aufgetreten ist

Erläuterung

- Beim Programmstart ist der Interrupt mit der Priorität 0 freigegeben. Wenn der Interrupt mit der Priorität 0 gesperrt ist, werden die Interrupts der Prioritäten 1 bis 8 nicht freigegeben, auch wenn sie auf freigegeben gesetzt sind.
- Die Interrupts der Prioritäten 1 bis 8 sind beim Programmstart gesperrt.
- Ein Interrupt kann nur ausgeführt werden, wenn folgende Bedingungen erfüllt sind:
 - Der Interrupt der Priorität 0 ist freigegeben.
 - Der Status der DEF ACT-Anweisung ist definiert worden.
 - Der Interrupt, der in der DEF ACT-Anweisung festgelegt wurde, ist durch die ACT-Anweisung freigegeben.
- Ein Rücksprung aus einer Interruptroutine kann entweder durch RETURN 0 oder RETURN 1 erfolgen. Sperren Sie den Interrupt, wenn der Rücksprung über RETURN 1 in die Zeile, die der Zeile mit dem Interruptaufruf folgt, erfolgte. Wird der Interrupt nicht gesperrt und die Interrupt-Bedingung ist erfüllt, erfolgt eine erneute Ausführung der Interrupt-Routine und die Zeile kann beim Rücksprung übersprungen werden.
- Auch wenn der Roboter sich in einer Interpolation befindet, wird ein mit DEF ACT definierter Interrupt ausgeführt.
- Während eines Interruptprozesses wird der entsprechende Interrupt auf gesperrt gesetzt.
- Ein Kommunikations-Interrupt hat eine höhere Priorität als ein mit DEF ACT definierter Interrupt.
- Die Reihenfolge der Prioritäten ist: COM > ACT > WTHIF (WTH) > Impulsausgang.

Steht in Beziehung zu folgenden Befehlen:

DEF ACT, RETURN

6.3.3 BASE (Base)

Funktion: Basis-Konvertierungsdaten

Dieser Befehl ermöglicht eine Verschiebung oder Drehung des Roboterkoordinatensystems. Dazu müssen die Basis-Transformationsdaten festgelegt werden. Beachten Sie, dass die Änderung der Basis-Transformationsdaten innerhalb eines Programms zu undefinierten Aktivitäten beim JOG-Betrieb u. Ä. führen kann.

Eingabeformat

```
BASE □ <Basis-Transformationsdaten>
```

<Basis-Transformationsdaten> Legt die Basis-Transformationsdaten in Form von Positionskonstanten oder -variablen fest

Programmbeispiel

10	BASE (50,100,0,0,0,90)	Eingabe der Basis-Transformationsdaten als Konstante
20	MVS P1	Position P1 mittels Linear-Interpolation anfahren
30	BASE P2	Eingabe der Basis-Transformationsdaten als Variable
40	MVS P1	Position P1 mittels Linear-Interpolation anfahren
50	BASE P_NBASE	Zurücksetzen der Basis-Transformationsdaten auf den Standardwert

Erläuterung

- Die X-, Y- und Z-Koordinaten geben die parallele Verschiebung des Basiskoordinatensystems in Bezug auf das Weltkoordinatensystem an. Die Basis-Transformationsdaten können ausschließlich mit dem BASE-Befehl geändert werden. Die Komponenten A, B und C geben dabei die Drehwinkel des Basiskoordinatensystems in Bezug auf das Weltkoordinatensystem an.
 X = Paralleler Abstand zur X-Achse
 Y = Paralleler Abstand zur Y-Achse
 Z = Paralleler Abstand zur Z-Achse
 A = Drehwinkel um die X-Achse
 B = Drehwinkel um die Y-Achse
 C = Drehwinkel um die Z-Achse
 L1 = Weg der Zusatzachse 1
 L2 = Weg der Zusatzachse 2
- Aus Sicht des Koordinatenursprungs werden Winkel in Uhrzeigerichtung positiv gewertet.
- Die Werte der Stellungsmerker sind ohne Bedeutung.
- Die Änderungen des mit dem BASE-Befehl geänderten Basiskoordinatensystems werden im Parameter MEXBS gespeichert und bleiben auch nach Ausschalten der Spannungsversorgung erhalten.
- Der Standardwert ist P_NBASE = (0,0,0,0,0,0)(0,0). Er wird ohne Berücksichtigung des verwendeten Robotertyps für 6 Achsen und dreidimensional berechnet.
- Die für den BASE-Befehl zugelassen Achsen sind vom Robotermodell abhängig (siehe Abschn. 9.8 „Standard-Basiskoordinaten“).

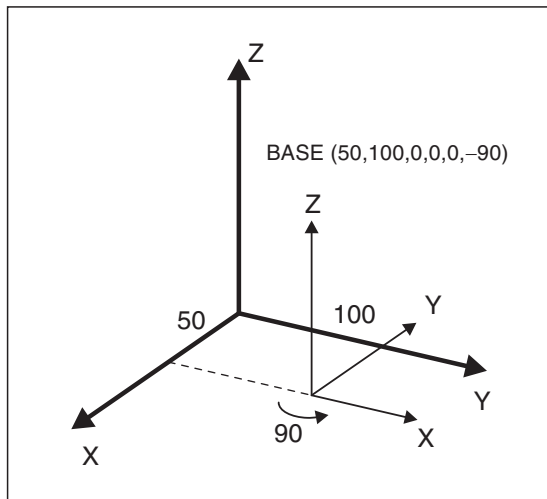


Abb. 6-1:
Basis-Konvertierungsdaten

R000877C

Steht in Beziehung zu folgenden Parametern:

MEXBS

Steht in Beziehung zu folgenden Systemvariablen:

P_BASE (aktuelle Basis-Konvertierungsdaten), P_NBASE (Grundeinstellung)

6.3.4 CALLP (Call P)

Funktion: Programm aufrufen

Führt das aufgerufene Programm aus (siehe auch GOSUB-Befehl für Unterprogrammaufrufe). Der Rücksprung ins Hauptprogramm erfolgt bei Ausführung der END-Anweisung oder der letzten Zeile des Unterprogramms.

Eingabeformat

```
CALLP □ "<Programmname>" [, <Parameter> [, <Parameter>] ...]
```

- <Programmname> Legt den Programmnamen als Zeichenkettenkonstante oder Zeichenkettenvariable fest
Weitere Hinweise zur Vergabe von Programmnamen finden Sie in Abschn. 4.2.1.
- <Parameter> Legt die Variablen fest, die beim Aufruf des Programmes übergeben werden
Es können maximal 16 Variablen übergeben werden.

Programmbeispiel

Übergabe der Variablen an das aufgerufene Programm

Hauptprogramm

10	M1 = 10	Weist M1 den Wert 10 zu
20	CALLP "10",M1,P1,P2	Aufruf des Programms 10 und Übergabe der Variablen M1, P1, P2
30	M1 = 1	Weist M1 den Wert 1 zu
40	CALLP "10",M1,P1,P2	Aufruf des Programms 10 und Übergabe der Variablen M1, P1, P2
	:	
100	CALLP "10",M2,P3,P4	Aufruf des Programms 10 und Übergabe der Variablen M2, P3, P4
	:	
150	END	Programmende

Programm "10"

10	FPRM M01,P01,P02	Definiert die Variablen M01, P01, P02
20	IF M01<> 0 THEN GOTO *LBL1	Sprung zur Marke LBL1, falls M01 ungleich 0 ist
30	MOV P01	Position mittels Gelenk-Interpolation anfahren
40	*LBL1	Legt die Sprungmarke „LBL1“ fest
50	MVS P02	Position mittels Linear-Interpolation anfahren
60	END	Rücksprung in Hauptprogramm

Bei Ausführung der Zeilen 20 und 40 des Hauptprogramms werden die Werte der Variablen M1, P1 und P2 in die Variablen M01, P01 und P02 des Unterprogramms übertragen. Bei Ausführung der Zeile 100 Hauptprogramms werden die Werte der Variablen M2, P3 und P4 in die Variablen M01, P01 und P02 des Unterprogramms übertragen.

Keine Übergabe der Variablen an das aufgerufene Programm

Hauptprogramm

10	MOV P1	Position mittels Gelenk-Interpolation anfahren
20	CALLP "20"	Aufruf des Programms 20
30	MOV P2	Position mittels Gelenk-Interpolation anfahren
40	CALLP "20"	Aufruf des Programms 20
50	END	Programmende

Programm "20"

10	MOV P1	Position P1 des Unterprogramms entspricht nicht der Position P1 des Hauptprogramms
20	MVS P002	Position mittels Linear-Interpolation anfahren
30	MOUT(17) = 1	Ausgang 17 auf „1“ setzen
40	END	Rücksprung in Hauptprogramm

Erläuterung

- Der Rücksprung ins Hauptprogramm aus einem über die CALLP-Anweisung aufgerufenen Programm (Unterprogramm) erfolgt mit Ausführung der END-Anweisung (analog zur RETURN-Anweisung beim Befehl GOSUB). Fehlt die END-Anweisung, erfolgt der Rücksprung ins Hauptprogramm nach Abarbeitung der letzten Unterprogrammzeile.
- Die Definition von Variablen zur Variablenübergabe erfolgt zu Beginn des Unterprogramms mit der FPRM-Anweisung.
- Weicht eine Variable in der CALLP-Anweisung von der im Programm definierten Variablen (FPRM) ab, erfolgt eine Fehlermeldung.
- Weicht die Anzahl der Variablen in der CALLP-Anweisung von der Anzahl der im Programm definierten Variablen ab, erfolgt eine Fehlermeldung.
- Wird das Programm zurückgesetzt, geht die Steuerung auf den Anfang des Hauptprogramms zurück.
- Das aufgerufene Programm hat keinen Einfluss auf die Anweisungen DEF ACT, DEF FN, DEF PLT und DIM im aufrufenden Programm. Sobald das aufgerufene Programm zurückspringt, werden sie wieder gültig.
- Die Geschwindigkeits-, Werkzeugdaten (TCP) und die OADL-Einstellung bleiben gültig, die über die Befehle ACCEL und SPD festgelegten Werte sind ungültig.
- Innerhalb eines Unterprogramms kann über die CALLP-Anweisung ein weiteres Unterprogramm aufgerufen werden. Es ist nicht möglich das aufrufende oder ein in einem anderen Programmplatz aktives Programm aufzurufen. Ein Programm kann sich nicht selber aufrufen.
- Mit der CALLP-Anweisung können bis zu 8 Programme von einem Programm aus aufgerufen werden.
- Variablenwerte können über Parameter vom aufrufenden Programm an das aufgerufene Programm übergeben werden. Die Ergebnisse, die im aufgerufenen Programm berechnet wurden, können nicht über die Parameter an das aufrufende Programm übergeben werden. Verwenden Sie externe Variablen, um die Werte zu übergeben.

Steht in Beziehung zu folgenden Befehlen:

FPRM

6.3.5 CHRSRCH (Character search)

Funktion: Zeichenkette suchen

Sucht eine Zeichenkette innerhalb einer Feldvariablen.

Eingabeformat

```
CHRSRCH □ <Zeichenketten-Feldvariable>, <Zeichenkette>,
          <Suchergebnis>
```

<Zeichenketten-Feldvariable>	Legt die zu durchsuchende Zeichenketten-Feldvariable fest
<Zeichenkette>	Legt die zu suchende Zeichenketten fest
<Suchergebnis>	Legt den Speicherort für die Nummer des gefundenen Feldelements fest

Programmbeispiel

10 DIM C1\$(10)	Deklariert eine Feldvariable
20 C1\$(1) = "ABCDEFGG"	Legt das erste Feldelement fest
30 C1\$(2) = "MELFA"	Legt das zweite Feldelement fest
40 C1\$(3) = "BCDF"	Legt das dritte Feldelement fest
50 C1\$(4) = "ABD"	Legt das vierte Feldelement fest
60 C1\$(5) = "XYZ"	Legt das fünfte Feldelement fest
70 C1\$(6) = "MELFA"	Legt das sechste Feldelement fest
80 C1\$(7) = "CDF"	Legt das siebte Feldelement fest
90 C1\$(8) = "ROBOT"	Legt das achte Feldelement fest
100 C1\$(9) = "FFF"	Legt das neunte Feldelement fest
110 C1\$(10) = "BCD"	Legt das zehnte Feldelement fest
120 CHRSRCH C1\$(1), "ROBOT", M1	Speichert die Elementnummer 8 in M1
130 CHRSRCH C1\$(1), "MELFA", M2	Speichert die Elementnummer 2 in M2

Erläuterung

- Die festgelegte Zeichenkette wird in der Zeichenketten-Feldvariablen gesucht. Nur wenn die gesamte Zeichenkette dem Feldelement entspricht, erfolgt die Speicherung der Elementnummer. Teile der Feldelemente werden nicht gefunden.
Die Ausführung der Anweisung CHRSRCH C1\$(1), "ROBO", M1 führt zu keinem Suchergebnis.
- Bleibt die Suche erfolglos, wird im Speicherort eine „0“ abgespeichert.
- Die Suche der Zeichenkette erfolgt beginnend mit dem ersten Element sequentiell. Die Elementnummer des ersten übereinstimmenden Feldelements wird gespeichert.
Bei der Suche über die Anweisung CHRSRCH C1\$(3), "MELFA", M2 im obigen Programmbeispiel wird M2 auf „2“ gesetzt, obwohl das Feldelement C1\$(6) dieselbe Zeichenkette enthält.
- Es können nur eindimensionale Zeichenketten-Feldvariablen durchsucht werden. Beim Durchsuchen mehrdimensionaler Feldvariablen erfolgt eine Fehlermeldung.

6.3.6 CLOSE (Close)

Funktion: Datei schließen

Schließt die festgelegte Datei (inklusive Kommunikationsleitungen).

Eingabeformat

```
CLOSE □ [[#]<Dateinummer>[, [[#]<Dateinummer> ...]
```

<Dateinummer> Legt die Dateinummer der zu schließenden Datei fest
Es dürfen nur numerische Konstanten verwendet werden. Fehlt die Dateinummer, werden alle geöffneten Dateien geschlossen.

Programmbeispiel

10 OPEN "COM1:" AS#1	„COM1:“ wird als Datei Nummer 1 geöffnet
20 PRINT #1,M1	Überträgt den Inhalt von M1 in Datei Nummer 1
100 INPUT #1,M2	Liest die Daten von Datei Nummer 1 in M2 ein
110 CLOSE #1	Datei Nummer 1 schließen
200 CLOSE	Alle geöffneten Dateien schließen

Erläuterung

- Die CLOSE-Anweisung schließt alle Dateien (inklusive der Kommunikationsleitungen), die mit der OPEN-Anweisung geöffnet wurden. Die Daten aus dem Pufferspeicher werden entfernt.

Pufferspeicher	Datenverarbeitung beim Schließen von Dateien
Empfangspuffer der Kommunikationsleitung	Die Daten des Pufferspeichers werden zerstört.
Sendepuffer der Kommunikationsleitung	Der Sendepuffer enthält keine Daten, da die Daten direkt nach Ausführung der PRINT-Anweisung verschickt wurden.
Pufferspeicher zum Laden von Dateien	Die Daten des Pufferspeichers werden zerstört.
Pufferspeicher zum Speichern von Dateien	Die Daten des Pufferspeichers werden in die Datei geschrieben. Danach wird die Datei geschlossen.

Tab. 6-3: Datenverarbeitung beim Schließen von Dateien

- Durch die Ausführung der END-Anweisung wird eine Datei ebenfalls geschlossen.
- Fehlt die Dateinummer, werden alle geöffneten Dateien geschlossen.

Steht in Beziehung zu folgenden Befehlen:

OPEN, PRINT, INPUT

6.3.7 CLR (Clear)

Funktion: Löschfunktion

Zurücksetzen der allgemeinen Ausgänge, der lokalen und externen numerischen Variablen.

Eingabeformat

CLR <Ausführung>

<Ausführung>

Die Ausführung kann als Konstante oder Variable vorgegeben werden.

- 0: Die allgemeinen Ausgangsbits, die lokalen und globalen Variablen werden zurückgesetzt.
- 1: Die allgemeinen Ausgänge werden auf ein voreingestelltes Bitmuster zurückgesetzt. Die Bitmuster werden über die Parameter ORST0 bis ORST224 vorgegeben (siehe auch Abschn. 9.15) (0: AUS, 1: EIN, *: HALTEN).
- 2: Alle lokalen numerischen Variablen und alle im Programm verwendeten numerischen Feldvariablen werden auf „0“ gesetzt.
- 3: Alle externen numerischen Variablen (externe Systemvariablen und benutzerdefinierte externe Variablen) und externen numerischen Feldvariablen werden auf „0“ gesetzt. Externe Positionsvariablen werden nicht zurückgesetzt.

Programmbeispiel

(1) Zurücksetzen der allgemeinen Ausgangssignale auf das vorgegebene Bitmuster

10 CLR 1 Zurücksetzen der allgemeinen Ausgangssignale

(2) Alle lokalen numerischen Variablen und alle im Programm verwendeten numerischen Feldvariablen werden auf „0“ gesetzt.

10 DIM MA(10) Deklariert die Feldvariable MA als eine Variable mit 10 Elementen

20 DEF INTE IVAL Deklariert IVAL als Namen einer numerischen Variablen

30 CLR 2 Setzt die Variablen MA(1) bis MA(10), IVAL und alle lokalen numerischen des Programms auf „0“

110 CLOSE #1 Datei Nummer 1 schließen

200 CLOSE Alle geöffneten Dateien schließen

(3) Alle externen numerischen Variablen und externen numerischen Feldvariablen werden auf „0“ gesetzt.

10 CLR 3 Setzt alle externen numerischen Variablen und Feldvariablen auf „0“

(4) Die Punkte (1) bis (3) werden gleichzeitig ausgeführt

10 CLR 0 Die allgemeinen Ausgangsbits, die lokalen und globalen Variablen werden zurückgesetzt.

Steht in Beziehung zu folgenden Parametern:

ORST0 bis ORST224

Steht in Beziehung zu folgenden Systemvariablen:

M_IN/M_INB/M_INW/M_OUT/M_OUTB/M_OUTW

6.3.8 CMP JNT (Compliance Joint)

Funktion: Achsenweichheit im Gelenk-Koordinatensystem aktivieren

Der Befehl legt fest, welche Achse im Gelenk-Koordinatensystem weich geschaltet werden soll. Er kann ausschließlich mit den Robotermodellen RH-5AH/10AH/15AH und RH-6SH/12SH verwendet werden.

Eingabeformat

```
CMP  JNT, <Achse>
```

<Achse> Legt über ein Bitmuster die Achse fest, für die die Weichheit eingestellt werden soll (1: aktiv, 0: inaktiv)
&B00000000: Achse 87654321

Programmbeispiel

10	MOV P1	Position P1 mittels Gelenk-Interpolation anfahren
20	CMPG 0.0,0.0,1.0,1.0, , , ,	Einstellung der Weichheit
30	CMP JNT, &B11	Achsenweichheit für J1 und J2 aktivieren
40	MOV P2	Position P2 mittels Gelenk-Interpolation anfahren
50	HOPEN 1	Öffnet Hand 1
60	MOV P1	Position P1 mittels Gelenk-Interpolation anfahren
70	CMP OFF	Achsenweichheit deaktivieren

Erläuterung

- Die Weichheit einer Roboterachse kann im Gelenk-Koordinatensystem festgelegt werden.
- Wird bei einem SCARA-Roboter beim senkrechten Einsetzen eines Bolzens in eine Bohrung die Weichheit für die Achsen J1 und J2 aktiviert, wird der Bolzen sanft in die Bohrung geführt (siehe Programmbeispiel oben).
- Der Wert der Weichheit entspricht einer Federkonstanten und wird über den Befehl CMPG eingestellt. Setzen Sie die Achsenweichheit bei einem SCARA-Roboter (z. B. RH-□AH) für die Achsen J1 und J2 auf „0.0“, um die Servowirkung bei der Steuerung des Roboters zu deaktivieren (servolose Steuerung, d. h. Federkonstante gleich 0). Eine servolose Steuerung der vertikalen Achsen ist nicht möglich, auch wenn der Wert auf „0.0“ eingestellt ist. Achten Sie darauf, dass die Achsen nicht außerhalb des zulässigen Bewegungsbereiches bewegt werden und dass die Positionsabweichung nicht zu groß wird.
- Die Einstellung der Weichheit bleibt auch bei einer Programmunterbrechung bis zur Ausführung des Befehls CMP OFF oder bis zum Aus- und Wiedereinschalten der Versorgungsspannung aktiviert.
- Ist die Weichheit aktiviert, kann der Roboter keine Position erreichen, die außerhalb des Verfahrwegbereiches der Gelenke liegt (nicht bei servolosem Betrieb).
- Ist die Abweichung der aktuellen Position von der Zielposition größer als 200 mm, unterbricht der Roboter die Verfahrbewegung und die Programmsteuerung springt in die nächste Zeile (nicht bei servolosem Betrieb).

- Eine gleichzeitige Verwendung der Befehle CMP JNT, POS und TOOL ist nicht möglich. Soll z. B. nach Ausführung des Befehls CMP JNT einer der Befehle CMP POS oder CMP TOOL ausgeführt werden, erfolgt eine Fehlermeldung. Heben Sie zuerst die Einstellung der Weichheit über den Befehl CMP OFF auf, bevor Sie einen der Befehle ausführen.
- Schaltet sich bei aktivierter Weichheitseinstellung die Servospannung ein, kann sich die Position des Roboters ändern.
- Bei aktivierter Weichheitseinstellung ist ein JOG-Betrieb möglich. Die Einstellung der Weichheit kann nicht über die Teaching Box deaktiviert werden. Die Ausführung der Anweisung muss innerhalb eines Programms oder im Menü zur Programmeditierung erfolgen.
- Heben Sie zur Änderung der Achsenauswahl die Weichheitseinstellung über den Befehl CMP OFF auf und führen Sie anschließend den Befehl CMP JNT erneut aus.

HINWEIS

Die Einstellung ist nur bei bestimmten Robotermodellen möglich. (Detaillierte Hinweise finden Sie im Technischen Handbuch des jeweiligen Roboters.)

**ACHTUNG:**

Führen Sie nach Aktivierung der Weichheitseinstellung über den Befehl CMP JNT den JOG-Betrieb im Gelenk-JOG-Modus aus. Die Wahl einer anderen JOG-Betriebsart kann aufgrund der unterschiedlichen Koordinatensysteme für die Weichheitseinstellung und den JOG-Betrieb zu Verfahrbewegungen in eine andere als die gewählte Richtung führen.

Beim Teachen von Positionen mit aktivierter Weichheitseinstellung muss zuerst die Servoversorgungsspannung ausgeschaltet werden. Ansonsten entspricht die geteachte Position nicht der aktuellen, sondern der ursprünglich geteachten Position.

Steht in Beziehung zu folgenden Systemvariablen:

M_CMPDST

Steht in Beziehung zu folgenden Befehlen:

CMP OFF, CMPG, CMP TOOL, CMP POS

6.3.9 CMP POS (Compliance Posture)

Funktion: Achsenweichheit im kartesischen Koordinatensystem aktivieren

Der Befehl legt fest, welche Achse im kartesischen Koordinatensystem weich geschaltet werden soll. Er kann nur mit den Robotermodellen RV-1A/2AJ, RV2A/3AJ, RV-4A/5AJ, RV-3SB/SJB, RV-6S/6SL/12S/12SL, RH-5AH/10AH/15AH und RH-6SH/12SH verwendet werden.

Eingabeformat

```
CMP  POS, <Achse>
```

<Achse> Legt über ein Bitmuster die Achse fest, für die die Weichheit eingestellt werden soll (1: aktiv, 0: inaktiv)
 &B00000000: L2, L1, C, B, A, Z, Y, X

Programmbeispiel

10	MOV P1	Position oberhalb der Einsetzposition mittels Gelenk-Interpolation anfahren
20	CMPG 0.5,0.5,1.0,0.5,0.5, , ,	Einstellung der Weichheit
30	CMP POS, &B011011	Achsenweichheit für die Achsen X, Y, A und B aktivieren
40	MVS P2	Einsetzposition mittels Linear-Interpolation anfahren
50	M_OUT(10) = 1	Spannfutter für die Werkstückaufnahme schließen
60	DLY 1.0	Wartezeit von 1 s bis das Spannfutter geschlossen ist
70	HOPEN 1	Öffnet Hand 1
80	MVS, -100	Position anfahren, die 100 mm in Werkzeuglängsrichtung von der aktuellen Position entfernt ist
90	CMP OFF	Achsenweichheit deaktivieren

Erläuterung

- Die Weichheit einer Roboterachse kann im kartesischen Koordinatensystem festgelegt werden.
- Wird beim senkrechten Einsetzen eines Bolzens in eine Bohrung die Weichheit für die Achsen X, Y, A und B aktiviert, wird der Bolzen sanft in die Bohrung geführt.
- Der Wert der Weichheit wird über den Befehl CMPG eingestellt.
- Die Einstellung der Weichheit bleibt auch bei einer Programmunterbrechung bis zur Ausführung des Befehls CMP OFF oder bis zum Aus- und Wiedereinschalten der Versorgungsspannung aktiviert.
- Ist die Weichheit aktiviert, kann der Roboter keine Position erreichen, die außerhalb des Verfahrwegbereiches der Gelenke liegt.
- Die Abweichung der aktuellen Position von der Zielposition kann aus dem Parameter M_CMPDIST ausgelesen werden. Der Parameter ermöglicht die Überprüfung des erfolgreichen Einsetzens eines Bolzens.
- Ist die Abweichung der aktuellen Position von der Zielposition größer als 200 mm, unterbricht der Roboter die Verfahrbewegung und die Programmsteuerung springt in die nächste Zeile.
- Eine gleichzeitige Verwendung der Befehle CMP JNT, POS und TOOL ist nicht möglich. Soll z. B. nach Ausführung des Befehls CMP JNT einer der Befehle CMP POS oder CMP TOOL ausgeführt werden, erfolgt eine Fehlermeldung. Heben Sie zuerst die Einstellung der Weichheit über den Befehl CMP OFF auf, bevor Sie einen der Befehle ausführen.
- Schaltet sich bei aktivierter Weichheitseinstellung die Servospannung ein, kann sich die Position des Roboters ändern.
- Bei aktivierter Weichheitseinstellung ist ein JOG-Betrieb möglich. Die Einstellung der Weichheit kann nicht über die Teaching Box deaktiviert werden. Die Ausführung der Anweisung muss innerhalb eines Programms oder im Menü zur Programmeditierung erfolgen.
- Heben Sie zur Änderung der Achsenauswahl die Weichheitseinstellung über den Befehl CMP OFF auf und führen Sie anschließend den Befehl CMP POS erneut aus.
- Der Betrieb des Roboters in der Nähe eines singulären Punktes kann zu einer Fehlermeldung oder zu einem Fehler der Steuerung führen. Vermeiden Sie den Betrieb des Roboters in der Nähe singulärer Punkte.
Tritt trotzdem einer dieser Fälle auf, deaktivieren Sie die Achsenweichheit bei ausgeschalteter Servoversorgungsspannung durch Ausführung des Befehls CMP OFF (oder schalten Sie die Versorgungsspannung aus und wieder ein), bewegen Sie den Roboter vom singulären Punkt weg und aktivieren Sie die Achsenweichheit erneut.

Beispiel ▾

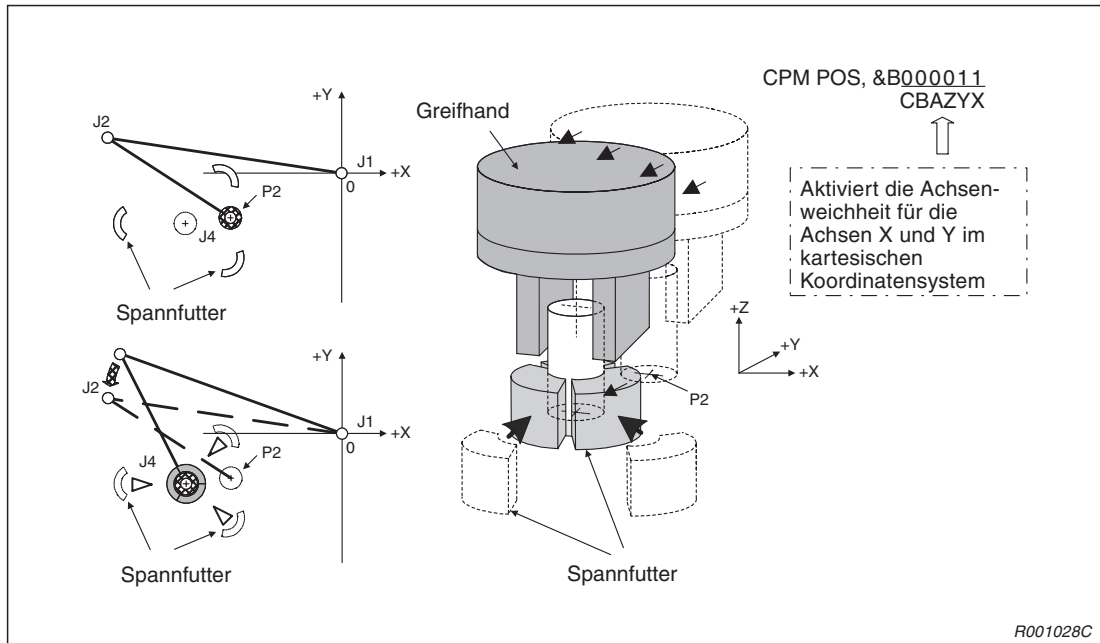


Abb. 6-2: Achsenweichheit im kartesischen Koordinatensystem beim Einsetzen eines Bolzens in ein Spannfutter



ACHTUNG:

Die Einstellung der Weichheit bleibt bis zur Ausführung des Befehls CMP OFF oder bis zum Aus- und Wiedereinschalten der Versorgungsspannung aktiviert. Besondere Vorsicht ist daher bei einem Programmwechsel und bei Ausführung des JOG-Betriebs geboten.

Führen Sie nach Aktivierung der Weichheitseinstellung über den Befehl CMP POS den JOG-Betrieb im XYZ-JOG-Modus aus. Die Wahl einer anderen JOG-Betriebsart kann aufgrund der unterschiedlichen Koordinatensysteme für die Weichheitseinstellung und den JOG-Betrieb zu Verfahrbewegungen in eine andere als die gewählte Richtung führen.

Beim Teachen von Positionen mit aktivierter Weichheitseinstellung muss zuerst die Servoversorgungsspannung ausgeschaltet werden. Ansonsten entspricht die geteachte Position nicht der aktuellen, sondern der ursprünglich geteachten Position.

Steht in Beziehung zu folgenden Systemvariablen:

M_CMPDST

Steht in Beziehung zu folgenden Befehlen:

CMP OFF, CMPG, CMP TOOL, CMP JNT

6.3.10 CMP TOOL (Compliance Tool)

Funktion: Achsenweichheit im Werkzeugkoordinatensystem aktivieren

Der Befehl legt fest, welche Achse im Werkzeugkoordinatensystem weich geschaltet werden soll. Er kann nur mit den Robotermodellen RV-1A/2AJ, RV2A/3AJ, RV-4A/5AJ, RV-3SB/SJB, RV-6S/6SL/12S/12SL, RH-5AH/10AH/15AH und RH-6SH/12SH verwendet werden.

Eingabeformat

```
CMP  TOOL, <Achse>
```

<Achse> Legt über ein Bitmuster die Achse fest, für die die Weichheit eingestellt werden soll
&B000000: C, B, A, Z, Y, X

Programmbeispiel

10	MOV P1	Position oberhalb der Einsetzposition mittels Gelenk-Interpolation anfahren
20	CMPG 0.5,0.5,1.0,0.5,0.5, , ,	Einstellung der Weichheit
30	CMP POS, &B011011	Achsenweichheit für die Achsen X, Y, A und B aktivieren
40	MVS P2	Einsetzposition mittels Linear-Interpolation anfahren
50	M_OUT(10) = 1	Spannfutter für die Werkstückaufnahme schließen
60	DLY 1.0	Wartezeit von 1 s bis das Spannfutter geschlossen ist
70	HOPEN 1	Öffnet Hand 1
80	MVS, -100	Position anfahren, die 100 mm in Werkzeuglängsrichtung von der aktuellen Position entfernt ist
90	CMP OFF	Achsenweichheit deaktivieren

Erläuterung

- Die Weichheit einer Roboterachse kann im Werkzeugkoordinatensystem festgelegt werden (siehe auch Abschn. 9.7).
- Wird beim senkrechten Einsetzen eines Bolzens in eine Bohrung die Weichheit für die Achsen X, Y, A und B aktiviert, wird der Bolzen sanft in die Bohrung geführt.
- Der Wert der Weichheit wird über den Befehl CMPG eingestellt.
- Die Einstellung der Weichheit bleibt auch bei einer Programmunterbrechung bis zur Ausführung des Befehls CMP OFF oder bis zum Aus- und Wiedereinschalten der Versorgungsspannung aktiviert.
- Ist die Weichheit aktiviert, kann der Roboter keine Position erreichen, die außerhalb des Verfahrwegbereiches der Gelenke liegt.
- Die Abweichung der aktuellen Position von der Zielposition kann aus dem Parameter M_CMPDIST ausgelesen werden. Der Parameter ermöglicht die Überprüfung des erfolgreichen Einsetzens eines Bolzens.
- Eine gleichzeitige Verwendung der Befehle CMP JNT, POS und TOOL ist nicht möglich. Soll z. B. nach Ausführung des Befehls CMP JNT einer der Befehle CMP POS oder CMP TOOL ausgeführt werden, erfolgt eine Fehlermeldung. Heben Sie zuerst die Einstellung der Weichheit über den Befehl CMP OFF auf, bevor Sie einen der Befehle ausführen.
- Schaltet sich bei aktivierter Weichheitseinstellung die Servospannung ein, kann sich die Position des Roboters ändern.
- Bei aktivierter Weichheitseinstellung ist ein JOG-Betrieb möglich. Die Einstellung der Weichheit kann nicht über die Teaching Box deaktiviert werden. Die Ausführung der Anweisung muss innerhalb eines Programms oder im Menü zur Programmeditierung erfolgen.
- Heben Sie zur Änderung der Achsauswahl die Weichheitseinstellung über den Befehl CMP OFF auf und führen Sie anschließend den Befehl CMP TOOL erneut aus.
- Bei fünfachsigem, vertikalen Knickarmrobotern (z. B. RV-5AJ) kann die Weichheit im Werkzeugkoordinatensystem nur für die Achsen X und Z eingestellt werden.
- Der Betrieb des Roboters in der Nähe eines singulären Punktes kann zu einer Fehlermeldung oder zu einem Fehler der Steuerung führen. Vermeiden Sie den Betrieb des Roboters in der Nähe singulärer Punkte.
Tritt trotzdem einer dieser Fälle auf, deaktivieren Sie die Achsenweichheit bei ausgeschalteter Servoversorgungsspannung durch Ausführung des Befehls CMP OFF (oder schalten Sie die Versorgungsspannung aus und wieder ein), bewegen Sie den Roboter vom singulären Punkt weg und aktivieren Sie die Achsenweichheit erneut.

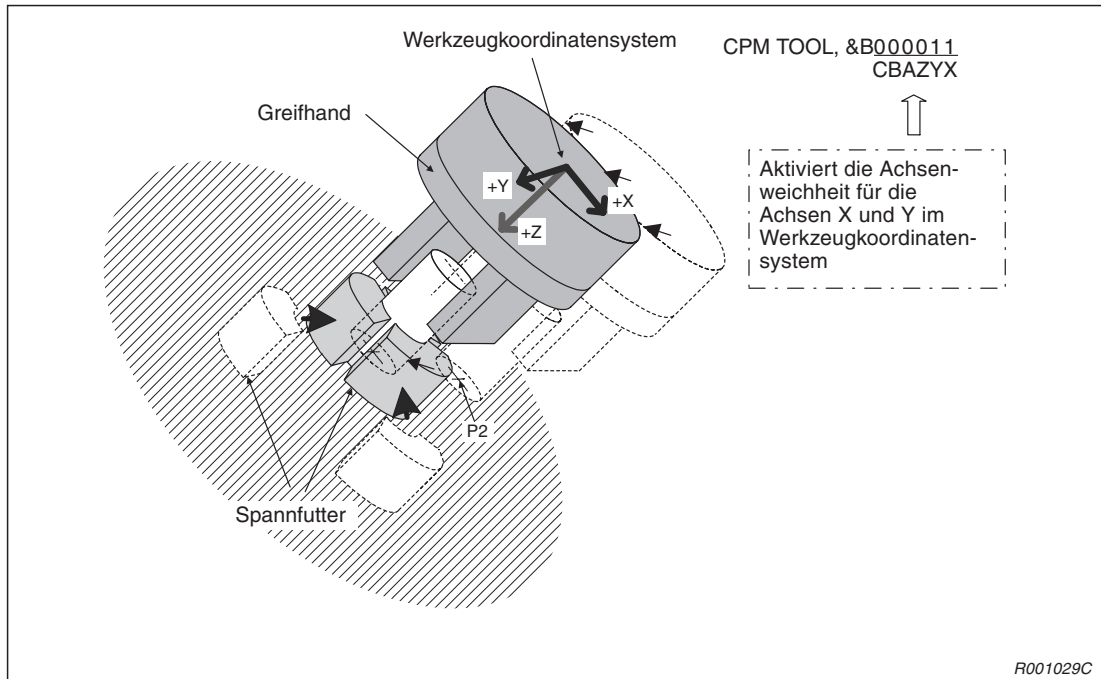
Beispiel ▾

Abb. 6-3: Achsenweichheit im Werkzeugkoordinatensystem beim Einsetzen eines Bolzens in ein Spannfutter

**ACHTUNG:**

Die Einstellung der Weichheit bleibt bis zur Ausführung des Befehls **CMP OFF** oder bis zum Aus- und Wiedereinschalten der Versorgungsspannung aktiviert. Besondere Vorsicht ist daher bei einem Programmwechsel und bei Ausführung des **JOG-Betriebs** geboten.

Führen Sie nach Aktivierung der Weichheitseinstellung über den Befehl **CMP TOOL** den **JOG-Betrieb** im Werkzeug-JOG-Modus aus. Die Wahl einer anderen JOG-Betriebsart kann aufgrund der unterschiedlichen Koordinatensysteme für die Weichheitseinstellung und den JOG-Betrieb zu Verfahrbewegungen in eine andere als die gewählte Richtung führen.

Beim Teachen von Positionen mit aktivierter Weichheitseinstellung muss zuerst die Servoversorgungsspannung ausgeschaltet werden. Ansonsten entspricht die geteachte Position nicht der aktuellen, sondern der ursprünglich geteachten Position.

Steht in Beziehung zu folgenden Systemvariablen:

M_CMPDST

Steht in Beziehung zu folgenden Befehlen:

CMP OFF, CMPG, CMP POS, CMP JNT

6.3.11 CMP OFF (Compliance OFF)

Funktion: Achsenweichheit deaktivieren

Der Befehl deaktiviert die eingestellte Weichheit. Er kann nur mit den Robotermodellen RV-1A/2AJ, RV2A/3AJ, RV-4A/5AJ, RV-3SB/SJB, RV-6S/6SL/12S/12SL, RH-5AH/10AH/15AH und RH-6SH/12SH verwendet werden.

Eingabeformat

CMP <input type="checkbox"/> OFF

Programmbeispiel

10	MOV P1	Position oberhalb der Einsetzposition mittels Gelenk-Interpolation anfahren
20	CMPG 0.5,0.5,1.0,0.5,0.5, , ,	Einstellung der Weichheit
30	CMP POS, &B011011	Achsenweichheit für die Achsen X, Y, A und B aktivieren
40	MVS P2	Einsetzposition mittels Linear-Interpolation anfahren
50	M_OUT(10) = 1	Spannfutter für die Werkstückaufnahme schließen
60	DLY 1.0	Wartezeit von 1 s bis das Spannfutter geschlossen ist
70	HOPEN 1	Öffnet Hand 1
80	MVS, -100	Position anfahren, die 100 mm in Werkzeuglängsrichtung von der aktuellen Position entfernt ist
90	CMP OFF	Achsenweichheit deaktivieren

Erläuterung

- Die über die Befehle CMP TOOL, CMP POS oder CMP JNT eingestellte Weichheit wird deaktiviert.
- Deaktivieren Sie die Weichheitseinstellung im JOG-Betrieb, indem Sie die Anweisung in einem Programm oder im Menü zur Programmeditierung ausführen.

Steht in Beziehung zu folgenden Befehlen:

CMPG, CMP TOOL, CMP POS, CMP JNT

6.3.12 CMPG (Compliance Gain)

Funktion: Achsenweichheit einstellen

Der Befehl legt den Wert der Weichheit einer Achse fest. Er kann nur mit den Robotermodellen RV-1A/2AJ, RV2A/3AJ, RV-4A/5AJ, RV-3SB/SJB, RV-6S/6SL/12S/12SL, RH-5AH/10AH/15AH und RH-6SH/12SH verwendet werden.

Eingabeformat

Für die Befehle CMP POS und CMP TOOL

```
CMPG □    [<Einstellung X-Achse>], [<Einstellung Y-Achse>]
          [<Einstellung Z-Achse>], [<Einstellung A-Achse>]
          [<Einstellung B-Achse>], [<Einstellung C-Achse>]
          [<Einstellung L1-Achse>], [<Einstellung L2-Achse>]
```

Für den Befehl CMP JNT

```
CMPG □    [<Einstellung J1-Achse>], [<Einstellung J2-Achse>]
          [<Einstellung J3-Achse>], [<Einstellung J4-Achse>]
          [<Einstellung J5-Achse>], [<Einstellung J6-Achse>]
          [<Einstellung J7-Achse>], [<Einstellung J8-Achse>]
```

<Einstellung X- bis L2-Achse>

<Einstellung J1- bis J8-Achse>

Der Grad der Weichheit für jede Achse kann als Konstante eingestellt werden.

Die Einstellung „1“ bedeutet Normalbetrieb, die Einstellung „0.2“ entspricht der größten Weichheit.

Bei fehlender Angabe wird der aktuelle Wert verwendet.

Erläuterung

- Der Grad der Weichheit kann für jede Achse festgelegt werden.
- Die Aktivierung der Weichheit erfolgt über die Befehle CMP JNT, CMP POS oder CMP TOOL.
- Über den Befehl CMPG wird eine Kraft ähnlich einer Federkraft eingestellt, deren Größe von der Abweichung zwischen Zielposition und der aktuellen Position abhängig ist. Die Federkonstante wird über den CMPG-Befehl eingestellt.
- Die Abweichung der aktuellen Position von der Zielposition kann aus dem Parameter M_CMPDIST ausgelesen werden. Der Parameter ermöglicht die Überprüfung des erfolgreichen Einsetzens eines Bolzens.
- Bei kleinen Einstellwerten kann die Position bei Aktivierung der Weichheit durch den Befehl CMP JNT, CMP POS oder den Befehl CMP TOOL aufgrund des Eigengewichts des Roboterarms nach unten sinken. Stellen Sie die Achsenweichheit schrittweise ein.
- Bei aktivierter Achsenweichheit kann der Grad der Weichheit auch während des Betriebs verändert werden.
- Bei einer Weichheitseinstellung kleiner als 0,2 wird der Wert automatisch auf 0,2 gesetzt. (Für die Roboter RV-1A/2AJ ist eine Einstellung auf 0,1 und für die Roboter RH-5AH/10AH/15AH eine Einstellung auf 0,0 möglich.)

6.3.13 CNT (Continuous)

Funktion: Roboterbewegung steuern

Der Befehl legt die Steuerung für eine kontinuierliche und gleichmäßige Bewegung fest und dient der Verkürzung der Zykluszeiten.

Eingabeformat

```
CNT □ <Freigeben/Sperren>[, <Numerischer Wert 1>]
      [, <Numerischer Wert 2>]
```

- <Freigeben/Sperren> Legt den Anfang und das Ende einer kontinuierlichen und gleichmäßigen oder einer beschleunigten und abgebremsten Roboterbewegung fest
1 = freigeben, 0 = gesperrt
- <Numerischer Wert 1> Legt den Anfangspunktabstand der kontinuierlichen Bewegung in mm fest
Der Standardwert ist der Startpunkt der Beschleunigung bzw. Abbremsung.
- <Numerischer Wert 2> Legt den Endpunktabstand der kontinuierlichen Bewegung in mm fest
Der Standardwert ist der Startpunkt der Beschleunigung bzw. Abbremsung.

Programmbeispiel

- | | | |
|-----|-----------------|-------------------------------------------------------------------------------------------------------------------------------------|
| 10 | CNT 0 | Sperren der CNT-Einstellung |
| 20 | MVS P1 | Position P1 mittels Linear-Interpolation und Beschleunigung/Verzögerung anfahren |
| 30 | CNT 1 | Freigeben der CNT-Einstellung |
| 40 | MVS P2 | Position P2 mittels Linear-Interpolation und kontinuierlicher gleichmäßiger Geschwindigkeit anfahren |
| 50 | CNT 1, 100, 200 | Anfangspunktabstand der kontinuierlichen Bewegung auf 100 mm und Endpunktabstand der kontinuierlichen Bewegung auf 200 mm festlegen |
| 60 | MVS P3 | Position P3 mittels Linear-Interpolation und kontinuierlicher gleichmäßiger Geschwindigkeit anfahren |
| 70 | CNT 1, 300 | Anfangspunktabstand der kontinuierlichen Bewegung auf 300 mm und Endpunktabstand der kontinuierlichen Bewegung auf 300 mm festlegen |
| 80 | MOV P4 | Position P4 mittels Gelenk-Interpolation und kontinuierlicher gleichmäßiger Geschwindigkeit anfahren |
| 90 | CNT 0 | Sperren der CNT-Einstellung |
| 100 | MOV P5 | Position P5 mittels Gelenk-Interpolation und Beschleunigung/Verzögerung anfahren |

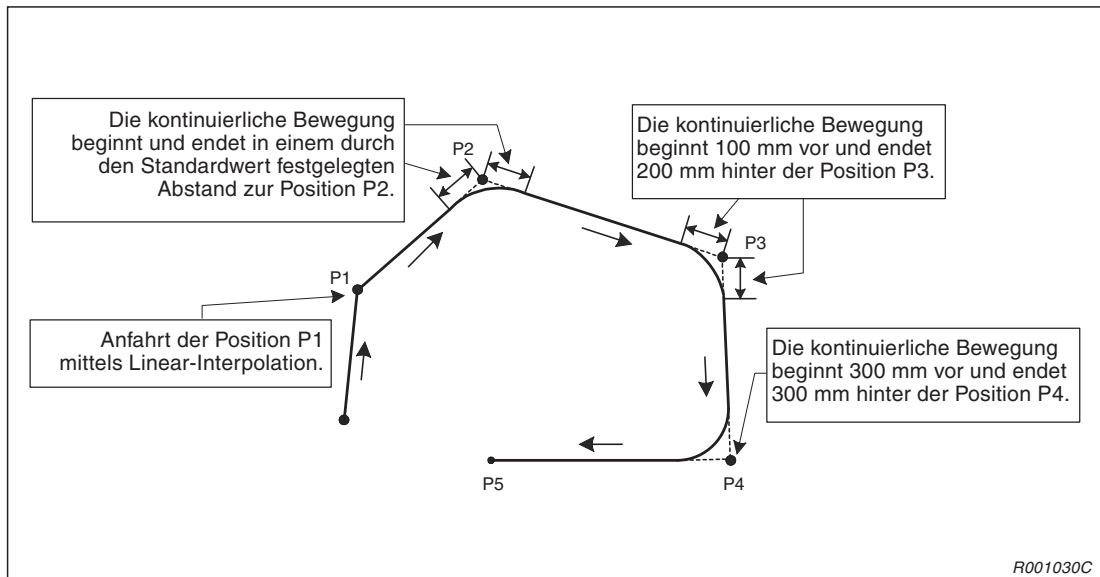


Abb. 6-4: Verfahrweg für die verschiedenen CNT-Einstellungen

Erläuterung

- Die Positionen zwischen den Befehlen CNT 1 und CNT 0 (Zeile 40 bis 80) werden mit kontinuierlicher gleichmäßiger Geschwindigkeit angefahren.
- Standardmäßig ist die CNT-Einstellung gesperrt.
- Der Übergang zwischen den Verfahrwegsegmenten beginnt bei fehlender Angabe der numerischen Werte 1 und 2 am Startpunkt der Beschleunigung bzw. Abbremsung.
- Wie folgende Abbildung zeigt, wird die Verfahrbewegung bei deaktivierter CNT-Einstellung vor der Zielposition bis zum Stopp abgebremsst. Bei der Anfahrt der nächsten Position wird der Roboter erneut beschleunigt. Bei aktivierter CNT-Einstellung erfolgt vor der Zielposition nur eine geringe Abbremsung des Roboters. Die Beschleunigung zur Anfahrt der nächsten Position setzt an diesem Punkt ein. Daher verläuft der Verfahrweg nicht durch jede Position, sondern durch die Punkte, die durch den Anfangs- und Endpunktabstand festgelegt wurden.

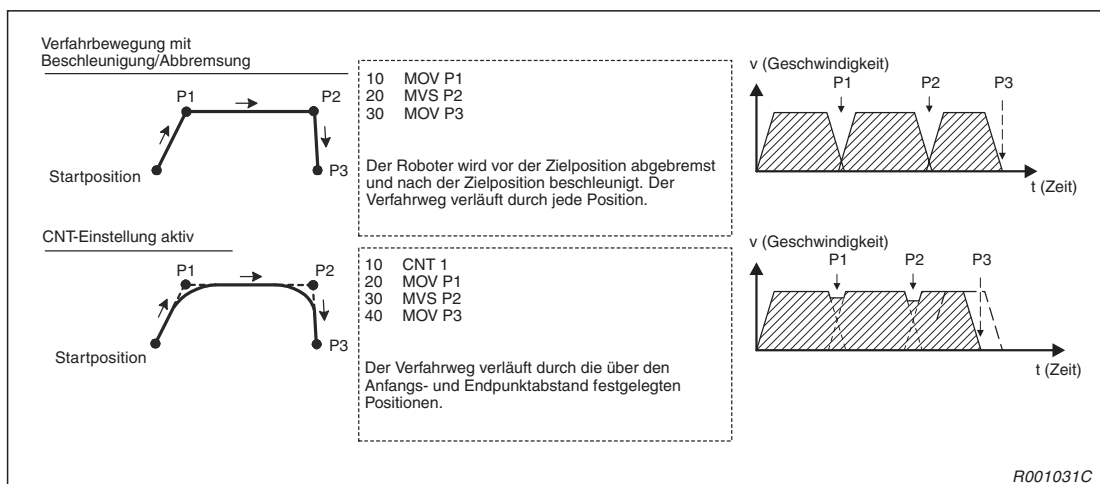


Abb. 6-5: Einfluss der CNT-Einstellung auf die Beschleunigung/Abbremsung

- Die Anfangs- und Endpunktabstände definieren die Entfernungen der Punkte von der Zielposition, durch die der Verfahrenweg verläuft. Der Übergang zwischen den Verfahrenwegsegmenten beginnt bei fehlender Angabe der numerischen Werte 1 und 2 am Startpunkt der Beschleunigung bzw. Abbremsung. Die Zielposition wird dabei nicht durchlaufen, die Zykluszeit jedoch minimiert. Um die Abstände der Start- und Endposition der kontinuierlichen Bewegung zur Zielposition zu verkleinern, sind die numerischen Werte 1 und 2 zu verringern.

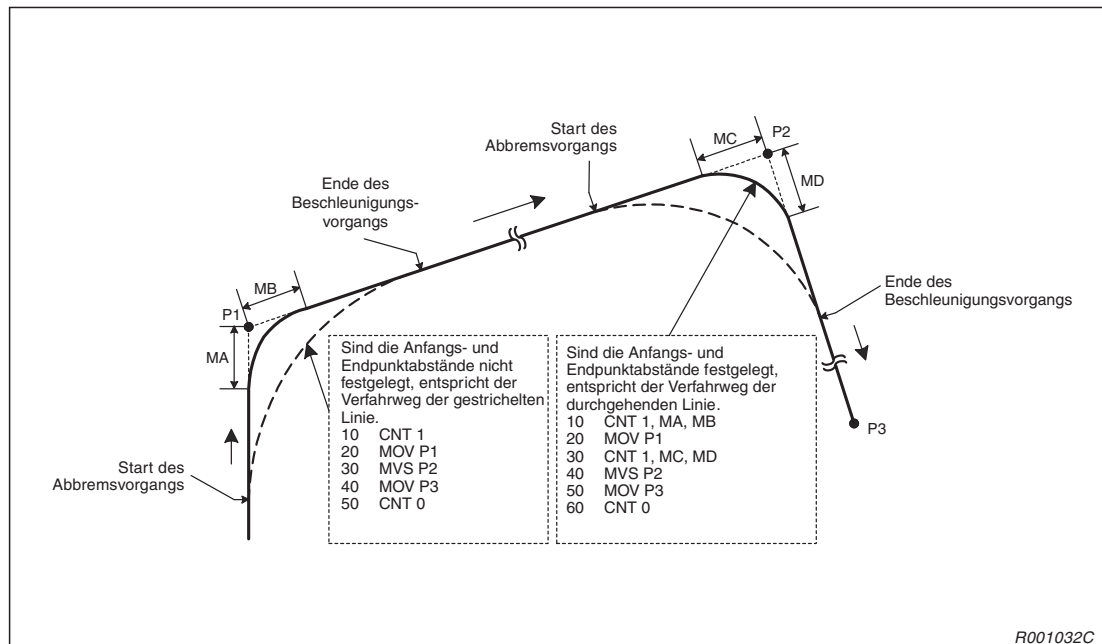


Abb. 6-6: Festlegung von Anfangs- und Endpunktabstand

- Fehlt die Angabe für den numerischen Wert 2, wird er auf den gleichen Wert wie der numerische Wert 1 gesetzt.
- Bei freigegebener CNT-Einstellung ist die FINE-Einstellung gesperrt.
- Bei kleinen numerischen Werten kann das Durchlaufen des Verfahrensweges länger dauern als bei deaktivierter CNT-Einstellung.

6.3.14 COLCHK (Col Check)

Funktion: Kollisionsüberwachung aktivieren

Der Befehl aktiviert die Kollisionsüberwachung. Bei aktivierter Kollisionsüberwachung führt ein Zusammenstoß des Roboters mit umliegenden Einrichtungen zum sofortigen Stopp des Roboters. Die Funktion dient zur Begrenzung von Schäden, die bei einem Zusammenstoß entstehen können. Einen vollständigen Schutz vor Schäden und Verformungen an Komponenten des Roboters oder der umliegenden Einrichtungen bietet die Funktion jedoch nicht. Der Befehl COLCHK ist ab Software-Version J2 verfügbar und kann nur mit den Robotermodellen RV-3SB/3SJB, RV-6S/6SL/12S/12SL und RH-6SH/12SH verwendet werden.

Eingabeformat

COLCHK ON [, NOERR] / OFF

ON	Aktiviert die Kollisionsüberwachung Bei einem Zusammenstoß erfolgt ein sofortiger Stopp des Roboters, die Fehlernummer 1010 wird ausgegeben und die Servos werden abgeschaltet. 1 = freigegeben, 0 = gesperrt
OFF	Deaktiviert die Kollisionsüberwachung
NOERR	Auch wenn die Kollisionsüberwachung anspricht, wird kein Fehler ausgegeben. Keine Angabe führt zur Ausgabe eines Fehlers.

Programmbeispiel 1

Bei einem Zusammenstoß erfolgt die Ausgabe einer Fehlermeldung

10	COLLVL 80,80,80,80,80,80,,	Ansprechschwelle für die Kollisionsüberwachung festlegen
20	COLCHK ON	Kollisionsüberwachung aktivieren
30	MOV P1	Position P1 mittels Gelenk-Interpolation anfahren
40	MOV P2	Position P2 mittels Gelenk-Interpolation anfahren
50	DLY 0.2	Der Abschluss des Positioniervorgangs erfolgt über die Timer-Einstellung. (Es kann auch die Anweisung FINE verwendet werden.)
60	COLCHK OFF	Kollisionsüberwachung deaktivieren
70	MOV P3	Position P3 mittels Gelenk-Interpolation anfahren

Programmbeispiel 2

Bei einem Zusammenstoß erfolgt der Aufruf eines Interrupt-Prozesses

10	DEF ACT 1,M_COLSTS(1) = 1 GOTO *HOME,S	Definiert bei einem Zusammenstoß einen Unterprogrammssprung zur Marke HOME
20	ACT 1 = 1	Interrupt 1 freigeben
30	COLCHK ON,NOERR	Kollisionüberwachung ohne Fehlerausgabe aktivieren
40	MOV P1	
50	MOV P2	Erfolgt während der Ausführung der Zeilen 40 bis 70 ein Zusammenstoß, wird der Interrupt-Prozess ausgeführt
60	MOV P3	
70	MOV P4	
80	ACT 1 = 0	Interrupt 1 sperren
	:	
1000	*HOME	Interrupt-Prozess bei einem Zusammenstoß
1010	COLCHK OFF	Kollisionsüberwachung deaktivieren
1020	SERVO ON	Schaltet die Servospannung ein
1030	PESC = P_COLDIR(1)*(-2)	Abstand der Ausweichposition festlegen
1040	PDST = P_FBC(1) + PESC	Ausweichposition festlegen
1050	MVS PDST	Ausweichposition mittels Linear-Interpolation anfahren
1060	ERROR 9100	Benutzerdefinierten, leichten Fehler ausgeben

Erläuterung

- Die Auslösung der Kollisionsüberwachung erfolgt über eine Erfassung des Drehmomentes während der Ausführung eines Bewegungsbefehls. Als Zusammenstoß wird das Überschreiten eines bestimmten Differenzbetrages zwischen dem Drehmoment-Ist- und dem Drehmoment-Sollwert gewertet. Der Roboter wird sofort gestoppt.

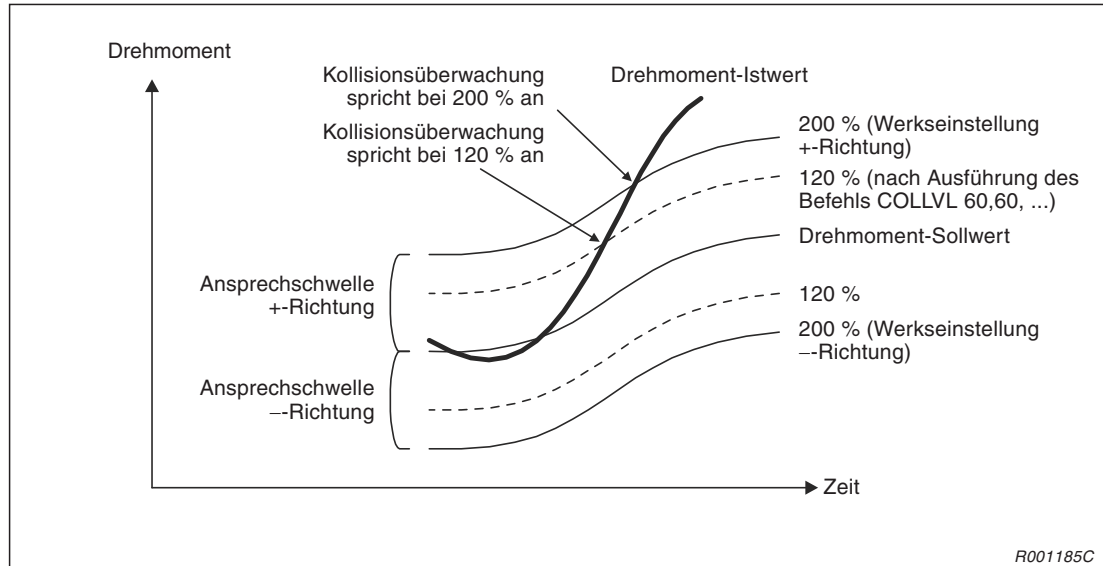


Abb. 6-7: Ansprechschwelle der Kollisionsüberwachung

- Direkt nach dem Einschalten der Spannungsversorgung ist die Kollisionsüberwachung deaktiviert. Geben Sie die Kollisionsüberwachung über den Parameter COL frei, bevor Sie sie verwenden.
- Die Ansprechschwelle der Kollisionsüberwachung kann über die Anweisung COLLVL eingestellt werden. Der Initialisierungswert ist im Parameter COLLVL festgelegt.
- Nach Aktivierung der Kollisionsüberwachung bleibt die Funktion solange aktiviert, bis die Anweisung COLCHK oder END ausgeführt, das Programm zurückgesetzt oder die Spannungsversorgung ausgeschaltet wird.
- Auch wenn die Kollisionsüberwachung über die Anweisung wieder deaktiviert ist, bleibt die mit dem Befehl COLLVL eingestellte Ansprechschwelle erhalten.
- Bei einer kontinuierlichen Programmausführung wird die vorhergehende Einstellung der Kollisionsüberwachung auch dann übernommen, wenn die Spannungsversorgung aus- und wieder eingeschaltet wird.
- Ist ein von der Roboterstatusvariablen M_COLSTS abhängiger Interrupt (Interrupt-Bedingung: $M_COLSTS(*) = 1$ mit Roboter Nummer = *) bei Definition des NOERR-Modus nicht freigegebenen (siehe Programmbeispiel 2), erfolgt die Ausgabe der Fehlermeldung 3950. Die Fehlermeldung 3960 wird ausgegeben, wenn der Interrupt bei aktiviertem NOERR-Modus geperrt wird.
- Ein Zusammenstoß im NOERR-Modus führt zu einer Abschaltung der Servoversorgung und der Roboter stoppt. Es erfolgt keine Fehlermeldung und der Betrieb kann fortgesetzt werden. Der Zusammenstoß wird in einer LOG-Datei registriert, wenn nicht gleichzeitig andere Fehler auftreten.
- Wird eine der Anweisungen COLCHK ON oder COLCHK ON,NOERR auf einen Roboter angewendet, der nicht über die Funktion der Kollisionsüberwachung verfügt, erfolgt die Ausgabe des leichten Fehlers 3970. Bei der Anweisung COLCHK OFF erfolgt weder eine Fehlermeldung noch eine Verarbeitung der Anweisung.

- Eine Verwendung der Kollisionsüberwachung ist nicht möglich, wenn die Achsenweichheit über die Anweisung CMP aktiviert oder eine Drehmomentgrenze über die Anweisung TORQ definiert ist. Bei Aktivierung der Kollisionsüberwachung erfolgt in diesen Fällen die Ausgabe der Fehlermeldung 3940. Ist die Kollisionsüberwachung aktiviert, erfolgt bei einer Ausführung der Anweisung CMP oder TORQ die Ausgabe der Fehlermeldung 3930.
- Wird die Anweisung COLCHK OFF direkt nach einem Verfahrbefehl ausgeführt, ist die Kollisionsüberwachung kurz vor dem Erreichen der Zielposition eventuell nicht mehr wirksam. Fügen Sie daher zwischen dem Verfahrbefehl und der Anweisung COLCHK OFF eine DLY- oder FINE- Anweisung ein, um die Vefahrbewegung mit aktiver Kollisionüberwachung abzuschließen (siehe Programmbeispiel 1).
- Eine falsche Einstellung der Parameter für die Hand- (HNDDATn) und Werkstückdaten (WRKDATn) kann zu einem ungewollten Ansprechen der Kollisionsüberwachung führen. Achten Sie daher auf eine korrekte Einstellung der Parameter.

Steht in Beziehung zu folgenden Befehlen:

COLLVL

Steht in Beziehung zu folgenden Systemvariablen:

M_COLSTS, J_COLMXL, P_COLDIR

Steht in Beziehung zu folgenden Parametern:

COL, COLLVL, COLLVLJG

6.3.15 COLLVL (Col Level)

Funktion: Ansprechschwelle der Kollisionsüberwachung

Der Befehl legt die Ansprechschwelle der Kollisionsüberwachung fest. Die Kollisionsüberwachung ist ab Software-Version J2 verfügbar und kann nur mit den Robotermodellen RV-3SB/3SJB, RV-6S/6SL/12S/12SL und RH-6SH/12SH verwendet werden.

Eingabeformat

```
COLLVL [<J1-Achse>], [<J2-Achse>], [<J3-Achse>], [<J4-Achse>],
        [<J5-Achse>], [<J6-Achse>], [<J7-Achse>], [<J8-Achse>]
```

<J1- bis J8-Achse> Legt die Ansprechschwelle für jede Achse in einem Bereich zwischen 1 und 500 % fest. Erfolgt keine Angabe, ist der zuletzt eingestellte Wert wirksam.
Für die Achsen J7 und J8 ist zur Zeit keine Einstellung möglich.
Der Initialisierungswert ist im Parameter COLLVL festgelegt.

Programmbeispiel

10	COLLVL 80,80,80,80,80,80,,	Ansprechschwelle für die Kollisionsüberwachung festlegen
20	COLCHK ON	Kollisionsüberwachung aktivieren
30	MOV P1	Position P1 mittels Gelenk-Interpolation anfahren
40	COLLVL ,50,50,,,,,	Ansprechschwelle der Achsen J2 und J3 für die Kollisionsüberwachung ändern
50	MOV P2	Position P2 mittels Gelenk-Interpolation anfahren
60	DLY 0.2	Nach Erreichen der Position P2 wird die Kollisionsüberwachung aktiviert.
70	COLCHK OFF	Kollisionsüberwachung deaktivieren
80	MOV P3	Position P3 mittels Gelenk-Interpolation anfahren

Erläuterung

- Stellen Sie zum Programmbetrieb die Ansprechschwelle für die Kollisionsüberwachung auf einen für die jeweilige Achse zulässigen Wert ein.
- Nach Einschalten der Spannungsversorgung sind die im Parameter COLLVL eingestellten Werte wirksam.
- Standardmäßig sind alle Achsen im Parameter COLLVL auf 200 % eingestellt.
- Je größer der eingestellte Wert, desto höher die Ansprechschwelle und desto niedriger somit die Ansprechempfindlichkeit.
- Stellen Sie die Ansprechschwelle nicht zu niedrig ein, da dies zu einem ungewollten Ansprechen der Kollisionsüberwachung führen kann. In Abhängigkeit der Geschwindigkeit und der Roboterstellung ist eine ungewollte Auslösung auch bei Verwendung des Standardwertes möglich. Erhöhen Sie in diesem Falle die Ansprechschwelle.
- Eine falsche Einstellung der Parameter für die Hand- (HNDDATn) und Werkstückdaten (WRKDATn) kann zu einem ungewollten Ansprechen der Kollisionsüberwachung führen. Achten Sie daher auf eine korrekte Einstellung der Parameter.
- Bei einer kontinuierlichen Programmausführung wird die vorhergehende Einstellung der Kollisionsüberwachung auch dann übernommen, wenn die Spannungsversorgung aus- und wieder eingeschaltet wird.
- Beim Zurücksetzen des Programmes oder bei Ausführung der Anweisung END wird die eingestellte Ansprechschwelle auf den im Parameter COLLVL festgelegten Wert gesetzt.
- Es erfolgt keine Fehlermeldung, wenn die Anweisung COLLVL auf einen Roboter angewendet wird, der nicht über die Funktion der Kollisionsüberwachung verfügt.
- Zur Zeit kann die Kollisionüberwachung nicht auf die Achsen J7 und J8 angewendet werden. Eine spätere Nutzung ist jedoch vorgesehen.

Steht in Beziehung zu folgenden Befehlen:

COLCHK

Steht in Beziehung zu folgenden Systemvariablen:

M_COLSTS, J_COLMXL, P_COLDIR

Steht in Beziehung zu folgenden Parametern:

COL, COLLVL

6.3.16 COM OFF (Communication OFF)

Funktion: Kommunikations-Interrupt sperren

Sperrt die Interrupts von Kommunikationskanälen

Eingabeformat

```
COM [( <Nummer des Kommunikationskanals> )]  OFF
```

<Nummer des Kommunikationskanals> Legt die Nummer des Kommunikationskanals fest (z. B. 1, 2 oder 3)

Programmbeispiel

Ein Programmbeispiel finden Sie unter dem Befehl ON COM GOSUB in Abschn. 6.3.54.

Erläuterung

- Nach Ausführung des COM OFF-Befehls bleibt der Interrupt auch bei anstehenden Nachrichten gesperrt.
- Informationen über Kommunikationskanäle sind unter dem Befehl OPEN in Abschn. 6.3.57 zu finden.

6.3.17 COM ON (Communication ON)

Funktion: Kommunikations-Interrupt freigeben

Gibt die Interrupts von Kommunikationskanälen frei

Eingabeformat

```
COM (<Nummer des Kommunikationskanals>)]  ON
```

<Nummer des Kommunikationskanals> Legt die Nummer des Kommunikationskanals fest (z. B. 1, 2 oder 3)

Programmbeispiel

Ein Programmbeispiel finden Sie unter dem Befehl ON COM GOSUB in Abschn. 6.3.54.

Erläuterung

- Informationen über Kommunikationskanäle sind unter dem Befehl OPEN in Abschn. 6.3.57 zu finden.

6.3.18 COM STOP (Communication STOP)

Funktion: Kommunikations-Interrupt stoppen

Stoppt die Interrupts von Kommunikationskanälen

Eingabeformat

```
COM [( <Nummer des Kommunikationskanals> )]  STOP
```

<Nummer des Kommunikationskanals> Legt die Nummer des Kommunikationskanals fest (z. B. 1, 2 oder 3)

Programmbeispiel

Ein Programmbeispiel finden Sie unter dem Befehl ON COM GOSUB in Abschn. 6.3.54.

Erläuterung

- Nach Ausführung des COM STOP-Befehls wird der Interrupt auch bei anstehenden Nachrichten nicht generiert. Die anstehenden Daten und der Interrupt werden aufgezeichnet und beim nächsten Öffnen des Kanals abgearbeitet.
- Informationen über Kommunikationskanäle sind unter dem Befehl OPEN in Abschn. 6.3.57 zu finden.

6.3.19 DEF ACT (Define act)

Funktion: Interrupt-Prozess definieren

Dieser Befehl legt die Bedingungen eines Interrupts fest. Dazu gehört die gleichzeitige Überwachung von Signalen, die Interrupt-Verarbeitung während der Programmausführung und die Definition des Interrupt-Prozesses, der bei Auftreten eines Interrupts ausgeführt werden soll.

Eingabeformat

DEF □ ACT □ <Priorität>, <Ausdruck> □ <Prozess> [, <Typ>]

<Priorität>	Gibt die Priorität des Interrupts an $1 \leq \text{Priorität} \leq 8$
<Ausdruck>	Folgende Formate können für den Interrupt-Status verwendet werden (siehe auch Syntaxdiagramm): <Num. Datentyp> <Vergleichsoperator> <Num. Datentyp> oder <Num. Datentyp> <Logischer Operator> <Num. Datentyp>. Die Angabe <Numerischer Datentyp> bezieht sich auf: <Numerische Konstanten> <Numerische Variablen> <Numerische Feldvariablen> <Komponentendaten>.
<Prozess>	Legt eine GOTO- oder GOSUB-Anweisung fest, die bei einem Interrupt ausgeführt wird
<Typ>	Bei fehlender Angabe: Stoppmethode 1 Die Stopposition des Roboters ergibt sich bei einer Geschwindigkeitsübersteuerung von 100 %. Wird der Wert verkleinert, ist die Zeit bis zum Stopp länger, die Stopposition jedoch bleibt dieselbe. S: Stoppmethode 2 (ab Software-Version E3) Unabhängig von der Einstellung der Geschwindigkeitsübersteuerung stoppt der Roboter in der kürzestmöglichen Zeit. L: Stopp nach Abarbeitung der Zeile Die Angabe „L“ legt fest, dass der Interrupt-Prozess nach Erreichen der Zielposition (Abarbeitung der Zeile) ausgeführt wird.

Programmbeispiel

10 DEF ACT 1,M_IN(17) = 1 GOSUB 100	Definiert einen Unterprogramm- sprung zu Zeile 100, wenn der Status des allgemeinen Eingangssignals Nummer 17 = EIN ist
20 DEF ACT 2,MFG1 AND MFG2 GOTO 200	Definiert einen Programmsprung zu Zeile 200, wenn das Resultat der UND-Verknüpfung von MFG1 und MFG2 „wahr“ ist
30 DEF ACT 3,M_TIMER(1) > 10.5 GOSUB *LBL	Definiert nach Ablauf von 10,5 s einen Unterprogramm- sprung zu Zeile 300
:	
100 M_TIMER(1) = 0	Zähler zurücksetzen
110 ACT 3 = 1	Interrupt 3 freigeben
120 RETURN 0	Sprung in die Zeile, aus der der Interrupt aufgerufen wurde
200 MOV P_SAFE	Rückzugspunkt mittels Gelenk- Interpolation anfahren
210 END	Unterprogrammende
300 *LBL	Marke LBL festgelegt
310 M_TIMER(1) = 0.0	Zähler zurücksetzen
320 ACT 3 = 0	Interrupt 3 sperren
330 RETURN 0	Sprung in die Zeile, aus der der Interrupt aufgerufen wurde

Erläuterung

- Die Prioritäten der Interrupts sind in aufsteigender Reihenfolge von 1 bis 8 festgelegt.
- Über die Priorität können bis zu 8 Interrupts unterschieden werden.
- Als Ausdruck dürfen einfache logische Operationen oder Vergleichsoperationen (ein Operand) verwendet werden. Klammerausdrücke sind nicht zulässig.
- Haben zwei Interrupts dieselbe Priorität, ist der später definierte Interrupt vorrangig.
- Der DEF ACT-Befehl definiert nur den Interrupt. Mit dem ACT-Befehl wird der Status des Interrupts festgelegt.
- Der Kommunikations-Interrupt (COM) hat eine höhere Priorität als Interrupts, die mit dem DEF ACT-Befehl definiert wurden.
- DEF ACT-Definitionen sind nur in dem Programm wirksam, in dem sie definiert wurden. In einem Unterprogramm müssen sie gegebenenfalls neu definiert werden.
- Wird ein Interrupt durch eine GOTO-Anweisung in einem DEF ACT-Befehl generiert, bleibt der Interrupt während der Abarbeitung des verbleibenden Programmteils erhalten und es werden nur Interrupts höherer Priorität akzeptiert. Der Interrupt kann durch die Ausführung der END-Anweisung deaktiviert werden.
- Ausdrücke mit Kombinationen aus logischen Operationen und Vergleichsoperationen wie z. B. (M1 AND &H001) = 1 sind nicht zulässig.

**ACHTUNG:**

Wählen Sie die Stoppmethode passend zum Anwendungszweck. Soll z. B. die Ausführung eines Bewegungsbefehls in kürzester Zeit mit kürzestem Weg gestoppt werden, wählen Sie die Einstellung „S“.

Folgende Diagramme zeigen die unterschiedlichen Stoppmethoden bei Unterbrechung einer Verfahrbewegung durch einen Interrupt-Prozess.

Stoppmethode	Übersteuerung 100 %	Übersteuerung 50 %
Stoppmethode 1 (bei fehlendem Argument) $S1 = S2$		
Stoppmethode 2 (S)		
Stopp nach Abarbeitung der Zeile (L) $S3 = S4$		

Tab. 6-4: Stoppmethoden

Steht in Beziehung zu folgenden Befehlen:

ACT

6.3.20 DEF ARCH (Define Arch)

Definiert einen Bogen für die Verfahrbewegung mittels Bogen-Interpolation über den Befehl MVA.

Eingabeformat

Ab Software-Version G2

```
DEF ARCH □ <Bogennummer>,
          [<Schrittweite bei Aufwärtsbewegung>],
          [<Schrittweite bei Abwärtsbewegung>],
          [<Hilfsschrittweite bei Aufwärtsbewegung>],
          [<Hilfsschrittweite bei Abwärtsbewegung>],
          [<Interpolationstyp>],
          [<Interpolationstyp 1>,<Interpolationstyp 2>]
```

<Bogennummer>

Gibt die Nummer des Bogens als Konstante oder Variable an
 $1 \leq \text{Bogennummer} \leq 4$

<Schrittweite bei Aufwärtsbewegung>
 <Schrittweite bei Abwärtsbewegung>
 <Hilfsschrittweite bei Aufwärtsbewegung>
 <Hilfsschrittweite bei Abwärtsbewegung>

Legt die Schrittweite als Konstante oder Variable fest
 (siehe folgende Abbildung)

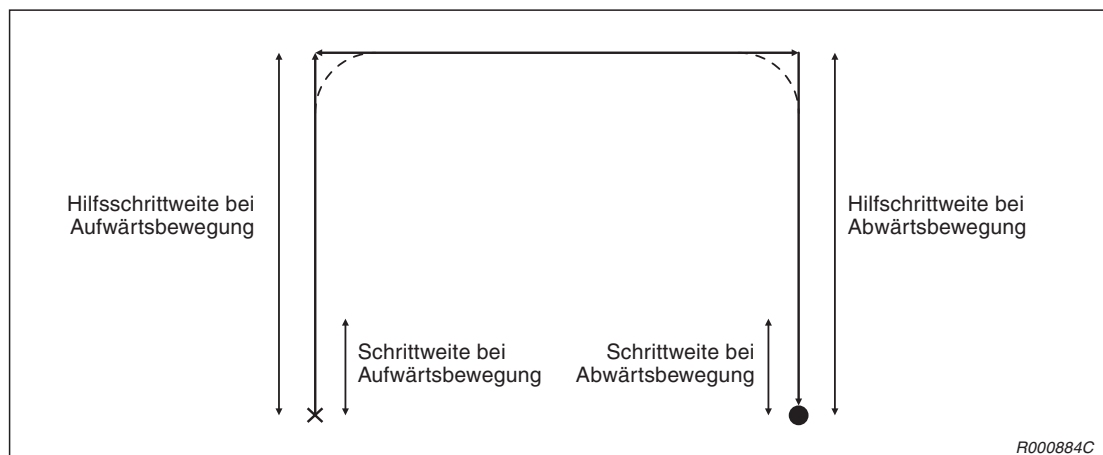


Abb. 6-8: Definition des Bogens

<Interpolationstyp>

Legt die Interpolationsart für die Aufwärts- und Abwärtsbewegungen fest

<Interpolationstyp 1>
 <Interpolationstyp 2>

Linear/Gelenk = 1/0
 Indirekt/direkt = 1/0
 3-Achsen-XYZ/äquivalente Drehung = 1/0

Für fehlende Argumente neben der Bogennummer werden die Standardwerte gesetzt. Die Standardwerte sind über folgende Parameter festgelegt. Prüfen Sie die Werte der Parameter vor ihrer Verwendung. Die Parameterwerte sind veränderbar.

Parameter	Bogennummer	Schrittweite bei Aufwärtsbewegung [mm]	Schrittweite bei Abwärtsbewegung [mm]	Hilfsschrittweite bei Aufwärtsbewegung [mm]	Hilfsschrittweite bei Abwärtsbewegung [mm]
ARCH1S	1	0.0	0.0	30.0	30.0
ARCH2S	2	10.0	10.0	30.0	30.0
ARCH3S	3	20.0	20.0	30.0	30.0
ARCH4S	4	30.0	30.0	30.0	30.0

Tab. 6-5: Standardwerte der Bogenparameter (Schrittweiten)

Für die Interpolationstypen gelten folgende Standardwerte:

Parameter	Bogennummer	Vertikaler Knickarmroboter (RV-1A/2AJ, RV-4A/5AJ usw.)			SCARA-Roboter (RH-5AH usw.)		
		Interpolationstyp	Interpolationstyp 1	Interpolationstyp 2	Interpolationstyp	Interpolationstyp 1	Interpolationstyp 2
ARCH1T	1	1	0	0	0	0	0
ARCH2T	2	1	0	0	0	0	0
ARCH3T	3	1	0	0	0	0	0
ARCH4T	4	1	0	0	0	0	0

Tab. 6-6: Standardwerte der Bogenparameter (Interpolationstypen)

Programmbeispiel

- 10 DEF ARCH 1,5,5,20,20 Definiert den Bogen
- 20 MVA P1,1 Position P1 mittels Bogen-Interpolation über den in Zeile 10 definierten Bogen anfahren
- 20 MVA P2,2 Position P2 mittels Bogen-Interpolation über den mit den Standardeinstellungen definierten Bogen anfahren

Erläuterung

- Wird der Befehl MVA ohne vorherige Definition des Bogens über den DEF ARCH-Befehl ausgeführt, erfolgt die Verfahrensbewegung entsprechend dem über die Parameter festgelegten Bogen.
- Der Befehl ermöglicht die Ausführung von Bewegungsbefehlen mit unterschiedlichen Schrittweiten in einem Programm.

Steht in Beziehung zu folgenden Befehlen:

MVA, ACCEL, OVRD, MVS (Erläuterung der Funktion „TYPE“)

6.3.21 DEF CHAR (Define Character)

Funktion: Zeichenkettenvariable definieren

Dieser Befehl definiert eine Zeichenkettenvariable. Er wird dann zur Variablendeklaration verwendet, wenn der Variablenname mit einem anderen Buchstaben als mit „C“ beginnen soll. Variablennamen, die mit „C“ beginnen, müssen nicht über den Befehl DEF CHAR deklariert werden.

Eingabeformat

```
DEF  CHAR  <Name der Zeichenkettenvariablen>
      [, <Name der Zeichenkettenvariablen>] ...
```

<Name der Zeichenkettenvariablen> Legt den Namen der Zeichenkettenvariablen fest

Programmbeispiel

10 DEF CHAR MESSAGE	Deklariert MESSAGE als Namen einer Zeichenkettenvariablen
20 MESSAGE = "WORKSET"	Schreibt die Zeichenkette WORKSET in die Zeichenkettenvariable MESSAGE
30 CMSG = "ABC"	Schreibt die Zeichenkette ABC in die Zeichenkettenvariable CMSG

Zeichenkettenvariablen, deren Variablenname mit „C“ beginnt, müssen nicht über den Befehl DEF CHAR deklariert werden

Erläuterung

- Der Variablenname darf aus maximal 8 Zeichen bestehen. Detaillierte Hinweise zu den verwendbaren Zeichentypen finden Sie in Abschn. 5.1.5.
- Bei Deklaration mehrerer Variablennamen dürfen maximal 127 Zeichen (inklusive des Befehls) in einer Zeile verwendet werden.
- Durch einen Unterstrich „_“ hinter dem „C“ wird eine im Basisprogramm definierte Variable zur globalen Variablen und kann programmübergreifend verwendet werden. Eine detaillierte Beschreibung benutzerdefinierter externer Variablen finden Sie auf Seite 5-16.

6.3.22 DEF FN (Define function)

Funktion: Funktion definieren

Definiert eine Funktion und legt den Namen fest

Eingabeformat

```
DEF □ FN <Identifizierungszeichen> <Name>
      [ (<Formalparameter> [, <Formalparameter>] ... ) ]
      = <Funktionsausdruck>
```

- <Identifizierungszeichen> Es werden vier Zeichen zur Identifizierung unterschieden:
 M: Typ Numerisch
 C: Typ Zeichenkette
 P: Typ Position
 J: Typ Gelenk
- <Name> Benutzerdefinierte Zeichenkette (max. 5 Zeichen)
- <Formalparameter> Legt die Variablen der Funktion fest
 Es können maximal 16 Variablen verwendet werden.
- <Funktionsausdruck> Legt die Rechenoperation fest.

Programmbeispiel

- 10 DEF FNMAVE(MA,MB) = (MA+MB)/2 Legt fest, dass durch FNMAVE der Durchschnitt von zwei numerischen Variablen gebildet wird
- 20 MDATA1 = 20 Weist MDATA1 den Wert 20 zu
- 30 MDATA2 = 30 Weist MDATA2 den Wert 30 zu
- 40 MAV = FNMAVE(MDATA1,MDATA2) Der Durchschnitt von 20 und 30 (= 25) wird der numerischen Variablen MAV zugewiesen.
- 50 DEF FNPADD(PA,PB) = (PA+PB) Legt fest, dass durch FNPADD die Summe von zwei Positionsvariablen gebildet wird
- 60 P10 = FNPADD(P1,P2) Weist P10 die Summe von P1 und P2 zu

Erläuterung

- Durch FN und <Name> wird der Name der Funktion festgelegt. Der Funktionsname kann bis zu 8 Zeichen lang sein.
Beispiel:
Numerischer Typ ... FNMAX Identifizierungszeichen: M
Zeichenkettentyp ... FNCAME\$ Identifizierungszeichen: C (Wird durch „\$“ abgeschlossen)
- Eine mit DEF FN definierte Funktion heißt benutzerdefinierte Funktion.
- Es können Funktionen bis zu maximal einer Zeilenlänge beschrieben werden.
- Im <Ausdruck> können fest definierte und schon vorher vom Benutzer definierte Funktionen verwendet werden. In diesem Fall können 16 Ebenen von benutzerdefinierten Funktionen verwendet werden.
- Wenn die Variablen im Funktionsausdruck nicht in den Formalparametern aufgeführt wurden, erfolgt für die Variablen eine Verarbeitung der augenblicklichen Werte. Es tritt eine Fehlermeldung auf, wenn die Anzahl oder der Typ der verwendeten Variablen (numerische oder Zeichenkette) von den deklarierten abweichen.
- Eine benutzerdefinierte Funktion steht nur in dem Programm zur Verfügung, in dem sie definiert worden ist. Sie kann von einem anderen Programm nicht durch einen CALLP-Befehl aufgerufen werden.

6.3.23 DEF INTE/FLOAT/DOUBLE (Define Integer/Float/Double)

Funktion: Numerische Variable deklarieren

Dieser Befehl definiert eine numerische Variable. Dabei steht INTE für den Typ Integer, FLOAT für den Typ Real mit einfacher Genauigkeit und DOUBLE für den Typ Real mit doppelter Genauigkeit.

Eingabeformat

```

DEF □ INTE □ <Name einer numerischen Variablen>
                [, <Name einer numerischen Variablen>] ...
DEF □ FLOAT □ <Name einer numerischen Variablen>
                [, <Name einer numerischen Variablen>] ...
DEF □ DOUBLE □ <Name einer numerischen Variablen>
                [, <Name einer numerischen Variablen>] ...
    
```

<Name einer numerischen Variablen> Legt den Variablennamen fest

Programmbeispiel

Deklaration einer Variablen vom Typ Integer

- 10 DEF INTE WORK1, WORK2 Deklariert WORK1 und WORK 2 als Namen zweier numerischer Variablen
- 20 WORK1 = 100 Setzt den Wert der numerischen Variablen WORK1 auf „100“
- 30 WORK2 = 10.562 Setzt den Wert der numerischen Variablen WORK2 auf „11“
- 40 WORK2 = 10.12 Setzt den Wert der numerischen Variablen WORK2 auf „10“

Deklaration einer Variablen vom Typ Real mit einfacher Genauigkeit

- 10 DEF FLOAT WORK3 Deklariert WORK3 als Namen einer numerischen Variablen
- 20 WORK3 = 123.468 Setzt den Wert der numerischen Variablen WORK3 auf „123,468000“

Deklaration einer Variablen vom Typ Real mit doppelter Genauigkeit

- 10 DEF DOUBLE WORK4 Deklariert WORK4 als Namen einer numerischen Variablen
- 20 WORK4 = 100/3 Setzt den Wert der numerischen Variablen WORK4 auf „33,333332061767599“

Erläuterung

- Der Variablenname kann bis zu 8 Zeichen lang sein. Detaillierte Hinweise zu den verwendbaren Zeichentypen finden Sie in Abschn. 5.1.5.
- Bei Deklaration mehrerer Variablennamen dürfen maximal 123 Zeichen (inklusive Befehl) in einer Zeile verwendet werden.
- Eine mit dem Befehl INTE deklarierte Variable ist vom Typ Integer (–32768 bis +32767).
- Eine mit dem Befehl FLOAT deklarierte Variable ist eine numerische Variable mit einfacher Genauigkeit ($\pm 1,70141E+38$).
- Eine mit dem Befehl DOUBLE deklarierte Variable ist eine numerische Variable mit doppelter Genauigkeit ($\pm 1,701411834604692E+308$).

6.3.24 DEF IO (Define IO)

Funktion: Ein-/Ausgangvariable definieren

Dieser Befehl definiert eine Ein-/Ausgangvariable. Sie dient zur Festlegung von Bitbreiten. Die Variablen M_IN und M_OUT werden für Einzelbitsignale, die Variablen M_INB und M_OUTB für 8-Bit breite Bytes und die Variablen M_INW und M_OUTW für 16-Bit breite Wörter verwendet.

Eingabeformat

```
DEF  IO  <E/A-Variablenname> = <Variablentyp>,  
                                <E/A-Bitnummer>] [, <Maskeninformation>]
```

- <E/A-Variablenname> Legt den Variablennamen fest
- <Variablentyp> Legt fest, ob die Variable vom Typ BIT (1 Bit), BYTE (8 Bit), WORD (16 Bit) oder INTEGER ist
- <E/A-Bitnummer> Legt die Nummer des Eingangs-/Ausgangsbits fest
- <Maskeninformation> Legt fest, ob ein bestimmtes Signal zugelassen wird

Programmbeispiel

Zuweisung des Ein-/Ausgangsbits Nr. 6 vom Typ BIT an die Ein-/Ausgangvariable mit dem Namen PORT1

```
10 DEF IO PORT1 = BIT,6            Weist der Ein-/Ausgangvariablen mit dem
   :                               Namen PORT1 das Ein-/Ausgangsbit Nr. 6 vom
   :                               Typ BIT zu
100 PORT1 = 1                      Schaltet Ausgangssignal 6 ein
   :
200 PORT1 = 2                      Schaltet Ausgangssignal 6 aus (da die niedrigste
   :                               Stelle des numerischen Werts „2“ gleich „0“ ist
210 M1 = PORT1                     Weist der Variablen M1 den Wert des
   :                               Eingangssignals 6 zu
```

Zuweisung des Ein-/Ausgangsbits Nr. 5 vom Typ BYTE an die Ein-/Ausgangvariable mit dem Namen PORT2 und Festlegung der Maskeninformation auf hexadezimal 0F

```
10 DEF IO PORT2 = BYTE,5,&H0F    Weist der Ein-/Ausgangvariablen mit dem
   :                               Namen PORT2 das Ein-/Ausgangsbit Nr. 5 vom
   :                               Typ BYTE zu und legt die Maske auf hexadezimal
   :                               „0F“ fest
100 PORT2 = &HFF                  Schaltet die Ausgangssignale 5 bis 8 ein
   :
200 M2 = PORT2                     Weist der Variablen M2 den Wert der
   :                               Eingangssignale 5 bis 8 zu
```

Zuweisung des Ein-/Ausgangsbits Nr. 8 vom Typ WORD an die Ein-/Ausgangsvariable mit dem Namen PORT3 und Festlegung der Maskeninformation auf hexadezimal 0FFF

```
10 DEF IO PORT3 = WORD,8,&H0FFF      Weist der Ein-/Ausgangsvariablen mit dem
      :                               Namen PORT3 das Ein-/Ausgangsbit Nr. 8
100 PORT3 = 9                        Schaltet die Ausgangssignale 8 und 11 ein
      :
200 M3 = PORT3                       Weist der Variablen M3 den Wert der
                                     Eingangssignale 8 bis 19 zu
```

Erläuterung

- Eingangssignale werden beim Zugriff auf die Variable eingelesen.
- Ausgangssignale werden geschrieben, wenn der Variablen ein Wert zugewiesen wird.
- Über die mit dem Befehl DEF IO definierten Variablen kann kein Zugriff auf die Ausgangssignale erfolgen. Verwenden Sie zu diesem Zweck die Variable M_OUT.
- Der Variablenname kann bis zu 8 Zeichen lang sein. Detaillierte Hinweise zu den verwendbaren Zeichentypen finden Sie in Abschn. 5.1.5.
- Ist eine Maske angegeben, wird nur ein bestimmtes Signal zugelassen.

Beispiel ▾

In Zeile 20 wird bei einem 8 Bit breiten Byte ein Maskierungsprozess mit dem hexadezimalen Wert 0F ausgeführt.

Für PORT 2 ergibt sich danach als

- Eingangssignal (M1 = PORT2), dass die Bits Nummer 5 bis 8 als Eingänge zugelassen und die Bits Nummer 9 bis 12 auf „0“ gesetzt werden

```
      gesperrt  zugelassen
0 0 0 0 1 1 1 1
12                               5 (E/A-Bitnummer)
```

- Ausgangssignal (PORT2 = M1), dass die Bits Nummer 5 bis 8 die aktuellen Daten ausgeben und die Bits Nummer 9 bis 12 die Signalzustände beibehalten

```
      Daten beibehalten  aktuelle Daten ausgeben
      * * * * 1 1 1 1
12                               5 (E/A-Bitnummer)
```

△

Beispiel ▾

Mit Hilfe der DEF IO-Funktion ist es möglich, sowohl einen Eingang als auch einen Ausgang zu definieren. Generell lässt sich die Funktion wie folgt beschreiben:

Wird die Variable auf der linken Seite eines mathematischen Ausdrucks benutzt, wird sie als Ausgang festgelegt.

Wird die Variable auf der rechten Seite eines mathematischen Ausdrucks eingesetzt, wird sie als Eingang festgelegt.

```
10 DEF IO PORT1 = BIT,6
20 PORT1 = 1                               Ausgang 6 wird eingeschaltet.
30 M1 = PORT1                              Das Eingangssignal von Eingang 6 wird in M1
                                             eingelesen.
40 IF PORT1 = 1 THEN GOTO 100             Sonderfall!
                                             Eingang 6 wird auf log. 1 geprüft
```

△

6.3.25 DEF JNT (Define Joint)

Funktion: Gelenkvariable definieren

Dieser Befehl definiert eine Gelenkvariable. Er wird dann zur Variablendeklaration verwendet, wenn der Variablenname mit einem anderen Buchstaben als mit „J“ beginnen soll. Variablennamen, die mit „J“ beginnen, müssen nicht über den Befehl DEF JNT deklariert werden.

Eingabeformat

```
DEF □ JNT □ <Gelenkvariablenname> [,<Gelenkvariablenname>] ...
```

<Gelenkvariablenname> Legt den Namen der Gelenkvariablen fest

Programmbeispiel

10 DEF JNT SAFE	Deklariert SAFE als Namen einer Gelenkvariablen
20 MOV J1	Gelenkvariablen, deren Variablenname mit „J“ beginnt, müssen nicht über den Befehl DEF JNT deklariert werden
30 SAFE = (-50,120,30,300,0,0,0,0)	Weist der Variablen SAFE die angegebenen Werte zu
40 MOV SAFE	Fährt die Position SAFE an

Erläuterung

- Der Variablenname darf aus maximal 8 Zeichen bestehen. Detaillierte Hinweise zu den verwendbaren Zeichentypen finden Sie in Abschn. 5.1.5.
- Bei Deklaration mehrerer Variablennamen dürfen maximal 127 Zeichen (inklusive des Befehls) in einer Zeile verwendet werden.
- Durch einen Unterstrich „_“ hinter dem „J“ wird eine im Basisprogramm definierte Variable zur globalen Variablen und kann programmübergreifend verwendet werden. Eine detaillierte Beschreibung benutzerdefinierter externer Variablen finden Sie auf Seite 5-16.

6.3.26 DEF PLT (Define pallet)

Funktion: Palette definieren

Definiert eine Palette

Eingabeformat

```
DEF □ PLT □ <Palettennummer>, <Bezugsposition>,
<Endpunkt A>, <Endpunkt B>,
[<Paletteneckpunkt, der gegenüber der
Bezugsposition liegt>],
<Anzahl der Gitterpunkte zwischen Bezugsposition
und Endpunkt A>,
<Anzahl der Gitterpunkte zwischen Bezugsposition
und Endpunkt B>,
<Bewegungsrichtung>
```

<Palettennummer>	Legt die Nummer der eingesetzten Palette fest $1 \leq \text{Palettennummer} \leq 8$
<Bezugsposition>	Legt den Anfangspunkt der Palette fest
<Endpunkt A>	Legt einen der Endpunkte der Palette fest Dient als Zwischenposition für kreisförmige Paletten
<Endpunkt B>	Legt einen der Endpunkte der Palette fest Dient als Endpunkt für kreisförmige Paletten
<Paletteneckpunkt, der gegenüber der Bezugsposition liegt>	Legt den Punkt fest, der gegenüber der Bezugsposition liegt Für kreisförmige Paletten ohne Bedeutung
<Anzahl der Gitterpunkte zwischen Bezugsposition und Endpunkt A>	Legt die Gitterpunkte der Palette zwischen Bezugsposition und Endpunkt A fest Legt bei kreisförmiger Palette die Positionen zwischen dem Anfangs- und Endpunkt fest
<Anzahl der Gitterpunkte zwischen Bezugsposition und Endpunkt B>	Legt die Gitterpunkte der Palette zwischen Bezugsposition und Endpunkt B fest Für kreisförmige Paletten ohne Bedeutung (1 usw., muss festgelegt werden)
<Bewegungsrichtung>	Die Eingabe von „1“, „2“ oder „3“ legt die Bewegungsrichtung fest. 1 = Zickzack 2 = Bewegungsrichtung beibehalten (z. B. von links nach rechts) 3 = kreisförmige Bewegung

Programmbeispiel

```
10 DEF PLT 1,P1,P2,P3, ,4,3,1    Palettendefinition mit 3 Punkten
20 DEF PLT 1,P1,P2,P3,P4,4,3,1  Palettendefinition mit 4 Punkten
```

Palettendefinition und Bewegungsrichtung

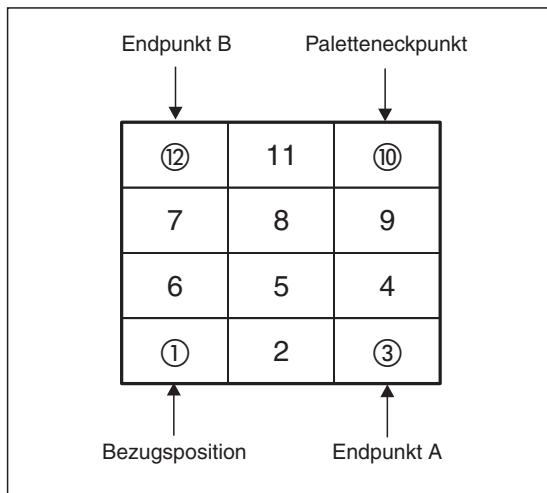


Abb. 6-9:
Palettendefinition mit
Bewegungsrichtung = 1 (zickzack)

R000693C

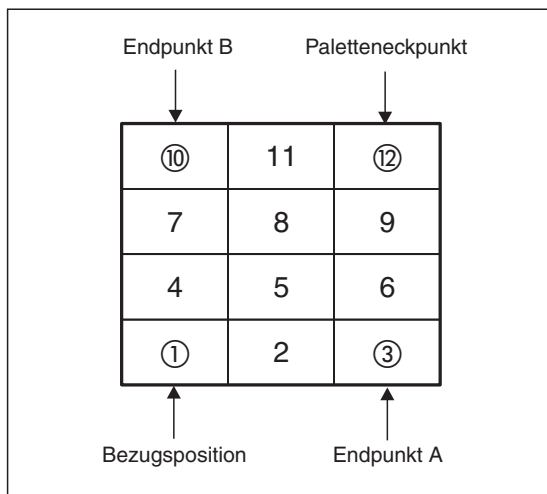


Abb. 6-10:
Palettendefinition mit
Bewegungsrichtung = 2
(Bewegungsrichtung beibehalten)

R000694C

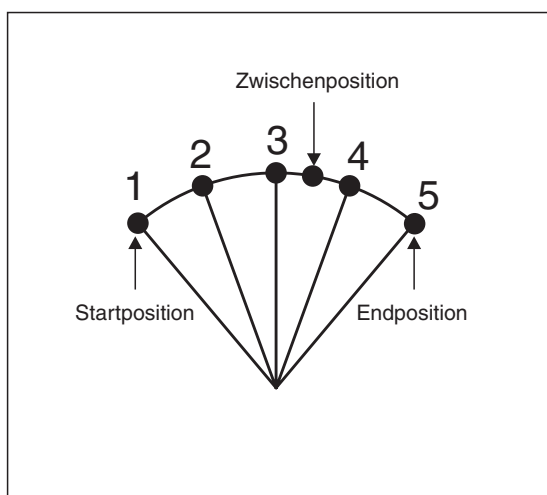


Abb. 6-11:
Palettendefinition mit
Bewegungsrichtung = 3
(kreisförmig)

R000695C

Erläuterung

- Es können 3 oder 4 Punkte einer Palette definiert werden. Die Festlegung von 3 Punkten ist einfacher zu programmieren; durch die Festlegung von 4 Punkten erreicht man eine höhere Präzision.
- Der Befehl steht nur innerhalb des ausgeführten Programms zur Verfügung. Er kann nicht von einem anderen Programm aufgerufen werden. Falls nötig, muss er erneut definiert werden.
- Die Anzahl der Zeilen- und Spaltengitterpunkte muss größer als Null sein. Ansonsten erfolgt eine Fehlermeldung.
- Wenn das Produkt $\langle \text{Anzahl der Gitterpunkte zwischen Bezugsposition und Endpunkt A} \rangle \times \langle \text{Anzahl der Gitterpunkte zwischen Bezugsposition und Endpunkt B} \rangle$ den Wert 32767 überschreitet, erfolgt eine Fehlermeldung.
- Die Angabe der Anzahl der Anzahl der Gitterpunkte zwischen Bezugsposition und Endpunkt B ist für kreisförmige Paletten ohne Bedeutung, darf aber nicht weggelassen werden. Der Paletteneckpunkt, der gegenüber der Bezugsposition liegt, ist wirkungslos, auch wenn er festgelegt wurde. Setzen Sie einen formalen Wert.
- Bei nach unten gerichtetem Handflansch müssen die Vorzeichen der Stellungsdaten (A, B und C) an der Bezugsposition, dem Endpunkt A und dem Endpunkt B übereinstimmen. Bei einem Knickarm-Roboter mit nach unten gerichtetem Handflansch gilt: $A = 180^\circ$ (oder -180°), $B = 0$ und $C = 180^\circ$ (oder -180°). Stimmen die Vorzeichen der Stellungsdaten A und C an den 3 Positionen nicht überein, kann die Hand in der Mittelposition rotieren. Gleichen Sie in einem solchen Fall die Vorzeichen über die Teaching Box an. Dieselbe Position kann über einen Drehwinkel von $+180^\circ$ oder -180° erreicht werden. Eine Änderung der Vorzeichen dürfte somit problemlos durchzuführen sein.

Steht in Beziehung zu folgenden Befehlen:

PLT

6.3.27 DEF POS (Define Position)

Funktion: Positionsvariable definieren

Dieser Befehl definiert eine XYZ-Positionsvariable. Er wird dann zur Variablendeklaration verwendet, wenn der Variablenname mit einem anderen Buchstaben als mit „P“ beginnen soll. Variablennamen, die mit „P“ beginnen, müssen nicht über den Befehl DEF POS deklariert werden.

Eingabeformat

```
DEF □ POS □ <Positionsvariablenname>
           [, <Positionsvariablenname>] ...
```

<Positionsvariablenname> Legt den Namen der Positionsvariablen fest

Programmbeispiel

<pre>10 DEF POS WORKSET 20 MOV P1</pre>	<p>Deklariert WORKSET als Namen einer Positionsvariablen</p> <p>Positionsvariablen, deren Variablenname mit „P“ beginnt, müssen nicht über den Befehl DEF POS deklariert werden</p>
<pre>30 WORKSET = (250,460,100,0,0,-90,0,0)(0,0)</pre>	<p>Weist der Variablen SAFE die angegebenen Werte zu</p>
<pre>40 MOV WORKSET</pre>	<p>Fährt die Position WORKSET an</p>

Erläuterung

- Der Variablenname darf aus maximal 8 Zeichen bestehen. Detaillierte Hinweise zu den verwendbaren Zeichentypen finden Sie in Abschn. 5.1.5.
- Bei Deklaration mehrerer Variablennamen dürfen maximal 127 Zeichen (inklusive des Befehls) in einer Zeile verwendet werden.
- Durch einen Unterstrich „_“ hinter dem „P“ wird eine im Basisprogramm definierte Variable zur globalen Variablen und kann programmübergreifend verwendet werden. Eine detaillierte Beschreibung benutzerdefinierter externer Variablen finden Sie auf Seite 5-16.

6.3.28 DIM (Dim)

Funktion: Dimension definieren

Legt die Anzahl der Elemente bei Feldvariablen fest. Es sind bis zu drei Dimensionen möglich.

Eingabeformat

```
DIM □ <Variablenname>(<max. Indexwert>, [, <max. Indexwert>])
      [, <Variablenname>(<max. Indexwert>
      [, <max. Indexwert>])]
```

<Variablenname> Legt den Namen für die Feldvariable fest

<maximaler Indexwert> Konstante, die die Anzahl der Elemente einer Feldvariablen festlegt
 $1 \leq \text{Maximaler Indexwert} \leq 999$
 Der maximale Indexwert darf nur Konstanten enthalten.
 Ausdrücke mit numerischen Operationen sind nicht erlaubt.

Programmbeispiel

10	DIM PDATA(10)	Deklariert die Positions-Feldvariable PDATA als eine Variable mit 10 Elementen
20	DIM MDATA#(5)	Deklariert die Feldvariable MDATA# als eine Variable mit 5 Elementen und doppelter Genauigkeit
30	DIM M1%(6)	Deklariert die Feldvariable M1% als eine Variable vom Typ Integer mit 6 Elementen
40	DIM M2!(4)	Deklariert die Feldvariable M2! als eine Feldvariable mit 4 Elementen und einfacher Genauigkeit
50	DIM CMOJI(7)	Deklariert die Zeichenketten-Feldvariable CMOJI als eine Variable mit 7 Elementen
20	DIM MD6(2,3), PD1(5,5)	Deklariert die zweidimensionale numerische Feldvariable MD6 vom Typ Real mit einfacher Genauigkeit als eine Variable mit 2×3 Elementen Deklariert die zweidimensionale Positions-Feldvariable PD1 als eine Variable mit 5×5 Elementen

Erläuterung

- Es sind ein-, zwei- und dreidimensionale Felder erlaubt.
- Numerische Variablen sind in die Typen Integer und Real mit einfacher oder doppelter Genauigkeit aufgeteilt. Der Variablentyp wird über ein Symbol hinter dem Variablennamen festgelegt. Fehlt die Angabe des Variablentyps, wird der Typ Real mit einfacher Genauigkeit definiert.
DIM MABC(10) deklariert die Feldvariable MABC als eine Variable vom Typ Real mit einfacher Genauigkeit und 10 Elementen.
- Die Indexwerte für den Zugriff auf eine Feldvariable beginnen bei 1. Die Variable PDATA in Zeile 40 des Programmbeispiel setzt sich aus den Elementen mit den Nummern 1 bis 10 zusammen.
- Der Wertebereich für die Indexwerte liegt zwischen 1 und 999. Ausdrücke mit numerischen Operationen sind nicht erlaubt. Der Systemspeicherbereich kann jedoch nicht überschritten werden.
Ist die Anzahl der Elemente eine reelle Zahl, wird die Zahl automatisch auf eine Integer-Zahl gerundet. In Abhängigkeit des freien Systemspeicherplatzes ist die Anzahl der Elemente von Feldvariablen begrenzt. Erfolgt eine Fehlermeldung, ist die Speicherkapazität überschritten.
- Bei Überschreitung des Wertebereiches für den maximalen Indexwert erfolgt bei der Ausführung der DIM-Anweisung eine Fehlermeldung.
- Bei Ausführung der DIM-Anweisung sind die Feldvariablen auf keine Standardwerte gesetzt, sondern nicht definiert.
- Es können keine Felder ohne DIM-Anweisung verwendet werden.
- Die DIM-Anweisung steht nur innerhalb des ausgeführten Programmes zur Verfügung. Sie kann nicht von einem anderen Programm aufgerufen werden. Bei Verwendung in einem Unterprogramm muss die DIM-Anweisung erneut definiert werden.
- Feldvariablen können wie normale Variablen verwendet werden. Ein Variablennamen oder eine Zeichenkette zur Festlegung der Elementnummer mit mehr als 8 Zeichen kann jedoch nicht auf der Teaching Box angezeigt werden.
- Durch einen Unterstrich „_“ als zweites Zeichen im Variablennamen wird eine Variable als globale Variable gekennzeichnet und kann programmübergreifend verwendet werden. Eine detaillierte Beschreibung benutzerdefinierter externer Variablen finden Sie auf Seite 5-16.

6.3.29 DLY (Delay)

Funktion: Verzögerung einstellen

Als einzelner Befehl wird zur festgelegten Zeit der Wartestatus erzeugt. Der Befehl dient zur Positionierung des Roboters und zur zeitabhängigen Steuerung von Ein- und Ausgangssignalen. Wird der DLY-Befehl für einen zusätzlichen Impulsausgang genutzt, wird die Impulsdauer festgelegt.

Eingabeformat

Als einzelner Befehl:

```
DLY □ <Zeit>
```

Für einen zusätzlichen Impulsausgang:

```
Beispiel: M_OUT(1) = 1 DLY □ <Zeit>
```

<Zeit> Legt die Dauer des Wartestatus oder die Impulsdauer in Sekunden fest
Der kleinste einstellbare Wert ist 0,01 s. Auch die Einstellung 0,00 s ist zulässig.

Programmbeispiel

Wartestatus festlegen

10 DLY 30 Wartezeit von 30 Sekunden

Ausgabe eines Impulses

20 M_OUT(17) = 1 DLY 0.5 Setzt das allgemeine Ausgangssignal 17 für 0,5 s auf „1“

30 M_OUTB(18) = 1 DLY 0.5 Setzt das Signal 18 von den allgemeinen Ausgangssignalen 18 bis 25 für 0,5 s auf „1“ und die Signale 19 bis 25 nach 0,5 s auf „1“

Positioniervorgang abschließen

10 MOV P1 Position P1 mittels Gelenk-Interpolation anfahren

20 DLY 0.1 Der Abschluss des Positioniervorgangs erfolgt über die Timer-Einstellung.

Handsteuerungsvorgang (Hand öffnen/schließen) abschließen

10 HOPEN 1 Öffnet Hand 1

20 DLY 0.1 Um sicherzustellen, dass die Hand geöffnet ist, erfolgt der Abschluss des Vorgangs über die Timer-Einstellung.

Erläuterung

- Der DLY-Befehl wird verwendet, um in Programmen Verzögerungszeiten zu erzeugen. Er dient ebenso zur zeitabhängigen Steuerung von Ein- und Ausgangssignalen, Bewegungsbefehlen und zur Festlegung einer Impulsdauer für ein Ausgangssignal in der OUT-Anweisung (siehe Zeile 20 im Programmbeispiel).
- Der Impulsausgang wird gleichzeitig mit Ausführung des in der nächsten Zeile stehenden Befehls gesetzt.
- Es können bis zu 50 Impulsausgänge gleichzeitig gesteuert werden. Wird dieser Wert überschritten, kommt es bei Ausführung des Befehls zu einer Fehlermeldung.
- Bei der Steuerung eines Impulsausgangs wird jedes Bit nach Ablauf der eingestellten Zeit [z. B. M_OUT (8 Bits) oder M_OUTW (16 Bits)] invertiert.
- Nach Ablauf der festgesetzten Zeit wird wieder der Zustand vor Ausführung des Befehls angenommen.
- Ein Programm endet ohne Berücksichtigung der eingestellten Zeitdauer für einen Impulsausgang mit Ausführung der END-Anweisung oder der letzten Programmzeile. Nach Ablauf der Impulsdauer wird der Ausgang abgeschaltet.
- Die Reihenfolge der Prioritäten ist:
COM > ACT > WTHIF (WTH) > Impulsausgang (Zeitintervall aktiv)
- Die Eingabe eines Stoppsignals während der Impulsausgabe führt nicht zur Unterbrechung der Impulsausgabe.

Beispiele ▾

Bei Eingabe eines Stoppsignals in Zeile 20, wird die Ausführung des Programms unterbrochen, der Ausgangssignalzustand bleibt jedoch erhalten.

```
10 M_OUT(17) = 1
20 DLY 10
30 M_OUT(17) = 0
```

Bei der Steuerung eines Impulsausgangs wird jedes Bit nach Ablauf der eingestellten Zeit [z. B. M_OUTB (8 Bits) oder M_OUTW (16 Bits)] invertiert.

```
M_OUTB(1) = 1 DLY 1.0
```

Das gezeigte Beispiel bewirkt die Ausgabe des Bitmusters 00000001 für eine Sekunde. Danach wird das Bitmuster 11111110 ausgegeben.

△

6.3.30 ERROR (Error)

Funktion: Fehler generieren

Im Anwendungsprogramm wird ein Fehler erzeugt.

Eingabeformat

```
ERROR □ <Fehlernummer>
```

<Fehlernummer> Es kann eine Konstante oder ein numerischer Ausdruck festgelegt werden.
 $9000 \leq \text{Fehlernummer} \leq 9299$

Programmbeispiel

Fehler 9000 generieren

```
100 ERROR 9000          Generiert den Fehler mit der Fehlernummer 9000
```

In Abhängigkeit des Variablenwertes von M1 werden unterschiedliche Fehler generiert.

```
40 IF M1 <> 0 THEN *LERR  Sprung zur Marke LERR, falls M1 ungleich 0 ist
```

```
:
```

```
140 *LERR                Sprungmarke „LERR“ festgelegt
```

```
150 MERR = 9000+M1*10    Berechnung der Fehlernummer in Abhängigkeit von M1
```

```
160 ERROR MERR           Generiert die berechnete Fehlernummer
```

```
170 END                  Programmende
```

Erläuterung

- Bei schweren und leichten Fehlern erfolgt eine Programmunterbrechung. Die Zeilen nach dem ERROR-Befehl werden nicht ausgeführt. Bei einer Warnung erfolgt keine Programmunterbrechung und das Programm wird in der nächsten Zeile weiter ausgeführt.
- Die Parameter UER1 bis UER20 ermöglichen die Definition von 20 Fehlermeldungen.
- Bei einer Fehlernummer außerhalb des gültigen Bereiches erfolgt die Meldung eines Systemfehlers.
- In Abhängigkeit der Fehlernummer reagiert das System wie in folgender Tabelle gezeigt:

Fehlernummer	Systemverhalten
9000–9099 (schwerer Fehler)	Die Programmausführung wird unterbrochen und die Servospannung abgeschaltet. Der Fehler wird bei einem Reset zurückgesetzt.
9100–9199 (leichter Fehler)	Die Programmausführung wird unterbrochen. Der Fehler wird bei einem Reset zurückgesetzt.
9200–9299 (Warnung)	Die Programmausführung wird nicht unterbrochen. Der Fehlerausgang wird bei einem Reset ausgeschaltet.

Tab. 6-7: Systemverhalten in Abhängigkeit der Fehlernummer

Steht in Beziehung zu folgenden Parametern:

UER1 bis UER20

6.3.31 END (End)

Funktion: Programmende

Der Befehl definiert die letzte Programmzeile. Er kennzeichnet explizit das Ende eines Programms, so dass ein weiteres Unterprogramm nach der END-Anweisung angefügt wird.

Eingabeformat

END

Programmbeispiel

10	MOV P1	Position P1 mittels Gelenk-Interpolation anfahren
20	GOSUB *ABC	Sprung zum Unterprogramm ABC
30	END	Programmende
	:	
100	*ABC	Sprungmarke „ABC“ festgelegt
110	M1 = 1	Numerische Variable M1 auf „1“ setzen
120	RETURN	Rücksprung in Zeile 20

Erläuterung

- Der Befehl definiert die letzte Programmzeile. Soll ein Programm in der Mitte unterbrochen und in den Pausestatus gesetzt werden, verwenden Sie den Befehl HLT.
- Bei einem über das Steuergerät gestarteten Programm erfolgt die Programmausführung kontinuierlich. Das Programm wird vom Beginn bis zur END-Anweisung und wieder von vorne ausgeführt. Eine Unterbrechung des Zyklus kann über die Taste END am Steuergerät erfolgen.
- Es können mehrere END-Anweisungen in einem Programm ausgeführt werden.
- Es muss nicht zwingend eine END-Anweisung an das Ende eines Programms gesetzt werden.
- Die Ausführung einer END-Anweisung in einem Unterprogramm, das durch einen CALLP-Befehl aufgerufen wird, übergibt die Steuerung wieder an das Programm, in dem der CALLP-Befehl ausgeführt wurde. Die Funktion entspricht der RETURN-Anweisung beim Befehl GOSUB.
- Eine END-Anweisung im Hauptprogramm schließt alle mit dem Befehl OPEN geöffneten Dateien.
- Bei Ausführung der END-Anweisung werden die durch folgende Befehle eingestellten Werte zurückgesetzt: SPD, ACCEL, OADL, JOVRD, OVRD, FINE und CNT.
- Das Ausführen der END-Anweisung führt nicht zwangsläufig zum Beenden des Programms. Durch das Umstellen des Parameters SLOTON auf den Wert 2 und Verändern des Parameters SLT1 von ..., REP, ..., ... auf ..., CYC, ..., ... bleibt der Roboter bei Erreichen von END stehen.

Steht in Beziehung zu folgenden Befehlen:

HLT, CALLP

6.3.32 FINE (Fine)

Funktion: Feinpositionierung

Legt den Status bei der Beendigung eines Interpolationsbefehls fest, wenn die CNT-Einstellung gesperrt ist. In Abhängigkeit des Robotermodells (z. B. RP-Serie) kann eine Positionierung über den Befehl DLY effektiver sein als über den Befehl FINE.

Eingabeformat

```
FINE  <Anzahl der Impulse> [, <Achse>]
```

<Anzahl der Impulse> Festlegung der Anzahl der Impulse zur Positionierung
Bei einer Einstellung von „0“ (Standardwert) ist die Feinpositionierung deaktiviert.

<Achse> Festlegung der Achse, für die die Feinpositionierung ausgeführt werden soll
Bei fehlender Angabe ist die Feinpositionierung für alle Achsen aktiviert.

Programmbeispiel

```
10 FINE 300 FINE-Einstellung auf 300 Impulse festlegen
20 MOV P1 Position P1 mittels Gelenk-Interpolation anfahren
30 FINE 100,2 Feinpositionierung für Achse 2 auf 100 Impulse festlegen
40 MOV P2 Position P2 mittels Gelenk-Interpolation anfahren
50 FINE 0 Feinpositionierung deaktivieren
60 MOV P3 Position P3 mittels Gelenk-Interpolation anfahren
70 FINE 100 Feinpositionierung für alle Achsen 100 Impulse festlegen
80 MOV P4 Position P2 mittels Gelenk-Interpolation anfahren
```

Erläuterung

- Die Beendigung von Bewegungsbefehlen wie z. B. MOV über die FINE-Anweisung findet nicht durch eine direkte Steuerung der Servomotoren statt. Es erfolgt vielmehr eine Prüfung, ob sich der Wert der Servorückmeldeimpulse innerhalb des gewünschten Bereiches befindet. Diese Methode ermöglicht eine genaue Positionierung des Roboters.
- Die Feinpositionierung über den Befehl DLY (Timer) ist einfacher zu definieren und kann in manchen Fällen bessere Ergebnisse als die Positionierung über den FINE-Befehl liefern.

Beispiel ▾

10	MOV P1	Position P1 mittels Gelenk-Interpolation anfahren
20	DLY 0.1	Der Abschluss des Positioniervorgangs erfolgt über die Timer-Einstellung.

△

- Während einer Programmabarbeitung ist die FINE-Einstellung solange gesperrt, bis sie durch das Programm freigegeben wird. Sobald die FINE-Einstellung freigegeben wurde, bleibt sie solange in diesem Zustand, bis sie erneut gesperrt wird.
- Nach Abarbeitung des Programms wird die FINE-Einstellung gesperrt (bei Ausführung der END-Anweisung oder bei Zurücksetzen des Programms während einer Programmunterbrechung).
- Ist die CNT-Einstellung freigegeben, wird der FINE-Befehl ignoriert. Er wird auch dann ignoriert, wenn er freigegeben ist (d. h. er wird als gesperrt interpretiert, die Einstellung bleibt jedoch erhalten).
- Die Festlegung der Anzahl der Impulse ist für Zusatzachsen (zusätzliche Servoachsen) nicht möglich. Hier kann der Wert des Parameters „INP“ des Servoverstärkers verwendet werden. Der Befehl FINE wird bei Einstellungen des Servoverstärker-Parameters auf Integer-Werte ungleich 0 aktiviert und bei der Einstellung 0 deaktiviert.

6.3.33 FOR-NEXT (For-Next)

Funktion: Programmschleife

Dieser Befehl bewirkt eine Wiederholung des Programmteils, der zwischen der FOR-Anweisung und der NEXT-Anweisung steht. Der Programmteil wird solange wiederholt, bis die Abbruchbedingungen erfüllt sind.

Eingabeformat

```
FOR □ <Zähler> = <Vorgabewert> TO <Endwert>[STEP<Schrittwert>]
:
NEXT □ [<Zähler 1> [,<Zähler 2>] ...]
```

<Zähler>	Der numerische Datentyp gibt die Anzahl der Wiederholungen der Programmschleife an. Gilt auch für Zähler 1 und Zähler 2 usw.
<Vorgabewert>	Gibt den Startwert des Zählers vor
<Endwert>	Gibt den Endwert des Zählers vor
<Schrittwert>	Legt die Schrittweite des Zählers fest Die Angabe des Wertes kann entfallen.

Programmbeispiel

Programm zur Addition der Zahlen 1 bis 10

10	MSUM = 0	Weist MSUM den Wert 0 zu
20	FOR M1 = 1 TO 10	Erhöht den Zähler der numerischen Variablen M1 von 1 bis 10 um 1
30	MSUM = MSUM + M1	Addiert M1 zu der numerischen Variablen MSUM
40	NEXT M1	Sprung zu Zeile 20

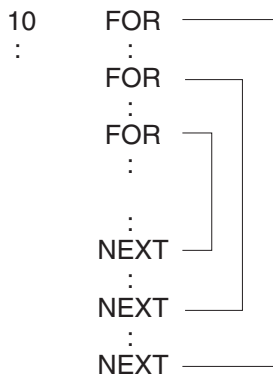
Speichert das Produkt zweier numerischer Variablen in eine zweidimensionale Feldvariable (Beispiel für verschachtelte FOR-NEXT-Programmschleifen)

10	DIM MBOX(10,10)	Reserviert Speicherplatz für eine 10 × 10 Feldvariable
20	FOR M1 = 1 TO 10 STEP 1	Erhöht den Zähler der numerischen Variablen M1 von 1 bis 10 um 1 und springt zu Zeile 70, sobald der Wert 10 überschritten ist („STEP 1“ kann weggelassen werden).
30	FOR M2 = 1 TO 10 STEP 1	Erhöht den Zähler der numerischen Variablen M2 von 1 bis 10 um 1 und springt zu Zeile 60, sobald der Wert 10 überschritten ist („STEP 1“ kann weggelassen werden).
40	MBOX(M1,M2) = M1 * M2	Ersetzt die Elemente der Feldvariablen MBOX(M1,M2) durch das Produkt M1 * M2.
50	NEXT M2	Sprung zu Zeile 30
60	NEXT M1	Sprung zu Zeile 20

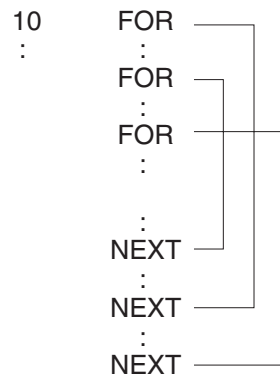
Erläuterung

- Es tritt ein Runtime-Fehler auf, wenn
 - der <Vorgabewert> größer als der <Endwert> und der <Schrittwert> positiv ist.
 - der <Vorgabewert> kleiner als der <Endwert> und der <Schrittwert> negativ ist.
- Erfolgt aus einer FOR-NEXT-Programmschleife ein Sprung über die GOTO-Anweisung, verringert sich der Speicherplatz (Stapelspeicher), der für die Programmstruktur zur Verfügung steht. Bei einer kontinuierlichen Ausführung des Programms kann eine Fehlermeldung auftreten. Bauen Sie das Programm so auf, dass die Schleife dann durchlaufen wird, wenn die FOR-Bedingung erfüllt ist.
- Werden FOR- und NEXT-Anweisungen nicht paarweise verwendet, tritt ein Runtime-Fehler auf.
- Steht die NEXT-Anweisung in unmittelbarer Beziehung zur nächsten FOR-Anweisung, können die Variablennamen in der NEXT-Anweisung weggelassen werden. „M2“ in Zeile 50 und „M1“ in Zeile 60 im Programmbeispiel können weggelassen werden. Dies führt zu einer höheren Verarbeitungsgeschwindigkeit.
- Programmebenen
Es ist möglich, FOR-NEXT-Programmschleifen zwischen weiteren FOR-NEXT-Anweisungen zu verwenden. Mit jeder FOR-NEXT-Programmschleife erhöht sich die Zahl der Programmebenen um 1. Ein Programm darf aus maximal 16 Programmebenen bestehen. Bei mehr als 16 Ebenen erfolgt eine Fehlermeldung.
- Schleifen-Verschachtelungen

Beispiel ▾



Richtig!



Falsch!



6.3.34 FPRM (FPRM)

Funktion: Parameter definieren

Legt im Unterprogramm die Reihenfolge, den Typ und die Anzahl von Parametern fest, die von der CALLP-Anweisung eines Hauptprogramms übergeben werden.

Eingabeformat

```
FPRM □ <Formalparameter>[, <Formalparameter>] ...
```

<Formalparameter> Variablenname im Unterprogramm
 Es können alle Variablen verwendet werden.
 Es dürfen maximal 16 Variablen verwendet werden.

Programmbeispiel

```
10 FPRM M1,P1,P2                      Festlegung der Datentypen, der Reihenfolge und
                                         der Anzahl
```

Wenn das Unterprogramm den Namen 20 trägt, so ist die Übergabe der Variablen M4, P3 und P5 vom Hauptprogramm an das Unterprogramm in die Variablen M100, P1 und P30 wie folgt zu definieren:

Hauptprogramm:

```
      :
120 CALLP "P20", M4, P3, P5
      :
```

Unterprogramm:

```
10 FPRM M100, P1, P30
```

Die Variableninhalte von M4 wird an M100, P3 an P1 und P5 an P30 übergeben.

Erläuterung

- Der FPRM-Befehl wird nicht benötigt, wenn im aufgerufenen Unterprogramm keine Parameter verwendet werden.
- Weicht die Anzahl oder der Typ der in der CALLP-Anweisung aufgeführten Formalparameter von denen im FPRM-Befehl ab, erfolgt eine Fehlermeldung.
- Rechenergebnisse eines Unterprogramms können nicht mittels temporärer Parameter in das Hauptprogramm übertragen werden. Verwenden Sie zu diesem Zweck externe Variablen.

Steht in Beziehung zu folgenden Befehlen:

CALLP

6.3.35 GETM (Get Mechanism)

Funktion: Mechanismus definieren

Dieser Befehl definiert einen Mechanismus oder zusätzliche Achsen für den Multitask-Betrieb. Die Zuordnung kann über den RELM-Befehl aufgehoben werden.

Eingabeformat

```
GETM □ <Mechanismusnummer>
```

<Mechanismusnummer> Festlegung der Mechanismusnummer über einen numerischen Wert oder eine Variable
 $1 \leq \text{Mechanismusnummer} \leq 3$
 Der Roboterarm in einem Standardsystem wird als Mechanismus 1 definiert.

Programmbeispiel

Programmplatz 2 wird über den Programmplatz 1 gestartet. Der Mechanismus 1 im Programmplatz 2 wird über Programmplatz 1 gesteuert.

```
10 RELM                Definition von Mechanismus 1 aufheben, um
                       Mechanismus 1 über Programmplatz 2 zu steuern
20 XRUN 2,"10"         Startet Programm 10 als Programmplatz 2
30 WAIT M_RUN(2) = 1   Wartestatus, bis Betriebssignal des Programmplatzes 2
                       gleich 1 ist
:
Programmplatz 2 (Programm 10)
10 GETM 1              Definition des Mechanismus 1
20 SERVO ON           Servospannung des Mechanismus 1 einschalten
30 MOV P1              Position 1 mittels Gelenk-Interpolation anfahren
40 MVS P2              Position 2 mittels Linear-Interpolation anfahren
50 P3 = P_CURR         Aktuelle Position von Mechanismus 1 in P3 übertragen
60 SERVO OFF          Servospannung des Mechanismus 1 ausschalten
70 RELM                Definition von Mechanismus 1 aufheben
80 END                Programmende
```

Erläuterung

- In der Regel, d. h. beim Einzelprogrammplatzbetrieb, wird der Befehl RELM nicht benötigt, da der Betrieb des Mechanismus 1 standardmäßig vordefiniert ist.
- Ein Mechanismus oder eine Zusatzachse kann nicht gleichzeitig über mehrere Programmplätze angesteuert werden.
- Wird beim Multitasking ein Mechanismus oder eine Zusatzachse nicht im Programmplatz 1 betrieben, muss nach Aufhebung der Mechanismusdefinition über den Befehl RELM in Programmplatz 1 eine Zuordnung über den Befehl GETM im Robotersteuerprogramm erfolgen. Es erfolgt eine Fehlermeldung, wenn der GETM-Befehl für einen Mechanismus ausgeführt wird, der bereits über den Befehl definiert worden ist.
- Befehle zum Ein- und Auschalten der Servomotoren, Interpolationsbefehle, Befehle zur Festlegung von Beschleunigungs- und Bremszeiten sowie die Befehle TOOL und BASE können nur nach Definition eines Mechanismus über den GETM-Befehl ausgeführt werden.
- Bei fehlender Angabe des Arguments werden den Roboterstatusvariablen, die eine Mechanismusdefinition benötigen, die Werte des aktuellen Mechanismus zugewiesen.
- Bei einem Programmstopp wird der Befehl RELM automatisch durch das System ausgeführt. Bei einem Neustart des Programms wird der Befehl GETM automatisch ausgeführt.
- Der Befehl kann nicht in einem kontinuierlich ausgeführten Programm verwendet werden.

Steht in Beziehung zu folgenden Befehlen:

RELM

6.3.36 GOSUB (Go Subroutine)

Funktion: Sprung zu einem Unterprogramm

Bewirkt einen Sprung zu einem Unterprogramm, das mit einer festgelegten Zeilennummer oder einer Marke beginnt
 Der Rücksprung muss über die RETURN-Anweisung erfolgen.

Eingabeformat

```
GOSUB □ <Sprungziel>
```

<Sprungziel> Legt eine Zeilennummer oder eine Marke fest

Programmbeispiel

10	GOSUB 300	Sprung zum Unterprogramm (Zeile 300)
20	GOSUB *LABLE	Sprung zum Unterprogramm LABLE
30	MOV P10	Position P10 anfahren
40	MOV P11	Position P11 anfahren
50	GOTO 10	Sprung zur Programmzeile 10
	:	
300	MOV P1	Position P1 anfahren
310	MOV P2	Position P2 anfahren
320	RETURN	Rücksprung zur Zeile 20
	:	
500	*LABLE	Sprungmarke „LABLE“ festgelegt
510	MOV P8	Position P8 anfahren
520	MOV P20	Position P20 anfahren
530	RETURN	Rücksprung zur Zeile 30
	:	

Erläuterung

- Der Rücksprung aus dem Unterprogramm muss mit der RETURN-Anweisung erfolgen. Ein Rücksprung über die GOTO-Anweisung verringert den für die Programmsteuerung zur Verfügung stehenden Speicherplatz (Stapelspeicher) und führt bei kontinuierlicher Ausführung zu einer Fehlermeldung.
- Aus einem Unterprogramm kann über die GOSUB-Anweisung der Sprung zu weiteren Unterprogrammen erfolgen. Dabei ist eine Verschachtelungstiefe von ca. 800 Sprüngen möglich.
- Als Sprungziel kann eine Zeilennummer oder eine Marke angegeben werden. Ist das angegebene Sprungziel nicht vorhanden, erfolgt eine Fehlermeldung.

Steht in Beziehung zu folgenden Befehlen:

RETURN

6.3.37 GOTO (Go To)

Funktion: Sprung zu einer Programmzeile oder Marke

Bewirkt einen unbedingten Sprung zu einer festgelegten Zeilennummer oder Marke.

Eingabeformat

`GOTO □ <Sprungziel>`

<Sprungziel> Legt eine Zeilennummer oder eine Marke fest

Programmbeispiel

100 GOTO 500	Sprung in die Programmzeile 500
:	
500 MOV P1	Position P1 anfahren
:	
700 GOTO *LABLE	Sprung zum Unterprogramm LABLE
:	
900 *LABLE	Sprungmarke „LABLE“ festgelegt

Erläuterung

- Als Sprungziel kann eine Zeilennummer oder eine Marke angegeben werden.
- Ist das angegebene Sprungziel nicht vorhanden, erfolgt eine Fehlermeldung.

6.3.38 HLT (Halt)

Funktion: Programmablauf stoppen

Der Befehl stoppt die Roboterbewegung und den Programmablauf. Das ausgeführte Programm wird in den Standby-Status gesetzt.

Eingabeformat

HLT

Programmbeispiel

Der Roboter wird während der Programmausführung ohne Angabe einer Bedingung gestoppt.

150 HLT	Stoppt den Roboter ohne Bedingung
---------	-----------------------------------

Der Roboter wird während der Programmausführung unter Angabe einer Bedingung gestoppt.

100 IF M_IN(18) = 1 THEN HLT	Stoppt den Roboter beim Einschalten des Eingangssignals 18
200 MOV P1 WTHIF M_IN(17) = 1,HLT	Position 1 anfahren und Programm stoppen, falls das Eingangsbit 17 gleich 1 ist

Erläuterung

- Der Befehl unterbricht den Programmablauf und stoppt den Roboter mit der definierten Abbremszeit.
- Bei Ausführung des HLT-Befehls im Multitask-Betrieb wird nur das Programm in dem Programmplatz unterbrochen, in dem der HLT-Befehl ausgeführt worden ist.
- Ein Neustart kann über die Teaching Box oder durch ein externes Start-Signal erfolgen. Der Programmstart beginnt eine Zeile nach dem HLT-Befehl. Wurde der HLT-Befehl in einer Verknüpfung ausgeführt, startet das Programm in der Zeile, in der es unterbrochen wurde.

Steht in Beziehung zu folgenden Befehlen:

END

6.3.39 HOPEN/HCLOSE (Hand Open/Hand Close)

Funktion: Handgreiferzustand festlegen

Legt den Handgreiferzustand (offen/geschlossen) fest.

Eingabeformat

```
HOPEN □ <Handnummer>[, <Start-Greifkraft>, <Halte-Greifkraft>,
      <Haltezeit für Start-Greifkraft>]
HCLOSE □ <Handnummer>
```

<Handnummer> Festlegung der Handnummer als Konstante oder Variable
 1 ≤ Handnummer ≤ 8

Programmbeispiel

```
10 HOPEN 1    Öffnet Hand 1
20 DLY 0.2    Wartezeit von 0,2 Sekunden ermöglicht ein sicheres Öffnen der Hand
30 HCLOSE 1   Schließt Hand 1
40 DLY 0.2    Wartezeit von 0,2 Sekunden ermöglicht ein sicheres Schließen der
               Hand
50 MOV PUP    Position PUP anfahren
```

Erläuterung

- Die Funktion der Magnetventile (einfach/doppelt) wird in Parameter HANDTYPE festgelegt.
- Bei Einstellung der Funktion auf „doppelt“ können Hand 1 bis Hand 4, bei Einstellung der Funktion auf „einfach“ können Hand 1 bis Hand 8 gesteuert werden.
- Das Handsteuersignal (auf/zu) nach Einschalten der Versorgungsspannung kann über Parameter HANDINIT festgelegt werden.
- Die Roboterstatusvariable M_HNDCQ ermöglicht die Abfrage des Handgreiferstatus. Auf das Signal kann auch über die Eingangssignale Nr. 900 bis 907 (bei Definition eines Mechanismus) zugegriffen werden.

```
10 HCLOSE 1                            Schließt Hand 1
20 IF M_HNDCQ(1) <> 1 GOTO 20        Wartestatus, bis die Roboterstatusvariable
                                     M_HNDCQ(1) gleich 1 ist
30 MOV P1                                Position 1 mittels Gelenk-Interpolation
                                     anfahren
```

- Auf den Handgreifer bezogene Parameter finden Sie im Kap. 9 „Parameter“.

Steht in Beziehung zu folgenden Systemvariablen:

M_IN/M_INB/M_INW (ab Signalnummer 900), M_OUT/M_OUTB/M_OUTW (ab Signalnummer 900), M_HNDCQ

Steht in Beziehung zu folgenden Befehlen:

LOADSET, OADL

Steht in Beziehung zu folgenden Parametern:

HANDTYPE, HANDINIT

6.3.40 IF ... THEN ... ELSE (If Then Else)

Funktion: WENN ... DANN ... SONST

WENN eine bestimmte Bedingung zutrifft, DANN führe Anweisung 1 aus, SONST führe Anweisung 2 aus.

Eingabeformat

```
IF □ <Ausdruck> □ THEN □ <Anweisung> □ [ELSE <Anweisung>]
```

Die Funktion ist ab Software-Version G1 verfügbar. Die BREAK-Anweisung ist ab Software-Version J1 verfügbar

```
IF □ <Ausdruck> □ THEN
  <Anweisung>
  <Anweisung>
  BREAK
  :
[ELSE]
  <Anweisung>
  <Anweisung>
  BREAK
  :
ENDIF
```

<Ausdruck> Beschreibt einen booleschen Ausdruck

<Anweisung> Anweisung nach THEN wird ausgeführt, wenn das Ergebnis des booleschen Ausdrucks „wahr“ ist. Anweisung nach ELSE wird ausgeführt, wenn das Ergebnis des booleschen Ausdrucks „unwahr“ ist.

Programmbeispiele

Gilt für Software-Versionen, die älter als Version G1 sind

100 IF M1 > 10 THEN 1000	Sprung zu Zeile 1000, falls M1 größer 10
110 IF M1 > 10 THEN GOTO 200 ELSE GOTO 300	Sprung zu Zeile 200, falls M1 größer 10, sonst Sprung zu Zeile 300 Die Anweisung GOTO nach THEN oder ELSE kann entfallen
:	
200 M1 = 10	Setzt M1 auf den Wert „10“
210 MOV P1	Position P1 anfahren
220 GOTO 400	Sprung zu Zeile 400
300 M1 = -10	Setzt M1 auf den Wert „-10“
310 MOV P2	Position P2 anfahren
320 GOTO 400	Sprung zu Zeile 400

Gilt für Software-Versionen ab Version G1

```
100 IF M1 > 10 THEN          Setzt M1 auf den Wert „10“, falls M1 größer 10
110   M1 = 10
120   MOV P1                  Position P1 anfahren
130 ELSE                      Setzt M1 auf den Wert „-10“, falls M1 kleiner gleich 10
140   M1 = -10
150   MOV P2                  Position P2 anfahren
160 ENDIF
```

HINWEIS

Bei Software-Versionen, die älter als die Version G1 sind, kann die Befehlszeile auch folgendermaßen aufgebaut werden:

```
250 IF M2 = 0 THEN GOSUB *SUB1 ELSE GOSUB *SUB2
```

Ausführung einer IF-Anweisung innerhalb einer THEN- oder ELSE-Anweisung (gilt für Software-Versionen ab Version G1)

```
300 IF M1 > 10 THEN
310   IF M2 > 20 THEN          Setzt M1 und M2 auf den Wert „10“, falls M1 größer 10
                               und M2 größer 20
320     M1 = 10
330     M2 = 10
340   ELSE                    Setzt M1 und M2 auf den Wert „0“, falls M1 größer 10
                               und M2 kleiner gleich 20
350     M1 = 0
360     M2 = 0
370   ENDIF
380 ELSE                      Setzt M1 und M2 auf den Wert „-10“, falls M1 kleiner
                               gleich 10
390   M1 = -10
400   M2 = -10
410 ENDIF
```

Nach einer THEN- oder ELSE-Anweisung ermöglicht die BREAK-Anweisung einen Sprung in die Zeile, die auf die ENDIF-Anweisung folgt (gilt für Software-Versionen ab Version J1)

```

300 IF M1 > 10 THEN
310   IF M2 > 20 THEN BREAK           Sprung zur Zeile 390, falls M1 größer 10 und
                                       M2 größer 20
320       M1 = 10
330       M2 = 10
340 ELSE
350       M1 = -10
360       IF M2 > 20 THEN BREAK       Sprung zur Zeile 390, falls M1 kleiner gleich 10
                                       und M2 größer 20
370       M2 = -10
380 ENDIF
390 IF M_BRKQC = 1 THEN HLT
400 MOV P1

```

Erläuterung

- Die IF ... THEN ... ELSE-Anweisungen müssen in einer Zeile aufgeführt sein.
- Ein IF ... THEN ... ELSE ... ENDIF-Programmblock kann in mehrere Zeilen aufgeteilt werden.
- Die ELSE-Anweisung kann entfallen.
- Ein IF ... THEN ... ELSE ... ENDIF-Programmblock muss die Anweisung ENDIF enthalten.
- Ein Rücksprung aus einem IF ... THEN ... ELSE ... ENDIF-Programmblock über die GOTO-Anweisung führt zu einer Fehlermeldung, wenn die Speicherplatzkapazität für die Programmsteuerung (Stapelspeicher) überschritten wird.
- Innerhalb eines IF ... THEN ... ELSE ... ENDIF-Programmblöcks können zwischen den Anweisungen IF und ELSE weitere IF ... THEN ... ELSE ... ENDIF-Programmblöcke ausgeführt werden. Dabei ist eine Verschachtelungstiefe von bis zu 8 Programmebenen möglich.
- Nach einer THEN- oder ELSE-Anweisung kann die GOTO-Anweisung entfallen.
 Beispiel: IF M1 > 10 THEN 200 ELSE 300
 Folgt eine GOTO-Anweisung auf eine THEN-Anweisung, kann entweder die GOTO- oder die THEN-Anweisung entfallen.
 Beispiel: IF M1 > 10 THEN GOTO 200 entspricht der Programmzeile
 IF M1 > 10 THEN 200 oder der Programmzeile
 IF M1 > 10 GOTO 200
- Nach einer THEN- oder ELSE-Anweisung ermöglicht die BREAK-Anweisung einen Sprung in die Zeile, die auf die ENDIF-Anweisung folgt (gilt für Software-Versionen ab Version J1).

6.3.41 INPUT # (Input)

Funktion: Eingabe

Mit dieser Anweisung können Daten aus Dateien oder Eingabegeräten im ASCII-Format eingelesen werden. Hinweise zu den entsprechenden Parametern finden Sie im Abschn. 9.16.

Eingabeformat

```
INPUT □ # <Dateinummer>, <Datename> [, <Datename>] ...
```

- <Dateinummer> Legt die Dateinummer fest
1 ≤ Dateinummer ≤ 8
- <Datename> Name der Variablen, in die die Daten übertragen werden
Es können alle Variablen verwendet werden.

Programmbeispiel

```
10 OPEN "COM1:" AS #1            „COM1:“ wird als Datei Nummer 1 geöffnet
20 INPUT #1,M1                    Erfolgt eine Eingabe von der Tastatur, wird dieser Wert
                                  in die numerische Variable M1 übertragen.
30 INPUT #1,CABC$                Erfolgt eine Eingabe von der Tastatur, wird dieser Wert
                                  in die Zeichenkettenvariable CABC$ übertragen.
                                  :
100 CLOSE #1                      Datei Nummer 1 schließen
```

Erläuterung

- Überträgt Eingangsdaten aus Dateien (oder von Eingabegeräten), die mittels OPEN-Anweisung geöffnet worden sind, in eine Variable. Ist die OPEN-Anweisung nicht ausgeführt worden, erfolgt eine Fehlermeldung.
- Der übertragene Datentyp und der Variablentyp müssen übereinstimmen.
- Werden mehrere Variablenamen angegeben, müssen sie durch Kommas getrennt werden.
- Bei Ausführung der INPUT-Anweisung wartet das System auf eine Eingabe. Bei Betätigung der Eingabetaste (CR und LF) werden die Eingangsdaten in die Variablen übertragen.
- Beim Senden von Werten an den Roboter ist vor den Daten die Buchstabenfolge PRN zu setzen.
Beispiel: PRN 50 für die Übergabe des Wertes 50
- Bei Eingabe mehrerer Elemente werden die Elemente der Reihe nach übertragen.
Beispiel:
Bei Eingabe einer Zeichenkette, eines numerischen Wertes und einer Position
10 INPUT #1,C1\$,M1,P1
PRN MELFA,125.75,(130.5,-117.2,55.1,16.2,0,0)(1,0) CR
„MELFA“ wird in C1\$ übertragen,
125.75 in M1 und
(130.5,-117.2,55.1,16.2,0,0)(1,0) in P1
Ist die Anzahl der Elemente größer als in der INPUT-Anweisung angegeben, werden die überzähligen Elemente nicht eingelesen.

Steht in Beziehung zu folgenden Befehlen:

OPEN, CLOSE, PRINT

6.3.42 JOVRD (J Override)

Funktion: Übersteuerung

Legt die Geschwindigkeit für die Gelenk-Interpolation fest

Eingabeformat

```
JOVRD □ <Übersteuerungswert>
```

<Übersteuerungswert> Legt den prozentualen Übersteuerungswert als reelle Zahl oder als numerischen Ausdruck fest
 $1 \leq \text{Übersteuerungswert} \leq 100.0$

Programmbeispiel

10	JOVRD 50	Übersteuerung auf den Wert 50 % einstellen
20	MOV P1	Position P1 anfahren
30	JOVRD M_NJOVRD	Standardwert einstellen

Erläuterung

- Der JOVRD-Befehl ist nur bei der Gelenk-Interpolation wirksam.
- Die aktuelle Arbeitsgeschwindigkeit ergibt sich folgendermaßen:

$$\text{Gelenk-Interpolation} = \frac{\text{Übersteuerungswert der T/B- oder des Steuergeräts}}{100} \times \text{Einstellwert des OVRD-Befehls} \times \text{Einstellwert des JOVRD-Befehls}$$

- Der Maximalwert der Arbeitsgeschwindigkeit ist 100 %. Der Standardwert der Arbeitsgeschwindigkeit beträgt 100 % der Standardeinstellung (M_NOVRD).
- Die Arbeitsgeschwindigkeit wird bei Ausführung der END-Anweisung und bei einem Reset auf den Standardwert zurückgesetzt.

Steht in Beziehung zu folgenden Systemvariablen:

M_JOVRD/M_NJOVRD/M_OPOVRD/M_OVRD/M_NOVRD
M_NJOVRD (Standardeinstellung), M_JOVRD (aktuelle Einstellung)

Steht in Beziehung zu folgenden Befehlen:

OVRD, SPD

6.3.43 JRC (Joint Roll Change)

Funktion: Gelenkposition verändern

Roboterarmachse:

Überschreibt die aktuelle Position durch Addition von $\pm 360^\circ$ zur aktuellen Gelenkposition der entsprechenden Achse (siehe Eingabeformat unter „Achsennummer“ unten).

Benutzerdefinierte Achse:

Überschreibt die aktuelle Position durch Addition oder Subtraktion eines festgelegten Wertes zur aktuellen Gelenkposition der gewählten Achse. Der Anwender kann den Wert in einem Parameter festlegen. Als Achse kann sowohl eine Gelenkachse als auch eine lineare Achse gewählt werden.

Die Grundposition eines Gelenkes kann verändert werden.

Eingabeformat

```
JRC □ <[+]1/-1/0> [, <Achsennummer>]
```

Ab Software-Version J1:

```
JRC □ <[+]<Numerischer Wert>/-<Numerischer Wert>/0>
      [, <Achsennummer>]
```

- <+1> Addiert einen definierten Wert zur aktuellen Position einer gewählten Gelenkachse
Der Wert, um den die Position verändert wird, ist im Parameter JRCQTT festgelegt. Bei der Standardachse ist dieser Wert auf 360° festgelegt.
 - <-1> Subtrahiert einen definierten Wert von der aktuellen Position einer gewählten Gelenkachse
Der Wert, um den die Position verändert wird, ist im Parameter JRCQTT festgelegt. Bei der Standardachse ist dieser Wert auf 360° festgelegt.
 - <0> Der aktuelle Wert einer festgelegten Achse wird als Grundposition in den Parameter JRCORG geschrieben. Dieser Befehl kann nur bei einer benutzerdefinierten Achse ausgeführt werden.
 - <Achsennummer> Die Zielachse wird durch Jx angegeben, wobei x eine Zahl zwischen 1 und 8 ist. Erfolgt keine Angabe, werden die Standardachsen verwendet.

 Robotertypen und Standardachsen:
 RV-A und RV-S: J6-Achse
 RH-A, RH-S und RP-A: J4-Achse

 Es können bei allen Robotertypen benutzerdefinierte Zusatzachsen als Zielachse definiert werden.

 Es können alle Achsen benutzerdefinierter Mechanismen als Zielachse definiert werden.
- Ab Software-Version J1:
- <Numerischer Wert> Legt den Wert ganzer Umdrehungen fest, die zu der aktuellen Position einer Gelenkachse addiert werden
 Beispiel: +3: Zunahme des Winkels der Zielachse um 1080°
 -2: Abnahme des Winkels der Zielachse um 720°

Programmbeispiel

10	MOV P1	Position 1 anfahren
20	JRC 1	Addition von 360° zur aktuellen Position der J6-Achse
30	MOV P1	Position 1 anfahren

Ab Software-Version J1:

10	MOV P1	Position 1 anfahren (Ausgangsposition)
20	JRC +1	Addition von 360° zur aktuellen Position der J6-Achse
30	MOV P1	Position 1 anfahren
40	JRC +1	Addition von 360° zur aktuellen Position der J6-Achse
50	MOV P1	Position 1 anfahren
60	JRC -2	Subtraktion von 720° von der aktuellen Position der J6-Achse (Rückkehr zur Ausgangsposition)

Erläuterung

- JRC = 1/-1 (JRC n/-n) erhöht/verringert den aktuellen Gelenkwinkel um einen festgelegten Wert.
- Über JRC = 0 wird die Grundposition einer festgelegten Achse mit der aktuellen Position überschrieben.
- Verwenden Sie den JRC-Befehl, muss im Voraus der Bewegungsbereich der Zielachse verändert werden. Dies ist notwendig, damit der Roboter bei Ausführung des Befehls den Bewegungsbereich nicht verlässt. Im Parameter MEJAR verändern Sie dazu die Minus- und Pluswerte. Der Bewegungsbereich der Drehachsen liegt zwischen -2340° und 2340°.
- Wird keine Achse festgelegt, wird als Zielachse automatisch die Standardachse eingestellt. Dies ist bei allen Robotern die letzte Rotationsachse.
- Existiert die festgelegte Achse bei dem verwendeten Roboter nicht, oder ist sie im Sinne des JRC-Befehls keine Zielachse, wird bei Ausführung des JRC-Befehls ein Fehler angezeigt.
- Ist keine Grundposition definiert, erfolgt eine Fehlermeldung, sobald der JRC-Befehl ausgeführt wird.
- Aufgrund des JRC-Befehls stoppt der Roboter. Bei Freigabe des CNT-Befehls und Verwendung des JRC-Befehls ist die Bewegung nicht mehr interpoliert.
- Bevor Sie den JRC-Befehl verwenden, müssen die folgenden Parameter eingestellt sein:
 - JRCEXE = 1 (JRC-Ausführung freigegeben)
 - Änderung des Bewegungsbereiches der Zielachse über MEJAR
 - Einstellung des Wertes in JRCQTT (nur bei benutzerdefinierten Achsen möglich)
 - Einstellung der Grundposition über JRCORG
- Bei JRCEXE = 0 kann der JRC-Befehl nicht ausgeführt werden.
- Die Bewegung ist über den Parameter JRCQTT festgelegt. Liegt dieser Parameter nicht innerhalb der Pulsdaten 0-MAX, wird bei der Initialisierung ein Fehler angezeigt. Der MAX-Wert ist definiert durch: $MAX = 2^{(\text{Anzahl der Encoder-Bits} + 15)} - 1$.

Beispiel ▾

Ein 13-Bit-Encoder (8192 Pulse): $MAX = 2^{(13 + 15)} - 1$
 $MAX = 0x0ffffff$
 Ein 14-Bit-Encoder (16384 Pulse): $MAX = 2^{(14 + 15)} - 1$
 $MAX = 0x1ffffff$



Pulsdaten werden in Bewegungsdaten konvertiert.

Für Drehachsen:

Pulsdaten = Bewegung(°/360) × Anzahl der Encoder-Pulse

Für lineare Achsen:

Pulsdaten = Bewegung (mm/360) × Anzahl der Encoder-Pulse

- Muss das Steuergerät während eines Updates initialisiert werden, speichern Sie vorher die Parameter im Grundzustand.
- Das schrittweise Ausführen (rückwärts) des Programms ist mit dem JRC-Befehl nicht möglich.
- Die Anweisung kann in einem kontinuierlich ausgeführtem Programm nicht verwendet werden.

Einsatzbereich

- RV-1A/2AJ, RV-4A/5AJ und vergleichbare Modelle, J6-Achse
- Benutzerdefinierte Achsen

Steht in Beziehung zu folgenden Parametern:

JRCEXE	Freigabe des JRC-Befehls (freigegeben/gesperrt = 1/0) Standardeinstellung JRCEXE = 0
JRCQTT	Festlegung der Werte, um die die festgelegten Achsen mit dem JRC-Befehl verschoben werden. Dieser Wert ist bei der Standardachse auf 360° festgelegt.
JRCORG	Festlegung der Grundposition für JRC = 0 (kann nur bei zusätzlichen und benutzerdefinierten Achsen eingestellt werden)

6.3.44 LABEL (Label)

Funktion: Sprungmarke

Legt ein Sprungziel fest

Eingabeformat

```
*<Name der Marke>
```

Ab Software-Version J1:

```
*<Name der Marke>[:<Befehlszeile>]
```

<Name der Marke>	Legt den Namen der Marke über eine Zeichenkette fest Das erste Zeichen muss ein Buchstabe sein. Die maximale Länge beträgt 8 Zeichen (Das (*)-Zeichen wird nicht mitgezählt.)
<Befehlszeile>	Legt eine Befehlszeile hinter dem Markennamen nach dem Doppelpunkt fest

Programmbeispiel

100 *SUB1	Sprungmarke „SUB1“ festgelegt
200 IF M1 = 1 THEN GOTO *SUB1	Sprung zum Unterprogramm SUB1, wenn M1 gleich 1

Ab Software-Version J1:

300 *LBL1: IF M_IN(19) = 0 THEN GOTO *LBL1	Durchläuft die Warteschleife in Zeile 300 bis M_IN(19) auf „1“ gesetzt wird.
--------------------------------------------	------------------------------------------------------------------------------------

Erläuterung

- Es erfolgt keine Fehlermeldung, wenn die Marke während eines Programmablaufes nicht aufgerufen wird.
- Ist die gleiche Marke in einem Programm mehrmals definiert, erfolgt eine Fehlermeldung.
- Reservierte Wörter dürfen nicht als Markennamen vergeben werden.
- Ein Markenname, in dem der Unterstrich verwendet wird, muss mit dem Zeichen „L“ beginnen (z. B. L_ABC, L12_345, *LABEL_1). Beginnt der Name mit einem anderen Zeichen (z. B. *A_LABEL) erfolgt eine Fehlermeldung. Fehlerhaft wären auch Markennamen wie *H_ABC, *ABC_123, *NG_ oder *_LABEL.
- Ab Software-Version J1 kann auf die Markendefinition und nach einem Doppelpunkt eine Befehlszeile folgen. Das Anhängen weiterer Befehle ist in dieser Zeile jedoch nicht möglich.

6.3.45 LOADSET (Load Set)

Funktion: Hand- und Werkstückbedingung einstellen

Legt die Hand- und Werkstückbedingungen für eine optimale Beschleunigung/Abbremsung (OADL) fest

Eingabeformat

LOADSET <Handnummer>, <Werkstücknummer>

- | | |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <Handnummer> | Legt die Hand für die Einstellung der Größe und des Gewichts für die optimale Beschleunigung/Abbremsung fest
($1 \leq \text{HNDDAT} \leq 8$)
Bei den Robotermodellen RV-S/RH-S ist auch eine Einstellung auf „0“ möglich (HNDDAT0). |
| <Werkstücknummer> | Legt das Werkstück für die Einstellung der Größe und des Gewichts für die optimale Beschleunigung/Abbremsung fest
($1 \leq \text{WRKDAT} \leq 8$)
Bei den Robotermodellen RV-S/RH-S ist auch eine Einstellung auf „0“ möglich (WRKDAT0). |

Programmbeispiel

- | | | |
|----|-------------|----------------------------------------------------------------------------------------------|
| 10 | LOADSET 1,1 | Optimale Beschleunigung/Abbremsung für Hand 1 (HNDDAT1) und Werkstück 1 (WRKDAT1) einstellen |
| 20 | OADL ON | Optimale Beschleunigung/Abbremsung freigeben |
| 30 | MOV P1 | Position 1 mittels Gelenk-Interpolation und optimaler Beschleunigung/Abbremsung anfahren |
| 40 | MOV P2 | Position 2 mittels Gelenk-Interpolation und optimaler Beschleunigung/Abbremsung anfahren |
| 50 | LOADSET 1,2 | Optimale Beschleunigung/Abbremsung für Hand 1 (HNDDAT1) und Werkstück 2 (WRKDAT2) einstellen |
| 60 | MOV P1 | Position 1 mittels Gelenk-Interpolation und optimaler Beschleunigung/Abbremsung anfahren |
| 70 | MOV P2 | Position 2 mittels Gelenk-Interpolation und optimaler Beschleunigung/Abbremsung anfahren |
| 80 | OADL OFF | Optimale Beschleunigung/Abbremsung sperren |

Für Robotermodelle RV-S/RH-S

- | | | |
|----|-------------|----------------------------------------------------------------------------------------------|
| 10 | LOADSET 1,1 | Optimale Beschleunigung/Abbremsung für Hand 1 (HNDDAT1) und Werkstück 1 (WRKDAT1) einstellen |
| 20 | OADL ON | Optimale Beschleunigung/Abbremsung freigeben |
| 30 | MOV P1 | Position 1 mittels Gelenk-Interpolation und optimaler Beschleunigung/Abbremsung anfahren |
| 40 | LOADSET 0,0 | Optimale Beschleunigung/Abbremsung für Hand 0 (HNDDAT0) und Werkstück 0 (WRKDAT0) einstellen |
| 50 | MOV P2 | Position 2 mittels Gelenk-Interpolation und optimaler Beschleunigung/Abbremsung anfahren |
| 60 | OADL OFF | Optimale Beschleunigung/Abbremsung sperren |

Erläuterung

- Die Funktion ermöglicht die Ausführung der Roboterbewegung für verschiedene Handdaten und Werkstücke mit optimaler Beschleunigung/Abbremsung.
- Beim Programmstart wird für die Hand die maximale Last gesetzt.
- Werden mehrere Variablennamen angegeben, müssen sie durch Kommas getrennt werden.
- Stellen Sie das Gewicht, die Maße (X, Y, Z) und den Schwerpunkt (X, Y, Z) der Hand in Parameter HANDDAT (HANDDAT 1 bis 8) ein.
- Stellen Sie das Gewicht, die Maße (X, Y, Z) und den Schwerpunkt (X, Y, Z) des Werkstücks in Parameter WRKDAT (WRKDAT 1 bis 8) ein.
- Die Hand- und Werkzeugbedingungen für die optimale Beschleunigung/Abbremsung werden beim Zurücksetzen des Programms und bei Ausführung der END-Anweisung auf die Standardwerte zurückgesetzt.
- Als Standardwert wird die Handbedingung auf den Lastnennwert und die Werkzeugbedingung auf „keine“ (0 kg) gesetzt.
- Bei den Robotermodellen RV-S/RH-S können die Standardwerte des Systems HNDDAT0, WRKDAT0 und HNDHOLD0 verändert werden.
- Weitere Hinweise zur Einstellung der optimalen Beschleunigung/Abbremsung finden Sie in Abschn. 9.17.1 „Optimale Beschleunigung/Abbremsung“ und Kap. 9 unter dem Parameter ACCMODE.

Steht in Beziehung zu folgenden Befehlen:

OADL, HOPEN/HCLOSE

Steht in Beziehung zu folgenden Parametern:

HNDDAT1 bis 8, WRKDAT1 bis 8, HNDHOLD1 bis 8

6.3.46 MOV (Move)

Funktion: Bewegung mit Gelenk-Interpolation

Bewegt die Handspitze mittels Gelenk-Interpolation zu einer festgelegten Position

Eingabeformat

```
MOV □ <Zielposition> [, <Abstand>]
      [□ TYPE □ <Konstante 1>, <Konstante 2>] □
      [<Verknüpfungsbedingung>]
```

- <Zielposition> Legt die Zielposition als Positionskonstante, -variable oder Gelenkvariable fest
- <Abstand> Legt den Verfahrenswegbetrag in Werkzeugrichtung auf der Z-Achse fest (Abstand zur Zielposition)
- TYPE
- <Konstante 1> Indirekte/direkte Anfahrt = 1/0 (Standardwert: 1)
- <Konstante 2> Durchfahren des singulären Punktes (0 = gesperrt, 1 = freigegeben)
- <Verknüpfungsbedingung> Es können die Verknüpfungen WTH und WTHIF verwendet werden.

Programmbeispiel

10	MOV P1 TYPE 1,0	Position 1 anfahren
20	MOV P2	Position 2 anfahren
30	MOV (PLT 1,10),100.0 WTH M_OUT(17) = 1	Palette 1 anfahren und Ausgangsbit 17 auf „1“ setzen
40	MOV P4+P5,50.0 WTHIF M_IN(18)=1,M_OUT(20) = 1	Position (P4+P5) anfahren, wenn Eingangsbit 18 gleich 1 ist, setze Ausgangsbit 20 auf „1“

Erläuterung

- Die Gelenkwinkel jeder Achse werden am Start- und Endpunkt gleichmäßig interpoliert. Der Weg der Handspitze kann daher nicht exakt vorhergesagt werden.

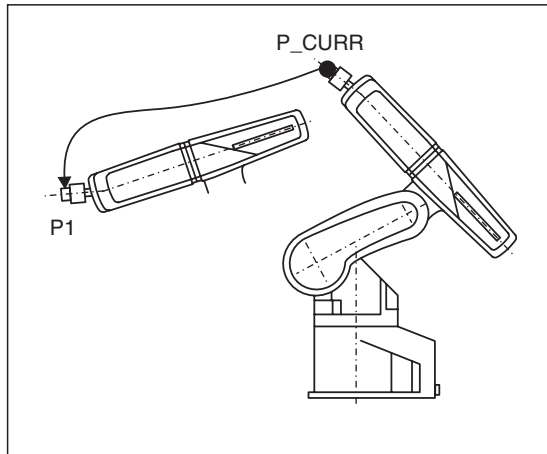


Abb. 6-12:
Verfahrweg bei Gelenk-Interpolation

R000885C

- Durch die Verknüpfungsbedingungen über die WTH- und WTHIF-Anweisung können die Verfahrbewegung und die Signalausgabe synchronisiert werden.
- Über die numerische Konstante 1 zur Festlegung des Interpolationstyps wird die Interpolation der Stellung definiert.
- Bei der indirekten Anfahrt erfolgt die Anfahrt der Position exakt mit der geteachten Stellung. In Abhängigkeit der geteachten Stellung kann die direkte Anfahrt gewählt werden.
- Bei der direkten Anfahrt wird die Stellung beim Start bis zur Stellung bei Erreichen der Zielposition mit weniger Bewegungen geändert.
- Die Auswahl zwischen direkter und indirekter Anfahrt ist bei einem Bewegungsbereich der Stellungsachse von 180° oder mehr von Bedeutung.
- Liegt die Zielposition bei angewählter direkter Anfahrt außerhalb des Bewegungsbereiches, ist es möglich, dass die Achse in umgekehrter Richtung mit indirekter Anfahrt bewegt wird.
- Die numerische Konstante 2 ist für die Gelenk-Interpolation bedeutungslos.
- Dieser Befehl kann in einem kontinuierlich ausgeführten Programm nicht verwendet werden.
- Wird die Ausführung eines MOV-Befehls unterbrochen und nach anschließendem JOG-Betrieb wieder fortgesetzt, kehrt der Roboter zu der Position der Unterbrechung zurück und führt die Verfahrbewegung fort. Die Auswahl der Interpolationsart (Gelenk- oder XYZ-Interpolation) zur Anfahrt der Position der Unterbrechung kann über den Parameter RETPATH eingestellt werden.
Weiterhin erlaubt die Funktion die Fortsetzung des Verfahrwegs von der aktuellen zur Zielposition, ohne Anfahrt der Position, bei der die Verfahrbewegung unterbrochen wurde (siehe auch Abschn. 9.11).

6.3.47 MVA (Move Arch)

Funktion: Bewegung mit Bogen-Interpolation

Bewegt die Handspitze mittels Bogen-Interpolation zu einer festgelegten Position

Eingabeformat

Ab Software-Version G2:

```
MVA □ <Zielposition> [, <Bogennummer>]
```

- <Zielposition> Legt die Zielposition als Positionskonstante oder -variable oder Gelenkvariable fest
- <Bogennummer> Die Bogennummer entspricht dem mit dem Befehl DEF ARCH definierten Bogen
 $1 \leq \text{Bogennummer} \leq 4$
 Fehlt die Angabe, wird der Wert auf „1“ gesetzt.

Programmbeispiel

- 10 DEF ARCH 1,5,5,20,20 Bogen definieren
- 20 OVRD 100,20,20 Übersteuerung festlegen
- 30 ACCEL 100,100,50,50,50,50 Beschleunigung/Abbremsung festlegen
- 40 MVA P1,1 Position P1 mittels Bogen-Interpolation über den in Zeile 10 definierten Bogen anfahren
- 50 MVA P2,2 Position P2 mittels Bogen-Interpolation über den mit den Standardeinstellungen definierten Bogen anfahren

Erläuterung

- Die Verfahrbewegung erfolgt von der Startposition entlang der Z-Achse an aufwärts, dann zu einer Position über der Zielposition und anschließend wieder nach unten zur Zielposition. Die Ausführung der bogenförmigen Verfahrbewegung erfolgt über eine einzelne Anweisung.

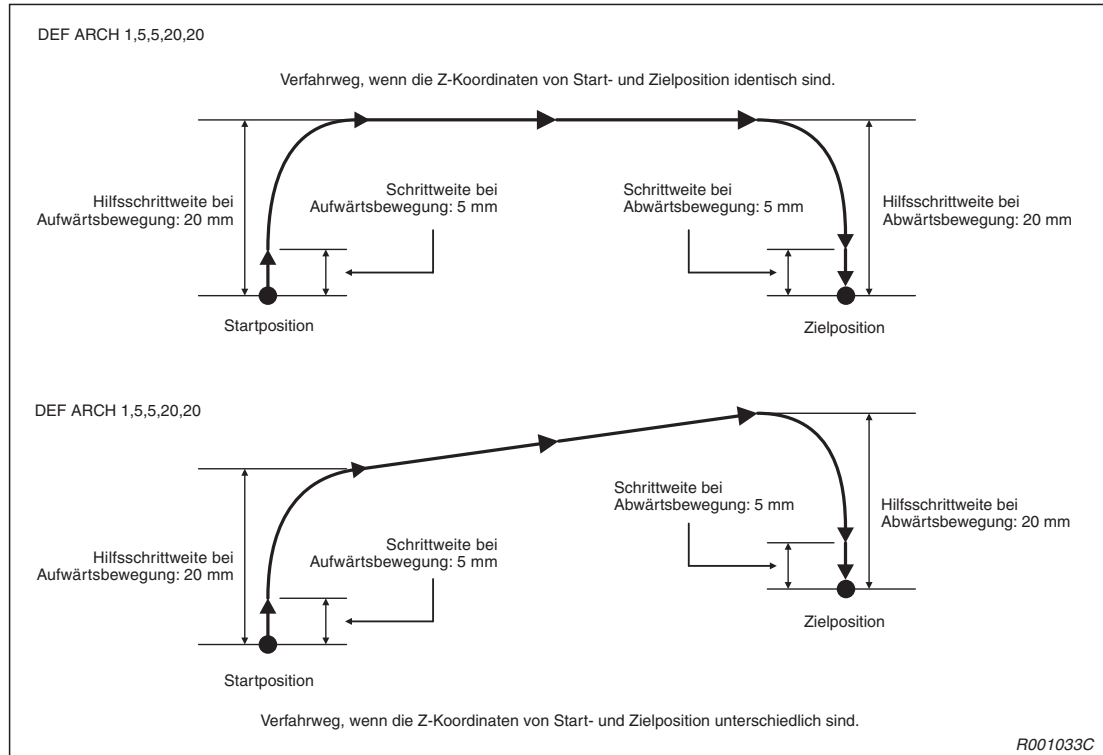


Abb. 6-13: Beispiele zur Bogen-Interpolation

- Wird der Befehl MVA ohne vorherige Definition des Bogens über den DEF ARCH-Befehl ausgeführt, erfolgt die Verfahrbewegung entsprechend dem über die Parameter festgelegten Bogen (siehe auch Abschn. 6.3.20 „DEF-ARCH“).
- Der Befehl kann nicht in einem kontinuierlich ausgeführtem Programm verwendet werden.
- Wird die Ausführung eines MVA-Befehls unterbrochen und nach anschließendem JOG-Betrieb wieder fortgesetzt, kehrt der Roboter zu der Position der Unterbrechung zurück und führt die Verfahrbewegung fort. Eine Änderung dieser Funktion sowie die Auswahl der Interpolationsart (Gelenk- oder XYZ-Interpolation) zur Anfahrt der Position der Unterbrechung kann über den Parameter RETPATH eingestellt werden (siehe auch Abschn. 9.11).

Steht in Beziehung zu folgenden Befehlen:

DEF ARCH, ACCEL, OVRD

Erläuterung

- Mittels Kreis-Interpolation bewegt sich die Handspitze des Roboters auf dem Kreisumfang des durch die 3 Punkte festgelegten Kreises (360°).

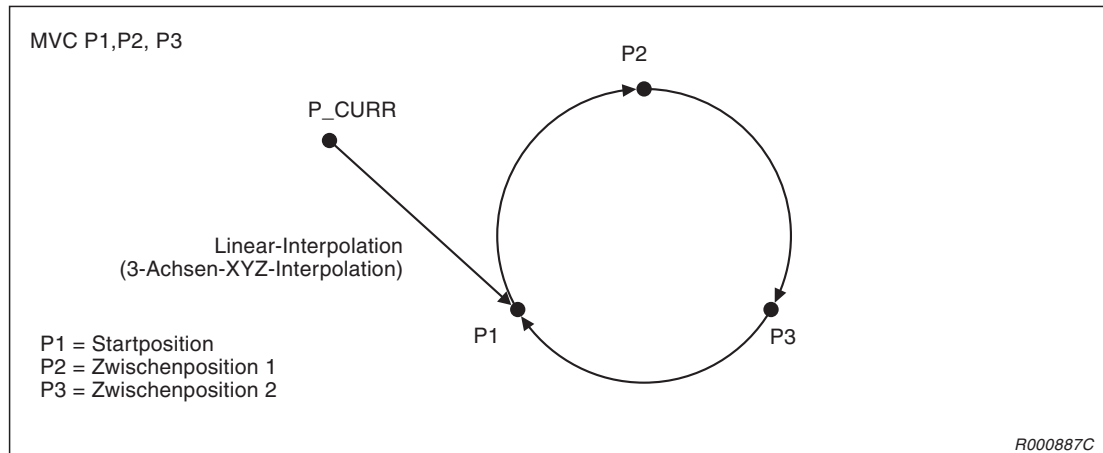


Abb. 6-14: Beispiel zur Kreis-Interpolation über zwei Zwischenpositionen

- Während der Kreis-Interpolation bleibt die Orientierung des Roboters im Startpunkt unverändert. Die Orientierungen bei Zwischenposition 1 und Zwischenposition 2 sind nicht definiert.
- Entspricht die momentane Position nicht der Startposition, fährt der Roboter die Startposition mittels Linear-Interpolation (3-Achsen-XYZ-Interpolation) an und führt anschließend die Kreis-Interpolation aus.
- Wird die Ausführung eines MVC-Befehls unterbrochen und nach anschließendem JOG-Betrieb wieder fortgesetzt, bewegt sich der Roboter mittels Gelenk-Interpolation zu der Position an der die Bewegung unterbrochen wurde und setzt dort die Kreisbogenbewegung fort. Die Auswahl der Interpolationsart (Gelenk- oder XYZ-Interpolation) zur Anfahrt der Position der Unterbrechung kann über den Parameter RETPATH eingestellt werden (siehe auch Abschn. 9.11).
- Der Befehl kann nicht in einem kontinuierlich ausgeführtem Programm verwendet werden.

Erläuterung

- Mittels Kreis-Interpolation bewegt sich der Roboterarm auf dem Kreisbogen, der durch die 3 Punkte festgelegt ist.

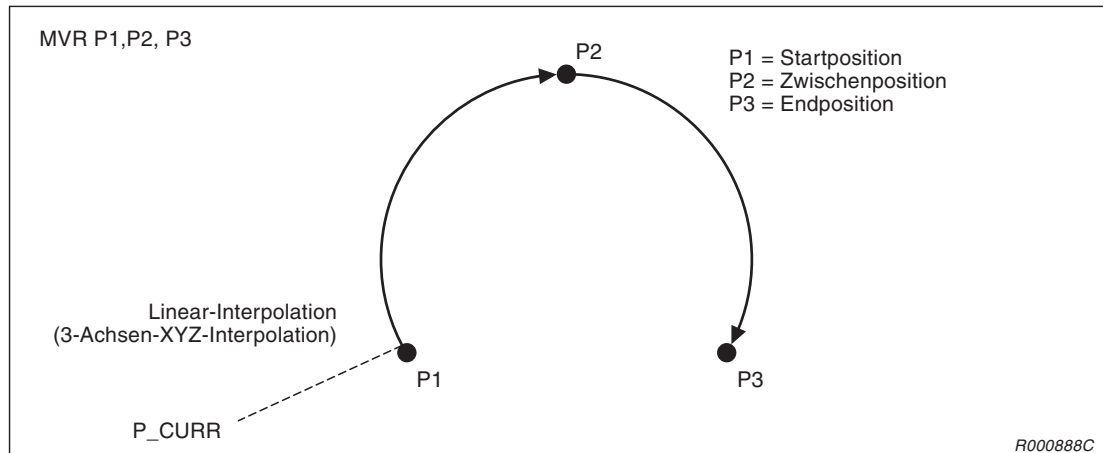


Abb. 6-15: Beispiel zur Kreis-Interpolation über eine Zwischenposition

- Die Roboterstellung wird vom Startpunkt zum Endpunkt interpoliert. Die Stellung der Zwischenposition hat keinen Einfluss.
- Entspricht die aktuelle Position nicht der Startposition, fährt der Roboter die Startposition mittels Linear-Interpolation an.
- Wird die Ausführung eines MVR-Befehls unterbrochen und nach anschließendem JOG-Betrieb wieder fortgesetzt, bewegt sich der Roboter mittels Gelenk-Interpolation zu der Position an der die Bewegung unterbrochen wurde und setzt dort die Kreisbogenbewegung fort. Die Auswahl der Interpolationsart (Gelenk- oder XYZ-Interpolation) zur Anfahrt der Position der Unterbrechung kann über den Parameter RETPATH eingestellt werden (siehe auch Abschn. 9.11).
- Weichen die Stellungsmerker der Start- und Endposition für eine andere Interpolationsmethode als die kartesische Interpolation für 3 Achsen voneinander ab, erfolgt eine Fehlermeldung.
- Der Roboter verfährt mit Linear-Interpolation, wenn zwei der drei Positionen gleich sind oder alle Positionen auf einer Geraden liegen. Es erfolgt keine Fehlermeldung.
- Wird über die numerische Konstante 2 die 3-Achsen-XYZ-Interpolation gewählt, ist die numerische Konstante 1 unwirksam und der Roboter bewegt sich mit der geteachten Orientierung.
- Die numerische Konstante 2 legt den Interpolationstyp der Stellung fest. Bei einer Interpolation im Koordinatensystem X, Y, Z, J4, J5 und J6 wird die 3-Achsen-XYZ-Interpolation verwendet, um den Roboter in die Nähe eines bestimmten Punktes zu bewegen.
- Dieser Befehl kann nicht in einem kontinuierlich ausgeführtem Programm verwendet werden.

6.3.50 MVR2 (Move R2)

Funktion: Kreis-Interpolation

Bewegt die Handspitze mittels 3D-Kreis-Interpolation von der Startposition zur Endposition. Der Kreisbogen wird durch die Startposition, die Referenzposition und die Endposition festgelegt. Die Roboterbewegung geht dabei nicht durch den Referenzpunkt.

Eingabeformat

```
MVR2 □ <Startposition>, <Endposition>, <Referenzposition>
      [□ TYPE □ <Konstante 1>, <Konstante 2>] □
      [<Verknüpfungsbedingung>]
```

- <Startposition> Legt den Startpunkt des Kreises fest
- <Endposition> Legt die Endposition auf dem Kreisumfang fest
- <Referenzposition> Legt die Referenzposition auf dem Kreisumfang fest
- TYPE
- <Konstante 1> 1/0 = indirekte/direkte Anfahrt (Standardwert: 0)
- <Konstante 2> Durchfahren des singulären Punktes
(0 = gesperrt, 1 = freigegeben)
- <Verknüpfungsbedingung> Es können die Verknüpfungen WTH und WTHIF verwendet werden.

Programmbeispiel

- 10 MVR2 P1,P2,P3 Bewegung auf dem durch P1, P2 und P3 festgelegten Kreisbogen
- 20 MVR2 P1,J2,P3 Bewegung auf dem durch P1, J2 und P3 festgelegten Kreisbogen
- 30 MVR2 P1,P2,P3 WTH M_OUT(17) = 1 Bewegung auf dem durch P1, P2 und P3 festgelegten Kreisbogen und Setzen des Ausgangsbits 17 auf „1“
- 40 MVR2 P3,(PLT 1,5),P4 WTHIF M_IN(20) = 1,M_OUT(21) = 1 Bewegung auf dem durch P3, PLT1,5 und P4 festgelegten Kreisbogen und Setzen des Ausgangsbits 21 auf „1“, wenn Eingangsbit 20 gleich 1 ist

Erläuterung

- Mittels Kreis-Interpolation bewegt sich die Roboterhand auf dem Kreisbogen, der durch die 3 Punkte festgelegt ist. Die Bewegung geht nicht durch die Referenzposition.

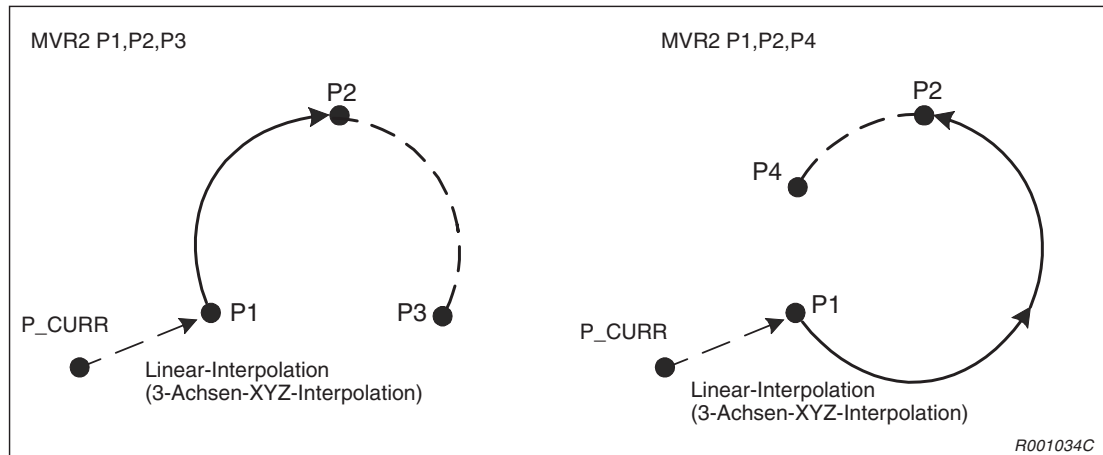


Abb. 6-16: Beispiel zur Kreis-Interpolation über einen Referenzpunkt

- Die Roboterstellung wird vom Startpunkt zum Endpunkt interpoliert. Die Stellung des Referenzpunktes hat keinen Einfluss.
- Entspricht die aktuelle Position nicht der Startposition, fährt der Roboter automatisch die Startposition mittels Linear-Interpolation an.
- Wird die Ausführung eines MVR2-Befehls unterbrochen und nach anschließendem JOG-Betrieb wieder fortgesetzt, bewegt sich der Roboter mittels Gelenk-Interpolation zu der Position an der die Bewegung unterbrochen wurde und setzt dort die Kreisbogenbewegung fort. Die Auswahl der Interpolationsart (Gelenk- oder XYZ-Interpolation) zur Anfahrt der Position der Unterbrechung kann über den Parameter RETPATH eingestellt werden (siehe auch Abschn. 9.11).
- Der Roboter bewegt sich in die Richtung entlang des Kreisbogens, die nicht durch die Referenzposition geht.
- Weichen die Stellungenmerker der Start- und Endposition für eine andere Interpolationsmethode als die kartesische Interpolation für 3 Achsen voneinander ab, erfolgt eine Fehlermeldung.
- Der Roboter verfährt mit Linear-Interpolation, wenn zwei der drei Positionen gleich sind oder alle Positionen auf einer Geraden liegen. Es erfolgt keine Fehlermeldung.
- Wird über die numerische Konstante 2 die 3-Achsen-XYZ-Interpolation gewählt, ist die numerische Konstante 1 unwirksam und der Roboter bewegt sich mit der geteachten Orientierung.
- Die numerische Konstante 2 legt den Interpolationstyp der Stellung fest. Bei einer Interpolation im Koordinatensystem X, Y, Z, J4, J5 und J6 wird die 3-Achsen-XYZ-Interpolation verwendet, um den Roboter in die Nähe eines bestimmten Punktes zu bewegen.
- Der Befehl kann nicht in einem kontinuierlich ausgeführtem Programm verwendet werden.

Erläuterung

- Mittels Kreis-Interpolation bewegt sich die Roboterhand auf dem Kreisbogen, der durch die 3 Punkte festgelegt ist.

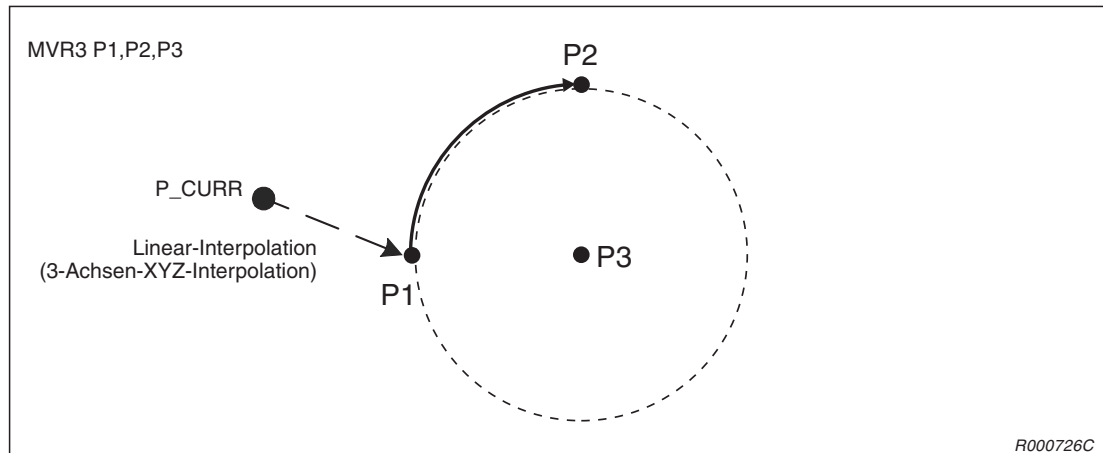


Abb. 6-17: Beispiel zur Kreis-Interpolation über einen Mittelpunkt

- Die Roboterstellung wird vom Startpunkt zum Endpunkt interpoliert. Die Stellung des Referenzpunktes hat keinen Einfluss.
- Entspricht die aktuelle Position nicht der Startposition, fährt der Roboter automatisch die Startposition mittels Linear-Interpolation an.
- Wird die Ausführung eines MVR3-Befehls unterbrochen und nach anschließendem JOG-Betrieb wieder fortgesetzt, bewegt sich der Roboter mittels Gelenk-Interpolation zu der Position an der die Bewegung unterbrochen wurde und setzt dort die Kreisbogenbewegung fort. Die Auswahl der Interpolationsart (Gelenk- oder XYZ-Interpolation) zur Anfahrt der Position der Unterbrechung kann über den Parameter RETPATH eingestellt werden (siehe auch Abschn. 9.11).
- Weichen die Stellungsmerker der Start- und Endposition für eine andere Interpolationsmethode als die 3-Achsen-XYZ-Interpolation voneinander ab, erfolgt eine Fehlermeldung.
- Wird über die numerische Konstante 2 die 3-Achsen-XYZ-Interpolation gewählt, ist die numerische Konstante 1 unwirksam und der Roboter bewegt sich mit der geteachten Orientierung.
- Die numerische Konstante 2 legt den Interpolationstyp der Stellung fest. Bei einer Interpolation im Koordinatensystem X, Y, Z, J4, J5 und J6 wird die 3-Achsen-XYZ-Interpolation verwendet, um den Roboter in die Nähe eines bestimmten Punktes zu bewegen.
- Der Zentriwinkel vom Start- bis zum Endpunkt liegt zwischen 0 und 180°.
- Legen Sie die Positionen so fest, dass die Differenz zwischen Mittelpunkt und Endpunkt und die Differenz zwischen Mittelpunkt und Startpunkt größer als 0,01 mm ist.
- Fallen der Start- und der Mittelpunkt oder der End- und der Mittelpunkt zusammen, oder liegen alle Positionen auf einer Geraden, erfolgt eine Fehlermeldung.
- Stimmen der Start- und Endpunkt oder alle 3 Punkte überein, erfolgt keine Fehlermeldung. Der nächste Befehl wird ausgeführt. Ändert sich in dieser Zeit die Stellung, wird nur die Stellung interpoliert.
- Der Befehl kann nicht in einem kontinuierlich ausgeführten Programm verwendet werden.

Erläuterung

- Dieser Befehl verfährt die Handspitze entlang einer geraden Linie zur festgelegten Position.
- Die Roboterstellung wird vom Startpunkt zum Endpunkt interpoliert.
- Erfolgt die Verfahrbewegung durch Angabe des Abstands oder Verfahrwegbetrags im Werkzeugkoordinatensystem hängt die Richtung des Verfahrwegs vom Werkzeugkoordinatensystem des Roboters ab (siehe auch Abschn. 9.7).
Folgende Abbildung zeigt den Verfahrweg des Roboters RV-1A bei Linear-Interpolation.

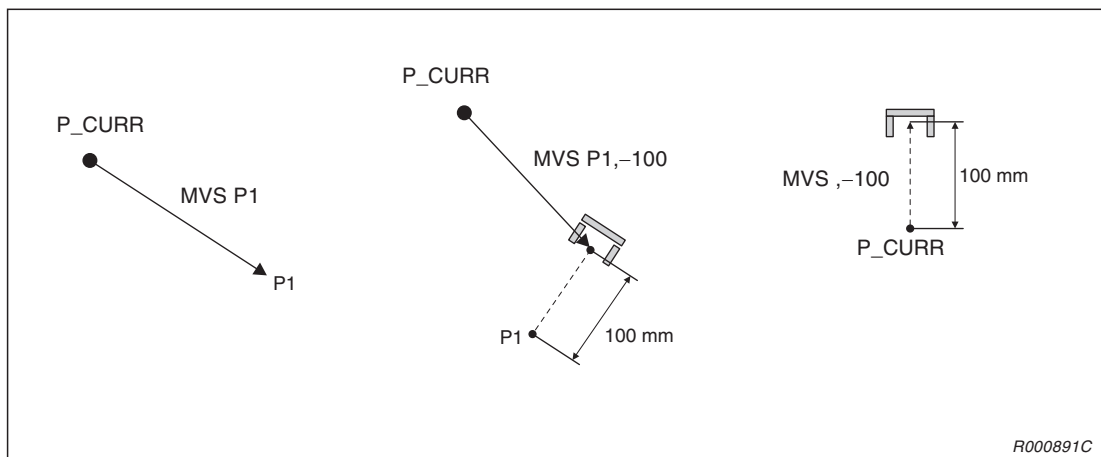


Abb. 6-18: Beispiel zur Linear-Interpolation

- Wird die Ausführung eines MVS-Befehls unterbrochen und nach anschließendem JOG-Betrieb wieder fortgesetzt, kehrt der Roboter zu der Position der Unterbrechung zurück und führt die Verfahrbewegung fort. Eine Änderung dieser Funktion sowie die Auswahl der Interpolationsart (Gelenk- oder XYZ-Interpolation) zur Anfahrt der Position der Unterbrechung kann über den Parameter RETPATH eingestellt werden (siehe auch Abschn. 9.11).
- Der Befehl kann nicht in einem kontinuierlich ausgeführtem Programm verwendet werden.
- Weichen die Stellungsmerker der Start- und Endposition für eine andere Interpolationsmethode als die 3-Achsen-XYZ-Interpolation voneinander ab, erfolgt eine Fehlermeldung.
- Wird über die numerische Konstante 2 die 3-Achsen-XYZ-Interpolation gewählt, ist die numerische Konstante 1 unwirksam und der Roboter bewegt sich mit der geteachten Orientierung.
- Die numerische Konstante 2 legt den Interpolationstyp der Stellung fest. Bei einer Interpolation im Koordinatensystem X, Y, Z, J4, J5 und J6 wird die 3-Achsen-XYZ-Interpolation verwendet, um den Roboter in die Nähe des singulären Punktes zu bewegen.

Singulärer Punkt

Singuläre Punkte markieren im Allgemeinen physikalische Umschlagpunkte oder kritische technische Daten, wie z. B. das Reißen eines auf Zug beanspruchten Seils.

Die Bedeutung singulärer Punkte beim Roboter wird nachfolgend anhand eines 6-achsigen Roboterarms erläutert.

Eine Bewegung von Stellung A über Stellung B zur Stellung C kann nicht mittels Linear-Interpolation (MVS) erfolgen. Diese Einschränkung gilt nur, wenn der Winkel der J4-Achse in allen drei Stellungen (A, B und C) 0° beträgt.

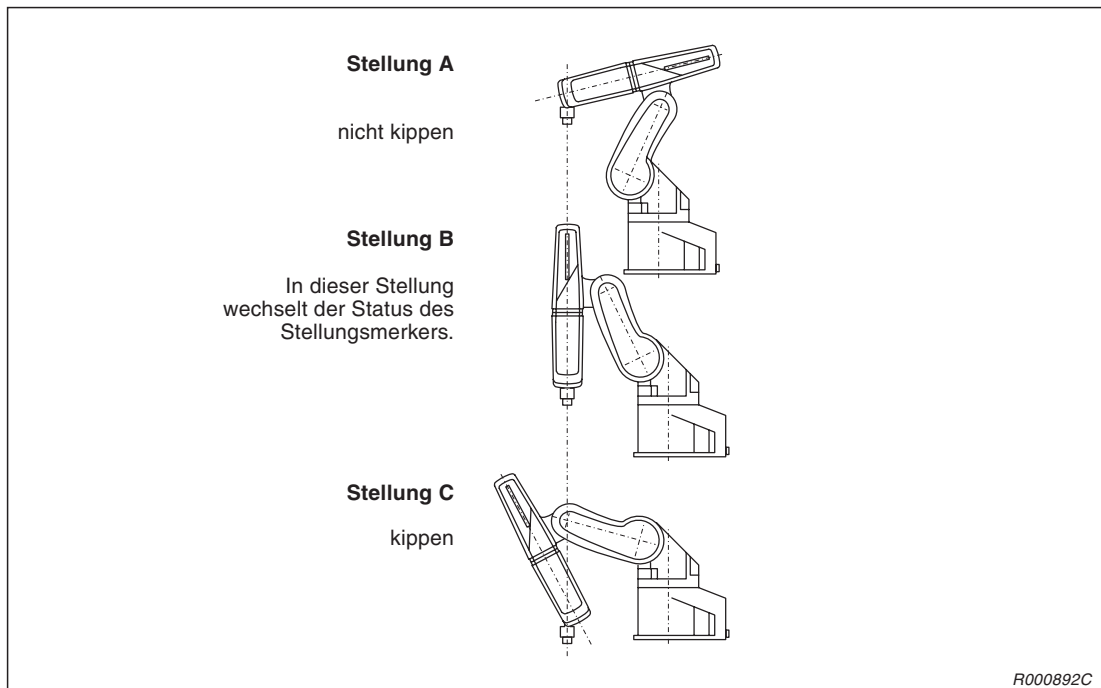


Abb. 6-19: Singulärer Punkt

Der Stellungsmerkerstatus der J5-Achse (Handgelenkneigungsachse) in Stellung A ist „nicht kippen“ und „kippen“ in Stellung C. Weiterhin ist in Stellung B das Handgelenk gestreckt und die Achsen J4 und J6 liegen auf einer Linie. Unter diesen Bedingungen ist keine Berechnung einer Position mittels Linear-Interpolation möglich.

Bei diesen Stellungen ist eine 3-Achsen-XYZ-Linear-Interpolation möglich, wenn im MVS-Befehl die Festlegung „TYPE 0, 1“ erfolgt. Da jedoch die Drehwinkel der Achsen J4, J5 und J6 am Start- und am Endpunkt gleichmäßig interpoliert werden, bleibt die Stellung streng genommen nicht erhalten. Die Roboterhand wird sich beim Verfahren von Stellung A nach C vor und zurück bewegen. Zu Minimierung der Handstellungsänderung kann in der Mitte des Verfahrenswegs eine Position hinzugefügt werden.

Ein weiterer singulärer Punkt ist erreicht, wenn sich der Mittelpunkt der J5-Achse mit nach oben gerichtetem Handflansch über dem Nullpunkt befindet. In diesem Fall liegen die Achsen J1 und J6 auf einer Geraden und eine Berechnung der Roboterposition ist nicht möglich.

6.3.53 OADL (Optimum Acceleration/Deceleration)

Funktion: Optimale Beschleunigung/Abbremsung

Legt die optimale Beschleunigungs-/Abbremszeit in Abhängigkeit von der Lasteinstellung der Hand fest

Dadurch ist eine Verkürzung der Taktzeiten möglich.

Ist die optimale Beschleunigung/Abbremsung freigegeben kann die Beschleunigungs-/Bremszeit wie folgt berechnet werden:

$$\text{Beschleunigungs-/Bremszeit [s]} = \frac{\text{Optimale Beschleunigungs-/Bremszeit [s]}}{\text{Einstellwert des ACCEL-Befehls [\%]} \times \text{M_SETADL [\%]}}$$

Eingabeformat

OADL <ON/OFF>

<ON/OFF>

ON: Einstellung für die optimale Beschleunigung/Abbremsung freigegeben

OFF: Einstellung für die optimale Beschleunigung/Abbremsung sperren

Programmbeispiel

10	LOADSET 1,1	Optimale Beschleunigung/Abbremsung für Hand 1 und Werkstück 1 einstellen
20	OADL ON	Gibt die optimale Beschleunigung/Abbremsung frei
30	MOV P1	Position 1 anfahren
40	MOV P2	Position 2 anfahren
50	HOPEN 1	Öffnet Hand 1
60	MOV P3	Position 3 anfahren
70	HCLOSE 1	Schließt Hand 1
80	MOV P4	Position 4 anfahren
90	OADL OFF	Sperrt die optimale Beschleunigung/Abbremsung

Parameter HNDHOLD ist auf „0, 1“ gesetzt.

Erläuterung

- Der Roboter bewegt sich mit der optimalen Beschleunigung/Abbremsung für die über den LOADSET-Befehl eingestellten Bedingungen für die Hand und das Werkstück.

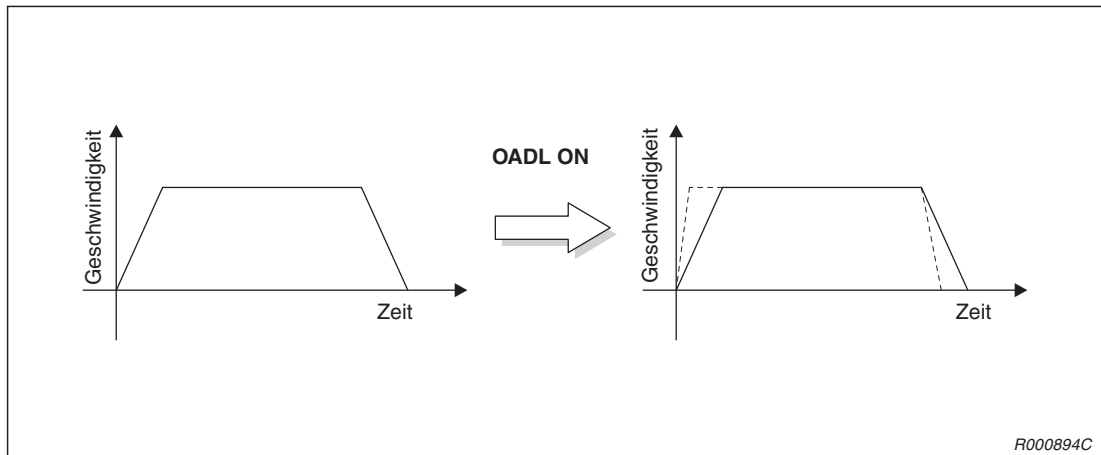


Abb. 6-20: Optimale Beschleunigung/Abbremsung

- Die Zuordnungen für das Öffnen/Schließen einer Hand und den Befehlen HOPEN oder HCLOSE erfolgen über die Parameter HNDHOLD 1 bis 8.
- Die OADL-StandardEinstellung kann über Parameter ACCMODE verändert werden (siehe auch Tab. 9-3 „Signalparameter“).
- Der OADL-Befehl ist so lange aktiv (OADL ON), bis er auf OFF gesetzt wird (OADL OFF) oder die END-Anweisung ausgeführt wird.
- In Abhängigkeit der Hand- und Werkstückbedingungen kann sich die Verfahrbewegung auch verlangsamen.
- Der Betrieb mit optimaler Beschleunigung/Abbremsung erfolgt über die Befehle LOADSET und OADL sowie die Parameter HNDDAT1(0) bis 8 und WRKDAT1(0) bis 8 (siehe auch Abschn. 9.17.1).
- Die Einstellzeit für die optimale Beschleunigungs-/Abbremszeit jeder Achse ist im Parameter JADL festgelegt. Der Parameter ist nur für die Robotermodelle RV-S verfügbar und vom jeweiligen Roboter abhängig (siehe auch Abschn. 9.2).

Steht in Beziehung zu folgenden Befehlen:

ACCEL, LOADSET, HOPEN/HCLOSE

Steht in Beziehung zu folgenden Parametern:

HNDDAT0 bis 8, WRKDAT0 bis 8, HNDHOLD1 bis 8, ACCMODE, JADL

6.3.54 ON COM GOSUB (ON Communication Go Subroutine)

Funktion: Sprung zu einem Unterprogramm

Legt den Sprung in ein Unterprogramm fest, wenn ein Interrupt von einem Kommunikationskanal anliegt

Eingabeformat

```
ON  COM  [ (<Dateinummer> ) ]  GOSUB  <Sprungziel>
```

<Dateinummer> Legt die Nummer des Kommunikationskanals
($1 \leq \text{Kommunikationskanal} \leq 3$) fest

<Sprungziel> Legt eine Zeilennummer oder eine Marke fest

Programmbeispiel

10	OPEN "COM1:" AS #1	Öffnet Kommunikationskanal 1 als Datei Nr. 1
20	ON COM(1) GOSUB *RECV	Springt zu Marke RECV, wenn auf dem Kommunikationskanal Nummer 1 ein Interrupt anliegt
30	COM(1) ON	Gibt den Kommunikations-Interrupt der Datei Nr. 1 frei
40		Liegt der Kommunikations-Interrupt der Datei Nr. 1 in diesem Bereich an, so erfolgt ein Sprung zur Marke RECV
100		
110		
120	MOV P1	Position P1 anfahren
130	COM(1) STOP	Ignoriert Interrupts während der Bewegung von P1 nach P2
140	MOV P2	Position P2 anfahren
150	COM(1) ON	Sind während der Verfahrbewegung von P1 nach P2 Interrupts aufgetreten, erfolgt nun deren Verarbeitung
160		Liegt der Kommunikations-Interrupt der Datei Nr. 1 in diesem Bereich an, so erfolgt ein Sprung zur Marke RECV
170		
260		
270	COM(1) OFF	Sperrt den Kommunikations-Interrupt der Datei Nr. 1
280	CLOSE #1	Schließt Datei Nummer 1
290	END	Programmende
	:	
	:	
3000	*RECV	Interruptprozedur
3010	INPUT #1,M0001	Speichert die empfangenen Daten in die Variablen M0001 und P0001
3020	INPUT #1,P0001	
3100	RETURN 1	Springt eine Zeile hinter die Zeile, in der der Interrupt aufgetreten ist

Erläuterung

- Bei fehlender Nummer des Kommunikationskanals wird der Standardwert 1 gesetzt.
- Die Prioritäten der Interrupts werden mit steigender Dateinummer kleiner.
- Bei anliegendem Interrupt wird die Roboterbewegung gestoppt. Mit der COM STOP-Anweisung kann der Interrupt ignoriert werden und die Roboterbewegung wird nicht unterbrochen.
- In der Grundeinstellung sind die Interrupts gesperrt. Geben Sie die Interrupts nach Ausführung des Befehls über den COM ON-Befehl wieder frei.
- Der Rücksprung aus dem Unterprogramm muss mit der RETURN-Anweisung erfolgen. Ein Rücksprung über die GOTO-Anweisung führt zu einer Fehlermeldung, wenn die Speicherplatzkapazität für die Programmsteuerung (Stapelspeicher) überschritten wird.

Steht in Beziehung zu folgenden Befehlen:

COM ON/COM OFF/COM STOP, RETURN, OPEN, INPUT, PRINT, CLOSE

6.3.55 ON GOSUB (ON GOSUB)

Funktion: Sprung zu einem Unterprogramm

Legt den Sprung zu einer festgelegten Zeilennummer oder einer Marke eines Unterprogramms fest

Eingabeformat

```
ON <Ausdruck> GOSUB [<Sprungziel>] [, [<Sprungziel>]] ...
```

<Ausdruck> Legt fest, zu welcher Zeilennummer oder Marke das Programm verzweigt wird

<Sprungziel> Legt eine Zeilennummer oder Marke fest
Die maximale Anzahl beträgt 32.

Programmbeispiel

Der durch 3 Bits festgelegte Wert des Eingangssignals 16 wird in M1 übertragen. In Abhängigkeit von M1 (1 bis 7) erfolgt eine Programmverzweigung.

```
10  M1 = M_INB(16) AND &H7  Schreibt die Eingangssignalbits 16 bis 18
                               als 8-Bit-Wort in die numerische Variable M1

20  ON M1 GOSUB 1000,*LSUB,2000,2000,2000,*L67,*L67  Springt zur Zeile 1000,
                                                         falls M1 = 1, springt zur
                                                         Marke LSUB, falls
                                                         M1 = 2, springt zur Zeile
                                                         2000, falls M1 = 3, 4
                                                         oder 5 und springt zur
                                                         Marke L67, falls M1 = 6
                                                         oder 7

1000                               Prozedur bei M1 = 1
1010
1200 RETURN                       Rücksprung

1210 *LSUB
1220                               Prozedur bei M1 = 2
1300 RETURN                       Rücksprung

1700 *L67
1710                               Prozedur bei M1 = 6 oder 7
1720 RETURN                       Rücksprung

2000                               Prozedur bei M1 = 3, 4 oder 5
2010
2020 RETURN                       Rücksprung
```

Erläuterung

- Der Wert des Ausdrucks legt fest, zu welcher Zeilennummer oder Marke das Programm verzweigt wird.
Beispiel: Ist der Wert des Ausdrucks 2, wird das zweite Sprungziel aufgerufen.
- Ist der Wert des Ausdrucks größer als die Anzahl der angegebenen Sprungziele, springt die Programmsteuerung in die nächste Zeile.
- Wird eine Zeilennummer oder Marke aufgerufen, die nicht existiert oder zweimal definiert ist, erfolgt eine Fehlermeldung.
- Der Rücksprung aus dem Unterprogramm muss mit der RETURN-Anweisung erfolgen. Ein Rücksprung über die GOTO-Anweisung führt zu einer Fehlermeldung, wenn die Speicherplatzkapazität für die Programmsteuerung (Stapelspeicher) überschritten wird.

Wert von <Ausdruck>	Steuerung
Reelle Zahl	Der Wert wird zu einer Integer-Zahl gerundet.
Wenn der Wert des Ausdrucks gleich 0 ist oder wenn der Wert größer als die Anzahl der Zeilen oder Marken ist	Steuerung springt in die nächste Zeile.
Wenn der Wert negativ oder größer als 32767 ist	ERROR
Zeilennummer oder Marke ist nicht angegeben	ERROR

Tab. 6-8: Werte des Ausdrucks und deren Verarbeitung

6.3.56 ON ... GOTO (On Go To)

Funktion: Programmverzweigung

Legt den Sprung zu einer festgelegten Zeilennummer oder einer Marke fest

Eingabeformat

```
ON  <Ausdruck>  GOTO  [<Sprungziel>][, [<Sprungziel>]] ...
```

<Ausdruck> Legt fest, zu welcher Zeilennummer oder Marke das Programm verzweigt wird

<Sprungziel> Legt eine Zeilennummer oder Marke fest
Die maximale Anzahl beträgt 32.

Programmbeispiel

In Abhängigkeit von M1 (1 bis 7) erfolgt eine Programmverzweigung.

100	ON M1 GOTO 1000,*LJMP,2000,2000,2000,*L67,*L67	Springt zur Zeile 1000, falls M1 = 1, springt zur Marke LJMP, falls M1 = 2, springt zur Zeile 2000, falls M1 = 3, 4 oder 5 und springt zur Marke L67, falls M1 = 6 oder 7
110		Diese Zeile wird ausgeführt, falls M1 keinem der Werte von 1 bis 7 entspricht (z. B. 0, 8 oder größer)
1000		Prozedur bei M1 = 1
1010	:	
1110	*LJMP	
1120		Prozedur bei M1 = 2
1130	:	
1700	*L67	
1710		Prozedur bei M1 = 6 oder 7
1720	:	
2000		Prozedur bei M1 = 3, 4 oder 5
2010	:	

Erläuterung

- Der Wert des Ausdrucks legt fest, zu welcher Zeilennummer oder Marke das Programm verzweigt wird.
Beispiel: Ist der Wert des Ausdrucks 2, wird das zweite Sprungziel aufgerufen.
- Ist der Wert des Ausdrucks größer als die Anzahl der angegebenen Sprungziele, springt die Programmsteuerung in die nächste Zeile.
- Wird ein Sprungziel aufgerufen, das nicht existiert oder zweimal definiert ist, erfolgt eine Fehlermeldung.

Wert von <Ausdruck>	Steuerung
Reelle Zahl	Der Wert wird zu einer Integer-Zahl gerundet.
Wenn der Wert des Ausdrucks gleich 0 ist oder wenn der Wert größer als die Anzahl der Zeilen oder Marken ist	Steuerung springt in die nächste Zeile.
Wenn der Wert negativ oder größer als 32767 ist	ERROR
Zeilennummer oder Marke ist nicht angegeben	ERROR

Tab. 6-9: Werte des Ausdrucks und deren Verarbeitung

6.3.57 OPEN (Open)

Funktion: Datei öffnen

Öffnet eine Datei oder einen Kommunikationkanal

Eingabeformat

```
OPEN  "<Dateibezeichnung>"  [FOR <Modus>] 
AS  [#]<Dateinummer>
```

<Dateibezeichnung> Gibt den Namen der Datei oder des Kommunikationskanals an
 Die Dateibezeichnung "<Dateiname des Kommunikationskanals>:" wird zum Öffnen von Kommunikationskanälen verwendet.
 Die Dateibezeichnung "<Dateiname>" wird zum Öffnen anderer Dateien verwendet.

<Modus> Legt die Methode fest, mit der auf eine Datei zugegriffen wird
 Die Angabe kann zum Öffnen von Kommunikationskanälen weggelassen werden.

- Keine Angabe = wahlfreier Modus
 Dieser Modus wird beim Zugriff auf die Kommunikationskanäle verwendet.
- INPUT = Eingabemodus
 Liest Daten von einer vorhandenen Datei ein
- OUTPUT = Ausgabemodus (neue Datei)
 Legt eine neue Datei an und schreibt Daten in diese Datei
- APPEND = Ausgabemodus (vorhandene Datei)
 Hängt Daten an das Ende einer vorhandenen Datei an

Dateibezeichnung	Dateiname	Zugriffsmethode
Dateiname	Maximal 16 Zeichen	INPUT, PRINT, APPEND
Kommunikationskanal	COM 1: Standardschnittstelle RS232 (Grundeinstellung), COM 2: Einstellung des : Parameters COMDEV, COM 8: Einstellung des Parameters COMDEV	Keine Angabe = Wahlfreier Modus

Tab. 6-10: Dateibezeichnung und Zugriffsmethode

<Dateinummer> Konstante im Bereich zwischen 1 und 8;
 für einen Interrupt eines Kommunikationskanals: 1 bis 3

Programmbeispiel

Öffnen eines Kommunikationskanals

10	OPEN "COM1:" AS #1	Öffnet den RS232C-Kommunikationskanal als Datei Nr. 1
20	MOV P_01	Position P_01 anfahren
30	PRINT #1,P_CURR	Sendet die Daten der aktuellen Position an einen an der RS232 angeschlossenen Empfänger, z. B. Terminal, SPS ... Format: (100.00,200.00,300.00,400.00)(7.0)
40	INPUT #1,M1,M2,M3	Liest die Daten „101.00,202.00,303.00“ im ASCII-Format ein und schreibt sie in die Variablen M1, M2 und M3
50	P_01.X = M1	Schreibt die Daten aus M1 in die X-Komponente der globalen Variablen P_01
60	P_01.Y = M2	Schreibt die Daten aus M2 in die Y-Komponente der globalen Variablen P_01
70	P_01.C = RAD(M3)	Schreibt die Daten aus M3 in die C-Komponente der globalen Variablen P_01
80	CLOSE	Schließt alle geöffneten Dateien
90	END	Programmende

Öffnen einer Datei

10	OPEN "temp.txt" FOR APPEND AS #1	Hängt die Daten der Datei temp.txt an das Ende der Datei Nr. 1
20	PRINT #1,"abc"	Schreibt die Daten „abc“ in Datei Nr. 1
30	CLOSE #1	Schließt die Datei Nr. 1

Erläuterung

- Die Datei wird über die Dateibezeichnung geöffnet und es wird eine Dateinummer festgelegt. Schreib- oder Lesevorgänge werden einer Datei durch Angabe der Dateinummer zugeordnet.
- Ein Kommunikationskanal wird wie eine Datei behandelt.

Steht in Beziehung zu folgenden Befehlen:

CLOSE, PRINT, INPUT

Steht in Beziehung zu folgenden Parametern:

COMDEV

6.3.58 OVRD (Override)

Funktion: Übersteuerung

Legt den Programmwert für die Geschwindigkeitsübersteuerung fest.

Eingabeformat

```
OVRD □ <Übersteuerungswert>
```

Ab Software-Version G2:

```
OVRD □ <Übersteuerungswert>
      [, <Übersteuerungswert bei Aufwärtsbewegung>]
      [, <Übersteuerungswert bei Abwärtsbewegung>
```

<Übersteuerungswert>

Legt den prozentualen Übersteuerungswert fest (Standardwert: 100)
 $1 \leq \text{Übersteuerungswert} \leq 100.0$
 Bei einer Einstellung eines Wertes außerhalb des Einstellbereiches erfolgt eine Fehlermeldung.

<Übersteuerungswert bei Aufwärtsbewegung>
 <Übersteuerungswert bei Abwärtsbewegung>

Legt den Übersteuerungswert für Auf- bzw. Abwärtsbewegungen bei Bogen-Interpolation (MVA) fest.

Programmbeispiel

10	OVRD 50	Übersteuerung auf den Wert 50 % einstellen
20	MOV P1	Position P1 mittels Gelenk-Interpolation anfahren
30	MVS P2	Position P2 mittels Linear-Interpolation anfahren
40	OVRD M_NOVRD	Standardwert einstellen
50	MOV P1	Position P1 mittels Gelenk-Interpolation anfahren
60	OVRD 30,10,10	Übersteuerung für Auf- bzw. Abwärtsbewegungen bei Bogen-Interpolation auf 10 % einstellen
70	MVA P3,3	Position P3 mittels Bogen-Interpolation über Bogen 3 anfahren

Erläuterung

- Dieser Befehl legt den prozentualen Übersteuerungswert für die Arbeitsgeschwindigkeit des Roboters fest.
- Der OVRD-Befehl ist unabhängig von der Art der Interpolation wirksam.
- Die aktuelle Arbeitsgeschwindigkeit ergibt sich folgendermaßen:

$$\begin{array}{l} \text{Gelenk-} \\ \text{Interpolation} \end{array} = \begin{array}{l} \text{Einstellung der T/B} \\ \text{oder des Steuergeräts} \end{array} \times \begin{array}{l} \text{Einstellwert des} \\ \text{OVRD-Befehls} \end{array} \times \begin{array}{l} \text{Einstellwert des} \\ \text{JOVRD-Befehls} \end{array}$$

$$\begin{array}{l} \text{Linear-} \\ \text{Interpolation} \end{array} = \begin{array}{l} \text{Einstellung der T/B} \\ \text{oder des Steuergeräts} \end{array} \times \begin{array}{l} \text{Einstellwert des} \\ \text{OVRD-Befehls} \end{array} \times \begin{array}{l} \text{Einstellwert des} \\ \text{SPD-Befehls} \end{array}$$

- Der Maximalwert der Arbeitsgeschwindigkeit ist 100 %. Der Standardwert der Arbeitsgeschwindigkeit beträgt 100 % der Standardeinstellung (M_NOVRD).
- Der Standardwert bleibt so lange wirksam, bis der OVRD-Befehl ausgeführt wird. Die so festgesetzte Arbeitsgeschwindigkeit kann durch einen weiteren OVRD-Befehl geändert werden.
- Die durch einen OVRD-Befehl festgelegte Arbeitsgeschwindigkeit bleibt so lange erhalten, bis erneut ein OVRD-Befehl, eine END-Anweisung oder ein Reset ausgeführt wird. Nach Ausführung der END-Anweisung oder eines Resets ist der Standardwert wieder gültig.
- Liegt der Übersteuerungswert außerhalb des Wertebereiches des Roboters, erfolgt eine Fehlermeldung. Der Wert muss zwischen 0 und 100 % liegen.

Steht in Beziehung zu folgenden Befehlen:

JOVRD (für Gelenk-Interpolation), SPD (für Linear- und Kreis-Interpolation)

Steht in Beziehung zu folgenden Parametern:

M_JOVRD/M_NJOVRD/M_OPOVRD/M_OVRD/M_NOVRD
M_NOVRD (Standardeinstellung), M_OVRD (aktuelle Einstellung)

6.3.59 PLT (Pallet)

Funktion: Koordinaten für Palette berechnen

Berechnet die Koordinaten eines Gitterpunktes der festgelegten Palette und weist die berechneten Koordinaten der festgelegten Position zu

Eingabeformat

```
PLT □ <Palettennummer>, <Gitterpunktnummer>
```

<Palettennummer>

Wählt eine vorher mit dem DEF PLT-Befehl definierte Palette aus
Die Angabe erfolgt als Konstante oder Variable.
 $1 \leq \text{Palettennummer} \leq 8$

<Gitterpunktnummer>

Legt die Positionsnummer für die berechneten Koordinaten fest
Die Angabe erfolgt als Konstante oder Variable.

Programmbeispiel (RV-Roboter)

100 DEF PLT 1,P1,P2,P3,P4,4,3,1	Definiert Palette Nummer 1
110	
120 M1 = 1	Setzt M1 auf „1“
130 *LOOP	Sprungmarke „LOOP“ festgelegt
140 MOV PICK,-50	Position anfahren, die um 50 mm in Werkzeugzeuglängsrichtung von der Aufnahmeposition entfernt liegt
150 OVRD 50	Übersteuerung auf den Wert 50 % einstellen
160 MVS PICK	Aufnahmeposition mittels Linear-Interpolation anfahren
170 HCLOSE 1	Schließt Hand 1
180 DLY 0.5	Wartezeit von 0,5 Sekunden ermöglicht ein sicheres Schließen der Hand
190 OVRD 100	Übersteuerung auf den Wert 100 % einstellen
200 MVS ,-50	Position anfahren, die 50 mm in Werkzeugzeuglängsrichtung von der aktuellen Position entfernt ist
210 PLACE = PLT 1,M1	Weist PLACE die Koordinaten des Gitterpunktes M1 zu
220 MOV PLACE,-50	Position anfahren, die um 50 mm in Werkzeugzeuglängsrichtung von der Ablageposition entfernt liegt
230 OVRD 50	Übersteuerung auf den Wert 50 % einstellen
240 MVS PLACE	Ablageposition mittels Linear-Interpolation anfahren
250 HOPEN 1	Öffnet Hand 1
260 DLY 0.5	Wartezeit von 0,5 Sekunden ermöglicht ein sicheres Öffnen der Hand
270 OVRD 100	Übersteuerung auf den Wert 100 % einstellen
280 MVS ,-50	Position anfahren, die 50 mm in Werkzeugzeuglängsrichtung von der aktuellen Position entfernt ist
290 M1 = M1 + 1	Erhöht den Wert von M1 um 1
300 IF M1 <= 12 THEN *LOOP	Wiederholt die Schleife ab der Marke LOOP, solange M1 kleiner gleich 12 ist
310 MOV PICK,-50	Position anfahren, die um 50 mm in Werkzeugzeuglängsrichtung von der Position PICK entfernt liegt
320 END	Programmende

Erläuterung

- Dieser Befehl berechnet die Koordinaten eines Gitterpunktes einer Palette, die vorher mit dem DEF PLT-Befehl definiert wurde, und weist sie einer Position zu.
- Die Palettennummern müssen im Bereich von 1 bis 8 liegen. Es können bis zu acht Paletten gleichzeitig definiert sein.
- Die Position des Gitterpunktes kann in Abhängigkeit der festgelegten Bewegungsrichtung (siehe DEF PLT-Befehl) unterschiedlich sein.
- Wird ein Gitterpunkt festgelegt, der außerhalb der Zeilen oder Spalten der definierten Palette liegt, erfolgt eine Fehlermeldung.
- Ist ein Palettengitterpunkt, der in einem Bewegungsbefehl als Zielposition angegeben ist, nicht in Klammern aufgeführt (richtig ist z. B. MOV (PLT 1,M1),-50), erfolgt eine Fehlermeldung.
- Eine detaillierte Beschreibung der Palettenfunktion finden Sie in Abschn. 4.4.

Steht in Beziehung zu folgenden Befehlen:

DEF PLT

6.3.60 PREC (Precision)

Funktion: hohe Verfahrweggenauigkeit

Die Verfahrwegtreue bei der Ausführung von Bewegungsbefehlen kann erhöht werden. Die Funktion ist nur für bestimmte Robotermodelle verfügbar. Zur Zeit kann der Befehl mit folgenden Robotermodellen verwendet werden: RV-1A/2AJ, RV-2A/3AJ, RV-4A/5AJ, RV-3SB/3SJB, RV-6S/6SL/12S/12SL und RH-6SH/12SH.

Eingabeformat

PREC <ON/OFF>

<ON/OFF>

ON: hohe Verfahrweggenauigkeit aktiviert
 OFF: hohe Verfahrweggenauigkeit deaktiviert

Programmbeispiel

10	PREC ON	Aktiviert die hohe Verfahrweggenauigkeit
20	MVS P1	Position P1 mittels Linear-Interpolation und mit hoher Verfahrweggenauigkeit anfahren
30	MVS P2	Position P2 mittels Linear-Interpolation und mit hoher Verfahrweggenauigkeit anfahren
40	PREC OFF	Deaktiviert die hohe Verfahrweggenauigkeit
50	MOV P1	Position P1 mittels Gelenk-Interpolation anfahren

Erläuterung

- Die Ausführung eines Bewegungsbefehls mit hoher Verfahrweggenauigkeit erfolgt über den Befehl PREC ON.
- Durch Aktivierung der hohen Verfahrweggenauigkeit nehmen die Beschleunigungs- und Bremszeiten und somit auch die Zykluszeiten zu.
- Die hohe Verfahrweggenauigkeit ist ab dem ersten Interpolationsbefehl nach Ausführung des Befehls PREC ON gültig.
- Die hohe Verfahrweggenauigkeit wird durch Ausführung des Befehls PREC OFF, der END-Anweisung oder durch Zurücksetzen des Programms deaktiviert.
- Nach Einschalten der Spannungsversorgung ist die hohe Verfahrweggenauigkeit deaktiviert.
- Im JOG-Betrieb ist die Funktion immer deaktiviert.

6.3.61 PRINT (Print)

Funktion: Daten übertragen

Überträgt Daten in eine Datei oder Kommunikationsleitung
Alle Daten werden im ASCII-Format übertragen.

Eingabeformat

```
PRINT □ #<Dateinummer> □ [, [<Ausdruck>;] ... [<Ausdruck> [ ; ]]
```

<Dateinummer> Bezieht sich auf die im OPEN-Befehl festgelegte Dateinummer
 $1 \leq \text{Dateinummer} \leq 8$

<Ausdruck> Legt eine numerische Variable, eine Positionsvariable
oder eine Zeichenkette fest

Programmbeispiel

10 OPEN "COM1" AS #1	Öffnet den RS232C-Kommunikationskanal als Datei Nr. 1
20 MDATA = 150	Setzt MDATA auf „150“
30 PRINT #1, "***PRINT TEST***"	Gibt die Zeichenkette ***PRINT TEST*** aus
40 PRINT #1	Gibt eine Leerzeile aus
50 PRINT #1, "MDATA =", MDATA	Gibt die Zeichenkette MDATA = und den Wert von MDATA aus, (150)
60 PRINT #1	Gibt eine Leerzeile aus
70 PRINT #1, "*****"	Gibt die Zeichenkette ***** aus
80 END	Programmende

Folgendes Ergebnis wird ausgegeben:

```
***PRINT TEST***
```

```
MDATA = 150
```

```
*****
```

Erläuterung

- Fehlt eine Angabe für <Ausdruck>, wird ein „Carriage Return“ ausgegeben.
- Ausgabeformat der Daten:
Der Platz für die Ausgabe von <Ausdruck> ist in Einheiten von 14 festgelegt. Werden bei der Ausgabe mehrere Ausdrücke angegeben, muss ein Komma zwischen den einzelnen Ausdrücken stehen.
Bei Trennung der Ausdrücke durch Semikolons werden sie ohne Zwischenraum ausgegeben.
- Nach jeder PRINT-Anweisung wird ein „Carriage Return“ ausgeführt.
- Fehlt der OPEN-Befehl, erfolgt eine Fehlermeldung.
- Enthalten die Daten ein Anführungszeichen, erfolgt die Ausgabe der Daten bis zu diesen Anführungszeichen.

Beispiel ▾

10 M1 = 123.5

20 P1 = (130.5,-117.2,55.1,16.2,0.0,0.0)(1,0)

nach Eingabe von

30 PRINT #1,"OUTPUT TEST",M1,P1

wird

OUTPUT TEST 123.5 (130.5,-117.2,55.1,16.2,0.0,0.0)(1,0)

ausgegeben

nach Eingabe von

30 PRINT #1,"OUTPUT TEST";M1;P1

wird

OUTPUT TEST123.5(130.5,-117.2,55.1,16.2,0.0,0.0)(1,0)

ausgegeben

Werden die Ausdrücke durch ein Komma oder ein Semikolon getrennt, wird kein „Carriage Return“ zugelassen. Die Ausdrücke werden in einer Zeile ausgegeben. Wird kein Komma oder Semikolon eingegeben, wird ein „Carriage Return“ zugelassen.

Nach Eingabe von

30 PRINT #1,"OUTPUT TEST",

40 PRINT #1,M1;

50 PRINT #1,P1

wird

OUTPUT TEST 123.5(130.5,-117.2,55.1,16.2,0.0,0.0)(1,0)

ausgegeben.

△

Steht in Beziehung zu folgenden Befehlen:

OPEN, CLOSE, INPUT

6.3.62 PRIORITY (Priority)

Funktion: Priorität festlegen

Die Einstellung legt die Anzahl der auszuführenden Zeilen für einen Durchgang im Multi-task-Betrieb fest.

Eingabeformat

Ab Software-Version C2:

```
PRIORITY □ <Anzahl der auszuführenden Zeilen>
           [, <Programmplatznummer>]
```

<Anzahl der auszuführenden Zeilen>	Anzahl der auszuführenden Zeilen für einen Durchgang Einstellbereich: 1 bis 31
<Programmplatznummer>	Legt den Programmplatz fest $1 \leq \text{Programmplatz} \leq 32$ Bei fehlender Angabe wird der aktuelle Programmplatz (Slot) gesetzt.

Programmbeispiel

Programmplatz 1 (Slot 1)

```
10 PRIORITY 3
```

Setzt die Anzahl der in einem Durchgang auszuführenden Zeilen für den aktuellen Programmplatz auf „3“

Programmplatz 2 (Slot 2)

```
10 PRIORITY 4
```

Setzt die Anzahl der in einem Durchgang auszuführenden Zeilen für diesen Programmplatz auf „4“

Erläuterung

- Programme in anderen Programmplätzen werden solange nicht ausgeführt, bis die eingestellte Anzahl von Zeilen abgearbeitet worden ist. Im Programmbeispiel oben werden zuerst 3 Zeilen des Programms in Programmplatz 1 und anschließend 4 Zeilen des Programms in Programmplatz 2 ausgeführt. Nach einem Durchgang beginnt dieser Zyklus von vorne.
- Die Standardeinstellung für alle Programmplätze ist „1“. Das heißt, die Programmsteuerung springt nach Ausführung jeweils einer Programmzeile in einem Programmplatz zum nächsten Programmplatz.
- Existiert kein Programm im angegebenen Programmplatz, erfolgt eine Fehlermeldung.
- Eine Änderung der Prioritäten ist auch dann möglich, wenn das Programm im entsprechenden Slot ausgeführt wird.

6.3.63 RELM (Release Mechanism)

Funktion: Mechanismuszuordnung aufheben

Der Befehl wird im Multitask-Betrieb zur Steuerung von Mechanismen über Programmplätze verwendet. Er dient zur Aufhebung der über den GETM-Befehl definierten Zuordnung eines Mechanismus.

Eingabeformat

RELM

Programmbeispiel

Programmplatz 2 wird über den Programmplatz 1 gestartet. Der Mechanismus 1 im Programmplatz 2 wird über Programmplatz 1 gesteuert.

10	RELM	Definition von Mechanismus 1 aufheben, um Mechanismus 1 über Programmplatz 2 zu steuern
20	XRUN 2,"10"	Startet Programm 10 als Programmplatz 2
30	WAIT M_RUN(2) = 1	Wartestatus, bis Betriebssignal des Programmplatzes 2 gleich 1 ist

:

Programmplatz 2 (Programm 10)

10	GETM 1	Definition des Mechanismus 1
20	SERVO ON	Servospannung des Mechanismus 1 einschalten
30	MOV P1	Position 1 mittels Gelenk-Interpolation anfahren
40	MVS P2	Position 2 mittels Linear-Interpolation anfahren
50	SERVO OFF	Servospannung des Mechanismus 1 ausschalten
60	RELM	Definition von Mechanismus 1 aufheben
70	END	Programmende

Erläuterung

- Dieser Befehl hebt die aktuelle Zuordnung eines Mechanismus auf.
- Bei einem Programmstopp durch ein Interrupt-Signal wird der Befehl RELM automatisch durch das System ausgeführt.
- Der Befehl kann nicht in einem kontinuierlich ausgeführtem Programm verwendet werden.

Steht in Beziehung zu folgenden Befehlen:

GETM

6.3.64 REM (Remarks)

Funktion: Kommentar

Ermöglicht dem Programmierer, einen Kommentar zu schreiben

Eingabeformat

```
REM □ [<Kommentar>]
```

<Kommentar>

Es können Zeichenketten bis zur Länge einer Zeile eingegeben werden.

Programmbeispiel

10	REM ***Hauptprogramm***	Legt die Zeichenkette ***Hauptprogramm***
20	'***Hauptprogramm***	als Kommentar fest
30	MOV P1	'Position P1 mittels Gelenk-Interpolation anfahren

Erläuterung

- Anstelle der REM-Anweisung kann wahlweise ein Apostroph (') verwendet werden.
- Ein Kommentar kann wie im obigen Programmbeispiel in Zeile 30 hinter einem Befehl in derselben Zeile stehen.

6.3.65 RESET ERR (Reset Error)

Funktion: Fehler zurücksetzen

Über den Befehl kann ein vom Steuergerät generierter Fehler zurückgesetzt werden. Eine Verwendung des Befehls im Initialisierungszustand ist nicht erlaubt. Tritt außer einer Warnmeldung ein Fehler auf, können andere als kontinuierlich ausgeführte Programme nicht mehr ausgeführt werden. Die Verwendung des Befehls ist also in kontinuierlich ausgeführten Programmen sinnvoll.

Eingabeformat

Ab Software-Version B1:

```
RESET ERR
```

Programmbeispiel

Befehlsausführung in einem kontinuierlich ausgeführtem Programm

```
10 IF M_ERR = 1 THEN RESET ERR      Tritt ein vom Steuergerät generierter Fehler  
                                     auf, wird der Fehler zurückgesetzt.
```

Erläuterung

- Der Befehl wird in Programmen, in denen die Startbedingung in den Programmplatzparametern auf „kontinuierliche Ausführung (ALWAYS)“ gesetzt ist, zum Zurücksetzen eines Roboter-Systemfehlers verwendet.
- Die Freigabe des Befehls erfolgt nach Einstellung des Parameters ALWENA von „0“ auf „7“ und anschließendem Aus- und Wiedereinschalten der Spannungsversorgung.

Steht in Beziehung zu folgenden Parametern:

ALWENA

Steht in Beziehung zu folgenden Systemvariablen:

M_ERR/M_ERRLVL/M_ERRNO

6.3.66 RETURN (Return)

Funktion: Rücksprung zum Hauptprogramm

Springt beim Rücksprung aus einem Unterprogramm in die Zeile nach dem GOSUB-Befehl

Springt beim Rücksprung aus einer Interrupt-Routine in die Zeile zurück, in der der Interrupt aufgetreten ist oder in die nächste Zeile

Eingabeformat

Beim Rücksprung aus einem Unterprogramm:

```
RETURN
```

Beim Rücksprung aus einer Interrupt-Routine:

```
RETURN <Rücksprungziel>
```

<Rücksprungziel> Legt die Zeile fest, zu der die Steuerung zurückspringt, nachdem eine Interrupt-Routine abgearbeitet wurde
 0 ... Springt in die Zeile, in der der Interrupt aufgetreten ist.
 1 ... Springt eine Zeile hinter die Zeile, in der der Interrupt aufgetreten ist (auch bei fehlender Angabe).

Programmbeispiel

Rücksprung aus einem Unterprogramm

10	'***Hauptprogramm***	Legt die Zeichenkette ***Hauptprogramm*** als Kommentar fest
20	GOSUB *SUB_INIT	Sprung zum Unterprogramm SUB_INIT
30	MOV P1	Position P1 mittels Gelenk-Interpolation anfahren
1000	'***SUB_INIT***	Legt die Zeichenkette ***SUB_INIT*** als Kommentar fest
1010	*SUB_INIT	Sprungmarke SUB_INIT festgelegt
1020	PSTART = P1	Weist PSTART den Wert von P1 zu
1030	M100 = 123	Weist M100 den Wert 123 zu
1040	RETURN	Springt eine Zeile hinter die Zeile, in der der Interrupt aufgetreten ist (Zeile 30)

Rücksprung aus einer Interrupt-Routine

<p>10 DEF ACT 1,M_IN(17) = 1 GOSUB 100</p> <p>20 ACT 1 = 1</p> <p style="padding-left: 20px;">:</p> <p>100</p> <p>110 ACT 1 = 0</p> <p>120 M_TIMER(1) = 0</p> <p>130 MOV P2</p> <p>140 WAIT M_IN(17) = 0</p> <p>150 ACT 1 = 1</p> <p>160 RETURN 0</p>	<p>Definiert einen Unterprogrammprung zu Zeile 100, wenn der Status des allgemeinen Eingangssignals Nummer 17 = EIN ist</p> <p>Interrupt 1 freigeben</p> <p>Interrupt-Routine für Interrupt 1</p> <p>Interrupt 1 sperren</p> <p>Zähler zurücksetzen</p> <p>Position P2 anfahren</p> <p>Wartestatus, bis Eingangsbit 17 gleich 0 ist</p> <p>Interrupt 1 erneut freigeben</p> <p>Sprung in die Zeile, aus der der Interrupt aufgerufen wurde</p>
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Erläuterung

- Der Rücksprung aus einem Unterprogramm oder einer Interrupt-Routine, die mit dem Befehl GOSUB aufgerufen wurde, erfolgt über die RETURN-Anweisung.
- Wird die RETURN-Anweisung ohne vorhergehende GOSUB-Anweisung ausgeführt, erfolgt eine Fehlermeldung.
- Der Rücksprung aus einem mit der GOSUB-Anweisung aufgerufenen Unterprogramm muss mit der RETURN-Anweisung erfolgen. Ein Rücksprung über die GOTO-Anweisung führt zu einer Fehlermeldung, wenn die Speicherplatzkapazität für die Programmsteuerung (Stapelspeicher) überschritten wird.
- Es erfolgt eine Fehlermeldung, wenn bei einem RETURN-Befehl in einem Unterprogramm ein Rücksprungziel angegeben wurde. Es erfolgt eine Fehlermeldung, wenn das Rücksprungziel in einer Interrupt-Routine nicht angegeben wurde.
- Sperren Sie den Interrupt, wenn der Rücksprung über RETURN 1 in die Zeile, die der Zeile mit dem Interruptaufruf folgt, erfolgte. Wird der Interrupt nicht gesperrt und die Interrupt-Bedingung ist erfüllt, erfolgt eine erneute Ausführung der Interrupt-Routine und die Zeile kann beim Rücksprung übersprungen werden. Eine detaillierte Beschreibung zur Definition von Interrupt-Prozessen finden Sie in Abschn. 6.3.19.

Steht in Beziehung zu folgenden Befehlen:

GOSUB, ON GOSUB, ON COM GOSUB, DEF ACT

6.3.67 SELECT CASE

Funktion: Prozess ausführen

Führt in Abhängigkeit einer Bedingung einen von mehreren Prozessen aus

Eingabeformat

```
SELECT  <Auswahl>  
  CASE  <Ausdruck>  
    [<Prozess>]  
    BREAK  
  CASE  <Ausdruck>  
    [<Prozess>]  
    BREAK  
  
    :  
  
  DEFAULT  
    [<Prozess>]  
    BREAK  
END  SELECT
```

<Auswahl>	Legt einen numerischen Ausdruck fest
<Ausdruck>	Legt einen numerischen Ausdruck fest Der Typ muss mit dem der Bedingung übereinstimmen.
<Prozess>	Legt die auszuführende Anweisung (außer GOTO-Anweisung) fest

Programmbeispiel

10	SELECT MCNT	Auswahl der numerischen Variablen MCNT
20	M1 = 10	Diese Zeile wird nicht ausgeführt.
30	CASE IS <= 10	Fahre Position P1 an, falls MCNT kleiner gleich 10 ist
40	MOV P1	
50	BREAK	Sprung hinter die END SELECT-Anweisung
60	CASE 11	Fahre Position P2 an, falls MCNT gleich 11 ist
70	MOV P2	
80	BREAK	Sprung hinter die END SELECT-Anweisung
90	CASE 13 TO 18	Fahre Position P4 an, falls MCNT größer gleich 13 oder kleiner gleich 18 ist
100	MOV P4	
110	BREAK	Sprung hinter die END SELECT-Anweisung
120	DEFAULT	Setzt Ausgangsbit 10 auf „1“, falls MCNT keinem der oben genannten Werte oder Wertebereiche entspricht
130	M_OUT(10) = 1	
140	BREAK	Sprung hinter die END SELECT-Anweisung
150	END SELECT	

Erläuterung

- Wird eine der Bedingungen der CASE-Anweisung erfüllt, wird der Prozess bis zur nächsten CASE-, DEFAULT- oder ENDSELECT-Anweisung ausgeführt.
- Wird keine der CASE-Bedingungen erfüllt, wird der DEFAULT-Prozess ausgeführt. Ist kein DEFAULT-Prozess definiert, springt das Programm eine Zeile hinter die END SELECT-Anweisung.
- Eine SELECT-Anweisung muss immer durch eine END-SELECT-Anweisung abgeschlossen werden. Ein Sprung über die GOTO-Anweisung aus dem CASE-Block der SELECT-CASE-Anweisung belegt Speicherplatz des für die Programmsteuerung reservierten Stapelspeichers. Bei einer kontinuierlichen Ausführung des Programms kann deshalb eine Fehlermeldung erfolgen.
- Bei Ausführung einer END SELECT-Anweisung, der keine SELECT-Anweisung vorausgeht, erfolgt eine Fehlermeldung.
- Ab Software-Version G1 können innerhalb eines SELECT-CASE-Blocks weitere SELECT-CASE-Blöcke ausgeführt werden. Eine Verschachtelung von bis zu 8 Programmebenen ist möglich.
- Die Ausführung von WHILE-WEND- oder FOR-NEXT-Schleifen innerhalb eines CASE-Blocks ist möglich.
- Verwenden Sie Vergleichsoperatoren (<, =, > usw.) mit der Anweisung CASE IS.

6.3.68 SERVO (Servo)

Funktion: Servo ein-/auschalten

Schaltet die Servospannung ein oder aus

Eingabeformat

Verwendung in einem normalen Programm:

```
SERVO □ <ON/OFF>
```

Verwendung in einem Programm mit der Startbedingung „ALWAYS“:

```
SERVO □ <ON/OFF> [ , <Mechanismusnummer> ]
```

<ON/OFF>

ON: Servoversorgung einschalten
OFF: Servoversorgung ausschalten

<Mechanismusnummer>

Festlegung der Mechanismusnummer als
Konstante oder Variable
 $1 \leq \text{Mechanismusnummer} \leq 3$
Ist nur bei der Startbedingung „ALWAYS“ aktiv

Programmbeispiel

10	SERVO ON	Schaltet die Servospannung ein
20	IF M_SVO <> 1 GOTO 20	Wartestatus, bis die Servoversorgung eingeschaltet ist
30	SPD M_NSPD	Geschwindigkeit auf Standardwert setzen
40	MOV P1	Position P1 mittels Gelenk-Interpolation anfahren
50	SERVO OFF	Schaltet die Servospannung aus

Erläuterung

- Die Servospannung wird für alle Achsen ein- oder ausgeschaltet.
- Die Servospannung wird für alle Zusatzachsen ebenfalls ein- oder ausgeschaltet.
- In einem kontinuierlich ausgeführtem Programm wird der Befehl freigegeben, wenn Parameter ALWENA von „0“ auf „7“ umgestellt und anschließend die Spannungsversorgung aus- und wieder eingeschaltet wird.

Steht in Beziehung zu folgenden Systemvariablen:

M_SVO (1: EIN, 0: AUS)

Steht in Beziehung zu folgenden Parametern:

ALWENA

6.3.69 SKIP (Skip)

Funktion: Sprung in die nächste Zeile

Die Programmsteuerung springt in die nächste Zeile.

Eingabeformat

SKIP

Programmbeispiel

10 MOV P1 WTHIF M_IN(17) = 1,SKIP

Fährt Position 1 mittels Gelenk-Interpolation an und unterbricht die Roboterbewegung, wenn das Eingangsbit Nummer 17 gleich 1 wird
Die Programmsteuerung springt in die nächste Zeile.

20 IF M_SKIPCQ = 1 THEN HLT

Unterbricht das Programm bei Ausführung der SKIP-Anweisung

Erläuterung

- Dieser Befehl wird mit anderen Befehlen in Verbindung mit WTH und WTHIF verwendet. Bei Ausführung des Befehls in Verbindung mit WTH oder WTHIF wird die Programmabarbeitung innerhalb der Zeile unterbrochen und die Programmsteuerung springt in die nächste Zeile. Die Ausführung einer SKIP-Anweisung kann über die Roboterstatusvariable M_SKIPCQ geprüft werden.

Steht in Beziehung zu folgenden Systemvariablen:

M_SKIPCQ (1: Sprung, 0: kein Sprung)

6.3.70 SPD (Speed)

Funktion: Geschwindigkeit festlegen

Der Befehl legt die Geschwindigkeit für lineare und kreisförmige Bewegungen fest. Weiterhin ist eine Einstellung der normalen Geschwindigkeit möglich.

Eingabeformat

```
SPD □ <Geschwindigkeitswert>
```

```
SPD □ M_NSPD (normaler Geschwindigkeitswert)
```

<Geschwindigkeitswert>

Legt die Geschwindigkeit als reelle Zahl in mm/s fest

Programmbeispiel

10	SPD 100	Legt die Geschwindigkeit auf 100 mm/s fest
20	MVS P1	Position P1 mittels Linear-Interpolation anfahren
30	SPD M_NSPD	Setzt die Geschwindigkeit auf den Standardwert, so dass die Verfahrbewegungen mit der maximal möglichen Geschwindigkeit erfolgen
40	MOV P2	Position P2 mittels Gelenk-Interpolation anfahren
50	MOV P3	Position P3 mittels Gelenk-Interpolation anfahren
60	OVRD 80	Übersteuerung auf den Wert 80 % einstellen, so dass bei Ausführung von Verfahrbewegungen mit der maximal möglichen Geschwindigkeit kein Fehler wegen Geschwindigkeitsüberschreitung auftritt.
70	MOV P4	Position P4 mittels Gelenk-Interpolation anfahren
80	OVRD 100	Übersteuerung auf den Wert 100 % einstellen

Erläuterung

- Der SPD-Befehl ist nur bei linearen und kreisförmigen Bewegungen des Roboters wirksam.
- Der aktuelle Übersteuerungswert ergibt sich aus:

$$\begin{matrix} \text{Aktueller} & & \text{Übersteuerungswert} & & \text{Einstellwert} & & \text{Einstellwert} \\ \text{Übersteuerungs-} & = & \text{der T/B oder des} & \times & \text{des} & \times & \text{des} \\ \text{wert} & & \text{Steugeräts} & & \text{OVRD-Befehls} & & \text{SPD-Befehls} \end{matrix}$$

- Bei Verwendung des Standardwerts (Grundeinstellung: 1000) verfährt der Roboter immer mit der maximal möglichen Geschwindigkeit.
- In Abhängigkeit der Roboterstellung kann es bei Verwendung des Standardwerts zu einer Fehlermeldung aufgrund einer Geschwindigkeitsüberschreitung kommen. Reduzieren Sie in diesem Fall die Geschwindigkeit dieser Verfahrbewegung, in dem Sie vor der Befehlszeile einen OVRD-Befehl einfügen.
- Der Standardwert ist so lange gültig, bis mit dem SPD-Befehl ein neuer Wert festgelegt wird. Dieser kann durch den SPD-Befehl wieder geändert werden.
- Die über den SPD-Befehl festgelegte Geschwindigkeit wird bei der Ausführung der END-Anweisung auf den Standardwert zurückgesetzt.

Steht in Beziehung zu folgenden Systemvariablen:

M_SPD/M_NSPD/M_RSPD

6.3.71 TITLE (Title)

Funktion: Programmtitel festlegen

Der Befehl legt einen Titel für ein Programm fest. Die in der Programmliste des Steuergerätes festgelegten Zeichen können über die PC-Support-Software oder COSIROP auf dem PC angezeigt werden. Der Befehl ist ab Software-Version J1 verfügbar.

Eingabeformat

```
TITLE □ "<Zeichenkette>"
```

<Zeichenkette> Legt den Programmtitel fest

Programmbeispiel

10	TITLE "ROBOT Loader Programm"	Programmtitel „ROBOT Loader Programm“ festlegen
20	MOV P1	Position P1 mittels Gelenk-Interpolation anfahren
30	MVS P2	Position P2 mittels Linear-Interpolation anfahren

Erläuterung

- Die maximale Zeichenzahl entspricht der für eine Programmzeile zulässigen Zeichenzahl. Allerdings können nur 20 Zeichen der Programmliste im Steuergerät über die Programmier-Software auf dem PC angezeigt werden.

6.3.72 TOOL (Tool)

Funktion: Werkzeug-Konvertierungsdaten

Der Befehl legt die Werkzeug-Konvertierungsdaten fest (Verschiebung des TCPs).

Eingabeformat

```
TOOL □ <Werkzeug-Konvertierungsdaten>
```

<Werkzeug-Konvertierungsdaten> Legt die Werkzeug-Konvertierungsdaten als Positionsausdruck fest (z. B. Positionskonstanten, Positionsvariablen usw.)

Programmbeispiel

Festlegung der Werkzeug-Konvertierungsdaten als numerischer Wert

10	TOOL (100,0,100,0,0,0)	Der TCP wird im Werkzeugkoordinatensystem um 100 mm in X-Richtung und um 100 mm in Z-Richtung verschoben.
20	MVS P1	Position P1 mittels Linear-Interpolation anfahren
30	TOOL P_NTOOL	Setzt Werkzeug-Konvertierungsdaten auf den Standardwert P_NTOOL

Festlegung der Werkzeug-Konvertierungsdaten als Variable

(Ist PTL01 = (100,0,100,0,0,0), entspricht dieses Beispiel dem Beispiel oben.)

10	TOOL PLT01	Der TCP wird im Werkzeugkoordinatensystem um 100 mm in X-Richtung und um 100 mm in Z-Richtung verschoben.
20	MVS P1	Position P1 mittels Linear-Interpolation anfahren

Erläuterung

- Der Befehl TOOL wird z. B. beim Einsatz von Doppelgreifern zur Festlegung des Werkzeugmittelpunktes an jeder Handspitze verwendet. Sind die Werkzeugmittelpunkte beider Handgreifer identisch, sollte die Festlegung über den Parameter MEXTL und nicht über den TOOL-Befehl erfolgen.
- Die mit dem TOOL-Befehl geänderten Werkzeug-Konvertierungsdaten werden im Parameter MEXTL gespeichert und bleiben auch nach Ausschalten der Spannungsversorgung erhalten.
- Es wird der Standardwert (P_NTOOL) verwendet, bis ein TOOL-Befehl ausgeführt wird. Ist der TOOL-Befehl ausgeführt, sind die Werkzeug-Konvertierungsdaten so lange gültig, bis der TOOL-Befehl erneut ausgeführt wird.
- Die mit dem TOOL-Befehl festgelegten Werkzeug-Konvertierungsdaten werden im Parameter MEXTL gespeichert. Der Wert bleibt auch nach Ausschalten der Spannungsversorgung des Steuergeräts erhalten.
- Werden beim Teachen und im Automatikbetrieb unterschiedliche Werkzeug-Konvertierungsdaten verwendet, kann der Roboter nicht vorhersehbare Positionen anfahren. Stellen Sie sicher, dass die Werte bei beiden Betriebsarten übereinstimmen.
- Die für den TOOL-Befehl zugelassenen Achsen sind vom Robotermodell abhängig (siehe Abschn. 9.7 „Standard-Werkzeugkoordinaten“).

Beispiel ▾ TOOL (0,65,145,0,0,0)

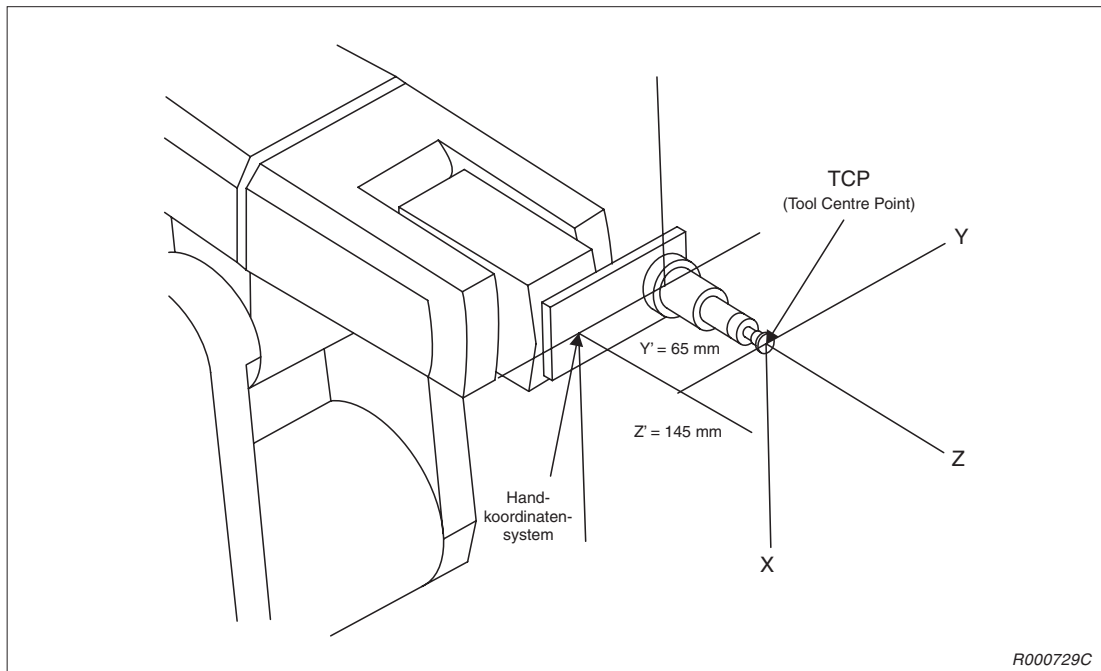


Abb. 6-21: Übergang vom Handkoordinatensystem zum Werkzeugkoordinatensystem TCP



Steht in Beziehung zu folgenden Parametern:

MEXTL, MEXTL1 bis 4 (siehe auch Abschn. 9.7)

Steht in Beziehung zu folgenden Systemvariablen:

P_NTOOL/P_TOOL

6.3.73 TORQ (Torque)

Funktion: Drehmomentgrenze definieren

Der Befehl legt die Drehmomentgrenze für eine Achse fest. Somit können Überlastungen vermieden werden. Beim Überschreiten der Drehmomentgrenze erfolgt die Meldung eines schweren Fehlers.

Eingabeformat

TORQ □ <Achsennummer> , <Drehmomentgrenze>

<Achsennummer>	Legt die Nummer der Achse fest $1 \leq \text{Achsennummer} \leq 6$
<Drehmomentgrenze>	Legt den prozentualen Grenzwert des Drehmoments, der von einer Achse erzeugt wird, fest $1 \leq \text{Grenzwert} \leq 100 \%$

Programmbeispiel

10 DEF ACT 1,M_FBD > 10 GOTO *SUB1,S	Definiert einen Unterprogramm- sprung zu Zeile 100, wenn die Abweichung zwischen der Soll- und der Istposition größer gleich 10 mm ist
20 ACT 1 = 1	Interrupt 1 freigeben
30 TORQ 3, 10	Legt die Drehmomentgrenze für Achse 3 auf 10 % des Maximalwerts fest
40 MVS P1	Position P1 mittels Linear- Interpolation anfahren
50 MOV P2	Position P2 mittels Linear- Interpolation anfahren
60 END	Programmende
100 *SUB1	Sprungmarke „SUB1“ festgelegt
110 ACT 1 = 0	Interrupt 1 sperren
120 MOV P_FBC	Soll- und Istposition abgleichen
140 M_OUT(10) = 1	Setzt das allgemeine Ausgangssignal 10 auf „1“
150 HLT	Programm stoppen, wenn die Abweichung zwischen Soll- und Istposition größer gleich 10 mm ist

Erläuterung

- Es wird das maximale Drehmoment für eine Achse festgelegt. Der Grenzwert wird in Prozent, bezogen auf den Standardwert, eingestellt. Der Standardwert ist durch den Hersteller vorgegeben.
- Der Bereich der Drehmomentbegrenzung ist vom Robotermodell abhängig. Die Einstellung wird für jede Servoachse vorgenommen und entspricht somit nicht zwingend der Drehmomentgrenze an der Handspitze des Roboters. Probieren Sie verschiedene Werte aus, bis Sie die gewünschte Einstellung gefunden haben.
- Bei einem Stopp des Roboters mit aktivierter Drehmomentbegrenzung, können Abweichungen zwischen Soll- und Istposition (aufgrund von Reibung usw.) auftreten. In diesem Fall erfolgt bei der Fortsetzung des Betriebs die Meldung eines schweren Fehlers. Gleichen Sie daher die Soll- und die Istposition ab, bevor Sie den Betrieb fortsetzen (siehe Zeile 110 im Programmbeispiel).
- Der Befehl kann nur für Standard-Roboterachsen verwendet werden und nicht für allgemeine Servoachsen (Zusatzachsen und benutzerdefinierte Achsen). Diese Drehmomentgrenzen sind servoseitig über Parameter einzustellen

Steht in Beziehung zu folgenden Systemvariablen:

P_FBC, M_FBD

6.3.74 WAIT (Wait)

Funktion: Wartestatus definieren

Der Befehl legt einen Wartestatus in Abhängigkeit von einer Variablen fest.

Eingabeformat

WAIT □ <numerische Variable>=<numerische Konstante>

<Numerische Variable>	Legt eine numerische Variable fest Es können auch Ein- und Ausgangsvariablen (z. B. M_IN, M_OUT) verwendet werden.
<Numerische Konstante>	Legt eine numerische Konstante fest

Programmbeispiel

Wartestatus in Abhängigkeit eines Signals

10 WAIT M_IN(1) = 1	Wartestatus, bis Eingangsbit 1 gleich 1 ist Entspricht der Befehlszeile 10 IF M_IN(1) = 0 THEN GOTO 10
20 WAIT M_IN(3) = 0	Wartestatus, bis Eingangsbit 3 gleich 0 ist

Wartestatus in Abhängigkeit einer Anwendung

30 WAIT M_RUN(2) = 1	Wartestatus, bis Betriebssignal der Anwendung 2 gleich 1 ist
----------------------	-----------------------------------------------------------------

Wartestatus in Abhängigkeit einer Variablen

40 WAIT M_01 = 1	Wartestatus, bis die externe Variable M_01 gleich 1 ist
------------------	------------------------------------------------------------

Erläuterung

- Der Befehl wird zur Unterbrechung eines Programms bis zu einer Signaleingabe und während des Multitaskings verwendet.
- Der WAIT-Befehl definiert einen Wartestatus. Die nächste Zeile wird erst dann ausgeführt, wenn die festgelegte Bedingung erfüllt ist.
- Im Multitask-Betrieb kann die Ausführung des WAIT-Befehls in mehreren Slots zu einer Verlängerung der Bearbeitungszeiten führen. Verwenden Sie in diesem Fall die IF-THEN-Anweisung anstelle des WAIT-Befehls.

Beispiel ▾

50 WAIT M_ABC = 0 ⇒ 50 IF M_ABC <> 0 THEN GOTO 50

△

6.3.75 WHILE ~ WEND (While End)

Funktion: Programmschleife

Solange die Schleifenbedingung „wahr“ ist, wird das Programm zwischen der WHILE- und der WEND-Anweisung wiederholt.

Eingabeformat

```
WHILE □ <Schleifenbedingung>
:
WEND
```

<Schleifenbedingung> Legt die Abarbeitung der Schleife über eine Vergleichsbedingung fest

Programmbeispiel

10	WHILE (M1 >= -5) AND (M1 <= 5)	Wiederholt den Programmblock, solange M1 zwischen -5 und +5 liegt und springt zu Zeile 50, wenn M1 außerhalb des Wertebereichs liegt
20	M1 = -(M1 + 1)	Addiert 1 zu M1 und kehrt das Vorzeichen um
30	PRINT #1,M1	Gibt den Wert von M1 aus
40	WEND	Springt zurück zur WHILE-Anweisung (Zeile 10)
50	END	Programmende

Erläuterung

- Der Programmblock zwischen WHILE und WEND wird wiederholt, solange die Schleifenbedingung „wahr“ (M1 zwischen -5 bis +5) ist.
- Ist die Schleifenbedingung „unwahr“ (M1 außerhalb von -5 bis +5), springt das Programm eine Zeile hinter die WEND-Anweisung.
- Ein Sprung über die GOTO-Anweisung aus einer WHILE-WEND-Schleife belegt Speicherplatz des für die Programmsteuerung reservierten Stapelspeichers. Bei einer kontinuierlichen Ausführung des Programms kann deshalb eine Fehlermeldung erfolgen. Bauen Sie das Programm so auf, dass die Schleife nur dann durchlaufen wird, wenn die Bedingung der WHILE-Anweisung erfüllt ist.

6.3.77 WTHIF (With If)

Funktion: Anweisung hinzufügen, wenn ...

Während einer Interpolationsbewegung wird eine bedingte, zusätzliche Anweisung ausgeführt.

Eingabeformat

```
WTHIF □ <Bedingung> , <Anweisung>
```

<Bedingung>	Legt die Bedingung fest, bei der die zusätzliche Anweisung ausgeführt wird (siehe auch ACT)
<Anweisung>	Legt die zusätzlich ausgeführte Anweisung fest (siehe auch WTH) Es dürfen folgende Operationen ausgeführt werden: <ul style="list-style-type: none"> – <num. Datentyp B><Substitutionsoperator><num. Datentyp A> Bsp.: M_OUT(1) = 1, P1 = P2 – HLT-Anweisung – SKIP-Anweisung

Programmbeispiel

10 MOV P1 WTHIF M_IN(17) = 1,HLT	Position 1 anfahren und Programm stoppen, falls das Eingangsbit Nummer 17 gleich 1 ist
20 MVS P2 WTHIF M_RSPD>200,M_OUT(17) = 1 DLY M1 + 2	Position 2 anfahren und das Ausgangsbit Nummer 17 für die Zeit von (M1 + 2) Sekunden auf „1“ setzen, falls die aktuelle Geschwindigkeit 200 mm/s übersteigt
30 MVS P3 WTHIF M_RATIO>15,M_OUT(1) = 1	Position 3 anfahren und das Ausgangsbit Nummer 1 auf „1“ setzen, sobald 15 % des Fahrweges zurückgelegt worden sind

Erläuterung

- Dieser Befehl wird dazu verwendet, während einer Interpolationsbewegung eine zusätzliche, bedingte Anweisung auszuführen.
- Die Anweisung wird mit Beginn der Roboterbewegung ausgeführt.
- In der Anweisung darf kein DLY-Befehl verwendet werden.

6.3.78 XCLR (X Clear)

Funktion: Programmauswahl zurücksetzen

Der Befehl setzt die Auswahl eines Programms in einen festgelegten Programmplatz (Slot/Task) während des Multitask-Betriebs zurück.

Eingabeformat

```
XCLR □ <Programmplatznummer>
```

<Programmplatznummer> Legt die Nummer des Programmplatzes fest

Programmbeispiel

10	XRUN 2,"1"	Auswahl des Programms 1 für Programmplatz 2
	:	
100	XSTP 2	Stoppt Programmplatz 2
110	WAIT M_WAI(2) = 1	Wartestatus, bis Programmplatz 2 gestoppt ist
150	XRST 2	Wartestatus von Programmplatz 2 zurücksetzen
	:	
200	XCLR 2	Auswahl von Programmplatz 2 zurücksetzen
210	END	Programmende

Erläuterung

- Entspricht die Programmplatznummer nicht dem ausgewählten Programmplatz, erfolgt eine Fehlermeldung.
- Wird das ausgewählte Programm während der Ausführung zurückgesetzt, erfolgt eine Fehlermeldung.
- Wird ein Programm zurückgesetzt, das sich im Wartestatus befindet, erfolgt eine Fehlermeldung.
- Die Festlegung des Programmnamens erfolgt in Anführungszeichen.
- In einem kontinuierlich ausgeführten Programm wird der Befehl freigegeben, wenn Parameter ALWENA von „0“ auf „7“ umgestellt und anschließend die Spannungsversorgung aus- und wieder eingeschaltet wird.

Steht in Beziehung zu folgenden Befehlen:

XLOAD, XRST, XRUN, XSTP

Steht in Beziehung zu folgenden Parametern:

ALWENA

6.3.79 XLOAD (X Load)

Funktion: Programm laden

Der Befehl lädt im Multitask-Betrieb ein Programm in einen festgelegten Programmplatz (Slot/Task).

Eingabeformat

```
XLOAD □ <Programmplatznummer> <Programmname>
```

<Programmplatznummer> Legt die Nummer des Programmplatzes fest

<Programmname> Legt den Namen des Programms fest

Programmbeispiel

10	IF M_PSA(2) = 0 THEN 60	Sprung zur Zeile 60, falls die Programmwählbarkeit des Programmplatzes 2 gesperrt ist
20	XLOAD 2,"10"	Auswahl des Programms 10 für Programmplatz 2
30	IF C_PRG(2) <> "10" THEN GOTO 30	Wartezeit bis das Programm gestartet wird
40	XRUN 2	Startet Programmplatz 2
50	WAIT M_RUN(2) = 1	Wartestatus, bis das Betriebssignal des Programmplatzes 2 gleich 1 ist
60		
70		Ist der Programmplatz 2 bereits aktiv, startet die Programmausführung hier

Erläuterung

- Es erfolgt eine Fehlermeldung, wenn das ausgewählte Programm nicht existiert.
- Ist das gewählte Programm bereits einem anderen Programmplatz zugewiesen, erfolgt bei der Programmausführung eine Fehlermeldung.
- Ist das gewählte Programm editiert worden, erfolgt bei der Programmausführung eine Fehlermeldung.
- Wird das gewählte Programm bereits ausgeführt, erfolgt bei der erneuten Programmausführung eine Fehlermeldung.
- Die Festlegung des Programmnamens erfolgt in Anführungszeichen.
- In einem kontinuierlich ausgeführten Programm wird der Befehl freigegeben, wenn Parameter ALWENA von „0“ auf „7“ umgestellt und anschließend die Spannungsversorgung aus- und wieder eingeschaltet wird.
- Wird der Befehl XRUN unmittelbar nach dem Befehl XLOAD ausgeführt, kann ein Fehler auftreten, da der Ladevorgang des Programms noch nicht abgeschlossen ist. Prüfen Sie daher den Abschluss des Ladevorgangs (siehe Zeile 30 im Programmbeispiel).

Steht in Beziehung zu folgenden Befehlen:

XCLR, XRST, XRUN, XSTP

Steht in Beziehung zu folgenden Parametern:

ALWENA

6.3.80 XRST (X Reset)

Funktion: Programm zurücksetzen

Der Befehl setzt die Steuerung des Programms des festgelegten Programmplatzes im Multi-task-Betrieb von der aktuellen Zeile auf den Programmanfang zurück.

Eingabeformat

```
XRST □ <Programmplatznummer>
```

<Programmplatznummer> Legt die Nummer des Programmplatzes fest

Programmbeispiel

10	XRUN 2	Startet Programmplatz 2
20	WAIT M_RUN(2) = 1	Wartestatus, bis Betriebssignal des Programmplatzes 2 gleich 1 ist
	:	
	:	
100	XSTP 2	Stoppt Programmplatz 2
110	WAIT M_WAI(2) = 1	Wartestatus, bis Programmplatz 2 gestoppt ist
	:	
150	XRST 2	Programmplatz 2 zurücksetzen
160	WAIT M_PSA(2) = 1	Wartestatus, bis Programmplatz 2 gewählt wurde
	:	
200	XRUN 2	Startet Programmplatz 2
210	WAIT M_RUN(2) = 1	Wartestatus, bis Betriebssignal des Programmplatzes 2 gleich 1 ist
	:	

Erläuterung

- Ein Zurücksetzen des Programms ist nur im gestoppten Zustand des Programmplatzes möglich.
- In einem kontinuierlich ausgeführten Programm wird der Befehl freigegeben, wenn Parameter ALWENA von „0“ auf „7“ umgestellt und anschließend die Spannungsversorgung aus- und wieder eingeschaltet wird.

Steht in Beziehung zu folgenden Befehlen:

XCLR, XLOAD, XRUN, XSTP

Steht in Beziehung zu folgenden Parametern:

ALWENA

Steht in Beziehung zu folgenden Systemvariablen:

M_PSA(Programmplatznummer) (1 = Auswahl freigegeben, 0 = Auswahl gesperrt)
 M_RUN(Programmplatznummer) (1 = Betrieb, 0 = kein Betrieb)
 M_WAI(Programmplatznummer) (1 = Pause, 0 = keine Pause)

6.3.81 XRUN (X Run)

Funktion: Programm starten

Der Befehl startet die parallele Ausführung der gewählten Programme im Multitask-Betrieb.

Eingabeformat

```
XRUN □ <Programmplatznummer> <Programmname> [, <Ausführung>]
```

<Programmplatznummer> Legt die Nummer des Programmplatzes fest

<Programmname> Legt den Namen des Programms fest

<Ausführung> Legt die Ausführung des Programms fest
0 = kontinuierlich, 1 = zyklisch

Programmbeispiel

Wahl des auszuführenden Programms über den Befehl XRUN (kontinuierliche Ausführung)

```
10 XRUN 2,"1"           Startet Programm 1 als Programmplatz 2
20 WAIT M_RUN(2) = 1    Wartestatus, bis Betriebssignal des Programmplatzes 2
                        gleich 1 ist
```

Wahl des auszuführenden Programms über den Befehl XRUN (zyklische Ausführung)

```
10 XRUN 3,"2",1        Startet Programm 2 als Programmplatz 3 im zyklischen
                        Betrieb
20 WAIT M_RUN(3) = 1    Wartestatus, bis Betriebssignal des Programmplatzes 3
                        gleich 1 ist
```

Wahl des auszuführenden Programms über den Befehl XLOAD (kontinuierliche Ausführung)

```
10 XLOAD 2,"1"         Auswahl des Programms 1 für
                        Programmplatz 2
20 IF C_PRG(2) <> "1" THEN GOTO 20    Wartezeit bis das Programm gestartet wird
30 XRUN 2               Startet Programmplatz 2
```

Wahl des auszuführenden Programms über den Befehl XLOAD (zyklische Ausführung)

```
10 XLOAD 3,"2"         Auswahl des Programms 2 für
                        Programmplatz 3
20 IF C_PRG(3) <> "1" THEN GOTO 20    Wartezeit bis das Programm gestartet wird
30 XRUN 3, ,1          Startet Programmplatz 3 im zyklischen
                        Betrieb
```

Erläuterung

- Es erfolgt eine Fehlermeldung, wenn das ausgewählte Programm nicht existiert.
- Wird der gewählte Programmplatz bereits verwendet, erfolgt bei der Programmausführung eine Fehlermeldung.
- Ist ein Programm noch in keinen Programmplatz geladen, erfolgt das Laden über diesen Befehl. Der Befehl XLOAD kann dann entfallen.
- Wird der XRUN-Befehl im Wartestatus eines in der Programmmitte gestoppten Programms ausgeführt, startet der Betrieb im kontinuierlichen Modus.
- Die Festlegung des Programmnamens erfolgt in Anführungszeichen.
- Fehlt die Angabe für die Ausführung, wird das Programm im aktuellen Modus ausgeführt.
- In einem kontinuierlich ausgeführten Programm wird der Befehl freigegeben, wenn Parameter ALWENA von „0“ auf „7“ umgestellt und anschließend die Spannungsversorgung aus- und wieder eingeschaltet wird.
- Wird der Befehl XRUN unmittelbar nach dem Befehl XLOAD ausgeführt, kann ein Fehler auftreten, da der Ladevorgang des Programms noch nicht abgeschlossen ist. Prüfen Sie daher den Abschluss des Ladevorgangs (siehe Zeile 20 im 3ten und 4ten Programmbeispiel).

Steht in Beziehung zu folgenden Befehlen:

XCLR, XLOAD, XRST, XSTP

Steht in Beziehung zu folgenden Parametern:

ALWENA

Steht in Beziehung zu folgenden Systemvariablen:

M_RUN(Programmplatznummer) (1 = Betrieb, 0 = kein Betrieb)

6.3.82 XSTP (X Stop)

Funktion: Programm stoppen

Der Befehl unterbricht die Ausführung des Programms des gewählten Programmplatzes (Slot/Task) im Multitask-Betrieb. Eine über den Programmplatz ausgeführte Verfahrbewegung des Roboters wird gestoppt.

Eingabeformat

```
XSTP □ <Programmplatznummer>
```

<Programmplatznummer> Legt die Nummer des Programmplatzes fest

Programmbeispiel

10	XRUN 2	Startet Programmplatz 2
	:	
100	XSTP 2	Stoppt Programmplatz 2
110	WAIT M_WAI(2) = 1	Wartestatus, bis Programmplatz 2 gestoppt ist
	:	
200	XRUN 2	Startet Programmplatz 2

Erläuterung

- Wird ein bereits gestoppter Programmplatz gestoppt, erfolgt keine Fehlermeldung.
- Der XSTP-Befehl stoppt auch ein kontinuierlich ausgeführtes Programm.
- In einem kontinuierlich ausgeführten Programm wird der Befehl freigegeben, wenn Parameter ALWENA von „0“ auf „7“ umgestellt und anschließend die Spannungsversorgung aus- und wieder eingeschaltet wird.

Steht in Beziehung zu folgenden Befehlen:

XCLR, XLOAD, XRST, XRUN

Steht in Beziehung zu folgenden Parametern:

ALWENA

Steht in Beziehung zu folgenden Systemvariablen:

M_WAI(Programmplatznummer) (1 = Pause, 0 = keine Pause)

6.3.83 SUBSTITUTE (Substitute)

Funktion: Daten ersetzen

Das Ergebnis einer Operation wird in eine Variable oder Feldvariable übertragen.

Eingabeformat 1

```
<Variablenname> = <Ausdruck 1>
```

Eingabeformat 2

```
<Variablenname> = <Ausdruck 1> DLY <Ausdruck 2>
```

<Variablenname> Legt den Namen der Variablen fest, in die die Daten übertragen werden (siehe auch Syntaxdiagramme der Variablentypen)

<Ausdruck 1> Daten, die in die Variable übertragen werden

<Ausdruck 2> Legt die Verzögerungszeit fest

Programmbeispiel

10	P100 = P1 + P2 * 2	Überträgt das Ergebnis der Operation P1 + P2 * 2 in die Variable P100
20	M_OUT(10) = 1	Setzt Ausgang 10 auf „1“
20	M_OUT(17) = 1 DLY 2.0	Setzt Ausgangsbit Nummer 17 für 2 Sekunden auf „1“

Erläuterung

- Wird der Befehl für einen Impulsausgang verwendet, wird der Impuls parallel zu den Befehlen der nachfolgenden Zeilen geschaltet.
- Bei einer Impulsausgabe über die Variablen M_OUTB oder M_OUTW werden die Bits in Gruppen von 8 oder 15 Bits invertiert. Es kann nicht jede beliebige Bitbreite ausgegeben werden.
- Wird während der festgesetzten Zeit eine END-Anweisung, die letzte Zeile des Programms oder ein NOT-HALT ausgeführt, behält der Impulsausgang seinen gegenwärtigen Zustand bei. Nach der abgelaufenen Verzögerungszeit werden die Bits wieder invertiert.

7 Roboterstatusvariablen

7.1 Allgemeine Hinweise

In den nachfolgenden Abschnitten finden Sie eine alphabetische Auflistung aller Roboterstatusvariablen und deren Anwendungsmöglichkeiten.

7.1.1 Beschreibung des verwendeten Formats

Funktion

Hier finden Sie eine Funktionsbeschreibung der Variablen.

Eingabeformat

Hier finden Sie das genaue Format zur Eingabe der Variablen. Die eckigen Klammern „[]“ kennzeichnen die wahlfreien Variablenwerte.

Systemstatusvariablen können in Vergleichsoperationen, als Bezugsparameter oder in Zuweisungsanweisungen verwendet werden. Die hier gezeigten Beispiele beschränken sich auf Anwendungen als Bezugsparameter oder Zuweisungsanweisungen.

Programmbeispiel

Hier finden Sie die Verwendung der Variablen in einem Beispielprogramm.

Erläuterung

Hier finden Sie eine detaillierte Beschreibung, Besonderheiten usw. der Variablen.

7.2 Detaillierte Variablenbeschreibung

In diesem Abschnitt finden Sie eine detaillierte Beschreibung sowie Programmbeispiele zur Anwendung der Roboterstatusvariablen.

7.2.1 C_DATE

Funktion: Datumseinstellungen lesen

Die Variable enthält die aktuellen Datumseinstellungen im Format Jahr/Monat/Tag.

Eingabeformat

```
Bsp.: <Zeichenkettensvariable> = C_DATE
```

Programmbeispiel

```
10 C1$ = C_DATE
```

Überträgt das aktuelle Datum, z. B. 2005/12/01 in die Zeichenkettensvariable C1\$

Erläuterung

- Die aktuellen Datumseinstellungen werden übertragen.
- Die Variable kann ausschließlich gelesen werden. Eine Änderung der Datumseinstellungen erfolgt über die Teaching Box.

Steht in Beziehung zu folgenden Variablen:

C_TIME

7.2.2 C_MAKER

Funktion: Herstellerangaben lesen

Die Variable enthält Herstellerangaben des Roboter-Steuergerätes.

Eingabeformat

```
Bsp.: <Zeichenkettensvariable> = C_MAKER
```

Programmbeispiel

```
10 C1$ = C_MAKER
```

Überträgt z. B. „COPYRIGHT 2002“ in die Zeichenkettensvariable C1\$

Erläuterung

- Es werden Herstellerangaben des Roboter-Steuergeräts übertragen.
- Die Variable kann ausschließlich gelesen werden.

Steht in Beziehung zu folgenden Variablen:

C_MECHA

7.2.3 C_MECHA

Funktion: Mechanismusangaben lesen

Die Variable enthält die Bezeichnung des verwendeten Mechanismus.

Eingabeformat

```
Bsp.: <Zeichenkettensvariable> = C_MECHA [( <Mechanismusnummer> )]
```

<Zeichenkettensvariable>

Legt eine Zeichenkettensvariable fest

<Mechanismusnummer>

Legt die Mechanismusnummer fest

$1 \leq \text{Mechanismusnummer} \leq 3$

Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

```
10 C1$ = C_MECHA
```

Falls der Robotertyp „RV-4A“ ist, wird „RV-4A“ in die Zeichenkettensvariable C1\$ übertragen.

Erläuterung

- Der aktuelle Mechanismustyp wird übertragen.
- Die Variable kann ausschließlich gelesen werden.

7.2.4 C_PRG

Funktion: Programmnamen lesen

Die Variable enthält die Programmnummer des ausgewählten Programms.

Eingabeformat

Bsp.: <Zeichenkettenvariable> = C_PRG [(<Numerischer Ausdruck>)]

<Zeichenkettenvariable>	Legt eine Zeichenkettenvariable fest
<Numerischer Ausdruck>	Legt die Programmplatznummer fest $1 \leq \text{Programmplatznummer} \leq 32$ Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

10 C1\$ = C_PRG(1) Falls die Programmnummer im Programmplatz 1 „10“ ist, wird der Wert „10“ in die Zeichenkettenvariable C1\$ übertragen

Erläuterung

- Die Programmnummer des ausgewähltem Programmplatzes wird übertragen.
- Im Einzelprogrammplatzbetrieb wird die Programmplatznummer auf „1“ gesetzt.
- Bei einer Auswahl des Programms über das Steuergerät wird dieser Wert gesetzt.
- Die Variable kann ausschließlich gelesen werden.
- Ist im ausgewählten Programmplatz kein Programm geladen, erfolgt bei Ausführung der Anweisung eine Fehlermeldung.

7.2.5 C_TIME

Funktion: Zeiteinstellungen lesen

Die Variable enthält die aktuellen Zeiteinstellungen in Stunden/Minuten/Sekunden im 24-Stunden-Format.

Eingabeformat

```
Bsp.: <Zeichenkettvariable> = C_TIME
```

Programmbeispiel

```
10 C1$ = C_TIME
```

 Überträgt die aktuelle Zeit, z. B. 01/05/20 in die Zeichenkettvariable C1\$

Erläuterung

- Die aktuellen Zeiteinstellungen werden übertragen.
- Die Variable kann ausschließlich gelesen werden. Eine Änderung der Datumseinstellungen erfolgt über die Teaching Box.

Steht in Beziehung zu folgenden Variablen:

C_DATE

7.2.6 C_USER

Funktion: Benutzerinformation lesen

Die Variable enthält die Zeichen des Parameters USERMSG.

Eingabeformat

```
Bsp.: <Zeichenkettvariable> = C_USER
```

Programmbeispiel

```
10 C1$ = C_USER
```

 Überträgt die Zeichen des Parameters USERMSG in die Zeichenkettvariable C1\$

Erläuterung

- Die Zeichen des Parameters USERMSG werden übertragen.
- Die Variable kann ausschließlich gelesen werden.
- Eine Änderung des Parameters erfolgt über die PC-Support-Software, über COSIROP oder die Teaching Box.

7.2.7 J_COLMXL

Funktion: Drehmomentabweichung lesen

Die Variable enthält die maximale Drehmomentabweichung zwischen dem Drehmoment-Istwert und dem Drehmoment-Sollwert bei aktivierter Kollisionsüberwachung.

Die Funktion der Kollisionsüberwachung ist ab Software-Version J2 verfügbar und kann nur mit den Robotermodellen RV-3SB/3SJB, RV-6S/6SL/12S/12SL und RH-6SH/12SH verwendet werden.

Eingabeformat

Bsp.: <Gelenkvariable> = J_COLMXL [(<Mechanismusnummer>)]

<Gelenkvariable>

Legt eine Gelenkvariable fest
Auch bei einer Impulskette wird eine Gelenkvariable verwendet.

<Mechanismusnummer>

Legt die Mechanismusnummer fest
 $1 \leq \text{Mechanismusnummer} \leq 3$
Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

10 M1 = 100

Initialisierungswert der Ansprechschwelle für die Kollisionsüberwachung für alle Achsen vorgeben

20 M2 = 100

30 M3 = 100

40 M4 = 100

50 M5 = 100

60 M6 = 100

70 COLLVL M1,M2,M3,M4,M5,M6,,

Ansprechschwelle für die Kollisionsüberwachung für alle Achsen einstellen

80 COLCHK ON

Kollisionsüberwachung aktivieren
(Berechnung der maximalen Drehmomentabweichung starten)

90 MOV P1

Position P1 mittels Gelenk-Interpolation anfahren

:

500 COLCHK OFF

Kollisionsüberwachung deaktivieren
(Berechnung der maximalen Drehmomentabweichung stoppen)

510 M1 = J_COLMXL(1).J1 + 10

Zulässige Ansprechschwelle mit einer Abweichung von 10 % für jede Achse berechnen

520 M2 = J_COLMXL(1).J2 + 10

530 M3 = J_COLMXL(1).J3 + 10

540 M4 = J_COLMXL(1).J4 + 10

550 M5 = J_COLMXL(1).J5 + 10

560 M6 = J_COLMXL(1).J6 + 10

570 GOTO 70

Erläuterung

- Die maximale Drehmomentabweichung zwischen dem Drehmoment-Istwert und dem Drehmoment-Sollwert bei aktivierter Kollisionsüberwachung wird in der Variablen J_COLMXL gespeichert.

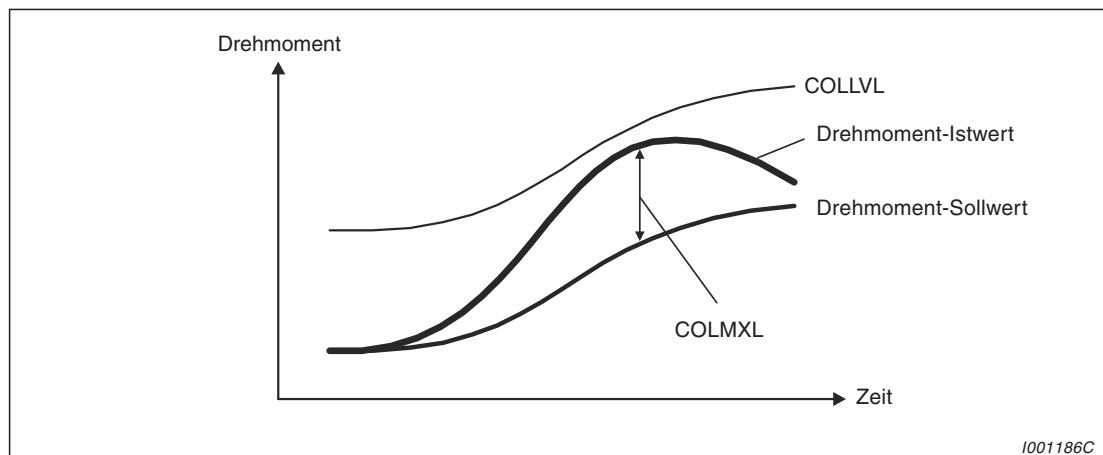


Abb. 7-1: Drehmomentabweichung zwischen Drehmoment-Istwert und -Sollwert

- Ein Wert der Variablen von 100 % entspricht dem zulässigen Wert der Drehmomentabweichung in der Werkseinstellung.
- Bei Robotern, die nicht über die Funktion der Kollisionsüberwachung verfügen, wird der Wert für jede Achse auf „0.0“ gesetzt.
- Wird die Servoversorgung während der Ausführung der Anweisung COLCHK ON oder COLLVL eingeschaltet, erfolgt ein Zurücksetzen der Variablen auf den Wert „0“.

Steht in Beziehung zu folgenden Befehlen:

COLCHK, COLLVL

Steht in Beziehung zu folgenden Variablen:

M_COLSTS, P_COLDIR

7.2.8 J_CURR

Funktion: Gelenkdaten lesen

Die Variable enthält die Gelenkdaten der aktuellen Position.

Eingabeformat

Bsp.: <Gelenkvariable> = J_CURR [(<Mechanismusnummer>)]

<Gelenkvariable>	Legt eine Gelenkvariable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest $1 \leq \text{Mechanismusnummer} \leq 3$ Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

10 J1 = J_CURR Überträgt die Gelenkdaten der aktuellen Position in die Gelenkvariable J1

Erläuterung

- Die Gelenkdaten der aktuellen Position des festgelegten Roboters werden übertragen.
- Die Variable kann ausschließlich gelesen werden.

Steht in Beziehung zu folgenden Variablen:

P_CURR

7.2.9 J_ECURRE

Funktion: Encoderdaten lesen

Die Variable enthält die Zahl der Encoderimpulse.

Eingabeformat

Bsp.: <Gelenkvariable> = J_ECURRE [(<Mechanismusnummer>)]

<Gelenkvariable>	Legt eine Gelenkvariable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest $1 \leq \text{Mechanismusnummer} \leq 3$ Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

10 J1 = J_ECURRE	Überträgt die Anzahl der Encoderimpulse in die Gelenkvariable J1
20 MA = J1, 1	Überträgt die Anzahl der Encoderimpulse der Achse J1 in die Variable MA

Erläuterung

- Die Anzahl der Encoderimpulse wird in eine Gelenkvariable übertragen. Legen Sie dann eine Achse fest und übertragen Sie die Encoderdaten in eine numerische Variable.
- Die Variable kann ausschließlich gelesen werden.

7.2.10 J_FBC/J_AMPFBC

Funktion: Gelenkdaten lesen

J_FBC: Die Variable enthält Gelenkdaten der aus den Servorückmeldeimpulsen abgeleiteten aktuellen Position.

J_AMPFBC: Die Variable enthält den aktuellen Wert der Servorückmeldeimpulse jeder Achse.

Eingabeformat

Bsp.: <Gelenkvariable> = J_FBC [(<Mechanismusnummer>)]

Bsp.: <Gelenkvariable> = J_AMPFBC [(<Mechanismusnummer>)]

<Gelenkvariable>

Legt eine Gelenkvariable fest

<Mechanismusnummer>

Legt die Mechanismusnummer fest

$1 \leq \text{Mechanismusnummer} \leq 3$

Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

10 J1 = J_FBC

Überträgt die Gelenkdaten der aktuellen Position, die aus den Servorückmeldeimpulsen ermittelt wurden, in die Gelenkvariable J1

20 J2 = J_AMPFBC

Überträgt den aktuellen Wert der Servorückmeldeimpulse in die Gelenkvariable J2

Erläuterung

- Die Variable J_FBC enthält die Gelenkdaten der aus den Encoderimpulsen ermittelten aktuellen Position des festgelegten Mechanismus.
- Die Variable J_FBC ermöglicht eine Prüfung der Verzögerungszeit zwischen den an die Servos übermittelten Steuerimpulsen und der eigentlichen Servoaktivität.
- Die Variable J_FBC ermöglicht die Erkennung einer durch die Anweisung CMP JNT hervorgerufenen Abweichung.
- Die Variable kann ausschließlich gelesen werden.

Steht in Beziehung zu folgenden Variablen:

P_FBC

7.2.11 J_ORIGIN

Funktion: Grundpositionsdaten lesen

Die Variable enthält die Gelenkdaten der Grundposition (Nullpunkt).

Eingabeformat

```
Bsp.: <Gelenkvariable> = J_ORIGIN [( <Mechanismusnummer> )]
```

<Gelenkvariable>	Legt eine Gelenkvariable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest $1 \leq \text{Mechanismusnummer} \leq 3$ Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

```
10 J1 = J_ORIGIN
```

Überträgt die Gelenkdaten der Grundposition in die Gelenkvariable J1

Erläuterung

- Die Gelenkdaten der eingestellten Grundposition werden übertragen.
- Die Variable ermöglicht eine Prüfung der Grundposition, z. B. bei Positionsabweichungen.
- Die Variable kann ausschließlich gelesen werden.

7.2.12 M_ACL/M_DACL/M_NACL/M_NDAACL/M_ACLSTS

Funktion: Beschleunigungs-/Abbremszeiten lesen

Die Variable enthält für die Beschleunigung/Abbremsung relevante Daten.

M_ACL: aktuelle Beschleunigungszeit (%)
 M_DACL: aktuelle Abbremszeit (%)
 M_NACL: Standardwert der Beschleunigungszeit (100 %)
 M_NDAACL: Standardwert der Abbremszeit (100 %)
 M_ACLSTS: aktueller Status der Beschleunigung/Abbremsung
 0 = gestoppt
 1 = beschleunigt
 2 = konstante Geschwindigkeit
 3 = bremst

Eingabeformat

```
Bsp.: <Numerische Variable> = M_ACL [( <Numerischer Ausdruck> ) ]
Bsp.: <Numerische Variable> = M_DACL [( <Numerischer Ausdruck> ) ]
Bsp.: <Numerische Variable> = M_NACL [( <Numerischer Ausdruck> ) ]
Bsp.: <Numerische Variable> = M_NDAACL [( <Numerischer Ausdruck> ) ]
Bsp.: <Numerische Variable> = M_ACLSTS [( <Numerischer Ausdruck> ) ]
```

<Numerische Variable>	Legt eine numerische Variable fest
<Numerischer Ausdruck>	Legt die Programmplatznummer fest $1 \leq \text{Programmplatznummer} \leq 32$ Bei fehlender Angabe wird der aktuelle Programmplatz gesetzt.

Programmeispiel

10 M1 = M_ACL	Überträgt die Beschleunigungszeit von Programmplatz 1 in die numerische Variable M1
20 M1 = M_DACL(2)	Überträgt die Abbremszeit von Programmplatz 2 in die numerische Variable M1
30 M1 = M_NACL	Überträgt den Standardwert der Beschleunigungszeit von Programmplatz 1 in die numerische Variable M1
40 M1 = M_NDAACL(2)	Überträgt den Standardwert der Abbremszeit von Programmplatz 2 in die numerische Variable M1
50 M1 = M_ACLSTS(3)	Überträgt den Status der Beschleunigung/Abbremsung von Programmplatz 3 in die numerische Variable M1

Erläuterung

- Die Beschleunigungs-/Abbremszeit wird als prozentualer Wert bezogen auf die maximale Beschleunigung/Abbremsung (Standardwert) des Roboters angegeben. Bei einem Wert von 50 % verdoppelt sich folglich die Beschleunigungs-/Abbremszeit, d. h. die Beschleunigung/Abbremsung ist geringer.
- Der Wert der Variablen M_NACL und M_NDAACL ist 100 %.
- Die Variablen können ausschließlich gelesen werden.

7.2.13 M_BRKCQ

Funktion: BREAK-Befehlsausführung lesen

Die Variable zeigt die Ausführung eines BREAK-Befehls an.

1: BREAK-Befehl wurde ausgeführt.

0: BREAK-Befehl wurde nicht ausgeführt.

Eingabeformat

Bsp.: <Numerische Variable> = M_BRKCQ [(<Numerischer Ausdruck>)]

<Numerische Variable>

Legt eine numerische Variable fest

<Numerischer Ausdruck>

Legt die Programmplatznummer fest
 $1 \leq \text{Programmplatznummer} \leq 32$
 Bei fehlender Angabe wird der aktuelle Programmplatz gesetzt.

Programmbeispiel

10 WHILE M1 <> 0

Wiederholt den Programmblock, solange M1 ungleich 0 ist und springt zu Zeile 40, wenn M1 gleich 0 ist

20 IF M2 = 0 THEN BREAK

Springt zur Zeile 40, falls M2 gleich 0 ist

30 WEND

Springt zurück zur WHILE-Anweisung (Zeile 10)

40 IF M_BRKCQ THEN HLT

Stoppt das Programm, wenn der BREAK-Befehl im WHILE-WEND-Programmblock ausgeführt wurde

Erläuterung

- Mit Hilfe der Variablen kann geprüft werden, ob ein BREAK-Befehl ausgeführt worden ist.
- Die Variable kann ausschließlich gelesen werden.
- Nach einem einmaligen Zugriff auf die Variable M_BRKCQ wird der BREAK-Status zurückgesetzt (d. h. die Variable wird auf „0“ gesetzt). Soll der Wert weiterhin erhalten bleiben, muss er in einer numerischen Variablen gespeichert werden.
- Der BREAK-Status wird auch dann zurückgesetzt, wenn ein Zugriff auf die Variable zur Anzeige auf der Teaching Box o. Ä. erfolgt.

7.2.14 M_BTIME

Funktion: Batterierestzeit lesen

Die Variable enthält die Restzeit der Batterie in Stunden.

Eingabeformat

```
Bsp.: <Numerische Variable> = M_BTIME
```

<Numerische Variable>

Legt eine numerische Variable fest

Programmbeispiel

```
10 M1 = M_BTIME
```

Überträgt die Restlaufzeit der Batterie in die numerische Variable M1

Erläuterung

- Die verbleibende Lebensdauer der Batterie wird übertragen.
- Als Standardwert für eine neue Batterie werden 14600 Stunden vorgegeben.
- Die Differenz – 14600 Stunden minus der kumulierten Einschaltzeit des Steuergerätes – wird in die Variable übertragen.
- Die Variable kann ausschließlich gelesen werden.
- Bei einem RESET wird der Wert auf 14600 zurückgesetzt.

7.2.15 M_CMPDST

Funktion: Positionsabweichung lesen

Die Variable enthält die Abweichung zwischen der Zielposition und der aktuellen Position nach Aktivierung der Achsenweichheit.

Eingabeformat

Bsp.: <Numerische Variable> = M_CMPDST [(<Mechanismusnummer>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest $1 \leq \text{Mechanismusnummer} \leq 3$ Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

10 MOV P1	Position P1 mittels Gelenk-Interpolation anfahren
20 CMPG 0.5,0.5,1.0,0.5,0.5, , ,	Einstellung der Weichheit
30 CMP POS, &B00011011	Achsenweichheit für die Richtungen X, Y und die Orientierungen A und B aktivieren
40 MVS P2	Position P2 mittels Linear-Interpolation anfahren
50 M_OUT(10) = 1	Setzt Ausgangsbit 10 auf „1“
60 MVS P2	Position P2 mittels Linear-Interpolation anfahren
70 M1 = M_CMPDST(1)	Setzt die Variable M1 auf den Wert der Abweichung zwischen der aktuellen Position und der Zielposition des Mechanismus 1
80 CMP OFF	Achsenweichheit deaktivieren

Erläuterung

- Die Variable wird bei aktivierter Achsenweichheit zur Überprüfung der Abweichung zwischen der aktuellen und der Zielposition verwendet.
- Die Variable kann ausschließlich gelesen werden.

7.2.17 M_COLSTS

Funktion: Status der Kollisionsüberwachung melden

Die Variable zeigt den Status der Kollisionsüberwachung an.

- 1: Kollision
- 0: keine Kollision

Die Variable M_COLSTS ist ab Software-Version J2 verfügbar und kann nur mit den Robotermodellen RV-3SB/3SJB, RV-6S/6SL/12S/12SL und RH-6SH/12SH verwendet werden.

Eingabeformat

Bsp.: DEF ACT 1, M_COLSTS [<Mechanismusnummer>] = 1 GOTO *LCOL

<Mechanismusnummer> Legt die Mechanismusnummer fest
 $1 \leq \text{Mechanismusnummer} \leq 3$
 Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

10	DEF ACT 1,M_COLSTS(1) = 1 GOTO *HOME,S	Definiert bei einem Zusammenstoß einen Unterprogramm sprung zur Marke HOME
20	ACT 1 = 1	Interrupt 1 freigeben
30	COLCHK ON,NOERR	Kollisionsüberwachung ohne Fehlerausgabe aktivieren
40	MOV P1	
50	MOV P2	Erfolgt während der Ausführung der Zeilen 40 bis 70 ein Zusammenstoß, wird der Interrupt-Prozess ausgeführt
60	MOV P3	
70	MOV P4	
80	ACT 1 = 0	Interrupt 1 sperren
	:	
1000	*HOME	Interrupt-Prozess bei einem Zusammenstoß
1010	COLCHK OFF	Kollisionsüberwachung deaktivieren
1020	SERVO ON	Schaltet die Servospannung ein
1030	PESC = P_COLDIR(1)*(-2)	Abstand der Ausweichposition festlegen
1040	PDST = P_FBC(1) + PESC	Ausweichposition festlegen
1050	MVS PDST	Ausweichposition mittels Linear-Interpolation anfahren
1060	ERROR 9100	Benutzerdefinierten, leichten Fehler ausgeben

Erläuterung

- Bei einem Zusammenstoß ist der Variablenwert „1“. Nach Beseitigung des Zusammenstoßes ist der Variablenwert „0“.
- Die Variable M_COLSTS kann zur Definition der Interrupt-Bedingung in der Anweisung DEF ACT verwendet werden. Für die Kollisionsüberwachung muss dabei auf der NOERR-Modus festgelegt sein.

7.2.18 M_CSTP

Funktion: Programmstatus lesen

Die Variable zeigt den Zyklusstopp eines Programmes an.

- 1: Zyklusstopp (Der Zyklus wurde über die END-Taste des Steuergerätes oder ein Zyklusstoppsignal unterbrochen.)
- 0: kein Zyklusstopp

Eingabeformat

```
Bsp.: <Numerische Variable> = M_CSTP
```

<Numerische Variable>

Legt eine numerische Variable fest

Programmbeispiel

```
10 M1 = M_CSTP
```

Überträgt bei einem Zyklusstopp den Wert „1“ in die numerische Variable M1

Erläuterung

- Wird im kontinuierlichen Betrieb die END-Taste auf dem Steuergerät betätigt, erfolgt ein Zyklusstopp. Die Variable M_CSTP wird auf „1“ gesetzt.
- Die Variable kann ausschließlich gelesen werden.

7.2.19 M_CYS

Funktion: Zyklusbetrieb lesen

Die Variable zeigt den Zyklusbetrieb eines Programmes an.

- 1: Zyklusbetrieb (wenn das Ausführungsformat im Programmplatzparameter auf „CYC“ gesetzt ist oder der zyklische Betrieb im Befehl XRUN festgelegt und der Befehl ausgeführt wurde)
0: kein Zyklusbetrieb

Eingabeformat

```
Bsp.: <Numerische Variable> = M_CYC
```

<Numerische Variable>

Legt eine numerische Variable fest

Programmbeispiel

```
10 M1 = M_CYC
```

Überträgt im Zyklusbetrieb den Wert „1“ in die numerische Variable M1

Erläuterung

- Das Ausführungsformat (kontinuierlich oder zyklisch) eines Programmes kann über einen Parameter oder den Startbefehl festgelegt werden. Das Ausführungsformat wird in die Variable M_CYC übertragen.
- Auch wenn das Ausführungsformat im Programmplatzparameter auf „CYC“ gesetzt ist, wird bei Festlegung des kontinuierlichen Betriebs im Startbefehl XRUN die Variable auf „0“ gesetzt.
- Die Variable kann ausschließlich gelesen werden.

7.2.20 M_DIN/M_DOUT

Funktion: CC-Link-Registerzugriff

Die Variable ermöglicht den Zugriff auf ein dezentrales Register in einem CC-Link-Netzwerk.

M_DIN: Lesezugriff auf ein Eingangsregister

M_DOUT: Lese- und Schreibzugriff auf ein Ausgangsregister

Eingabeformat

Bsp.: <Numerische Variable> = M_DIN [(<Numerischer Ausdruck 1>)]
 Bsp.: <Numerische Variable> = M_DOUT [(<Numerischer Ausdruck 2>)]

<Numerische Variable>	Legt eine numerische Variable fest, der der Registerinhalt zugewiesen wird.
<Numerischer Ausdruck 1>	Legt die Registernummer fest (≥ 6000)
<Numerischer Ausdruck 2>	Legt die Registernummer fest (≥ 6000)

Programmbeispiel

10 M1 = M_DIN(6000)	Überträgt den Inhalt des CC-Link-Eingangsregisters in die numerische Variable M1 (wenn die CC-Link-Stationennummer gleich 1 ist)
20 M1 = M_DOUT(6000)	Überträgt den Inhalt des CC-Link-Ausgangsregisters in die numerische Variable M1
30 M_DOUT(6000) = 100	Schreibt den Wert „100“ in das CC-Link-Ausgangsregister

Erläuterung

- Detaillierte Hinweise finden Sie im Handbuch des CC-Link-Schnittstellenmoduls.
- Signalnummern ab 6000 werden für den CC-Link-Zugriff verwendet.
- Die Variable M_DIN kann ausschließlich gelesen werden.

7.2.21 M_ERR/M_ERRLVL/M_ERRNO

Funktion: Fehlerdaten lesen

Die Variable ermöglicht den Zugriff auf die Daten eines vom Roboter generierten Fehlers.

M_ERR: zeigt an, ob ein Fehler aufgetreten ist (1: Fehler, 0: kein Fehler)

M_ERRLVL: Fehlerklassen (Warnung/leichter Fehler/schwerer Fehler 1/schwerer Fehler 2)

M_ERRNO: enthält die Nummer des generierten Fehlers

Eingabeformat

Bsp.: <Numerische Variable> = M_ERR
 Bsp.: <Numerische Variable> = M_ERRLVL
 Bsp.: <Numerische Variable> = M_ERRNO

<Numerische Variable> Legt eine numerische Variable fest

Programmbeispiel

```
10 IF M_ERR = 0 THEN 10      Wartestatus, bis ein Fehler auftritt
20 M2 = M_ERRLVL            Überträgt die Fehlerklasse in M2
30 M3 = M_ERRNO            Überträgt die Fehlernummer in M3
```

Erläuterung

- In der Regel werden Programme unterbrochen, wenn ein Fehler (nicht bei Warnmeldungen) auftritt. Mit Hilfe der Variablen kann der Fehlerstatus von Programmen angezeigt werden, deren Startbedingung in den Programmplatzparametern auf „ALWAYS“ gesetzt ist. Ein Programm mit der Startbedingung „ALWAYS“ wird auch dann nicht unterbrochen, wenn in einem anderen Programm ein Fehler auftritt.
- Je nach Fehlerklasse reagiert das System wie folgt:

Fehlerklasse	Beschreibung
1	Warnung
2	Der Betrieb wird unterbrochen.
3	Der Betrieb wird unterbrochen und die Servoversorgung abgeschaltet. Ein Zurücksetzen des Fehlers ist möglich.
4	Der Betrieb wird unterbrochen und die Servoversorgung abgeschaltet. Ein Zurücksetzen des Fehlers ist nicht möglich.

Tab. 7-1: Fehlerklassen

- Die Variablen können ausschließlich gelesen werden.

Steht in Beziehung zu folgenden Befehlen:

RESET ERR

7.2.22 M_EXP

Funktion: Basis des natürlichen Logarithmus lesen

Die Variable enthält den Wert der Basis des natürlichen Logarithmus (2,718281828459045).

Eingabeformat

```
Bsp.: <Numerische Variable> = M_EXP
```

<Numerische Variable>

Legt eine numerische Variable fest

Programmbeispiel

```
10 M1 = M_EXP
```

Überträgt den Wert der Basis des natürlichen Logarithmus (2,718281828459045) in die numerische Variable M1

Erläuterung

- Die Variable dient zur Berechnung von Exponential- und Logarithmusfunktionen.
- Die Variable kann ausschließlich gelesen werden.

7.2.23 M_FBD

Funktion: Differenz zwischen Soll- und Istposition lesen

Die Variable enthält die Differenz zwischen der durch den Befehlswert vorgegebenen Sollposition und der durch die Encoderimpulse gemeldeten Istposition.
Die Variable ist ab Software-Version J1 verfügbar.

Eingabeformat

Bsp.: <Numerische Variable> = M_FBD [(<Mechanismusnummer>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest $1 \leq \text{Mechanismusnummer} \leq 3$ Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

10 DEF ACT 1,M_FBD > 10 GOTO *SUB1,S	Definiert einen Unterprogramm- sprung zu Zeile 100, wenn die Abweichung zwischen der Soll- und der Istposition größer gleich 10 mm ist
20 ACT 1 = 1	Interrupt 1 freigeben
30 TORQ 3, 10	Legt die Drehmomentgrenze für Achse 3 auf 10 % des Maximalwerts fest
40 MVS P1	Position P1 mittels Linear- Interpolation anfahren
50 MOV P2	Position P2 mittels Gelenk- Interpolation anfahren
100 *SUB1	Sprungmarke „SUB1“ festgelegt
110 MOV P_FBC	Soll- und Istposition abgleichen
120 M_OUT(10) = 1	Setzt das allgemeine Ausgangssignal 10 auf „1“
130 HLT	Programm stoppen, wenn die Abweichung zwischen Soll- und Istposition größer 10 mm ist

Erläuterung

- Die Variable enthält die Differenz zwischen der durch den Befehlswert vorgebenen Position und der durch den Motor rückgemeldeten Position. Bei Verwendung des TORQ-Befehls sollte die Variable zur Vermeidung von zu großen Positionsabweichungen (Fehler H960, H970 usw.) zusammen mit dem DEF-ACT-Befehl verwendet werden.
- Die Variable kann ausschließlich gelesen werden.

Steht in Beziehung zu folgenden Befehlen:

TORQ

Steht in Beziehung zu folgenden Variablen:

P_FBC

7.2.24 M_G

Funktion: Erdbeschleunigung lesen

Die Variable enthält den Wert der Erdbeschleunigung (9,80665).

Eingabeformat

```
Bsp.: <Numerische Variable> = M_G
```

<Numerische Variable>

Legt eine numerische Variable fest

Programmbeispiel

```
10 M1 = M_G
```

Überträgt den Wert der Erdbeschleunigung (9,80665) in die numerische Variable M1

Erläuterung

- Die Variable dient zur Berechnung von Funktionen mit Hilfe der Erdbeschleunigung.
- Die Variable kann ausschließlich gelesen werden.

7.2.25 M_HNDCQ

Funktion: Handgreiferzustand lesen

Die Variable enthält Daten über den Handgreiferzustand.

Eingabeformat

```
Bsp.: <Numerische Variable> = M_HNDCQ [( <Numerischer Ausdruck > )]
```

<Numerische Variable>

Legt eine numerische Variable fest

<Numerischer Ausdruck>

Legt die Handeingangsnummer fest
 $1 \leq \text{Handeingangsnummer} \leq 8$
 (entspricht den Eingangssignalen 900 bis 907)

Programmbeispiel

```
10 M1 = M_HNDCQ
```

Überträgt Handgreiferzustand der Hand 1 in die numerische Variable M1

Erläuterung

- Die Variable enthält ein Bit des Handgreiferzustands (z. B. ein Sensorsignal).
- Die Angabe M_HNDCQ(1) bezeichnet im Grundzustand die Eingangssignalnummer 900, d. h. sie entspricht der Angabe M_IN(900), und kann unparametriert werden.
- Die Variable kann ausschließlich gelesen werden.

7.2.26 M_IN/M_INB/M_INW

Funktion: Eingangssignal lesen

Die Variablen enthalten die Werte verschiedener Eingangssignale.

M_IN: Lesezugriff auf ein Eingangssignalbit
 M_INB: Lesezugriff auf ein Eingangssignalbyte (8 Bit)
 M_INW: Lesezugriff auf ein Eingangssignalwort (16 Bit)

Eingabeformat

```
Bsp.: <Numerische Variable> = M_IN [( <Numerischer Ausdruck> ) ]
Bsp.: <Numerische Variable> = M_INB [( <Numerischer Ausdruck> ) ]
Bsp.: <Numerische Variable> = M_INW [( <Numerischer Ausdruck> ) ]
```

<Numerische Variable>	Legt eine numerische Variable fest
<Numerischer Ausdruck>	Legt die Eingangssignalnummer fest $0 \leq \text{Eingangssignalnummer} \leq 32767$ 0 bis 255: Standardeingänge (in der Regel 32 Adressen: 0 bis 31) 900 bis 907: Handeingang 2000 bis 5071: Eingangssignale in einem PROFIBUS-Netzwerk 6000 bis 8047: dezentrales Eingangsregister in einem CC-Link-Netzwerk

Programmbeispiel

10 M1 = M_IN(0)	Überträgt den Wert des Eingangssignals 0 (0 oder 1) in die numerische Variable M1
20 M2 = M_INB(0)	Überträgt den 8-Bit-Wert des Eingangssignals, beginnend mit Bit 0, in die numerische Variable M2
30 M3 = M_INB(3) AND &H7	Überträgt den 3-Bit-Wert des Eingangssignals, beginnend mit Bit 3, in die numerische Variable M3
40 M4 = M_INW(5)	Überträgt den 16-Bit-Wert des Eingangssignals, beginnend mit Bit 5, in die numerische Variable M4

Erläuterung

- Die Variablen enthalten die Zustände verschiedener Eingangssignale.
- M_INB und M_INW enthalten 8-Bit- oder 16-Bit-Informationen, beginnend mit dem festgelegten Bit.
- Auch wenn Signalnummern bis 32767 verwendet werden können, muss über die Nummern ein sinnvoller Zugriff auf die vorhandenen Eingänge erfolgen. Bei Angabe von Signalnummern, denen keine Hardware zugeordnet ist, wird ein undefinierter Zustand gesetzt.
- Die Variablen können ausschließlich gelesen werden.

7.2.27 M_JOVRD/M_NJOVRD/M_OPOVRD/M_OVRD/M_NOVRD

Funktion: Übersteuerungswerte lesen

Die Variablen enthalten Übersteuerungswerte.

M_JOVRD:	Wert der über die JOVRD-Anweisung festgelegte Übersteuerung für die Gelenk-Interpolation
M_NJOVRD:	Standard-Übersteuerungswert für die Gelenk-Interpolation (100 %)
M_OPOVRD:	über das Steuergerät eingestellter Übersteuerungswert
M_OVRD:	Wert der über die OVRD-Anweisung festgelegten aktuellen Übersteuerung
M_NOVRD:	Standard-Übersteuerungswert (100 %)

Eingabeformat

```
Bsp.: <Numerische Variable> = M_JOVRD [( <Numerischer Ausdruck> )]
Bsp.: <Numerische Variable> = M_NJOVRD [( <Numerischer Ausdruck> )]
Bsp.: <Numerische Variable> = M_OPOVRD
Bsp.: <Numerische Variable> = M_OVRD [( <Numerischer Ausdruck> )]
Bsp.: <Numerische Variable> = M_NOVRD [( <Numerischer Ausdruck> )]
```

<Numerische Variable>	Legt eine numerische Variable fest
<Numerischer Ausdruck>	Legt die Programmplatznummer fest $1 \leq \text{Programmplatznummer} \leq 32$ Bei fehlender Angabe wird der aktuelle Programmplatz (Slot) gesetzt.

Programmbeispiel

10	M1 = M_OVRD	Überträgt den über die OVRD-Anweisung festgelegten aktuellen Übersteuerungswert in die numerische Variable M1
20	M2 = M_NOVRD	Überträgt den Standard-Übersteuerungswert von 100 % in die numerische Variable M2
30	M3 = M_JOVRD	Überträgt den Übersteuerungswert für Gelenk-Interpolation in die numerische Variable M3
40	M4 = M_NJOVRD	Überträgt den Standard-Übersteuerungswert für Gelenk-Interpolation in die numerische Variable M4
50	M5 = M_OPOVRD	Überträgt den über das Steuergerät festgelegten Übersteuerungswert in die numerische Variable M5
60	M6 = M_OVRD(2)	Überträgt den aktuellen Übersteuerungswert von Programmplatz 2 in die numerische Variable M6

Erläuterung

- Fehlt die Angabe der Programmplatznummer, wird der Wert des aktuell ausgewählten Programmplatzes übertragen.
- Die Variablen können ausschließlich gelesen werden.

7.2.28 M_LDFACT

Funktion: Lastverhältnis lesen

Die Variable enthält das Lastverhältnis der einzelnen Gelenkachsen.
Die Variable ist ab Software-Version J1 verfügbar.

Eingabeformat

Bsp.: <Numerische Variable> = M_LDFACT [(<Achsennummer>)]

<Numerische Variable> Legt eine numerische Variable im Bereich von 0 bis 100 % fest

<Achsennummer> Legt die Achsennummer fest
1 ≤ Achsennummer ≤ 8

Programmbeispiel

10	ACCEL 100,100	Beschleunigung/Abbremsung für Standardlast einstellen
20	MOV P1	Position P1 mittels Gelenk-Interpolation anfahren
30	MOV P2	Position P2 mittels Gelenk-Interpolation anfahren
40	IF M_LDFACT(2) > 90 THEN	
50	ACCEL 50,50	Beschleunigung/Abbremsung auf 50 % reduzieren
60	M_SETADL(2) = 50	Zusätzlich wird die Beschleunigung/Abbremsung der Achse J2 auf 50 % verkleinert (d. h.: 50 % × 50 % = 25 %)
70	ELSE	
80	ACCEL 100,100	Beschleunigung/Abbremsung wieder auf Standardlast einstellen
90	ENDIF	
100	GOTO 20	Sprung zu Zeile 20

Erläuterung

- Die Variable ermöglicht einen Zugriff auf das Lastverhältnis der einzelnen Achsen.
- Das Lastverhältnis wird aus den einzelnen Servomotorströmen und der Zeit, in der die Ströme fließen, ermittelt.
- Das Lastverhältnis steigt, wenn der Roboter über einen längeren Zeitraum mit einer schweren Last in einer belastungsintensiven Stellung betrieben wird.
- Erreicht das Lastverhältnis 100 % erfolgt die Meldung eines Überlastfehlers. Im Programmbeispiel erfolgt bei einem Lastverhältnis von 90 % eine Verringerung der Beschleunigung/Abbremsung auf 25 %.
- Folgende Maßnahmen dienen der Verkleinerung des Lastverhältnisses:
 - Beschleunigungs-/Bremszeit vergrößern
 - weniger belastungsintensive Stellungen für die Verfahrbewegungen wählen
 - Servoversorgungsspannung abschalten

Steht in Beziehung zu folgenden Befehlen:

ACCEL, OVRD

Steht in Beziehung zu folgenden Variablen:

M_SETADL

7.2.29 M_LINE

Funktion: Zeilennummer lesen

Die Variable enthält die Nummer der Zeile, die gerade ausgeführt wird.

Eingabeformat

Bsp.: <Numerische Variable> = M_LINE [(<Numerischer Ausdruck>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Numerischer Ausdruck>	Legt die Programmplatznummer fest $1 \leq \text{Programmplatznummer} \leq 32$ Bei fehlender Angabe wird der aktuelle Programmplatz gesetzt.

Programmbeispiel

10 M1 = M_LINE(2)	Überträgt die Nummer der Zeile, die gerade im Programmplatz 2 ausgeführt wird, in die numerische Variable M1
-------------------	--------------------------------------------------------------------------------------------------------------

Erläuterung

- Die Variable kann im Multitask-Betrieb zur Überwachung der aktuell ausgeführten Zeile durch andere Programmplätze verwendet werden.
- Die Variable kann ausschließlich gelesen werden.

7.2.30 M_MODE

Funktion: Betriebsart lesen

Die Variable enthält die über den MODE-Umschalter eingestellte Betriebsart.

- 1: TEACH
- 2: AUTO (OP)
- 3: AUTO (Ext.)

Eingabeformat

Bsp.: <Numerische Variable> = M_MODE

<Numerische Variable>

Legt eine numerische Variable fest

Programmbeispiel

10 M1 = M_MODE

Überträgt die über den MODE-Schalter eingestellte Betriebsart in die numerische Variable M1

Erläuterung

- Die Variable kann im Multitask-Betrieb zur Überwachung kontinuierlich ausgeführter Programme (Startbedingung: ALWAYS) verwendet werden.
- Die Variable kann ausschließlich gelesen werden.

7.2.31 M_ON/M_OFF

Funktion: Signalzustand schalten

Die Variable überträgt den Wert „1“ (M_ON) oder „0“ (M_OFF).

Eingabeformat

Bsp.: <Numerische Variable> = M_ON
Bsp.: <Numerische Variable> = M_OFF

<Numerische Variable>

Legt eine numerische Variable fest

Programmbeispiel

10 M1 = M_ON

Setzt M1 auf „1“

20 M2 = M_OFF

Setzt M2 auf „0“

Erläuterung

- Die Variable überträgt den Signalzustand „1“ oder „0“.
- Die Variable kann ausschließlich gelesen werden.

7.2.32 M_OPEN

Funktion: Dateistatus lesen

Die Variable zeigt an, ob eine Datei geöffnet ist oder nicht. Der Status des RS232C-Kanals auf der Rechnerseite wird übertragen.

Eingabeformat

Ab Software-Version H7:

Bsp.: <Numerische Variable> = M_OPEN [<Dateinummer>]

- <Numerische Variable> Legt eine numerische Variable fest
- <Dateinummer> Gibt die Nummer eines mit dem OPEN-Befehl geöffneten Kommunikationskanals als Konstante im Bereich zwischen 1 und 8 an (Standardwert: 1)
Bei Angabe einer Dateinummer größer gleich 9 erfolgt eine Fehlermeldung.

Programmbeispiel

- 100 OPEN "COM2:" AS #1 Öffnet den Kommunikationskanal COM 2 als Datei Nr. 1
- 110 IF M_OPEN(1) <> 1 THEN GOTO 110 Wartestatus, bis die Datei Nummer 1 geöffnet wird

Erläuterung

- Die Variable kann ausschließlich gelesen werden.
- Der Variablenwert ist vom Typ der mit dem OPEN-Befehl geöffneten Datei abhängig:

Dateityp	Beschreibung	Variablenwert
Datei	Die Variable zeigt an, ob eine Datei geöffnet ist oder nicht. Nach Ausführung des Befehls OPEN ist der Variablenwert solange „1“, bis einer der Befehle CLOSE oder END ausgeführt oder das Programmende erreicht wird.	1: bereits geöffnet -1: nicht definierte Dateinummer (nicht geöffnet)
Kommunikationskanal RS232C	Die Variable zeigt den Status der RS232C-Kommunikationsleitung an. ① Es wird der Zustand des CTS-Signals angezeigt. (Die Funktion kann nur nach Freigabe des RTS-Signals am rechnerseitigen Leitungsende und bei Verwendung des Mitsubishi-Kabels RV-CAB□ genutzt werden.)	1: bereits verbunden (CTS-Signal EIN) 0: nicht verbunden (CTS-Signal AUS) -1: nicht definierte Dateinummer (nicht geöffnet)

Tab. 7-2: Variablenwert und Bedeutung

① Informationen zur Verwendung der Funktion beim Anschluss an das ETHERNET finden Sie im Handbuch der ETHERNET-Schnittstellenkarte.

Steht in Beziehung zu folgenden Befehlen:

OPEN

Steht in Beziehung zu folgenden Parametern:

COMDEV

7.2.33 M_OUT/M_OUTB/M_OUTW

Funktion: Ausgangssignal lesen/schreiben

Die Variablen ermöglichen den Schreib- und Lesezugriff auf externe Ausgangssignale.

M_OUT: Schreib-/Lesezugriff auf ein Ausgangssignalbit

M_OUTB: Schreib-/Lesezugriff auf ein Ausgangssignalbyte (8 Bit)

M_OUTW: Schreib-/Lesezugriff auf ein Ausgangssignalwort (16 Bit)

Eingabeformat

```
Bsp.: M_OUT [( <Numerischer Ausdruck > )] = <Numerische Variable >
Bsp.: M_OUTB [( <Numerischer Ausdruck > )] = <Numerische Variable >
Bsp.: M_OUTW [( <Numerischer Ausdruck > )] = <Numerische Variable >
```

<Numerische Variable>

Legt eine numerische Variable fest

<Numerischer Ausdruck>

Legt die Ausgangssignalnummer fest

$0 \leq \text{Ausgangssignalnummer} \leq 32767$

0 bis 255: Standardausgänge

900 bis 907: Handausgang

2000 bis 5071: Ausgangssignale in einem PROFIBUS-Netzwerk

6000 bis 8047: dezentrales Ausgangsregister in einem CC-Link-Netzwerk

Programmbeispiel

10 M_OUT(2) = 1

Einschalten des Ausgangssignals 2 (1 Bit)

20 M_OUTB(2) = &HFF

Einschalten von 8 Bits, beginnend mit dem Bit 2

30 M_OUTW(2) = &HFFFF

Einschalten von 16 Bits, beginnend mit dem Bit 2

40 M4 = M_OUTB(2) AND &H0F

Überträgt den 4-Bit-Wert des Ausgangssignals, beginnend mit dem Bit 2, in die numerische Variable M4

Erläuterung

- Die Variablen ermöglichen den Schreib- und Lesezugriff auf externe Ausgangssignale.
- Signalnummern von 900 bis 907 werden für Handsignale verwendet.
- Signalnummern ab 2000 bis 5071 werden für den PROFIBUS-Zugriff verwendet.
- Signalnummern ab 6000 bis 8047 werden für den CC-Link-Zugriff verwendet.

7.2.34 M_PI

Funktion: Kreiszahl lesen

Die Variable enthält den Wert der Kreiszahl π (3,14159265358979).

Eingabeformat

```
Bsp.: <Numerische Variable> = M_PI
```

<Numerische Variable>

Legt eine numerische Variable fest

Programmbeispiel

```
10 M1 = M_PI
```

Überträgt den Wert der Kreiszahl π
(3,14159265358979) in die numerische Variable M1

Erläuterung

- Die Variable ist vom Typ Real.
- Die Variable kann ausschließlich gelesen werden.

7.2.35 M_PSA

Funktion: Programmwählbarkeit lesen

Die Variable zeigt an, ob ein Programm in einem Programmplatz ausgewählt werden kann.

Eingabeformat

```
Bsp.: <Numerische Variable> = M_PSA [( <Numerischer Ausdruck> )]
```

<Numerische Variable>	Legt eine numerische Variable fest
<Numerischer Ausdruck>	Legt die Programmplatznummer fest $1 \leq \text{Programmplatznummer} \leq 32$ Bei fehlender Angabe wird der aktuelle Programmplatz gesetzt.

Programmbeispiel

10 M1 = M_PSA(2) Überträgt den Status der Programmwählbarkeit von Programmplatz 2 in die numerische Variable M1

Erläuterung

- Die Variable ermöglicht eine Abfrage, ob ein Programm über einen Programmplatz ausgewählt werden kann.
- Die Variable kann ausschließlich gelesen werden.

7.2.36 M_RATIO

Funktion: Annäherung an die Zielposition lesen

Die Variable ermöglicht während der Roboterbewegung eine Überwachung des bereits zur Zielposition zurückgelegten Verfahrenweges im Bereich von 0 bis 100 %.

Eingabeformat

Bsp.: <Numerische Variable> = M_RATIO [(<Numerischer Ausdruck>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Numerischer Ausdruck>	Legt die Programmplatznummer fest $1 \leq \text{Programmplatznummer} \leq 32$ Bei fehlender Angabe wird der aktuelle Programmplatz gesetzt.

Programmbeispiel

10 MOV P1 WTHIF M_RATIO > 80, M_OUT(1) = 1	Position P1 mittels Gelenk-Interpolation anfahren und Ausgangsbit 1 auf „1“ setzen, sobald 80 % des Verfahrenweges zurückgelegt worden sind
--------------------------------------------	---------------------------------------------------------------------------------------------------------------------------------------------

Erläuterung

- Die Variable wird z. B. dazu verwendet, an spezifischen Positionen des Verfahrenweges bestimmte Prozeduren auszuführen.
- Die Variable kann ausschließlich gelesen werden.

7.2.37 M_RDST

Funktion: Abstand zur Zielposition lesen

Die Variable ermöglicht während der Roboterbewegung eine Überwachung des Abstands zur Zielposition in Millimetern.

Eingabeformat

Bsp.: <Numerische Variable> = M_RDST [(<Numerischer Ausdruck>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Numerischer Ausdruck>	Legt die Programmplatznummer fest $1 \leq \text{Programmplatznummer} \leq 32$ Bei fehlender Angabe wird der aktuelle Programmplatz gesetzt.

Programmbeispiel

<pre>10 MOV P1 WTHIF M_RDST < 10, M_OUT(1) = 1</pre>	Position P1 mittels Gelenk-Interpolation anfahren und Ausgangsbit 1 auf „1“ setzen, sobald der Abstand zur Zielposition kleiner als 10 mm ist
----------------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------

Erläuterung

- Die Variable wird z. B. dazu verwendet, an spezifischen Positionen des Verfahrensweges bestimmte Prozeduren auszuführen.
- Die Variable kann ausschließlich gelesen werden.

7.2.38 M_RUN

Funktion: Programmstatus lesen

Die Variable enthält den Status eines Programms.

1: Ausführung

0: keine Ausführung (unterbrochen oder gestoppt)

Eingabeformat

Bsp.: <Numerische Variable> = M_RUN [(<Numerischer Ausdruck>)]

<Numerische Variable>

Legt eine numerische Variable fest

<Numerischer Ausdruck>

Legt die Programmplatznummer fest

$1 \leq \text{Programmplatznummer} \leq 32$

Bei fehlender Angabe wird der aktuelle Programmplatz gesetzt.

Programmbeispiel

10 M1 = M_RUN(2)

Überträgt den Status des Programms in Programmplatz 2 in die numerische Variable M1

Erläuterung

- Der Wert der Variablen ist „1“, wenn das Programm ausgeführt wird und „0“, wenn das Programm unterbrochen oder gestoppt ist.
- Verwenden Sie die Variable M_RUN oder M_WAI, um den Stoppzustand eines Programmes zu überprüfen (falls die aktuell ausgeführte Zeile die oberste Programmzeile ist).
- Die Variable kann ausschließlich gelesen werden.

7.2.39 M_SETADL

Funktion: Beschleunigungs-/Abbremszeit jeder Achse einstellen

Die Variable ermöglicht eine Einstellung der Beschleunigungs-/Abbremszeit jeder einzelnen Gelenkachse. Dadurch kann die Motorbelastung einer hoch belasteten Achse gezielt reduziert werden. Die Variable ist auch beim Betrieb mit optimaler Beschleunigung/Abbremsung (OADL ON) wirksam.

Die Einstellung der Geschwindigkeit des gesamten Roboters über die Befehle OVRD bzw. SPD oder der Beschleunigungs-/Abbremszeit über den Befehl ACCEL kann in vielen Fällen zu einer unnötigen Vergrößerung der Taktzeiten führen. Der Initialisierungswert ist im Parameter JADL festgelegt.

Die Variable M_SETADL ist ab Software-Version J2 verfügbar und kann nur mit den Robotermodellen RV-3SB/3SJB und RV-6S/6SL/12S/12SL verwendet werden.

Eingabeformat

Bsp.: M_SETADL [(<Achsennummer>)] = <Numerische Variable>

<Achsennummer>	Legt die Achsennummer fest $1 \leq \text{Achsennummer} \leq 8$
<Numerische Variable>	Legt eine die Standard-Beschleunigungs-/Abbremszeit im Bereich von 1 bis 100 % fest Der Initialisierungswert ist im Parameter JADL festgelegt.

Programmbeispiel

10 ACCEL 100,50	Abbremsung auf 50 % reduzieren
20 IF M_LDFACT(2) > 90 THEN	Bei Überschreitung des Lastverhältnisses von 90 %, gehe zur Zeile 30
30 M_SETADL(2) = 70	Beschleunigung der Achse J2 auf 70 % (100 % × 70 %) und Abbremsung auf 35 % (50 % × 70 %) verringern
40 ENDIF	
50 MOV P1	Position P1 mittels Gelenk-Interpolation anfahren
60 MOV P2	Position P2 mittels Gelenk-Interpolation anfahren
70 M_SETADL(2) = 100	Beschleunigung der Achse J2 wieder auf 100 % und Abbremsung auf 50 % einstellen
80 MOV P3	Position P3 mittels Gelenk-Interpolation anfahren
90 ACCEL 100,100	Beschleunigung/Abbremsung wieder auf Standardlast einstellen
100 MOV P4	Position P4 mittels Gelenk-Interpolation anfahren

Erläuterung

- Die Variable ermöglicht eine Einstellung der Beschleunigungs-/Abbremszeit für jede Gelenkachse. Dabei entsprechen 100 % der kürzesten Beschleunigungs-/Abbremszeit.
- Mit Hilfe der Variablen kann die Beschleunigungs-/Abbremszeit und somit die Belastung einer Achse, die eine Überlast- oder Überhitzungsfehlermeldung hervorruft, gezielt eingestellt werden.
- Die Einstellung der Variablen wirkt sich gleichzeitig auf die Beschleunigungs- und Bremszeit aus.
- Wird die Variable im Betrieb mit optimaler Beschleunigung/Abbremsung gemeinsam mit dem Befehl ACCEL verwendet, ergeben sich die Werte für die Beschleunigung-/Abbremsung aus der Multiplikation der einzelnen Faktoren (siehe auch Abschn. 6.3.53).
- Die Einstellung der Beschleunigungs-/Bremszeit erfolgt über den Befehl ACCEL. Da über die Variable M_SETADL eine Einstellung der einzelnen Achsen erfolgt und der berechnete Wert der Beschleunigungs-/Bremszeit auch von der Motorbelastung abhängt, kann der Istwert der Beschleunigungs-/Bremszeit etwas vom vorher ermittelten Sollwert abweichen.

Steht in Beziehung zu folgenden Befehlen:

ACCEL, OVRD, SPD

Steht in Beziehung zu folgenden Variablen:

M_LDFACT

7.2.40 M_SKIPCQ

Funktion: SKIP-Befehlsausführung lesen

Die Variable zeigt die Ausführung eines SKIP-Befehls an.

1: SKIP-Befehl wurde ausgeführt.

0: SKIP-Befehl wurde nicht ausgeführt.

Eingabeformat

Bsp.: <Numerische Variable> = M_SKIPCQ [(<Numerischer Ausdruck>)]

<Numerische Variable>

Legt eine numerische Variable fest

<Numerischer Ausdruck>

Legt die Programmplatznummer fest

$1 \leq \text{Programmplatznummer} \leq 32$

Bei fehlender Angabe wird der aktuelle Programmplatz gesetzt.

Programmbeispiel

<pre>10 MOV P1 WTHIF M_IN(10) = 1,SKIP</pre>	<p>Fährt Position 1 mittels Gelenk-Interpolation an und unterbricht die Roboterbewegung, wenn das Eingangsbit Nummer 17 gleich 1 wird Die Programmsteuerung springt in die nächste Zeile.</p>
<pre>20 IF M_SKIPCQ = 1 THEN GOTO 1000</pre>	<p>Springt bei Ausführung der SKIP-Anweisung zur Zeile 1000</p>
<pre> :</pre>	
<pre>1000 END</pre>	<p>Programmende</p>

Erläuterung

- Mit Hilfe der Variablen kann geprüft werden, ob ein SKIP-Befehl ausgeführt worden ist.
- Nach der Prüfung wird der SKIP-Status zurückgesetzt. (Die Variable wird auf „0“ gesetzt.) Soll der Status erhalten bleiben, ist der Wert in eine numerische Variable zu übertragen.
- Die Variable kann ausschließlich gelesen werden.

7.2.41 M_SPD/M_NSPD/M_RSPD

Funktion: Geschwindigkeit lesen

Die Variable ermöglicht eine Überwachung der Geschwindigkeit während der Linear- und Gelenk-Interpolation.

M_SPD: aktuell eingestellte Geschwindigkeit

M_NSPD: Systemstandardwert (optimale Geschwindigkeit)

M_RSPD: aktuelle Geschwindigkeit

Eingabeformat

```
Bsp.: <Numerische Variable> = M_SPD [( <Numerischer Ausdruck> )]
Bsp.: <Numerische Variable> = M_NSPD [( <Numerischer Ausdruck> )]
Bsp.: <Numerische Variable> = M_RSPD [( <Numerischer Ausdruck> )]
```

<Numerische Variable>

Legt eine numerische Variable fest

<Numerischer Ausdruck>

Legt die Programmplatznummer fest

$1 \leq \text{Programmplatznummer} \leq 32$

Bei fehlender Angabe wird der aktuelle Programmplatz gesetzt.

Programmbeispiel

10 M1 = M_SPD

Überträgt den zuletzt eingestellten Geschwindigkeitswert in die numerische Variable M1

20 SPD M_NSPD

Setzt die Geschwindigkeit auf den Systemstandardwert zur Steuerung mit optimaler Geschwindigkeit

Erläuterung

- Die Variable M_RSPD enthält die aktuelle Verfahrgeschwindigkeit des Roboters.
- Die Variablen können zur geschwindigkeitsabhängigen Steuerung von Multitask-Programmen oder in Kombination mit den angehängten Anweisungen WTH und WTHIF verwendet werden.
- Die Variablen können ausschließlich gelesen werden.

7.2.42 M_SVO

Funktion: Status der Servoversorgung lesen

Die Variable zeigt den Status der Servoversorgungsspannung an.

- 1: Servoversorgung eingeschaltet
- 0: Servoversorgung ausgeschaltet

Eingabeformat

```
Bsp.: <Numerische Variable> = M_SVO [(<Mechanismusnummer>)]
```

<Numerische Variable>	Legt eine numerische Variable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest $1 \leq \text{Mechanismusnummer} \leq 3$ Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

```
10 M1 = M_SVO(1)      Überträgt den Status der Servoversorgung von  
                      Mechanismus 1 in die numerische Variable M1
```

Erläuterung

- Mit Hilfe der Variablen kann der Status der Servoversorgungsspannung eines Mechanismus geprüft werden.
- Die Variable kann ausschließlich gelesen werden.

7.2.43 M_TIMER

Funktion: Zeitdauer schreiben/lesen

Die Bezugszeit wird in Millisekunden gemessen. Die Variable dient zur genauen Erfassung von Vorgangsdauern und zur genauen Zeitmessung.

Eingabeformat

Bsp.: <Numerische Variable> = M_TIMER [(<Numerischer Ausdruck>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Numerischer Ausdruck>	Legt die Timer-Nummer zwischen 1 und 8 fest

Programmbeispiel

10	M_TIMER(1) = 0	Setzt Timer 1 auf „0“
20	MOV P1	Position P1 mittels Gelenk-Interpolation anfahren
30	MOV P2	Position P2 mittels Gelenk-Interpolation anfahren
40	M1 = M_TIMER(1)	Schreibt die Dauer für die Verfahrbewegung von P1 nach P2 in ms in die numerische Variable M1 (Bsp.: Eine Zeit von 5,346 s entspricht einem Wert von M1 von 5346)
50	M_TIMER(1) = 1.5	Setzt Timer 1 auf 1,5 s

Erläuterung

- Wird der Variablen ein Wert zugewiesen, erfolgt die Zuweisung in Sekunden.
- Die Auflösung in Millisekunden ermöglicht eine hochgenaue Zeiterfassung.

7.2.44 M_TOOL

Funktion: Werkzeugnummer schreiben/lesen

Ein Zugriff auf die Werkzeugdaten der verschiedenen Werkzeuge kann über die Parameter MEXTL1 bis 4 oder die Variable M_TOOL erfolgen.

Die Variable M_TOOL ist ab Software-Version J1 verfügbar.

Eingabeformat

Werkzeugnummer lesen

```
Bsp.: <Numerische Variable> = M_TOOL [( <Mechanismusnummer> )]
```

Werkzeugnummer schreiben

```
Bsp.: M_TOOL [( <Mechanismusnummer> )] = <Numerischer Ausdruck>
```

<Numerische Variable>	Legt eine numerische Variable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest $1 \leq \text{Mechanismusnummer} \leq 3$ Bei fehlender Angabe wird der Standardwert „1“ gesetzt.
<Numerischer Ausdruck>	Legt die Werkzeugnummer zwischen 1 und 4 fest Der Wert muss in Klammern angegeben werden.

Programmbeispiel

Festlegung der Werkzeugdaten

10	TOOL (0,0,100,0,0,0)	Werkzeugdaten (0,0,100,0,0,0) festlegen und in die Variable MEXTL übertragen
20	MOV P1	Position P1 mittels Gelenk-Interpolation anfahren
30	M_TOOL = 2	Werkzeugdaten von Werkzeug 2 aktivieren (MEXTL2)
40	MOV P2	Position P2 mittels Gelenk-Interpolation anfahren

Umschalten der Werkzeugnummer

10	IF M_IN(900) = 1 THEN	Werkzeugdaten über Handeingangssignal umschalten
20	M_TOOL = 1	Werkzeugdaten von Werkzeug 1 aktivieren
30	ELSE	
40	M_TOOL = 2	Werkzeugdaten von Werkzeug 2 aktivieren
50	ENDIF	
60	MOV P1	Position P1 mittels Gelenk-Interpolation anfahren

Erläuterung

- Die Werkzeugdaten werden in den Parametern MEXTL1, MEXTL2, MEXTL3 und MEXTL4 eingestellt und in den Parameter MEXTL übertragen.
- Die Werkzeuge 1 bis 4 entsprechen den Parametern MEXTL1 bis 4.
- Beim Lesezugriff auf die Variable M_TOOL wird die aktuelle Werkzeugnummer übertragen.
- Ist der eingelesene Wert 0, bedeutet dies, dass die aktuellen Werkzeugdaten nicht über die Parameter MEXTL1 bis 4 aktiviert wurden.
- Die Auswahl der Werkzeugdaten kann auch über die Teaching Box erfolgen (siehe auch Abschn. 3.3).

Steht in Beziehung zu folgenden Befehlen:

TOOL

Steht in Beziehung zu folgenden Parametern:

MEXTL, MEXTL1, MEXTL2, MEXTL3, MEXTL4

7.2.45 M_UAR

Funktion: Aufenthalt im benutzerdefinierten Bereich prüfen

Die Variable zeigt an, ob der Mechanismus sich innerhalb des benutzerdefinierten Bereichs befindet. Dabei entsprechen die Bits 0 bis 7 den benutzerdefinierten Bereichen 1 bis 8.

1: innerhalb des benutzerdefinierten Bereichs

0: außerhalb des benutzerdefinierten Bereichs

Eingabeformat

Bsp.: <Numerische Variable> = M_UAR [(<Mechanismusnummer>)]

<Numerische Variable>

Legt eine numerische Variable fest

<Mechanismusnummer>

Legt die Mechanismusnummer fest

$1 \leq \text{Mechanismusnummer} \leq 3$

Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

10 M1 = MUAR(1)

M1 zeigt an, ob der Mechanismus sich innerhalb oder außerhalb des benutzerdefinierten Bereichs befindet
Der Wert „4“ zeigt z. B. an, dass der Roboter sich im benutzerdefinierten Bereich 3 befindet.

Erläuterung

- Eine detaillierte Beschreibung zur Anwendung benutzerdefinierter Bereiche finden Sie in Abschn. 9.9.
- Die Variable kann ausschließlich gelesen werden.

7.2.46 M_WAI

Funktion: Wartestatus lesen

Die Variable zeigt an, ob sich das Programm im ausgewählten Programmplatz im Wartestatus befindet.

1: Wartestatus (Das Programm ist unterbrochen.)

0: kein Wartestatus (Das Programm wird abgearbeitet oder befindet sich im Stoppzustand.)

Eingabeformat

Bsp.: <Numerische Variable> = M_WAI [(<Numerischer Ausdruck>)]

<Numerische Variable>

Legt eine numerische Variable fest

<Numerischer Ausdruck>

Legt die Programmplatznummer fest

$1 \leq \text{Programmplatznummer} \leq 32$

Bei fehlender Angabe wird der aktuelle Programmplatz gesetzt.

Programmbeispiel

10 M1 = M_WAI(1)

M1 zeigt an, ob sich das Programm im Programmplatz 1 im Wartestatus befindet

Erläuterung

- Mit Hilfe der Variablen M_WAI kann der Wartestatus eines Programms überprüft werden.
- Verwenden Sie die Variable M_RUN oder M_WAI, um den Stoppzustand eines Programmes zu überprüfen (falls die aktuell ausgeführte Zeile die oberste Programmzeile ist).
- Die Variable kann ausschließlich gelesen werden.

Steht in Beziehung zu folgenden Variablen:

M_WUPOV, M_WUPRT, M_WUPST

7.2.47 M_WUPOV

Funktion: Übersteuerung im Warmlaufbetrieb lesen

Die Variable enthält den Wert der Übersteuerung in Prozent, der dem Geschwindigkeitssollwert im Warmlaufbetrieb zur Reduzierung der Betriebsgeschwindigkeit überlagert wird. Die Statusvariable ist ab Software-Version J8 verfügbar.

Eine detaillierte Beschreibung des Warmlaufbetriebs finden Sie in Abschn. 9.20.

Eingabeformat

Bsp.: <Numerische Variable> = M_WUPOV [(<Mechanismusnummer>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest $1 \leq \text{Mechanismusnummer} \leq 3$ Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

```
10 M1 = M_WUPOV(1)      Überträgt den Übersteuerungswert für den
                        Warmlaufbetrieb von Mechanismus 1 in die
                        numerische Variable M1
```

Erläuterung

- Mit Hilfe der Statusvariablen kann der Übersteuerungswert im Warmlaufbetrieb (Betrieb mit verminderter Geschwindigkeit), der dem Geschwindigkeitssollwert im Warmlaufbetrieb zur Reduzierung der Betriebsgeschwindigkeit überlagert wird, geprüft werden.
- Ist der Warmlaufbetrieb deaktiviert, der [MODE]-Umschalter am Steuergerät auf „TEACH“ eingestellt oder der Mechanismus gesperrt, ist der Wert auf 100 gesetzt.
- Bei Umschaltung vom Normal- in den Warmlaufbetrieb oder einer Aktivierung des Warmlaufbetriebs direkt nach dem Einschalten, wird der Wert, der im ersten Element des Parameters WUPOVRD eingestellt ist, als Startwert der Variablen verwendet. Mit fortschreitender Betriebsdauer des Roboters vergrößert sich der Wert der Variablen M_WUPOV. Bei Beendigung des Warmlaufbetriebs wird der Wert der Variablen M_WUPOV auf 100 gesetzt.
- Die aktuelle Übersteuerung im Warmlaufbetrieb ergibt sich wie folgt:

Gelenk- Inter- polation	=	Übersteuerungs- wert der T/B oder des Steuergeräts	×	Einstellwert des OVRD- Befehls	×	Einstellwert des JOVRD- Befehls	×	Übersteuerungs- wert für den Warmlaufbetrieb
-------------------------------	---	----------------------------------------------------------	---	--------------------------------------	---	---------------------------------------	---	----------------------------------------------------

Linear- Inter- polation	=	Übersteuerungs- wert der T/B oder des Steuergeräts	×	Einstellwert des OVRD- Befehls	×	Einstellwert des SPD- Befehls	×	Übersteuerungs- wert für den Warmlaufbetrieb
-------------------------------	---	----------------------------------------------------------	---	--------------------------------------	---	-------------------------------------	---	----------------------------------------------------

- Die Variable kann ausschließlich gelesen werden.

7.2.48 M_WUPRT

Funktion: Restzeit einer Achse im Warmlaufbetrieb

Die Variable enthält die Restzeit in Sekunden, in der eine Achse im Warmlaufbetrieb verfahren wird, bis der Warmlaufbetrieb beendet ist. Die Statusvariable ist ab Software-Version J8 verfügbar.

Eine detaillierte Beschreibung des Warmlaufbetriebs finden Sie in Abschn. 9.20.

Eingabeformat

Bsp.: <Numerische Variable> = M_WUPRT [(<Mechanismusnummer>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest $1 \leq \text{Mechanismusnummer} \leq 3$ Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

```
10 M1 = M_WUPRT(1)
```

Überträgt die Restzeit der für den Warmlaufbetrieb festgelegten Achse von Mechanismus 1, in der die Achse bis zur Beendigung des Warmlaufbetriebs im Warmlaufbetrieb verfahren wird, in die numerische Variable M1

Erläuterung

- Mit Hilfe der Statusvariablen kann die Restzeit, die eine über den Parameter WUPAXIS festgelegte Achse für den Warmlaufbetrieb (Betrieb mit verminderter Geschwindigkeit) im Warmlaufbetrieb verfahren wird, bis der Warmlaufbetrieb beendet ist, überprüft werden.
- Bei deaktiviertem Warmlaufbetrieb ist der Wert der Variablen „0“.
- Bei Umschaltung vom Normal- in den Warmlaufbetrieb oder einer Aktivierung des Warmlaufbetriebs direkt nach dem Einschalten, wird der Wert, der im ersten Element des Parameters WUPTIME eingestellt ist, als Startwert der Variablen verwendet. Mit fortschreitender Betriebsdauer des Roboters verkleinert sich der Wert der Variablen M_WUPRT. Erreicht der Wert „0“, wird der Warmlaufbetrieb beendet.
- Sind mehrere Achsen für einen Warmlaufbetrieb festgelegt, enthält die Variable den Wert der Achse mit der kürzesten Betriebsdauer.

Beispiel ▾

Die Achse A arbeitet bis zur Beendigung des Warmlaufbetriebs noch 20 s (bei M_WUPRT = 20) im Warmlaufbetrieb. Wird in dieser Zeit eine Achse B nach einer Betriebspause vom Normal- in den Warmlaufbetrieb umgeschaltet, so ist dies die Achse mit der kürzesten Betriebsdauer (Betriebsdauer = 0 s). Die Restzeit der Achse B bis zur Beendigung des Warmlaufbetriebs beträgt 60 s (Werkseinstellung = 60 s). Somit wird dieser Wert in die Statusvariable übertragen (M_WUPRT = 60 s).



- Die Variable kann ausschließlich gelesen werden.

7.2.49 M_WUPST

Funktion: Zeit bis zur Wiederholung des Warmlaufbetriebs

Die Variable enthält die Zeit in Sekunden, bis nach Beendigung eines Warmlaufbetriebs ein neuer Warmlaufbetrieb aktiviert wird. Die Statusvariable ist ab Software-Version J8 verfügbar.

Eine detaillierte Beschreibung des Warmlaufbetriebs finden Sie in Abschn. 9.20.

Eingabeformat

Bsp.: <Numerische Variable> = M_WUPST [(<Mechanismusnummer>)]

<Numerische Variable>	Legt eine numerische Variable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest $1 \leq \text{Mechanismusnummer} \leq 3$ Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

```
10 M1 = M_WUPST(1)
```

Überträgt die Zeit, bis für Mechanismus 1 nach Beendigung eines Warmlaufbetriebs ein erneuter Warmlaufbetrieb aktiviert wird, in die numerische Variable M1

Erläuterung

- Mit Hilfe der Statusvariablen kann die Zeit, bis eine über den Parameter WUPAXIS festgelegte Achse nach Beendigung des Warmlaufbetriebs erneut im Warmlaufbetrieb (Betrieb mit verminderter Geschwindigkeit) verfahren wird, überprüft werden.
- Bei deaktiviertem Warmlaufbetrieb entspricht der Wert der Statusvariablen der im zweiten Element des Parameters WUPTIME (Wiederholschwelle) festgelegten Zeit.
- Beim Verfahren einer Achse nach Beendigung des Warmlaufbetriebs wird der im zweiten Element des Parameters WUPTIME (Wiederholschwelle) festgelegte Wert als Anfangszeit in die Statusvariable M_WUPST übertragen. Während der Betriebspause der für den Warmlaufbetrieb festgelegten Achse verringert sich der Wert. Erreicht der Wert „0“, wird ein erneuter Warmlaufbetrieb aktiviert.
- Sind mehrere Achsen für einen Warmlaufbetrieb festgelegt, wird der Wert der Achse mit der längsten Betriebspause in die Statusvariable übertragen.
- Die Variable kann ausschließlich gelesen werden.

7.2.50 P_BASE/P_NBASE

Funktion: Basis-Konvertierungsdaten lesen

Die Variable enthält Werte, die auf die Basis-Konvertierungsdaten bezogen sind.

P_BASE: aktuell eingestellte Basis-Konvertierungsdaten

P_NBASE: Standardwert der Basis-Konvertierungsdaten (0, 0, 0, 0, 0, 0) (0, 0)

Eingabeformat

```
Bsp.: <Positionsvariable> = P_BASE [ (<Mechanismusnummer> ) ]
Bsp.: <Positionsvariable> = P_NBASE
```

<Positionsvariable>

Legt eine Positionsvariable fest

<Mechanismusnummer>

Legt die Mechanismusnummer fest

$1 \leq \text{Mechanismusnummer} \leq 3$

Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

10 P1 = P_BASE

Überträgt die aktuell eingestellten Basis-Konvertierungsdaten in die Positionsvariable P1

20 BASE P_NBASE

Setzt die Basis-Konvertierungsdaten auf den Standardwert zurück

Erläuterung

- Der Standardwert der Basis-Konvertierungsdaten ist auf (0, 0, 0, 0, 0, 0) (0, 0) gesetzt.
- Beachten Sie, dass eine Änderung der Basiskonvertierungsdaten die geteachten Positionen beeinflussen kann.
- Eine Änderung des Basiskoordinatensystems erfolgt über den BASE-Befehl. Fehlerhaft eingestellte Basis-Konvertierungsdaten können zu undefinierten Aktivitäten des Roboters führen. Stellen Sie daher die Werte besonders gewissenhaft ein.
- Die Variable kann ausschließlich gelesen werden.

7.2.51 P_COLDIR

Funktion: Fahrwegrichtung bei einem Zusammenstoß lesen

Die Variable enthält die Richtung des Fahrweges bei Anprechen der Kollisionsüberwachung. Die Variable ist ab Software-Version J2 verfügbar und kann nur mit den Robotermodellen RV-3SB/SJB, RV-6S/6SL/12S/12SL und RH-6SH/12SH verwendet werden.

Eingabeformat

Bsp.: <Positionsvariable> = P_COLDIR [(<Mechanismusnummer>)]

<Positionsvariable>	Legt eine Positionsvariable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest $1 \leq \text{Mechanismusnummer} \leq 3$ Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

Bei einem Zusammenstoß erfolgt der Aufruf eines Interrupt-Prozesses

10	DEF ACT 1,M_COLSTS(1) = 1 GOTO *HOME,S	Definiert bei einem Zusammenstoß einen Unterprogrammssprung zur Marke HOME
20	ACT 1 = 1	Interrupt 1 freigeben
30	COLCHK ON,NOERR	Kollisionüberwachung ohne Fehlerausgabe aktivieren
40	MOV P1	
50	MOV P2	Erfolgt während der Ausführung der Zeilen 40 bis 70 ein Zusammenstoß, wird der Interrupt-Prozess ausgeführt
60	MOV P3	
70	MOV P4	
80	ACT 1 = 0	Interrupt 1 sperren
	:	
1000	*HOME	Interrupt-Prozess bei einem Zusammenstoß
1010	COLCHK OFF	Kollisionsüberwachung deaktivieren
1020	SERVO ON	Schaltet die Servospannung ein
1030	PESC = P_COLDIR(1)*(-2)	Abstand der Ausweichposition festlegen
1040	PDST = P_FBC(1) + PESC	Ausweichposition festlegen
1050	MVS PDST	Ausweichposition mittels Linear-Interpolation anfahren
1060	ERROR 9100	Benutzerdefinierten, leichten Fehler ausgeben

Erläuterung

- Die Variable dient zur Festlegung der Bewegungsrichtung des Roboters für die automatische Anfahrt einer Ausweichposition nach einem Zusammenstoß.
- Die Berechnung der Bewegungsrichtung im Augenblick des Zusammenstoßes erfolgt indem die Komponente der Position der Hauptbewegungsrichtung auf „1“ gesetzt und eine Anpassung der anderen Komponenten im gleichen Verhältnis durchgeführt wird. Ist das Verhältnis z. B. $X/Y = 2/-1$ ergibt sich $P_COLDIR = (1,-0.5,0,0,0,0)(0,0)$.
- Stellungsdaten und Stellungsmerker sind auf „0“ gesetzt: $(*,*,*,0,0,0,0)(0,0)$.
- Der berechnete Wert bei einem Zusammenstoß bleibt bis zum nächsten Zusammenstoß erhalten.
- Spricht die Kollisionsüberwachung durch Einwirkung äußerer Kräfte an, während der Roboter im Stillstand ist, werden die Werte für alle Achsen auf „0“ gesetzt.
- Da die Berechnung der Bewegungsrichtung mit Bezug auf die Zielposition eines Bewegungsbefehls erfolgt, werden alle Komponenten der Variablen bei einem Zusammenstoß nahe der Zielposition auf „0.0“ gesetzt.
- Die Variable kann ausschließlich gelesen werden.
- Bei Robotern, die nicht über die Funktion der Kollisionsüberwachung verfügen, sind alle Komponenten der Variablen auf „0.0“ gesetzt.

Steht in Beziehung zu folgenden Befehlen:

COLCHK, COLLVL

Steht in Beziehung zu folgenden Variablen:

M_COLSTS, J_COLMXL

7.2.52 P_CURR

Funktion: aktuelle Position lesen

Die Variable enthält die aktuelle Position (X, Y, Z, A, B, C, L1, L2) (FL1, FL2).

Eingabeformat

Bsp.: <Positionsvariable> = P_CURR [(<Mechanismusnummer>)]

<Positionsvariable>	Legt eine Positionsvariable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest $1 \leq \text{Mechanismusnummer} \leq 3$ Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

10	DEF ACT 1, M_IN(10) = 1 GOTO 1000	Definiert einen Sprung zu Zeile 1000, wenn der Status des allgemeinen Eingangssignals Nummer 10 = EIN ist
20	ACT 1 = 1	Interrupt 1 freigeben
30	MOV P1	Position P1 mittels Gelenk-Interpolation anfahren
40	MOV P2	Position P2 mittels Gelenk-Interpolation anfahren
50	ACT 1 = 0	Interrupt 1 sperren
	:	
1000	P100 = P_CURR	Lädt die aktuelle Position in die Positionsvariable P100, wenn ein Interrupt anliegt
1010	MOV P100,-100	Position anfahren, die 100 mm in Werkzeuglängsrichtung von der Position P100 entfernt ist
1020	END	Programmende

Erläuterung

- Die Variable überträgt die aktuelle Position.
- Die Variable kann ausschließlich gelesen werden.

Steht in Beziehung zu folgenden Variablen:

J_CURR

7.2.53 P_FBC

Funktion: aus der Servorückmeldung ermittelte Position lesen

Die Variable enthält die aus den Daten der Servorückmeldung ermittelte aktuelle Position (X, Y, Z, A, B, C, L1, L2) (FL1, FL2).

Eingabeformat

```
Bsp.: <Positionsvariable> = P_FBC [( <Mechanismusnummer> )]
```

<Positionsvariable>	Legt eine Positionsvariable fest
<Mechanismusnummer>	Legt die Mechanismusnummer fest $1 \leq \text{Mechanismusnummer} \leq 3$ Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

```
10 P1 = P_FBC
```

Überträgt die aus den Daten der Servorückmeldung ermittelte Position in die Positionsvariable P1

Erläuterung

- Die Variable überträgt die aus der Servorückmeldung ermittelte aktuelle Position.
- Die Variable kann ausschließlich gelesen werden.

Steht in Beziehung zu folgenden Befehlen:

TORQ

Steht in Beziehung zu folgenden Variablen:

J_FBC/J_AMPFBC, M_FBD

7.2.54 P_SAFE

Funktion: Rückzugspunkt lesen

Die Variable enthält die Position des Rückzugspunktes (XYZ-Koordinaten des Parameters JSAFE).

Eingabeformat

```
Bsp.: <Positionsvariable> = P_SAFE [(<Mechanismusnummer>)]
```

<Positionsvariable>

Legt eine Positionsvariable fest

<Mechanismusnummer>

Legt die Roboternummer fest

$1 \leq \text{Mechanismusnummer} \leq 3$

Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

```
10 P1 = P_SAFE
```

Überträgt die Daten des Rückzugspunktes in die Positionsvariable P1

Erläuterung

- Die Variable überträgt die aus den Gelenkdaten des Parameters JSAFE ermittelten XYZ-Koordinaten des Rückzugspunktes.
- Die Variable kann ausschließlich gelesen werden.

7.2.55 P_TOOL/P_NTOOL

Funktion: Werkzeug-Konvertierungsdaten lesen

Die Variable enthält die Werte der Werkzeug-Konvertierungsdaten.

P_TOOL: aktuell eingestellte Werkzeug-Konvertierungsdaten

P_NTOOL: Standardwert der Werkzeug-Konvertierungsdaten (0, 0, 0, 0, 0, 0, 0, 0) (0, 0)

Eingabeformat

```
Bsp.: <Positionsvariable> = P_TOOL [ (<Mechanismusnummer> ) ]
Bsp.: <Positionsvariable> = P_NTOOL
```

<Positionsvariable>

Legt eine Positionsvariable fest

<Mechanismusnummer>

Legt die Mechanismusnummer fest

$1 \leq \text{Mechanismusnummer} \leq 3$

Bei fehlender Angabe wird der Standardwert „1“ gesetzt.

Programmbeispiel

10 P1 = P_TOOL

Überträgt die aktuell eingestellten Werkzeug-Konvertierungsdaten in die Positionsvariable P1

20 BASE P_NTOOL

Setzt die Basis-Konvertierungsdaten auf den Standardwert zurück

Erläuterung

- Die Variable P_TOOL enthält die über den TOOL-Befehl oder den Parameter MEXTL festgelegten Werkzeug-Konvertierungsdaten.
- Eine Änderung der Werkzeugdaten erfolgt über den TOOL-Befehl.
- Die Variablen können ausschließlich gelesen werden.

7.2.56 P_ZERO

Funktion: Initialisierungswerte lesen

Die Variable enthält die Daten (0, 0, 0, 0, 0, 0, 0, 0) (0, 0).

Eingabeformat

```
Bsp.: <Positionsvariable> = P_ZERO
```

<Positionsvariable>

Legt eine Positionsvariable fest

Programmbeispiel

```
10 P1 = P_ZERO
```

Überträgt die Daten (0, 0, 0, 0, 0, 0, 0, 0) (0, 0)
in die Positionsvariable P1

Erläuterung

- Die Variable dient zur Initialisierung von Positionsvariablen.
- Die Variable kann ausschließlich gelesen werden.

8 Funktionen

8.1 Allgemeine Hinweise

In den nachfolgenden Abschnitten finden Sie eine alphabetische Auflistung aller Funktionen und deren Anwendungsmöglichkeiten.

8.1.1 Beschreibung des verwendeten Formats

Funktion

Hier finden Sie eine Beschreibung der Funktion.

Eingabeformat

Hier finden Sie das genaue Format zur Eingabe der Funktion.

Programmbeispiel

Hier finden Sie die Verwendung der Funktion in einem Beispielprogramm.

Erläuterung

Hier finden Sie eine detaillierte Beschreibung, Besonderheiten usw. der Funktion.

8.2 Detaillierte Funktionsbeschreibung

In diesem Abschnitt finden Sie eine detaillierte Beschreibung sowie Programmbeispiele zur Anwendung der Funktionen.

8.2.1 ABS

Funktion: Betrag berechnen

Die Funktion bildet den Betrag des angegebenen Wertes.

Eingabeformat

```
<Numerische Variable> = ABS (<Numerischer Ausdruck>)
```

Programmbeispiel

10	P2.C = ABS(P1.C)	Überträgt die C-Komponente von P1 ohne Vorzeichen in die C-Komponente der Positionsvariablen P2
20	MOV P2	Position P2 mittels Gelenk-Interpolation anfahren
30	M2 = -100	Weist der Variablen M2 den Wert „-100“ zu
40	M1 = ABS(M2)	Weist der Variablen M1 den Wert „100“ zu

Erläuterung

- Die Funktion ABS bildet den Betrag des angegebenen Wertes, so dass das Ergebnis ein positives Vorzeichen hat.

Steht in Beziehung zu folgenden Funktionen:

SGN

8.2.2 ALIGN

Funktion: axiale Ausrichtung

Setzt den Wert der Position mit dem kleinstmöglichen senkrechten oder waagerechten Abstand zur Stellung (A, B, C) der aktuellen Position. Das Ergebnis der ALIGN-Funktion ist ein numerischer Wert. Die Funktion beinhaltet auch Bewegungsbefehle wie z. B. den MOV-Befehl.

Eingabeformat

```
<Positionsvariable> = ALIGN (<Position>)
```

Programmbeispiel

10	P1 = P_CURR	Überträgt die Daten der aktuellen Position in die Positionsvariable P1
20	P2 = ALIGN(P1)	Überträgt die Position mit dem kleinstmöglichen senkrechten oder waagerechten Abstand zur Stellung der aktuellen Position in P2
30	MOV P2	Richtet den Roboterarm axial aus

Erläuterung

- Die Funktion ALIGN setzt den Wert der Position mit dem kleinstmöglichen senkrechten oder waagerechten Abstand ($0^\circ, \pm 90^\circ, \pm 180^\circ$) zur Stellung (A, B, C) der aktuellen Position.
- Da das Operationsergebnis eine Positionsvariable ist, erfolgt eine Fehlermeldung, wenn auf der linken Seite der Gleichung eine Gelenkvariable angegeben wird.
- Die Funktion kann nicht bei 5-achsigen Robotern verwendet werden, da die Orientierungsdaten A, B und C eine andere Bedeutung haben.

Beispiel ▾

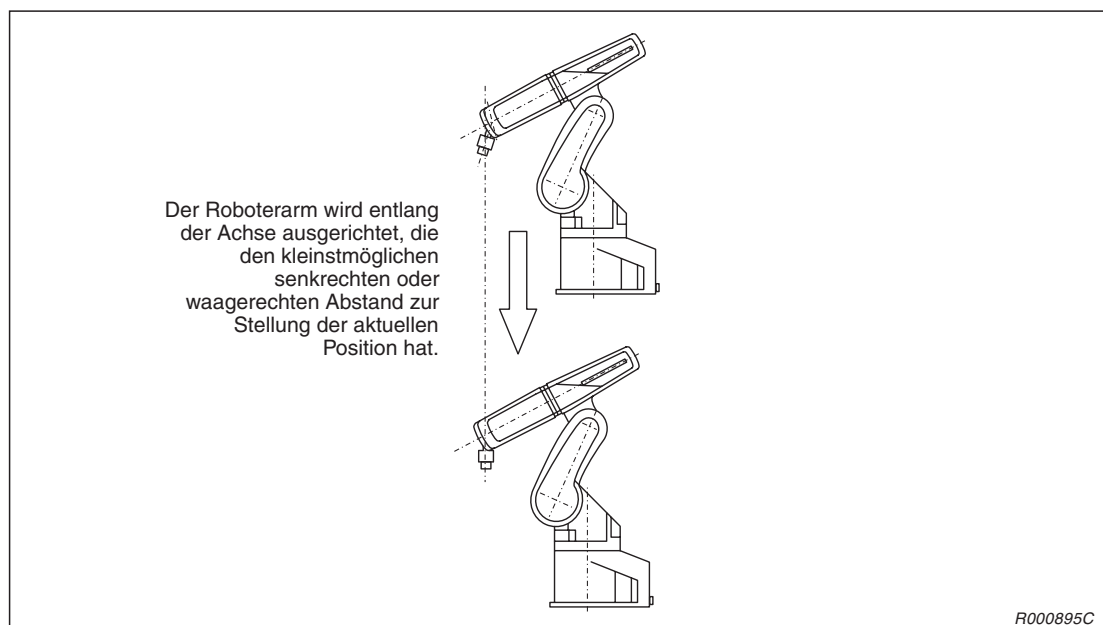


Abb. 8-1: Axiale Ausrichtung des Roboterarms



8.2.3 ASC

Funktion: ASCII-Code erzeugen

Die Funktion erzeugt den ASCII-Code für das erste Zeichen in der Zeichenkette.

Eingabeformat

```
<Numerische Variable> = ASC (<Zeichenkette>)
```

Programmbeispiel

```
10 M1 = ASC("A")
```

Weist der numerischen Variablen M1 den Wert „&H41“ zu

Erläuterung

- Die Funktion ASC bildet den ASCII-Code des ersten Zeichens einer Zeichenkette.
- Bei Anwendung der ASC-Funktion auf eine Leerkette erfolgt eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

CHR\$, VAL, CVI, CVS, CVD

8.2.4 ATN/ATN2

Funktion: Arcus Tangens berechnen

Die Funktion berechnet den Arcus Tangens.

Eingabeformat

```
<Numerische Variable> = ATN (<Numerischer Ausdruck>)
```

```
<Numerische Variable> = ATN2 (<Numerischer Ausdruck 1>,
                               <Numerischer Ausdruck 2>)
```

<Numerische Variable> Legt eine numerische Variable zur Übertragung des berechneten Wertes im Bogenmaß (Radiant) fest

<Numerischer Ausdruck> Quotient $\Delta X/\Delta Y$

<Numerischer Ausdruck 1> ΔX

<Numerischer Ausdruck 2> ΔY

Programmbeispiel

10 M1 = ATN(100/100) Weist der numerischen Variablen M1 den Wert des Arcus Tangens von $\pi/4$ in Radiant zu

20 M2 = ATN2(-100,100) Weist der numerischen Variablen M2 den Wert des Arcus Tangens von $-\pi/4$ in Radiant zu

Erläuterung

- Die Funktionen berechnen den Arcus Tangens. Die Einheit ist Radiant.
- Der Wertebereich der Funktion ATN ist $-\pi/2$ bis $\pi/2$.
- Der Wertebereich der Funktion ATN2 ist $-\pi$ bis π .
- Geht <Numerischer Ausdruck 2> gegen 0, ergibt der ATN2 bei einem positiven <Numerischen Ausdruck 1> $\pi/2$ und bei einem negativen <Numerischen Ausdruck 1> $-\pi/2$.
- Auf die Argumente <Numerischer Ausdruck 1> und <Numerischer Ausdruck 2> der Funktion ATN2 darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Nicht erlaubt sind z. B.: M1 = ATN2(MAX(MA,MB),100)
 M1 = ATN2(CINT(10.2),100)

Steht in Beziehung zu folgenden Funktionen:

SIN, COS, TAN

8.2.5 BIN\$

Funktion: binäre Zeichenkette erzeugen

Die Funktion wandelt einen Wert in eine binäre Zeichenkette um.

Eingabeformat

```
<Zeichenkettenvariable> = BIN$ (<Numerischer Ausdruck>)
```

Programmbeispiel

10	M1 = &B11111111	Weist der numerischen Variablen M1 den Wert der binären Zeichenkette zu
20	C1\$ = BIN\$(M1)	Weist der Zeichenkettenvariablen die Zeichenkette „11111111“ zu

Erläuterung

- Die Funktion BIN\$ konvertiert den angegebenen Wert in eine binäre Zeichenkette.
- Ist der numerische Ausdruck keine Integer-Zahl, wird der Wert gerundet und anschließend in eine binäre Zeichenkette umgewandelt.
- Die Umkehrfunktion von BIN\$ ist VAL.

Steht in Beziehung zu folgenden Funktionen:

HEX\$, STR\$, VAL

8.2.6 CALARC

Funktion: Kreisbogen berechnen

Die Funktion berechnet den über drei Punkte festgelegten Kreisbogen.

Eingabeformat

```
<Numerische Variable 4> = CALARC (<Position 1>, <Position 2>,
                                   <Position 3>,
                                   <Numerische Variable 1>,
                                   <Numerische Variable 2>,
                                   <Numerische Variable 3>,
                                   <Positionsvariable 1>)
```

<Position 1>	Legt den Startpunkt des Kreisbogens fest
<Position 2>	Legt die Zwischenposition auf dem Kreisbogen fest
<Position 3>	Legt den Endpunkt des Kreisbogens fest (Die drei Punkte entsprechen denen im MVR-Befehl.)
<Numerische Variable 1>	Berechnung des Radius des festgelegten Kreisbogens in mm
<Numerische Variable 2>	Berechnung des Zentriwinkels des festgelegten Kreisbogens in Radiant
<Numerische Variable 3>	Berechnung der Kreisbogenlänge des festgelegten Kreisbogens in mm
<Positionsvariable 1>	Berechnung der Mittelpunktkoordinaten des festgelegten Kreisbogens in mm (Die Daten werden als Positionsdaten übertragen. A, B, C sind gleich „0“.)
<Numerische Variable 4>	Übertragener Wert: 1: Berechnung ausgeführt -1: Zwei der Positionen 1 bis 3 sind deckungsgleich oder alle drei Positionen liegen auf einer Geraden. -2: Alle drei Punkte sind annähernd deckungsgleich.

Programmbeispiel

10 M1 = CALARC(P1,P2,P3,M10,M20,M30,P10)	Berechnung des Kreisbogens
20 IF M1 <> 1 THEN END	Programm bei Fehler in der Berechnung des Kreisbogens beenden
30 MR = M10	Überträgt den Radius in die numerische Variable MR
40 MRD = M20	Überträgt den Zentriwinkel in die numerische Variable MRD
50 MARCLEN = M30	Überträgt die Bogenlänge in die numerische Variable MARCLEN
60 PC = P10	Überträgt die Position des Mittelpunktes in die Positionsvariable PC

Erläuterung

- Die Funktion berechnet die Daten des über die Positionen 1, 2 und 3 festgelegten Kreisbogens.
- Bei erfolgreicher Berechnung des Kreisbogens wird die numerische Variable 4 auf „1“ gesetzt.
- Sind 2 der drei Positionen deckungsgleich oder liegen alle drei Positionen auf einer Geraden, wird die numerische Variable 4 auf „-1“ gesetzt. In diesem Fall wird die Entfernung zwischen Start- und Endpunkt als Bogenlänge, „-1“ als Radius, „0“ als Zentriwinkel und (0, 0, 0) als Mittelpunktposition übertragen.
- Tritt bei der Berechnung des Kreisbogens ein Fehler auf, wird die numerische Variable 4 auf „-2“ gesetzt. In diesem Fall wird „0“ als Bogenlänge, „-1“ als Radius, „0“ als Zentriwinkel und (0, 0, 0) als Mittelpunktposition übertragen.
- Auf die Argumente <Position 1>, <Position 2>, <Position 3>, <Numerische Variable 1>, <Numerische Variable 2>, <Numerische Variable 3> und <Positionsvariable 1> der Funktion CALARC darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

8.2.7 CHR\$

Funktion: Zeichen erzeugen

Die Funktion erzeugt ein Zeichen entsprechend dem angegebenen Wert.

Eingabeformat

```
<Zeichenkettenvariable> = CHR$ (<Numerischer Ausdruck>)
```

Programmbeispiel

```
10 M1 = &H40           Weist der numerischen Variablen M1 den Wert  
                        „&H40“ zu  
20 C1$ = CHR$(M1+1)    Weist der Zeichenkettenvariablen C1$ die Zeichenkette  
                        „A“ zu
```

Erläuterung

- Die Funktion CHR\$ erzeugt das Zeichen, das dem angegebenen numerischen Wert entspricht.
- Ist der numerische Ausdruck keine Integer-Zahl, wird der Wert gerundet und anschließend in das entsprechende Zeichen umgewandelt.

Steht in Beziehung zu folgenden Funktionen:

ASC

8.2.8 CINT

Funktion: Integer-Zahl berechnen

Die Funktion berechnet eine Integer-Zahl.

Eingabeformat

```
<Numerische Variable> = CINT (<Numerischer Ausdruck>)
```

Programmbeispiel

```
10 M1 = CINT(1.5)      Weist der numerischen Variablen M1 den Wert „2“ zu  
20 M2 = CINT(1.4)      Weist der numerischen Variablen M2 den Wert „1“ zu  
30 M3 = CINT(-1.4)     Weist der numerischen Variablen M3 den Wert „-1“ zu  
40 M4 = CINT(-1.5)     Weist der numerischen Variablen M4 den Wert „-2“ zu
```

Erläuterung

- Die Funktion CINT rundet den Wert eines numerischen Ausdrucks zu einer Integer-Zahl.

Steht in Beziehung zu folgenden Funktionen:

INT, FIX

8.2.9 CKSUM

Funktion: Prüfsumme erzeugen

Die Funktion erzeugt die Prüfsumme einer Zeichenkette.

Eingabeformat

```
<Numerische Variable> = CKSUM (<Zeichenkette>,
                                <Numerischer Ausdruck 1>,
                                <Numerischer Ausdruck 2>)
```

- | | |
|--------------------------|----------------------------------------------------------------------------------|
| <Zeichenkette> | Legt die Zeichenkette fest, aus der die Prüfsumme gebildet werden soll |
| <Numerischer Ausdruck 1> | Legt die Position des ersten Zeichens fest, ab dem die Prüfsummenbildung startet |
| <Numerischer Ausdruck 1> | Legt die Position des ersten Zeichens fest, bei dem die Prüfsummenbildung endet |

Programmbeispiel

```
10 M1 = CKSUM("ABCDEFGH",1,3)    Weist der numerischen Variablen M1 den Wert
                                &H41 ("A") + &H42 ("B") + &H43 ("C") = &HC6 zu
```

Erläuterung

- Die Funktion CKSUM addiert die Werte einer Zeichenkette vom angegebenen Startpunkt bis zum angegebenen Endpunkt und erzeugt so die Prüfsumme in einem Wertebereich zwischen 0 bis 255.
- Liegt der angegebene Startpunkt außerhalb der Zeichenkette, erfolgt eine Fehlermeldung.
- Liegt der angegebene Endpunkt außerhalb der Zeichenkette, erfolgt die Prüfsummenbildung vom angegebenen Startzeichen bis zum letzten Zeichen der Zeichenkette.
- Ist das Ergebnis größer als 255, wird der Wert modifiziert.
- Auf die Argumente <Zeichenkette>, <Numerischer Ausdruck 1> und <Numerischer Ausdruck 2> der Funktion CKSUM darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

8.2.10 COS

Funktion: Cosinus berechnen

Die Funktion berechnet den Cosinus.

Eingabeformat

```
<Numerische Variable> = COS (<Numerischer Ausdruck>)
```

Programmbeispiel

```
10 M1 = COS(RAD(60))
```

Weist der numerischen Variablen M1 den Wert „0.5“ zu

Erläuterung

- Die Funktion berechnet den Cosinus des numerischen Ausdrucks. Die Einheit des Arguments ist Radiant.
- Als Definitionsbereich ist der gesamte gültige Zahlenbereich zugelassen.
- Der Wertebereich der Funktion COS ist –1 bis 1.

Steht in Beziehung zu folgenden Funktionen:

SIN, TAN, ATN/ATN2

8.2.11 CVI

Funktion: 2 Zeichen einer Zeichenkette umwandeln

Die Funktion berechnet die Integer-Werte der ersten beiden Zeichen einer Zeichenkette.

Eingabeformat

```
<Numerische Variable> = CVI (<Zeichenkette>)
```

Programmbeispiel

```
10 M1 = CVI("10ABC")
```

Weist der numerischen Variablen M1 den Wert „&H3130“ zu

Erläuterung

- Die Funktion CVI wandelt die ersten beiden Zeichen einer Zeichenkette in Integer-Werte um.
- Besteht die Zeichenkette aus einem oder weniger Zeichen, erfolgt eine Fehlermeldung.
- Die Umwandlung eines numerischen Ausdrucks in eine Zeichenkette mit zwei Zeichen erfolgt mit der Funktion MKI.
- Die Funktion dient bei der Übertragung numerischer Daten an externe Geräte zur Datenreduzierung.

Steht in Beziehung zu folgenden Funktionen:

ASC, CVS, CVD, MKI\$, MKS\$, MKD\$

8.2.12 CVS

Funktion: 4 Zeichen einer Zeichenkette umwandeln

Die Funktion berechnet den Real-Wert mit einfacher Genauigkeit der ersten 4 Zeichen einer Zeichenkette.

Eingabeformat

```
<Numerische Variable> = CVS (<Zeichenkette>)
```

Programmbeispiel

```
10 M1 = CVS("FFFF")           Weist der numerischen Variablen M1 den Wert  
                               „12689.6“ zu
```

Erläuterung

- Die Funktion CVS wandelt die ersten 4 Zeichen einer Zeichenkette in einen Real-Wert mit einfacher Genauigkeit um.
- Besteht die Zeichenkette aus drei oder weniger Zeichen, erfolgt eine Fehlermeldung.
- Die Umwandlung eines numerischen Ausdrucks in eine Zeichenkette mit 4 Zeichen erfolgt mit der Funktion MKS.

Steht in Beziehung zu folgenden Funktionen:

ASC, CVI, CVD, MKI\$, MKS\$, MKD\$

8.2.13 CVD

Funktion: 8 Zeichen einer Zeichenkette umwandeln

Die Funktion berechnet den Real-Wert mit doppelter Genauigkeit der ersten 8 Zeichen einer Zeichenkette.

Eingabeformat

```
<Numerische Variable> = CVD (<Zeichenkette>)
```

Programmbeispiel

```
10 M1 = CVD("FFFFFFFF")    Weist der numerischen Variablen M1 den Wert  
                          „+3.52954E+30“ zu
```

Erläuterung

- Die Funktion CVD wandelt die ersten 8 Zeichen einer Zeichenkette in einen Real-Wert mit doppelter Genauigkeit um.
- Besteht die Zeichenkette aus sieben oder weniger Zeichen, erfolgt eine Fehlermeldung.
- Die Umwandlung eines numerischen Ausdrucks in eine Zeichenkette mit 8 Zeichen erfolgt mit der Funktion MKD.

Steht in Beziehung zu folgenden Funktionen:

ASC, CVI, CVS, MKI\$, MKS\$, MKD\$

8.2.14 DEG

Funktion: Radiant in Grad umwandeln

Die Funktion wandelt einen in Radiant angegebenen Winkel in Grad um.

Eingabeformat

```
<Numerische Variable> = DEG (<Numerischer Ausdruck>)
```

Programmbeispiel

10 P1 = P_CURR	Weist der Positionsvariablen P1 den Wert der aktuellen Position zu
20 IF DEG(P1.C) < 170 OR DEG(P1.C) > -150 THEN *NOERR	Sprung zur Marke „NOERR“, falls die C-Komponente der Position P1 innerhalb des Bereichs von -150° bis 170° liegt
30 ERROR(9100)	Generiert den Fehler mit der Fehlernummer 9100
40 *NOERR	Definiert die Marke „NOERR“

Erläuterung

- Die Funktion DEG wandelt einen in Radiant (rad) angegebenen Winkel in einen Winkel mit der Einheit Grad (deg) um.
- Bei einem Zugriff auf die Winkel von Positionsdaten über Positionskonstanten, erfolgt die Anzeige der Winkel in Grad (500, 0, 600, 180, 0, 180)(7, 0). Bei einem direkten Zugriff auf die Winkel über die Angabe der Komponente (z. B. über P1.C) ist die Einheit Radiant. Eine Umstellung der Einheit auf Grad kann durch Einstellung des Parameters PRGMDEG auf „1“ erfolgen.

Steht in Beziehung zu folgenden Funktionen:

RAD

8.2.15 DIST

Funktion: Abstand berechnen

Die Funktion berechnet den Abstand zwischen zwei Punkten (Positionsvariablen).

Eingabeformat

```
<Numerische Variable> = DIST (<Position 1>,<Position 2>)
```

Programmbeispiel

```
10 M1 = DIST(P1,P2)
```

Weist der numerischen Variablen M1 den Wert des Abstands zwischen den Positionen P1 und P2 zu

Erläuterung

- Die Funktion DIST berechnet den Abstand zwischen zwei Positionen in mm.
- Für die Berechnung werden nur die XYZ-Koordinaten verwendet. Die Stellungsdaten sind ohne Bedeutung.
- Gelenk-Variablen können zur Berechnung nicht verwendet werden. Wird die Funktion mit Gelenk-Variablen verwendet, erfolgt bei der Ausführung eine Fehlermeldung.
- Auf die Argumente <Position 1> und <Position 2> der Funktion DIST darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

8.2.16 EXP

Funktion: Exponentialfunktion berechnen

Die Funktion berechnet die Exponentialfunktion mit der Basis „e“.

Eingabeformat

```
<Numerische Variable> = EXP (<Numerischer Ausdruck>)
```

Programmbeispiel

```
10 M1 = EXP(2)
```

Weist der numerischen Variablen M1 den Wert „e²“ zu

Erläuterung

- Die Funktion EXP berechnet den Wert der Exponentialfunktion bei einem numerischen Ausdruck.

Steht in Beziehung zu folgenden Funktionen:

LN

8.2.17 FIX

Funktion: Integer-Anteil bilden

Die Funktion berechnet den ganzzahligen Anteil eines numerischen Ausdrucks.

Eingabeformat

```
<Numerische Variable> = FIX (<Numerischer Ausdruck>)
```

Programmbeispiel

```
10 M1 = FIX(5.5)
```

Weist der numerischen Variablen M1 den Wert „5“ zu

Erläuterung

- Die Funktion FIX berechnet den Integer-Anteil eines numerischen Ausdrucks.
- Bei einem positiven numerischen Ausdruck liefert die Funktion dasselbe Ergebnis wie die Funktion INT.
- Bei einem negativen numerischen Ausdruck werden die Stellen hinter dem Komma folgendermaßen gerundet: $\text{FIX}(-2.3) = -2.0$.

Steht in Beziehung zu folgenden Funktionen:

CINT, INT

8.2.18 FRAM

Funktion: Koordinatensystem berechnen

Die Funktion berechnet ein Koordinatensystem (Fläche) über drei Punkte. Verwenden Sie zur Berechnung einer Palette die Befehle DEF PLT und PLT.

Eingabeformat

```
<Position 4> = FRAM (<Position 1>,<Position 2>,<Position 3>)
```

<Position 1>	Legt die XYZ-Koordinaten des Flächenursprungs der über drei Positionen definierten Fläche als Variable oder Konstante fest
<Position 2>	Legt einen Punkt auf der X-Achse in der über drei Positionen definierten Fläche als Variable oder Konstante fest
<Position 3>	Legt einen Punkt in positiver Richtung auf der Y-Achse in der über drei Positionen definierten Fläche als Variable oder Konstante fest
<Position 4>	Positionsvariable zur Ablage des Ergebnisses Der Stellungsmerker entspricht ab Software-Version J1 dem der <Position 1>

Programmbeispiel

10 BASE P_NBASE	Setzt die Basis-Konvertierungsdaten auf den Standardwert zurück
20 P10 = FRAM(P1,P2,P3)	Erzeugt ein über die Positionen P1, P2 und P3 definiertes Koordinatensystem
30 P10 = INV(P10)	Weist der Positionsvariablen P10 die inverse Matrix von P10 zu
40 P10.X = P1.X	Überträgt die Daten der X-Komponente aus P1 in die X-Komponente von P10
50 P10.Y = P1.Y	Überträgt die Daten der Y-Komponente aus P1 in die Y-Komponente von P10
60 P10.Z = P1.Z	Überträgt die Daten der Z-Komponente aus P1 in die Z-Komponente von P10
70 BASE P10	Definiert die Position P10 als Ursprung des Roboter-Koordinatensystem

Erläuterung

- Die Funktion FRAM kann zur Definition des Basis-Koordinatensystems verwendet werden.
- Die Funktion definiert den Ursprung und die Neigung einer Fläche über die XYZ-Koordinaten der drei angegebenen Positionen. Das Ergebnis wird in eine Positionsvariable übertragen. Die XYZ-Koordinaten der Position entsprechen denen der Position 1. Die Stellungsdaten A, B, und C geben die über die drei Positionen festgelegte Neigung der Fläche wieder.
- Da bei Ausführung der Funktion Positionsdaten übertragen werden, darf als Variable 4 keine Gelenkvariable verwendet werden. Bei Verwendung einer Gelenkvariablen erfolgt eine Fehlermeldung.
- Auf die Argumente <Position 1>, <Position 2> und <Position 3> der Funktion FRAM darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Nicht erlaubt ist z. B.: $P10 = \text{FRAM}(\text{FPRM}(P01,P02,P03),P04,P05)$

- Weitere Hinweise zu dieser Funktion finden Sie im Abschn. 9.9.

8.2.19 HEX\$

Funktion: numerischen Ausdruck in hexadezimale Zeichenkette umwandeln

Die Funktion wandelt den Wert eines numerischen Ausdrucks (zwischen -32768 und 32767) in eine hexadezimale Zeichenkette um.

Eingabeformat

```
<Zeichenkettvariable> = HEX$ (<Numerischer Ausdruck>
                               [, <Anzahl der Zeichen>])
```

<Zeichenkettvariable>	Legt eine Zeichenkettvariable fest
<Numerischer Ausdruck>	Legt den umzuwandelnden numerischen Ausdruck fest
<Anzahl der Zeichen>	Legt die Anzahl der auszugebenden Zeichen fest

Programmbeispiel

10	C1\$ = HEX\$(&H41FF)	Weist der Zeichenkettvariablen C1\$ die Zeichenkette „41FF“ zu
20	C2\$ = HEX\$(&H41FF,2)	Weist der Zeichenkettvariablen C2\$ die Zeichenkette „FF“ zu

Erläuterung

- Bei Angabe der <Anzahl der Zeichen> wird die festgelegte Zeichenkettenlänge, beginnend mit dem äußersten rechten Zeichen, ausgegeben.
- Ist der numerische Ausdruck keine Integer-Zahl, wird der Wert gerundet und anschließend in eine hexadezimale Zeichenkette umgewandelt.
- Die Umkehrung der Funktion HEX\$ erfolgt über die Funktion VAL.
- Auf das Argument <Anzahl der Zeichen> der Funktion HEX\$ darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Nicht erlaubt ist z. B.: C1\$ = HEX\$(ASC("a"),1)

Steht in Beziehung zu folgenden Funktionen:

BIN\$, STR\$, VAL

8.2.20 INT

Funktion: Integer-Zahl erzeugen

Die Funktion erzeugt die größtmögliche Integer-Zahl, die kleiner als der Wert des numerischen Ausdrucks ist.

Eingabeformat

```
<Numerische Variable> = INT (<Numerischer Ausdruck>)
```

Programmbeispiel

10 M1 = INT(3.3) Weist der numerischen Variablen M1 den Wert „3“ zu

Erläuterung

- Bei einem positiven numerischen Ausdruck liefert die Funktion dasselbe Ergebnis wie die Funktion FIX.
- Bei einem negativen numerischen Ausdruck werden die Stellen hinter dem Komma folgendermaßen gerundet: $\text{INT}(-2.3) = -3.0$.

Steht in Beziehung zu folgenden Funktionen:

CINT, FIX

8.2.21 INV

Funktion: Position invertieren

Die Funktion erzeugt die inverse Matrix der angegebenen Position.

Eingabeformat

```
<Positionsvariable> = INV (<Positionsvariable>)
```

Programmbeispiel

```
10 P1 = INV(P2)           Weist der Positionsvariablen P1 die inverse
                          Matrix von P2 zu
```

Erläuterung

- Die Funktion INV wird bei relativen Rechenoperationen mit Positionen verwendet.
- Als Argument dürfen keine Gelenkvariablen verwendet werden. Bei Verwendung von Gelenkvariablen erfolgt eine Fehlermeldung.
- Da bei Ausführung der Funktion Positionsdaten übertragen werden, darf auf der linken Seite der Gleichung keine Gelenkvariable verwendet werden. Bei Verwendung einer Gelenkvariablen erfolgt eine Fehlermeldung.

8.2.22 JTOP

Funktion: Gelenk- in Positionsdaten umwandeln

Die Funktion wandelt die angegebenen Gelenkdaten in Positionsdaten um.

Eingabeformat

```
<Positionsvariable> = JTOP (<Gelenkvariable>)
```

Programmbeispiel

```
10 P1 = JTOP(J1)         Weist die über die Gelenkvariable J1 festgelegte
                          Position der Positionvariablen P1 im XYZ-Format zu
```

Erläuterung

- Als Argument dürfen keine Positionsvariablen verwendet werden. Bei Verwendung von Positionsvariablen erfolgt eine Fehlermeldung.
- Da bei Ausführung der Funktion Positionsdaten übertragen werden, darf auf der linken Seite der Gleichung keine Gelenkvariable verwendet werden. Bei Verwendung einer Gelenkvariablen erfolgt eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

PTOJ

8.2.23 LEFT\$

Funktion: Teil einer Zeichenkette erzeugen

Die Funktion erzeugt einen Teil der angegebenen Zeichenkette, beginnend mit dem linken Zeichen.

Eingabeformat

```
<Zeichenkettenvariable> = LEFT$(<Zeichenkette>,  
                                <Numerischer Ausdruck>)
```

<Zeichenkettenvariable> Legt eine Zeichenkettenvariable fest

<Numerischer Ausdruck> Legt die Anzahl der auszugebenden Zeichen fest

Programmbeispiel

10 C1\$ = LEFT\$("ABC",2) Weist der Zeichenkettenvariablen C1\$ die
Zeichenkette „AB“ zu

Erläuterung

- Die Funktion erzeugt einen Teil der Zeichenkette. Die Länge der erzeugten Zeichenkette, beginnend mit dem linken Zeichen, ist im zweiten Argument festgelegt.
- Ist die Anzahl der auszugebenden Zeichen negativ oder größer als die Länge der Zeichenkette, erfolgt eine Fehlermeldung.
- Auf die Argumente <Zeichenkette> und <Numerischer Ausdruck> der Funktion LEFT\$ darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

MID\$, RIGHT\$

8.2.24 LEN

Funktion: Länge einer Zeichenkette berechnen

Die Funktion berechnet die Länge einer Zeichenkette.

Eingabeformat

```
<Numerische Variable> = LEN (<Zeichenkette>)
```

Programmbeispiel

```
10 M1 = LEN("ABCDEFGG")      Weist der numerischen Variablen M1 den Wert „7“ zu
```

Erläuterung

- Die Funktion LEN berechnet die Länge der angegebene Zeichenkette.

Steht in Beziehung zu folgenden Funktionen:

LEFT\$, MID\$, RIGHT\$

8.2.25 LN

Funktion: natürlichen Logarithmus berechnen

Die Funktion berechnet den natürlichen Logarithmus (Basis: e).

Eingabeformat

```
<Numerische Variable> = LN (<Numerischer Ausdruck>)
```

Programmbeispiel

```
10 M1 = LN(2)                Weist der numerischen Variablen M1 den Wert  
                             „0.693147“ zu
```

Erläuterung

- Die Funktion LN berechnet den natürlichen Logarithmus des angegebenen numerischen Ausdrucks.
- Ist der numerische Ausdruck Null oder negativ, erfolgt eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

EXP, LOG

8.2.26 LOG

Funktion: dekadischen Logarithmus berechnen

Die Funktion berechnet den dekadischen Logarithmus (Basis: 10).

Eingabeformat

```
<Numerische Variable> = LOG (<Numerischer Ausdruck>)
```

Programmbeispiel

```
10 M1 = LOG(2)           Weist der numerischen Variablen M1 den Wert  
                        „0.301030“ zu
```

Erläuterung

- Die Funktion LOG berechnet den dekadischen Logarithmus des angegebenen numerischen Ausdrucks.
- Ist der numerische Ausdruck Null oder negativ, erfolgt eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

EXP, LN

8.2.27 MAX

Funktion: Maximalwert berechnen

Die Funktion berechnet den Maximalwert.

Eingabeformat

```
<Numerische Variable> = MAX (<Numerischer Ausdruck 1>,  
                             <Numerischer Ausdruck 2>, ...)
```

Programmbeispiel

```
10 M1 = MAX(2,1,3,4,10,100)   Weist der numerischen Variablen M1 den Wert  
                              „100“ zu
```

Erläuterung

- Die Funktion MAX berechnet den maximalen Wert der angegebenen numerischen Ausdrücke.
- Die maximale Zeilenlänge wird durch die in einer Zeile zulässige Anzahl von Zeichen begrenzt (123 Zeichen).
- Auf die Argumente <Numerischer Ausdruck 1>, <Numerischer Ausdruck 2> und <...> der Funktion MAX darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

MIN

8.2.28 MID\$

Funktion: Teil einer Zeichenkette erzeugen

Die Funktion erzeugt, beginnend mit der festgelegten Position, einen Teil einer Zeichenkette.

Eingabeformat

```
<Zeichenkettenvariable> = MID$ (<Zeichenkette>,  
                                <Numerischer Ausdruck 1>  
                                <Numerischer Ausdruck 2>])
```

<Zeichenkettenvariable> Legt eine Zeichenkettenvariable fest

<Numerischer Ausdruck 1> Legt die Position des ersten Zeichens fest

<Numerischer Ausdruck 2> Legt die Länge der Zeichenkette fest

Programmbeispiel

```
10 C1$ = MID$("ABCDEFGH",3,2)    Weist der Zeichenkettenvariablen C1$ die  
                                Zeichenkette „CD“ zu
```

Erläuterung

- Die Funktion erzeugt einen Teil der Zeichenkette. Die Länge der erzeugten Zeichenkette, ist im numerischen Ausdruck 2, die Position des ersten Zeichens im numerischen Ausdruck 1 festgelegt.
- Ist die Angabe für die Anzahl der auszugebenden Zeichen oder die Position des ersten Zeichens Null oder negativ, erfolgt eine Fehlermeldung.
- Liegt die Position eines der zu erfassenden Zeichen außerhalb der Zeichenkette, erfolgt eine Fehlermeldung.
- Auf die Argumente <Zeichenkette>, <Numerischer Ausdruck 1> und <Numerischer Ausdruck 2> der Funktion MID\$ darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

LEFT\$, RIGHT\$, LEN

8.2.29 MIN

Funktion: Minimalwert berechnen

Die Funktion berechnet den Minimalwert.

Eingabeformat

```
<Numerische Variable> = MIN (<Numerischer Ausdruck 1>,
                               <Numerischer Ausdruck 2>, ...)
```

Programmbeispiel

```
10 M1 = MIN(2,1,3,4,10,100)           Weist der numerischen Variablen M1 den Wert
                                     „1“ zu
```

Erläuterung

- Die Funktion MIN berechnet den minimalen Wert der angegebenen numerischen Ausdrücke.
- Die maximale Zeilenlänge wird durch die in einer Zeile zulässige Anzahl von Zeichen begrenzt (123 Zeichen).
- Auf die Argumente <Numerischer Ausdruck 1>, <Numerischer Ausdruck 2> und <...> der Funktion MIN darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

MAX

8.2.30 MIRROR\$

Funktion: Bits einer Zeichenkette spiegeln

Die Funktion spiegelt die Bits einer Zeichenkette.

Eingabeformat

```
<Numerische Variable> = MIN (<Numerischer Ausdruck 1>,
                               <Numerischer Ausdruck 2>, ...)
```

Programmbeispiel

```
10 C1$ = MIRROR$("BJ")                Weist der Zeichenkettenvariablen C1$ den Wert
                                     „RB“ zu
                                     "BJ" = &H42,&H4A = &B0100 0010,&B0100 1010
                                     gespiegelt: &B0101 0010,&B0100 0010 =
                                     &H52,&H42 => Ausgabe = "RB"
```

Erläuterung

- Die Funktion spiegelt die binären Bits der Zeichen einer Zeichenkette. Das Ergebnis ist eine Zeichenkette, die dem gespiegelten Wert entspricht.

8.2.31 MKI\$

Funktion: 2-Byte-Zeichenkette erzeugen

Die Funktion wandelt einen numerischen Ausdruck vom Typ Integer in eine 2-Byte-Zeichenkette um.

Eingabeformat

```
<Zeichenkettenvariable> = MKI$ (<Numerischer Ausdruck>)
```

Programmbeispiel

10 C1\$ = MKI\$(20299)	Weist der Zeichenkettenvariablen C1\$ die Zeichenkette „OK“ zu
20 M1 = CVI(C1\$)	Weist der numerischen Variablen M1 den Wert „20299“ zu

Erläuterung

- Die Funktion wandelt die beiden niederwertigsten Bytes eines numerischen Ausdrucks vom Typ Integer in eine 2-Byte-Zeichenkette um.
- Die Umwandlung einer 2-Byte-Zeichenkette in einen numerischen Ausdruck erfolgt mit der Funktion CVI.
- Die Funktion dient bei der Übertragung numerischer Daten an externe Geräte zur Datenreduzierung.

Steht in Beziehung zu folgenden Funktionen:

ASC, CVI, CVS, CVD, MKS\$, MKD\$

8.2.32 MKS\$

Funktion: 4-Byte-Zeichenkette erzeugen

Die Funktion wandelt einen numerischen Ausdruck vom Typ Real mit einfacher Genauigkeit in eine 4-Byte-Zeichenkette um.

Eingabeformat

```
<Zeichenkettenvariable> = MKS$ (<Numerischer Ausdruck>)
```

Programmbeispiel

```
10 C1$ = MKS$(100.1)
```

```
20 M1 = CVS(C1$)
```

Weist der numerischen Variablen M1 den Wert „100.1“ zu

Erläuterung

- Die Funktion wandelt die vier niederwertigsten Bytes eines numerischen Ausdrucks vom Typ Real mit einfacher Genauigkeit in eine 4-Byte-Zeichenkette um.
- Die Umwandlung einer 4-Byte-Zeichenkette in einen numerischen Ausdruck erfolgt mit der Funktion CVS.
- Die Funktion dient bei der Übertragung numerischer Daten an externe Geräte zur Datenreduzierung.

Steht in Beziehung zu folgenden Funktionen:

ASC, CVI, CVS, CVD, MKI\$, MKD\$

8.2.33 MKD\$

Funktion: 8-Byte-Zeichenkette erzeugen

Die Funktion wandelt einen numerischen Ausdruck vom Typ Real mit doppelter Genauigkeit in eine 8-Byte-Zeichenkette um.

Eingabeformat

```
<Zeichenkettenvariable> = MKD$ (<Numerischer Ausdruck>)
```

Programmbeispiel

10 C1\$ = MKD\$(10000.1)

20 M1 = CVD(C1\$)

Weist der numerischen Variablen M1 den Wert „10000.1“ zu

Erläuterung

- Die Funktion wandelt die acht niederwertigsten Bytes eines numerischen Ausdrucks vom Typ Real mit doppelter Genauigkeit in eine 8-Byte-Zeichenkette um.
- Die Umwandlung einer 8-Byte-Zeichenkette in einen numerischen Ausdruck erfolgt mit der Funktion CVD.
- Die Funktion dient bei der Übertragung numerischer Daten an externe Geräte zur Datenreduzierung.

Steht in Beziehung zu folgenden Funktionen:

ASC, CVI, CVS, CVD, MKI\$, MKS\$

8.2.34 POSCQ

Funktion: Position prüfen

Die Funktion prüft, ob die angegebene Position innerhalb der zulässigen Verfahrweggrenze liegt.

Eingabeformat

```
<Numerische Variable> = POSCQ (<Positionsvariable>)
```

Programmbeispiel

10 M1 = POSCQ(P1)

Weist der numerischen Variablen eine „1“ zu, falls die Position P1 innerhalb der zulässigen Verfahrweggrenze liegt

Erläuterung

- Liegt die Position innerhalb der zulässigen Verfahrweggrenze, wird die numerische Variable auf „1“ gesetzt, liegt sie außerhalb auf „0“.
- Das Argument kann Positions- oder Gelenkdaten enthalten.

8.2.35 POSMID

Funktion: Mittelposition berechnen

Die Funktion berechnet bei Linear-Interpolation die Mittelposition zwischen zwei Punkten.

Eingabeformat

```
<Positionsvariable> = POSMID (<Positionsvariable 1>,  
                               <Positionsvariable 2>,  
                               <Numerischer Ausdruck 1>,  
                               <Numerischer Ausdruck 2>)
```

Programmbeispiel

```
10 P1 = POSMID(P2,P3,0,0)  Weist der Positionsvariablen P1 die Daten der Position  
                           (inklusive der Stellungsdaten) in der Mitte zwischen  
                           P2 und P3 zu
```

Erläuterung

- Die Positionsvariable 1 gibt den Startpunkt, die Positionsvariable 2 den Endpunkt der Linear-Interpolation an.
- Die numerischen Ausdrücke 1 und 2 entsprechen den TYPE-Angaben des MVS-Befehls.
- Zwischen dem Start- und Endpunkt muss mit dem angegebenen Interpolationstyp eine Linear-Interpolation ausführbar sein. Weichen z. B. die Stellungsmerker der Start- und der Endposition voneinander ab, erfolgt eine Fehlermeldung.
- Auf die Argumente <Positionsvariable 1>, <Positionsvariable 2>, <Numerischer Ausdruck 1> und <Numerischer Ausdruck 2> der Funktion POSMID darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

8.2.36 PTOJ

Funktion: Positions- in Gelenkdaten umwandeln

Die Funktion wandelt die angegebenen Positionsdaten in Gelenkdaten um.

Eingabeformat

```
<Gelenkvariable> = PTOJ (<Positionsvariable>)
```

Programmbeispiel

```
10 J1 = PTOJ(P1)
```

Weist die über die Positionsvariable P1 im XYZ-Format festgelegte Position der Gelenkvariablen J1 zu

Erläuterung

- Als Argument dürfen keine Gelenkvariablen verwendet werden. Bei Verwendung von Gelenkvariablen erfolgt eine Fehlermeldung.
- Da bei Ausführung der Funktion Gelenkdaten übertragen werden, darf auf der linken Seite der Gleichung keine Positionsvariable verwendet werden. Bei Verwendung einer Positionsvariablen erfolgt eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

JTOP

8.2.37 RAD

Funktion: Grad in Radiant umwandeln

Die Funktion wandelt einen in Grad angegebenen Winkel in Radiant um.

Eingabeformat

```
<Numerische Variable> = RAD (<Numerischer Ausdruck>)
```

Programmbeispiel

10	P1 = P_CURR	Weist der Positionsvariablen P1 den Wert der aktuellen Position zu
20	P1.C = RAD(90)	Weist der C-Komponente der Position P1 90 Grad zu
30	MOV P1	Position P1 mittels Gelenk-Interpolation anfahren und den Winkel der C-Achse von der aktuellen Position aus um 90 Grad drehen

Erläuterung

- Die Funktion RAD wandelt einen in Grad (deg) angegebenen Winkel in einen Winkel mit der Einheit Radiant (rad) um.
- Die Funktion kann dazu verwendet werden, die Stellungsdaten A, B und C einer Position auf definierte Werte zu setzen. Weiterhin findet sie Anwendung in der Berechnung trigonometrischer Funktionen.

Steht in Beziehung zu folgenden Funktionen:

DEG

8.2.38 RDFL1

Funktion: Stellungsmerker in Zeichenkette umwandeln

Die Funktion wandelt den Stellungsmerker der festgelegten Position in eine Zeichenkette („R“/„L“, „A“/„B“ und „N“/„F“) um.

Eingabeformat

```
<Zeichenkettenvariable> = RDFL1 (<Positionsvariable>,
                                   <Numerischer Ausdruck>)
```

<Zeichenkettenvariable>	Legt eine Zeichenkettenvariable fest
<Positionsvariable>	Legt die Position fest, deren Stellungsmerker in eine Zeichenkette umgewandelt wird
<Numerischer Ausdruck>	Legt fest, welcher Stellungsmerker in eine Zeichenkette umgewandelt wird 0 = „R“/„L“ 1 = „A“/„B“ 2 = „N“/„F“

Programmbeispiel

10 P1 = (100,0,100,180,0,180)(7,0)	Der Stellungsmerker ist auf „7“ (&B111), also RAN gesetzt
20 C1\$ = RDFL1(P1,1)	Weist der Zeichenkettenvariablen C1\$ das Zeichen „A“ zu

Erläuterung

- Argument 1 gibt die Positionsvariable, Argument 2 den Stellungsmerker an, der in eine Zeichenkette umgewandelt wird.
- Die Funktion ist auf den Stellungsmerker FL1 von Positionsdaten anwendbar.
- Auf die Argumente <Positionsvariable> und <Numerischer Ausdruck> der Funktion RDFL1 darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

RDFL2, SETFL1, SETFL2

8.2.39 RDFL2

Funktion: Multirotationsdaten erzeugen

Die Funktion erzeugt die Multirotationsdaten der festgelegten Gelenkachse.

Eingabeformat

```
<Numerische Variable> = RDFL2 (<Positionsvariable>,
                                <Numerischer Ausdruck>)
```

<Numerische Variable>	Legt eine numerische Variable fest
<Positionsvariable>	Legt die Position fest, deren Multirotationsdaten erzeugt werden
<Numerischer Ausdruck>	Legt das Gelenk fest, dessen Multirotationsdaten erzeugt werden $1 \leq \text{numerischer Ausdruck} \leq 8$

Programmbeispiel

```
10 P1 = (100,0,100,180,0,180)(7,&H00100000)
```

```
20 M1 = RDFL2(P1,6)
```

Weist der numerischen Variablen M1 den Wert „1“ zu

Erläuterung

- Die Funktion RDFL2 erzeugt die Multirotationsdaten des über Argument 2 festgelegten Gelenks. Die Festlegung der Position erfolgt über Argument 1.
- Der Wertebereich der Funktion liegt zwischen -8 und 7 .
- Die Funktion ist auf den Stellungsmerker FL2 von Positionsdaten anwendbar.
- Stellungsmerker FL2 ist als 32-Bit-Information aufgebaut. Die Multirotationsdaten jeder der 8 Achsen sind über 4 Bit festgelegt.
- Negative Multirotationswerte zwischen -1 und -8 werden zur Anzeige auf der Teaching Box in den Bereich F bis 8 (4-Bit-hexadezimal mit Vorzeichen) umgewandelt und dargestellt.

		Achse								Angezeigter Wert und Anzahl der Rotationen						
		8	7	6	5	4	3	2	1							
Multirotation von J6 ist +1	FL2	0	0	1	0	0	0	0	0	...	-2	-1	0	+1	+2	...
Multirotation von J6 ist -1	FL2	0	0	F	0	0	0	0	0	...	E	F	0	1	2	...

Tab. 8-1: Anzeige von Multirotationsdaten auf der Teaching Box

- Auf die Argumente <Positionsvariable> und <Numerischer Ausdruck> der Funktion RDFL1 darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Steht in Beziehung zu folgenden Befehlen:

JRC

Steht in Beziehung zu folgenden Funktionen:

RDFL1, SETFL1, SETFL2

8.2.40 RND

Funktion: Zufallszahl erzeugen

Die Funktion erzeugt eine Zufallszahl.

Eingabeformat

<Numerische Variable> = RND (<Numerischer Ausdruck>)

<Numerische Variable>	Eine numerische Variable im Bereich zwischen 0.0 und 1.0 wird übertragen
<Numerischer Ausdruck>	Legt den Startwert der Zufallszahl fest Bei einer Einstellung auf „0“ werden nachfolgende Zahlen ohne Festlegung des Startwertes generiert.

Programmbeispiel

10 DIM MRND(10)	Deklariert die Feldvariable MRND als eine Variable mit 10 Elementen
20 C1\$ = RIGHT\$(C_TIME,2)	Setzt den Startwert über den TIMER, um verschiedene Zufallszahlen zu erhalten
30 MRNDBS = CVI(C1)	Weist der Variablen MRNDBS den Integer-Wert der Zeichenkettenvariablen C1 zu
40 MRND(1) = RND(MRNDBS)	Setzt den Startwert und weist dem ersten Element der Feldvariablen die erste Zufallszahl zu
50 FOR M1 = 2 TO 10	Erzeugt weitere 9 Zufallszahlen
60 MRND(M1) = RND(0)	
70 NEXT M1	

Erläuterung

- Die Funktion RND erzeugt eine Zufallszahl. Der Startwert wird über das Argument gesetzt.
- Ist das Argument „0“, wird kein Startwert gesetzt und die nächste Zufallszahl erzeugt.
- Bei gleichen Startwerten ergibt sich die gleiche Reihe an Zufallszahlen.

8.2.41 RIGHT\$

Funktion: Teil einer Zeichenkette erzeugen

Die Funktion erzeugt einen Teil der angegebenen Zeichenkette, beginnend mit dem rechten Zeichen.

Eingabeformat

```
<Zeichenkettensvariable> = RIGHT$ (<Zeichenkette>,  
                                     <Numerischer Ausdruck>)
```

<Zeichenkettensvariable> Legt eine Zeichenkettensvariable fest

<Numerischer Ausdruck> Legt die Anzahl der auszugebenden Zeichen fest

Programmbeispiel

```
10 C1$ = RIGHT$("ABCDEFG",3)    Weist der Zeichenkettensvariablen C1$ die  
                               Zeichenkette „EFG“ zu
```

Erläuterung

- Die Funktion erzeugt einen Teil der Zeichenkette. Die Länge der erzeugten Zeichenkette, beginnend mit dem rechten Zeichen, ist im zweiten Argument festgelegt.
- Ist die Anzahl der auszugebenden Zeichen negativ oder größer als die Länge der Zeichenkette, erfolgt eine Fehlermeldung.
- Auf die Argumente <Zeichenkette> und <Numerischer Ausdruck> der Funktion RIGHT\$ darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

LEFT\$, MID\$, LEN

8.2.42 SETFL1

Funktion: Stellungsmerker ändern

Die Funktion ändert den Stellungsmerker der festgelegten Position über eine Zeichenkette (z. B. „RAN“).

Eingabeformat

```
<Positionsvariable> = SETFL1 (<Positionsvariable>,<Zeichenkette>)
```

<Positionsvariable>	Legt die Positionsvariable fest, deren Stellungsmerker geändert werden soll
<Zeichenkette>	Legt fest, welcher Stellungsmerker geändert werden soll Es können mehrere Stellungsmerkerkennungen geändert werden. „R“ oder „L“: rechts/links „A“ oder „B“: oben/unten „N“ oder „F“: nicht kippen/kippen

Programmbeispiel

10 MOV P1	Positions P1 mittels Gelenk-Interpolation anfahren
20 P2 = SETFL1(P1,"LBF")	Stellungsmerker der Position P1 auf „links/unten/kippen“ setzen und der Position P2 zuweisen
30 MOV P2	Positions P2 mittels Gelenk-Interpolation anfahren

Erläuterung

- Der Stellungsmerker der im Argument 1 angegebenen Position wird auf die im Argument 2 angegebenen Werte der Zeichenkette gesetzt und in die Positionsvariable links vom Gleichheitszeichen übertragen.
- Die Funktion ändert nur die Daten des Stellungsmerkers FL1. Die durch das Argument 1 festgelegten Positionsdaten (X, Y, Z, A, B, C und FL2) bleiben unverändert.
- Der Stellungsmerker wird über die Zeichenkette, beginnend mit dem letzten Zeichen, festgelegt. Ist z. B. die Zeichenkette „LR“ vorgegeben, ergibt sich für den resultierenden Stellungsmerker „L“.
- Die Änderung eines Stellungsmerkers über einen numerischen Wert kann nach folgendem Schema erfolgen: P1.FL1 = 7.
- Die Stellungsmerker haben in Abhängigkeit vom Robotermodell unterschiedliche Bedeutungen. Eine detaillierte Beschreibung finden Sie im Technischen Handbuch des Roboters.

In der Positionskonstante (100, 0, 300, 180, 0, 180)(7, 0) ist der Stellungsmerker auf „7“ gesetzt. Die aktuelle Position wird über ein Bitmuster dargestellt.

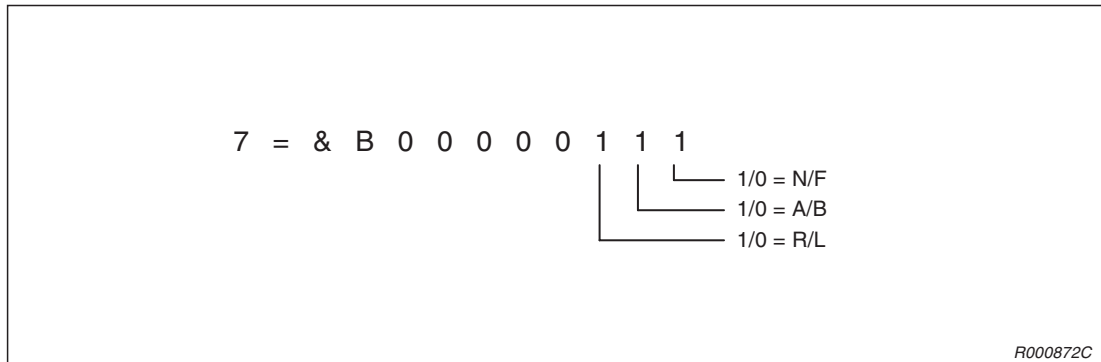


Abb. 8-2: Bedeutung der Stellungsmerker

- Auf die Argumente <Positionsvariable> und <Zeichenkette> der Funktion SETFL1 darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

RDFL1, RDFL2, SETFL2

8.2.43 SETFL2

Funktion: Multirotationsdaten ändern

Die Funktion ändert die Multirotationsdaten der festgelegten Position.

Eingabeformat

```
<Positionsvariable> = SETFL2 (<Positionsvariable>,
                               <Numerischer Ausdruck 1>,
                               <Numerischer Ausdruck 2>)
```

- <Positionsvariable> Legt die Positionsvariable fest, deren Multirotationsdaten geändert werden sollen
- <Numerischer Ausdruck 1> Legt das Gelenk fest, dessen Multirotationsdaten geändert werden sollen
1 ≤ numerischer Ausdruck 1 ≤ 8
- <Numerischer Ausdruck 2> Legt die Multirotationsdaten fest
-8 ≤ numerischer Ausdruck 2 ≤ 7

Programmbeispiel

- 10 MOV P1 Position P1 mittels Gelenk-Interpolation anfahren
- 20 P2 = SETFL2(P1,6,1) Multirotationsdaten des Gelenks 6 der Position P1 auf „1“ setzen
- 30 MOV P2 Position P2 mittels Gelenk-Interpolation anfahren

Erläuterung

- Die Multirotationsdaten des im Argument 2 angegebenen Gelenks der Positionsvariablen werden auf die im Argument 3 angegebenen Werte gesetzt und in die Positionsvariable links vom Gleichheitszeichen übertragen.
- Die Funktion ändert nur die Daten des Stellungsmerkers FL2. Die durch das Argument 1 festgelegten Positionsdaten (X, Y, Z, A, B, C und FL1) bleiben unverändert.

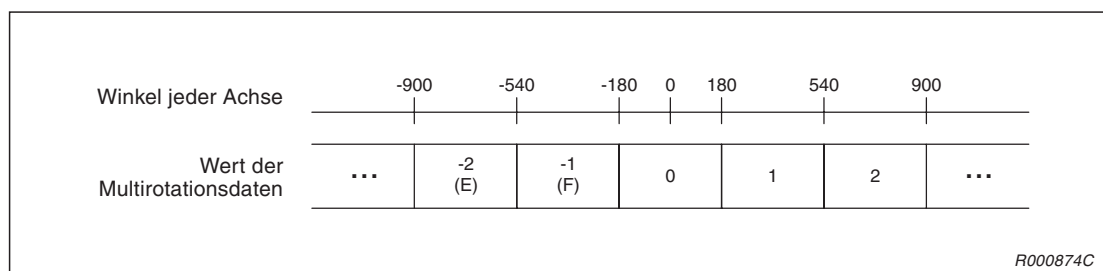


Abb. 8-3: Wert der Multirotationsdaten

- Auf die Argumente <Positionsvariable>, <Numerischer Ausdruck 1> und <Numerischer Ausdruck 2> der Funktion SETFL2 darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

RDFL1, RDFL2, SETFL1

8.2.44 SETJNT

Funktion: Gelenkvariable ändern

Die Funktion ändert die Werte der festgelegten Gelenkvariablen. Die Funktion ist ab Software-Version J2 verfügbar.

Eingabeformat

```
<Gelenkvariable> = SETJNT (<J1-Achse> [, <J2-Achse>
                             [, <J3-Achse> [, <J4-Achse>
                             [, <J5-Achse> [, <J6-Achse>
                             [, <J7-Achse> [, <J8-Achse>]]]]]]])
```

<Gelenkvariable> Legt eine Gelenkvariable fest

<J1-Achse> bis <J8-Achse> Die Einheit der Achsendaten ist RAD
(Für direkt angetriebene Achsen ist die Einheit mm.)

Programmbeispiel

<pre>10 J1 = J_CURR 20 FOR M1 = 0 TO 60 STEP 10 30 M2 = J1.J3 + RAD(M1) 40 J2 = SETJNT(J1.J1,J1.J2,M2) 50 MOV J2 60 NEXT M1 70 M0 = RAD(0) 80 M90 = RAD(90) 90 J3 = SETJNT(M0,M0,M90,M0,M90,M0) 100 MOV J3</pre>	<p>Überträgt die Gelenkdaten der aktuellen Position in die Gelenkvariable J1</p> <p>Erhöht den Wert der J3-Achse bei jedem Schleifendurchlauf um 10 Grad. Die Werte der J4-Achse und der nachfolgenden Achsen bleiben unverändert.</p> <p>Position J2 mittels Gelenk-Interpolation anfahren und den Winkel der J3-Achse von der aktuellen Position aus um 10 Grad drehen</p> <p>Sprung zu Zeile 20</p> <p>Weist der numerischen Variablen M0 den Wert 0 Grad zu</p> <p>Weist der numerischen Variablen M90 den Wert 90 Grad zu</p> <p>Festlegung der Gelenkvariablen J3</p> <p>Position J3 mittels Gelenk-Interpolation anfahren</p>
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

Erläuterung

- Die Funktion ermöglicht die Änderung der Winkel jeder einzelnen Achse einer Gelenkvariablen.
- Die Gelenkvariable kann durch Argumente beschrieben werden.
- Außer dem Argument für die J1-Achse können alle Argumente weggelassen werden. Dabei müssen auch alle nachfolgenden Argumente weggelassen werden. Beschreibungen wie z. B. SETJNT(10,10,,,,,10) sind nicht erlaubt.
- Auf die Argumente der Funktion SETJNT darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

SETPOS

Steht in Beziehung zu folgenden Parametern:

AXUNT, PRGMDEG

8.2.45 SETPOS

Funktion: Positionsvariable ändern

Die Funktion ändert die Werte der festgelegten Positionsvariablen. Die Funktion ist ab Software-Version J2 verfügbar.

Eingabeformat

```
<Positionsvariable> = SETPOS  (<X-Achse>[, <Y-Achse>
                               [, <Z-Achse>[, <A-Achse>
                               [, <B-Achse>[, <C-Achse>
                               [, <L1-Achse>[, <L2-Achse>]]]]]])
```

<Positionsvariable>	Legt eine Positionsvariable fest
<X-Achse> bis <Z-Achse>	Die Einheit der Achsendaten ist mm
<A-Achse> bis <C-Achse>	Die Einheit der Achsendaten ist RAD (Die Einheit kann über Parameter PRGMDEG auf „DEG“ geändert werden.)
<L1-Achse> und <L2-Achse>	Die Einheit hängt von der Einstellung des Parameters AXUNT ab.

Programmbeispiel

10 P1 = P_CURR	Überträgt die aktuelle Position in die Positionsvariable P1
20 FOR M1 = 0 TO 100 STEP 10	
30 M2 = P1.Z + M1	
40 P2 = SETPOS(P1.X,P1.Y,M2)	Erhöht den Wert der Z-Achse bei jedem Schleifendurchlauf um 10 mm. Die Werte der A-Stellungsdaten und der nachfolgenden Daten bleiben unverändert.
50 MOV J2	Position J2 mittels Gelenk-Interpolation anfahren und den Wert der Z-Achse von der aktuellen Position aus um 10 mm verschieben
60 NEXT M1	Sprung zu Zeile 20

Erläuterung

- Die Funktion ermöglicht die Änderung der Werte jeder einzelnen Komponente einer Positionsvariablen.
- Die Positionsvariable kann durch Argumente beschrieben werden.
- Außer dem Argument für die X-Achse können alle Argumente weggelassen werden. Dabei müssen auch alle nachfolgenden Argumente weggelassen werden. Beschreibungen wie z. B. SETPOS(10,10,,,,,10) sind nicht erlaubt.
- Auf die Argumente der Funktion SETPOS darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

Steht in Beziehung zu folgenden Funktionen:

SETJNT

Steht in Beziehung zu folgenden Parametern:

AXUNT, PRGMDEG

8.2.46 SGN

Funktion: Vorzeichen prüfen

Die Funktion prüft das Vorzeichen des angegebenen numerischen Ausdrucks.

Eingabeformat

```
<Numerische Variable> = SGN (<Numerischer Ausdruck>)
```

Programmbeispiel

```
10 M1 = -1           Weist der numerischen Variablen M1 den Wert „-1“ zu
20 M2 = SGN(M1)     Weist der numerischen Variablen M2 den Wert „-1“ zu
```

Erläuterung

- Die Funktion SGN prüft das Vorzeichen eines numerischen Ausdrucks nach folgendem Schema:
Positiver Wert ⇒ 1
Null ⇒ 0
Negativer Wert ⇒ -1

8.2.47 SIN

Funktion: Sinus berechnen

Die Funktion berechnet den Sinus.

Eingabeformat

```
<Numerische Variable> = SIN (<Numerischer Ausdruck>)
```

Programmbeispiel

```
10 M1 = SIN(RAD(60))   Weist der numerischen Variablen M1 den Wert  
                          „0.866025“ zu
```

Erläuterung

- Die Funktionen berechnet den Sinus des numerischen Ausdrucks. Die Einheit des Arguments ist Radiant.
- Als Definitionsbereich ist der gesamte gültige Zahlenbereich zugelassen.
- Der Wertebereich der Funktion SIN ist -1 bis 1.

Steht in Beziehung zu folgenden Funktionen:

COS, TAN, ATN/ATN2

8.2.48 SQR

Funktion: Quadratwurzel berechnen

Die Funktion berechnet die Quadratwurzel.

Eingabeformat

```
<Numerische Variable> = SQR (<Numerischer Ausdruck>)
```

Programmbeispiel

```
10 M1 = SQR(2)           Weist der numerischen Variablen M1 den Wert  
                        „1.414214“ zu
```

Erläuterung

- Die Funktion berechnet die Quadratwurzel des angegebenen numerischen Ausdrucks.
- Bei negativem Argument erfolgt eine Fehlermeldung.

8.2.49 STRPOS

Funktion: Zeichenkette suchen

Die Funktion sucht die angegebene Zeichenkette innerhalb einer anderen Zeichenkette.

Eingabeformat

```
<Numerische Variable> = STRPOS (<Zeichenkette 1>,<Zeichenkette 2>)
```

Programmbeispiel

```
10 M1 = STRPOS("ABCDEFG", "DEF")   Weist der numerischen Variablen M1 den  
                                   Wert „4“ zu
```

Erläuterung

- Die Funktion STRPOS ermittelt den Wert des ersten Auftretens der Zeichenkette 2 innerhalb der Zeichenkette 1.
- Ist die Länge der Zeichenkette 2 gleich 0, erfolgt eine Fehlermeldung.
- Kann die gesuchte Zeichenkette nicht gefunden werden, ist das Suchergebnis „0“.
- Auf die Argumente <Zeichenkette 1> und <Zeichenkette 2> der Funktion STRPOS darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

8.2.50 STR\$

Funktion: Zahl in Zeichenkette umwandeln

Die Funktion wandelt eine Zahl in eine Zeichenkette um.

Eingabeformat

```
<Zeichenkettvariable> = STR$ (<Numerischer Ausdruck>)
```

Programmbeispiel

10 C1\$ = STR\$(123) Weist der Zeichenkettvariablen C1\$ die Zeichenkette „123“ zu

Erläuterung

- Die Funktion wandelt den angegebenen numerischen Ausdruck in eine Zeichenkette um.
- Die Umkehrung der Funktion STR\$ erfolgt über die Funktion VAL.

Steht in Beziehung zu folgenden Funktionen:

BIN\$, HEX\$, VAL

8.2.51 TAN

Funktion: Tangens berechnen

Die Funktion berechnet den Tangens.

Eingabeformat

```
<Numerische Variable> = TAN (<Numerischer Ausdruck>)
```

Programmbeispiel

10 M1 = TAN(RAD(60)) Weist der numerischen Variablen M1 den Wert „1.732051“ zu

Erläuterung

- Die Funktionen berechnet den Tangens des numerischen Ausdrucks. Die Einheit des Arguments ist Radiant.
- Als Definitionsbereich ist der gesamte gültige Zahlenbereich zugelassen.
- Der Wertebereich der Funktion ist der gesamte gültige Zahlenbereich.

Steht in Beziehung zu folgenden Funktionen:

SIN, COS, ATN/ATN2

8.2.52 VAL

Funktion: Zeichkette in Zahl umwandeln

Die Funktion wandelt eine Zeichenkette in eine Zahl um.

Eingabeformat

```
<Numerische Variable> = VAL (<Zeichenkette>)
```

Programmbeispiel

10	M1 = VAL("15")	Weist der numerischen Variablen M1 den Wert der Zeichenkette „15“ zu
20	M2 = VAL("&B1111")	Weist der numerischen Variablen M2 den Wert der Zeichenkette „&B1111“ zu
30	M3 = VAL("&HF")	Weist der numerischen Variablen M3 den Wert der Zeichenkette „&HF“ zu

Erläuterung

- Die Funktion wandelt die angegebene Zeichenkette in einen numerischen Ausdruck um.
- Die Zeichenkette kann binär (&B) und hexadezimal (&H) dargestellt werden.
- Im Programmbeispiel wird in alle Variablen M1, M2 und M3 der Wert „15“ übertragen.

Steht in Beziehung zu folgenden Funktionen:

BIN\$, HEX\$, STR\$

8.2.53 ZONE

Funktion: Position prüfen

Die Funktion prüft, ob die Position innerhalb eines durch zwei Punkte definierten Quaders liegt.

Eingabeformat

```
<Numerische Variable> = ZONE (<Position 1>,<Position 2>,  
                               <Position 3>)
```

<Position 1>	Legt die Position fest, deren Lage geprüft werden soll
<Position 2>	Legt die erste Position zur Bereichsdefinition fest
<Position 3>	Legt die zweite Position zur Bereichsdefinition fest

Programmbeispiel

10 M1 = ZONE(P1,P2,P3)	Weist der numerischen Variablen M1 das Prüfergebnis zu
20 IF M1 = 1 THEN MOVE P_SAFE ELSE END	Fährt die Position P_SAFE an, falls die Position P1 innerhalb des durch die Positionen P2 und P3 definierten Quaders liegt und beendet das Programm, falls die Position außerhalb liegt

Erläuterung

- Die Funktion ZONE prüft, ob die Position 1 innerhalb eines durch die Positionen 2 und 3 definierten Quaders liegt. Die Positionen 2 und 3 bilden zwei Punkte auf der Diagonalen des erzeugten Quaders. Liegt die Position 1 innerhalb des Quaders, wird eine „1“, ansonsten eine „0“ in die numerische Variable übertragen.

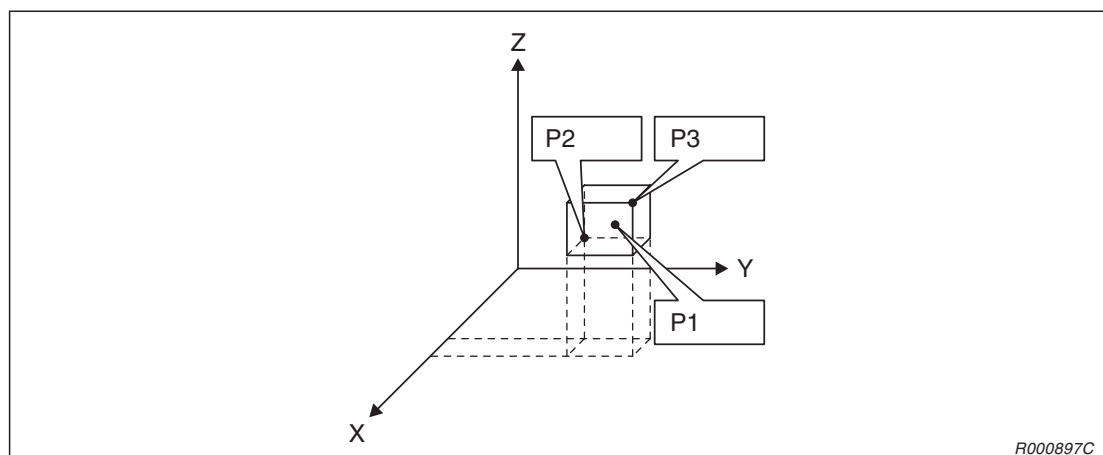


Abb. 8-4: Definition eines quaderförmigen Prüfbereiches

- Das Prüfergebnis wird ermittelt, indem für jedes Element der Position 1 (X, Y, Z, A, B, C, L1 und L2) untersucht wird, ob es zwischen den entsprechenden Werten der Positionen 2 und 3 liegt.
- Bei den Stellungsdaten (A, B und C) wird geprüft, ob die Werte der Position 1 innerhalb des Bereichs liegen, der bei Rotation vom Winkel der Position 2 zum Winkel der Position 3 in positiver Richtung durchfahren wird.

Beispiel ▽

Sind die A-Komponenten der Position 2 und 3: P2.A = -100, P3.A = +100, liegt die Position 1 mit der A-Komponente P1.A = 50 innerhalb des Bereiches. Diese Prüfung wird in gleicher Weise auch für die B- und C- Komponente durchgeführt.

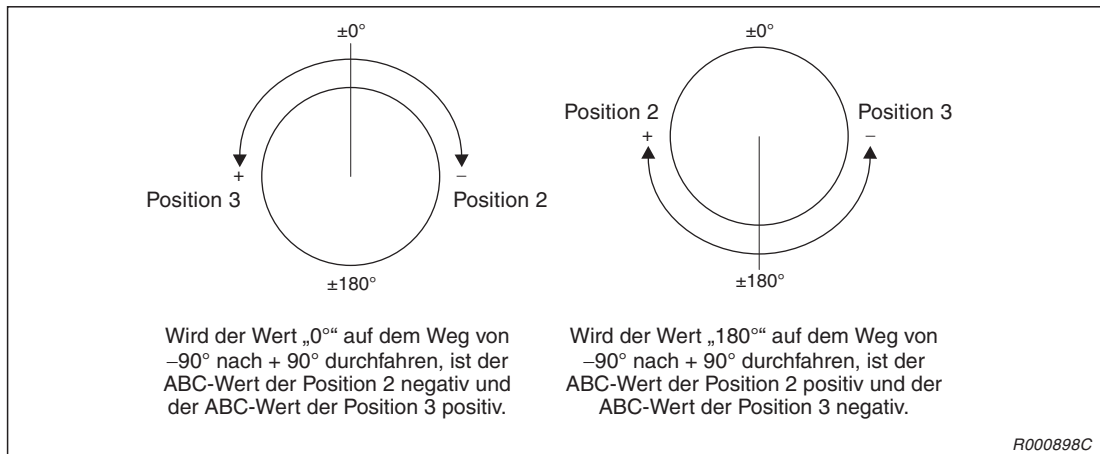


Abb. 8-5: Prüfung der Stellungendaten



- Komponenten, die nicht geprüft werden sollen oder fehlen, müssen auf folgende Werte gesetzt werden:
 Bei Einheit Grad:
 - Position 2 wird auf -360° gesetzt
 - Position 3 wird auf 360° gesetzt
 Bei Einheit mm:
 - Position 2 wird auf -10000 gesetzt
 - Position 3 wird auf 10000 gesetzt
- Auf die Argumente <Position 1>, <Position 2> und <Position 3> der Funktion ZONE darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

8.2.54 ZONE2

Funktion: Position prüfen

Die Funktion prüft, ob die Position innerhalb eines durch zwei Punkte definierten Zylinders liegt.

Eingabeformat

Ab Software-Version G5:

```
<Numerische Variable> = ZONE2    (<Position 1>,<Position 2>,  
                                   <Position 3>,  
                                   <Numerischer Ausdruck>)
```

<Position 1>	Legt die Position fest, deren Lage geprüft werden soll
<Position 2>	Legt die erste Position zur Bereichsdefinition fest
<Position 3>	Legt die zweite Position zur Bereichsdefinition fest
<Numerischer Ausdruck>	Legt den Radius des Bereichs an beiden Endpunkten fest

Programmbeispiel

10 M1 = ZONE2(P1,P2,P3,50)	Weist der numerischen Variablen M1 das Prüfergebnis zu
20 IF M1 = 1 THEN MOVE P_SAFE ELSE END	Fährt die Position P_SAFE an, falls die Position P1 innerhalb des durch die Positionen P2 und P3 definierten Zylinders liegt und beendet das Programm, falls die Position außerhalb liegt

Erläuterung

- Die Funktion ZONE2 prüft, ob die Position 1 innerhalb eines durch die Positionen 2 und 3 definierten Zylinders liegt. Die Positionen 2 und 3 bilden zwei Endpunkte des Zylinders. Der numerische Ausdruck legt einen Radius um diese Eckpunkte fest. Liegt die Position 1 innerhalb des Rechtecks, wird eine „1“, ansonsten eine „0“ in die numerische Variable übertragen.
- Die Funktion prüft, ob die XYZ-Komponenten einer Position innerhalb des festgelegten Bereiches liegen. Die Stellungsdaten werden nicht geprüft.

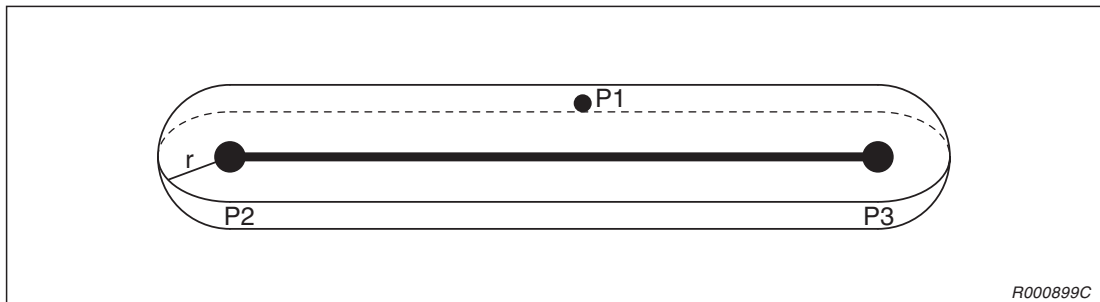


Abb. 8-6: Definition des zylindrischen Prüfbereiches

- Auf die Argumente <Position 1>, <Position 2>, <Position 3> und <Numerischer Ausdruck> der Funktion ZONE2 darf keine weitere Funktion angewendet werden. Bei einer solchen Verschachtelung erfolgt bei der Ausführung eine Fehlermeldung.

9 Parameter

9.1 Allgemeines

Folgende Tabelle zeigt die anwendungsspezifische Einteilung der Parameter der Steuergeräte CR1, CR2, CR2A, CR2B und CR3. Mit Hilfe der Parameter können verschiedene Funktionen und Grundeinstellungen beeinflusst werden.

Anwendung	Beschreibung	Seite
Bewegungsparameter	Diese Parameter dienen der Einstellung des Bewegungsbereiches, des Koordinatensystems und auf die Hand bezogener Größen.	9-2
Signalparameter	Diese Parameter dienen der Einstellung von Größen zur Beeinflussung von Signalen.	9-15
Betriebsparameter	Diese Parameter dienen der Einstellung von Größen zur Beeinflussung des Steuergeräte- und des Taching-Box-Betriebs usw.	9-17
Befehlsparameter	Diese Parameter dienen der Einstellung von Größen zur Beeinflussung der Programmausführung.	9-21
Kommunikationsparameter	Diese Parameter dienen der Einstellung von Größen zur Beeinflussung der Kommunikation.	9-27

Tab. 9-1: Anwendungsspezifische Einteilung der Parameter

Parameter zur Steuerung von Ein- und Ausgängen sind in Kap. 10 erläutert.

Die Änderung einer Parametereinstellung wird erst nach Aus- und Wiedereinschalten der Spannungsversorgung des Steuergerätes aktiv.

Eine genaue Beschreibung der Vorgehensweise zur Einstellung von Parametern finden Sie in Abschn. 3.13.1.



GEFAHR:

Nehmen Sie Parametereinstellungen nur mit besonderer Sorgfalt vor. Prüfen Sie die Auswirkungen einer neuen Einstellung auf die beeinflusste Funktion. Fehlerhaft eingestellte Parameter können zu undefinierten Aktivitäten des Roboters führen. Es besteht Verletzungsgefahr!

9.2 Bewegungsparameter

Diese Parameter dienen der Einstellung des Bewegungsbereiches, des Koordinatensystems und auf die Hand bezogener Größen.

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Verfahrweggrenzen für Gelenkbewegungen	MEJAR	Reelle Zahl 16	Legt die Verfahrweggrenzen für jedes einzelne Gelenk fest Um eine Begrenzung durch die mechanischen Anschläge zu vermeiden, wird eine Überschreitung der Verfahrweggrenzen nicht empfohlen. (-J1, +J1, -J2, +J2, ... -J8, +J8) Einheit: Grad	Abhängig vom Mechanismus
Verfahrweggrenzen für XYZ-Bewegungen	MEPAR	Reelle Zahl 6	Legt die Verfahrweggrenzen für das XYZ-Koordinatensystem fest Der Parameter kann dazu verwendet werden, Kollisionen des Roboters mit umliegenden Einrichtungen zu verhindern. (-X, +X, -Y, +Y, -Z, +Z) Einheit: mm	(-X, +X, -Y, +Y, -Z, +Z) = -10000, 10000, -10000, 10000, -10000, 10000
Standardwerkzeugkoordinaten (siehe auch Abschn. 9.7)	MEXTL	Reelle Zahl 6	Legt den Werkzeugmittelpunkt TCP fest Verwenden Sie diesen Parameter, wenn Sie einen Handgreifer installieren und der Werkzeugmittelpunkt angepasst werden muss. Der Parameter ermöglicht eine Überwachung der Stellung an der Handspitze für den XYZ- oder den Werkzeug-JOG-Betrieb. (X, Y, Z, A, B, C) Einheit: mm oder Grad	(X, Y, Z, A, B, C) = 0.0,0.0,0.0,0.0,0.0,0.0
Werkzeugkoordinaten 1 (siehe auch Abschn. 7.2.44)	MEXTL1	Reelle Zahl 6	Diese Werkzeugdaten sind wirksam, wenn die Variable M_TOOL auf „1“ gesetzt ist.	(X, Y, Z, A, B, C) = 0.0,0.0,0.0,0.0,0.0,0.0
Werkzeugkoordinaten 2 (siehe auch Abschn. 7.2.44)	MEXTL2	Reelle Zahl 6	Diese Werkzeugdaten sind wirksam, wenn die Variable M_TOOL auf „2“ gesetzt ist.	(X, Y, Z, A, B, C) = 0.0,0.0,0.0,0.0,0.0,0.0
Werkzeugkoordinaten 3 (siehe auch Abschn. 7.2.44)	MEXTL3	Reelle Zahl 6	Diese Werkzeugdaten sind wirksam, wenn die Variable M_TOOL auf „3“ gesetzt ist	(X, Y, Z, A, B, C) = 0.0,0.0,0.0,0.0,0.0,0.0
Werkzeugkoordinaten 4 (siehe auch Abschn. 7.2.44)	MEXTL4	Reelle Zahl 6	Diese Werkzeugdaten sind wirksam, wenn die Variable M_TOOL auf „4“ gesetzt ist	(X, Y, Z, A, B, C) = 0.0,0.0,0.0,0.0,0.0,0.0
Standardbasiskoordinaten (siehe auch Abschn. 9.8)	MEXBS	Reelle Zahl 6	Legt das Roboterkoordinatensystem in Beziehung zum Weltkoordinatensystem fest In der Werkseinstellung sind Roboter- und Weltkoordinatensystem identisch. Eine Einstellung des Parameters ist nur dann nötig, wenn der Aufbau der gesamten Anlage verändert wird oder, wenn die gesamte Anlage auf ein Koordinatensystem bezogen ist. (X, Y, Z, A, B, C) Einheit: mm oder Grad	(X, Y, Z, A, B, C) = 0.0,0.0,0.0,0.0,0.0,0.0

Tab. 9-2: Übersicht der Bewegungsparameter (1)

Parameter	Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Benutzerdefinierte Verfahrweggrenze		Über zwei Punkte wird ein quaderförmiger Bereich festgelegt. Ein Eindringen in diesen Bereich wird als Verfahrwegüberschreitung definiert und ein korrespondierendes Signal kann geschaltet werden. Es können 8 Bereiche definiert werden. Für Komponenten, die nicht geprüft werden sollen oder fehlen, müssen die entsprechenden Positionskordinaten auf folgende Werte gesetzt werden: Ist die Einheit Grad, muss AREA*P1 auf -360° und AREA*P2 auf 360° gesetzt werden. Ist die Einheit mm, muss AREA*P12 auf -10000 und AREA*P2 auf 10000 gesetzt werden.	
AREA1P1 : AREA8P1	Reelle Zahl 8	Festlegung des 1. Bereichspunktes Setzen Sie die Elemente in folgender Reihenfolge: X, Y, Z, A, B, C	(X, Y, Z, A, B, C) = 0.0,0.0,0.0,-360.0, -360.0,-360.0
AREA1P2 : AREA8P2	Reelle Zahl 8	Festlegung des 2. Bereichspunktes Setzen Sie die Elemente in folgender Reihenfolge: X, Y, Z, A, B, C	(X, Y, Z, A, B, C) = 0.0,0.0,0.0,+360.0, +360.0,+360.0
AREA1ME : AREA8ME	Ganze Zahl 1	Zuweisung der Begrenzungsbereiche an die Mechanismen 1 bis 4 Standardmäßig ist der Wert auf „1“ gesetzt.	0
AREA1AT : AREA8AT	Ganze Zahl 1	Festlegung der Bereichsprüfmethode: Gesperrt/Ausgabe eines In-Bereich-Signals/Fehler = 0/1/2 Gesperrt: Die Funktion ist deaktiviert. Ausgabe eines In-Bereich-Signals: Das Signal USRAREA wird eingeschaltet. Fehler: Es erfolgt eine Fehlermeldung.	0
USRAREA		Definition der Signalnummer zur Ausgabe des Statussignals (siehe auch Kap. 10)	-1, -1

Tab. 9-2: Übersicht der Bewegungsparameter (2)

Parameter	Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung	
Verfahrwegbegrenzungsebene (siehe auch Abschn. 9.10) Die Funktion bei der Einstellung „-1 (Der zugelassene Arbeitsbereich liegt auf der Seite, auf der der Nullpunkt des Basiskoordinatensystems nicht liegt.)“ ist ab Software-Version J1 verfügbar.		Die Verfahrweggrenzen werden über eine Ebene definiert. Die Ebene wird über die Koordinaten X1, Y1, Z1 bis X3, Y3, Z3 festgelegt. Bei Überschreitung dieser Bereichsgrenzen erfolgt eine Fehlermeldung. Folgende 3 Parametertypen können verwendet werden:		
SFC1P : SFC8P	Reelle Zahl 9	Über SFC1P bis SFC8P können 8 Begrenzungsebenen definiert werden. Setzen Sie die dazu nötigen 9 Elemente in folgender Reihenfolge: X1, Y1, Z1: Flächenursprung X2, Y2, Z2: Position auf der X-Achse in der Ebene X3, Y3, Z3: Position in positiver Richtung auf der Y-Achse in der Ebene Einheit: mm	(X1, Y1, Z1, X2, Y2, Z2, X3, Y3, Z3) = 0.0,0.0,0.0,0.0,0.0,0.0, 0.0,0.0,0.0	
SFC1ME : SFC8ME	Ganze Zahl 1	Zuweisung der Begrenzungsebenen an die Mechanismen 1 bis 3 Standardmäßig ist der Wert auf „1“ gesetzt.	0	
SFC1AT : SFC8AT	Ganze Zahl 1	Freigabe der Begrenzungsebenen: 0: gesperrt 1: freigegeben (Der zugelassene Arbeitsbereich liegt auf der Seite, auf der auch der Nullpunkt des Basiskoordinatensystems liegt.) -1: freigegeben (Der zugelassene Arbeitsbereich liegt auf der Seite, auf der der Nullpunkt des Basiskoordinatensystems nicht liegt.)	0 (gesperrt)	
Position des Rückzugpunktes	JSAFE	Reelle Zahl 8	Festlegung der Position des Rückzugpunktes Der Roboter fährt die Position des Rückzugpunktes bei Ausführung des Befehls MOV P_SAFE oder bei anliegendem externen SAFEPOS-Signal an. (J1, J2, J3, J4, J5, J6, J7, J8) Einheit: Grad	Nicht zulässig
Benutzerdefinierter Nullpunkt	USERORG	Reelle Zahl 8	Festlegung des benutzerdefinierten Nullpunkts (J1, J2, J3, J4, J5, J6, J7, J8) Einheit: Grad	(J1, J2, J3, J4, J5, J6, J7, J8) = 0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0
Fehlermeldung bei Annäherung an den singulären Punkt (siehe auch Abschn. 9.18.) Dieser Parameter ist ab Software-Version G8 verfügbar.	MESNGLSW	Ganze Zahl 1	Freigabe einer Fehlermeldung bei Annäherung an den singulären Punkt. (gesperrt/freigegeben = 0/1) Ist die Fehlermeldung über den Parameter freigegeben, ertönt der Warnton auch dann, wenn der Summer über den Parameter BZR (Summer EIN/AUS) ausgeschaltet ist.	1 (freigegeben)

Tab. 9-2: Übersicht der Bewegungsparameter (3)

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
JOG-Einstellung	JOGJSP	Reelle Zahl 3	Festlegung der Geschwindigkeit für den Gelenk-JOG- und den Schrittbetrieb (Einstellung der Werte H/L, maximaler Übersteuerungswert)	Abhängig vom Mechanismus
	JOGPSP	Reelle Zahl 3	Festlegung der Geschwindigkeit für den Linear-JOG- und den Schrittbetrieb (Einstellung der Werte H/L, maximaler Übersteuerungswert) Der maximale Wert von 250 mm/s kann nicht überschritten werden.	Abhängig vom Mechanismus
Geschwindigkeitsbegrenzung für den JOG-Betrieb	JOGSPMX	Reelle Zahl 1	Geschwindigkeitsbegrenzung im TEACH-Modus Einheit: mm/s (max. 250 mm/s)	250.0
Automatische Rückkehr nach einem Interrupt	RETPATH	Ganze Zahl 1	Bewirkt die Fortsetzung des Programms nach Auftreten eines Interrupts von der Interruptposition aus Ist die Funktion deaktiviert und der Roboter wird nach einem Interrupt im JOG-Betrieb zu einer anderen Position bewegt, erfolgt die Weiterführung der Roboterbewegung bei Fortsetzung des Programms von der aktuellen Position aus. Der Roboter kehrt nicht zu der Position zurück, an der er sich zum Zeitpunkt des Interrupts befunden hat. 0: Funktion deaktiviert 1: Rückkehr mittels Gelenk-Interpolation 2: Rückkehr mittels Linear-Interpolation HINWEISE: Die Einstellung „2“ ist ab Software-Version H4 möglich. Führen Sie bei Rückkehr mittels Linear-Interpolation kürzere Kreisbewegungen über die 3-Achsen-XYZ-Interpolation aus. Für die Kreis-Interpolation (MVC, MVR, MVR2, MVR3) kann die Funktion ab Software-Version H4 genutzt werden. Bei der Kreis- und Bogen-Interpolation (MVA) ist die Funktion bei der Einstellung „0“ dieselbe wie bei der Einstellung „1“.	1 (aktiviert)

Tab. 9-2: Übersicht der Bewegungsparameter (4)

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung										
Montagerichtung Dieser Parameter ist ab Software-Version H4 verfügbar. Der Parameter kann mit folgenden Robotermodellen verwendet werden: RV-1A/2AJ, RV-4A/5AJ	MEGDIR	Reelle Zahl 4	<p>Legt die Richtung und Größe der auf den Roboter wirkenden Erdbeschleunigung in Abhängigkeit der Montagerichtung für die X-, Y- und Z-Achse des Basiskoordinatensystems fest (Einheit: mm/s²). Der Parameter besteht aus 4 Elementen: Montagerichtung, Erdbeschleunigung in X-, Y- und in Z-Richtung</p> <table border="1"> <thead> <tr> <th>Montagerichtung</th> <th>Montagerichtung, Erdbeschleunigung in X-, Y- und in Z-Richtung</th> </tr> </thead> <tbody> <tr> <td>Bodenmontage</td> <td>(0.0, 0.0, 0.0, 0.0)</td> </tr> <tr> <td>Wandmontage</td> <td>(1.0, 0.0, 0.0, 0.0)</td> </tr> <tr> <td>Deckenmontage</td> <td>(2.0, 0.0, 0.0, 0.0)</td> </tr> <tr> <td>Spezielle Montage ①</td> <td>(3.0, ***, ***, ***)</td> </tr> </tbody> </table> <p>① Die Zeichen „***“ entsprechen einem numerischen Wert (siehe folgendes Beispiel)</p> <p>Beispiel: Der Roboter wird 30° nach vorne geneigt montiert. Für die Erdbeschleunigung in X-Richtung ergibt sich: $X_g = 9,8 \sin 30^\circ = 4,9$ Für die Erdbeschleunigung in Z-Richtung ergibt sich: $Z_g = 9,8 \cos 30^\circ = 8,5$ Der Wert muss auf $-8,5$ gesetzt werden, da der Vektor in die negative Richtung der Z-Achse zeigt. Für die Erdbeschleunigung in Y-Richtung ergibt sich: $Y_g = 0,0$ Daraus ergibt sich die Einstellung des Parameters: (3.0, 4.9, 0.0, -8.5)</p>	Montagerichtung	Montagerichtung, Erdbeschleunigung in X-, Y- und in Z-Richtung	Bodenmontage	(0.0, 0.0, 0.0, 0.0)	Wandmontage	(1.0, 0.0, 0.0, 0.0)	Deckenmontage	(2.0, 0.0, 0.0, 0.0)	Spezielle Montage ①	(3.0, ***, ***, ***)	0.0, 0.0, 0.0, 0.0
Montagerichtung	Montagerichtung, Erdbeschleunigung in X-, Y- und in Z-Richtung													
Bodenmontage	(0.0, 0.0, 0.0, 0.0)													
Wandmontage	(1.0, 0.0, 0.0, 0.0)													
Deckenmontage	(2.0, 0.0, 0.0, 0.0)													
Spezielle Montage ①	(3.0, ***, ***, ***)													

Tab. 9-2: Übersicht der Bewegungsparameter (5)

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Initialisierungsstatus der HAND (siehe auch Abschn. 9.14)	HANDINIT	Ganze Zahl 8	Festlegung der Ausgänge der Schnittstellenkarte für die pneumatische Greifhand nach Einschalten der Spannungsversorgung Der Parameter legt den Handgreiferzustand nach der Initialisierung über die 900er Signale fest. Soll die Einstellung des Handgreiferzustandes nach der Initialisierung über allgemeine E/A (andere als die 900er Signale) oder CC-Link (ab 6000) (im Parameter HANDTYPE sind andere Signale als die 900er festgelegt) erfolgen, verwenden Sie zur Festlegung der Zustände nicht den Parameter HANDINIT, sondern die Parameter ORST*. Die über die Parameter ORST* gesetzten Ausgangsbitmuster entsprechen dem Initialisierungszustand der Signale nach dem Einschalten der Spannungsversorgung.	1,0,1,0,1,0,1,0
Handausführung (siehe auch Abschn. 9.13)	HANDTYPE	Zeichenkette 8	Festlegung der Handausführung (Einfach-/Doppelmagnetspule = S/D) und Signalnummer Geben Sie erst den Handtyp, dann die Signalnummer an: z. B. D900. Bei einer Einstellung von D900 werden die Signale Nr. 900 und 901 ausgegeben. Ist über die Einstellung „D“ eine Doppelmagnetspule gewählt, setzen Sie die Signalnummern so, dass keine Überlappung der Signale auftritt.	D900, D902, D904, D906, , , ,

Tab. 9-2: Übersicht der Bewegungsparameter (6)

Parameter	Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung	
Hand- und Werkstückbedingungen (Bei optimaler Beschleunigung/ Abbremsung und aktivierter Kollisions- überwachung) (siehe auch Abschn. 9.17.1)		Einstellung der Hand- und Arbeits- bedingungen, wenn über OADL ON die optimale Beschleunigung/ Abbremsung gewählt wurde Es können bis zu 8 Bedingungen definiert werden. Die Auswahl von Kombinationen der Bedingungen erfolgt über den Befehl LOADSET.		
HNDDAT0	Reelle Zahl 7	Einstellung der Startbedingung für die Hand (Festlegung im Werkzeug- koordinatensystem) Nach dem Einschalten der Span- nungsversorgung ist dieser Wert wirksam. Der Parameter muss vor Aktivierung der Kollisionsüberwa- chung im JOG-Betrieb eingestellt werden. Ist der Parameter nicht ein- gestellt, kann ein fehlerhaftes An- sprechen der Kollisionsüberwachung erfolgen. (Gewicht, Größe X, Größe Y, Größe Z, Schwerpunkt X, Schwerpunkt Y, Schwerpunkt Z) Einheit: kg, mm	Für die Robotermodelle RV-3SB/6SJB: 3.50,284.00,284.00, 286.00,0.00,0.00,75.00 RV-6S/6SL: 6.00,213.00,213.00, 17.00,0.00,0.00,130.00 RV-12S/12SL: 12.00,265.00,265.00, 22.00,0.00,0.00,66.00 RH-6SH: 6.00,99.00,99.00, 76.00,0.00,0.00,38.00 RH-12SH: 12.00,225.00,225.00, 30.00,0.00,0.00,15.00	
HNDDAT1 : HNDDAT8	Reelle Zahl 7	Einstellung der Startbedingung für die Hand (Festlegung im Werkzeug- koordinatensystem) (Gewicht, Größe X, Größe Y, Größe Z, Schwerpunkt X, Schwerpunkt Y, Schwerpunkt Z) Einheit: kg, mm	Standardlast, 0.0,0.0,0.0,0.0,0.0,0.0	
WRKDAT0	Reelle Zahl 7	Einstellung der Startbedingung für das Werkstück (Festlegung im Werkzeugkoordinatensystem) Nach dem Einschalten der Span- nungsversorgung ist dieser Wert wirksam. (Gewicht, Größe X, Größe Y, Größe Z, Schwerpunkt X, Schwerpunkt Y, Schwerpunkt Z) Einheit: kg, mm	Für die Robotermodelle RV-S/RH-S: 0.0,0.0,0.0,0.0,0.0,0.0	
WRKDAT1 : WRKDAT8	Reelle Zahl 7	Einstellung der Werkstückbedingun- gen (Festlegung im Werkzeugkoordinatensystem) (Gewicht, Größe X, Größe Y, Größe Z, Schwerpunkt X, Schwerpunkt Y, Schwerpunkt Z) Einheit: kg, mm	0.0,0.0,0.0,0.0,0.0,0.0	
HNDHOLD1 : HNDHOLD8	Ganze Zahl 2	Festlegung, ob die Hand beim Be- fehl HOPEN (HCLOSE) geschlos- sen wird oder nicht (Einstellung für Öffnen, Einstellung für Schließen) (nicht schließen/schließen = 0/1)	0, 1	
Maximale Be- schleuni- gung/Ab- bremsung (siehe auch Abschn.9.17.1) Dieser Para- meter ist ab Software- Version G1 verfügbar.	ACCMODE	Ganze Zahl 1	Festlegung des Startwertes und der optimalen Beschleunigung/Verzöge- rung (gesperrt/freigegeben = 0/1)	Für die Robotermodelle RH-A/RH-S/RV-S: 1, für alle anderen Robotermodelle: 0

Tab. 9-2: Übersicht der Bewegungsparameter (7)

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Einstellzeit für die optimale Beschleunigungs-/ Abbremszeit Dieser Parameter ist ab Software-Version J2 verfügbar. Der Parameter kann mit folgenden Robotermodellen verwendet werden: RV-S	JADL	Reelle Zahl 8	<p>Festlegung des Startwertes (nach Einschalten der Spannungsversorgung) der Einstellzeit für die optimale Beschleunigungs-/Abbremszeit beim Betrieb mit optimaler Beschleunigung/Abbremsung. Dieser Wert wird dem berechneten Wert für die optimale Beschleunigung/Abbremsung überlagert. Ein großer Einstellwert führt bei den Robotermodellen RV-S zu einer Verkürzung der Zykluszeiten. Gleichzeitig steigt jedoch die Wahrscheinlichkeit von Überlast- und Überhitzungsfehlern. Verkleinern Sie in solchen Fällen die Einstellwerte. Werkseitig sind die Werte so eingestellt, dass Überlast- und Überhitzungsfehler vermieden werden. Die Werte sind für die Beschleunigungs- und die Abbremszeit wirksam.</p> <ul style="list-style-type: none"> ● Definition eines Überlastfehlers: Ein Überlastfehler tritt dann auf, wenn Größe der Last bei einer Verfahrbewegung mit hoher Geschwindigkeit eine Überhitzung des Motors zur Folge haben könnte. ● Definition eines Überhitzungsfehlers: Ein Überhitzungsfehler tritt dann auf, wenn der Encoder aufgrund der hohen Temperatur beschädigt werden könnte. <p>Einheit: %</p>	<p>Für die Robotermodelle RV-3SB/3SJB: 100,100,100,100, 100,100,100,100, RV-6S: 50,50,50,50,50,50,50,50 RV-6SL: 35,35,35,35,35,35,35,35 RV-12S: 50,50,50,50,50,50,50,50 RV-12SL: 35,35,35,35,35,35,35,35</p>
Kollisionsüberwachung Dieser Parameter ist ab Software-Version J2 verfügbar. Der Parameter kann mit folgenden Robotermodellen verwendet werden: RV-S/RH-S	COL	Ganze Zahl 3	<p>Freigabe der Kollisionsüberwachung und Aktivierung direkt nach Einschalten der Spannungsversorgung</p> <p>1. Element: Kollisionsüberwachung 0 = deaktiviert 1 = aktiviert</p> <p>2. Element: Aktivierung direkt nach Einschalten der Spannungsversorgung 0 = deaktiviert 1 = aktiviert</p> <p>3. Element: Freigabe im JOG-Betrieb 0 = deaktiviert 1 = aktiviert 2 = NOERR-Modus</p> <p>Im NOERR-Modus erfolgt keine Fehlerausgabe, auch wenn die Kollisionsüberwachung anspricht. Die Servoversorgung wird jedoch abgeschaltet. Verwenden Sie diesen Modus, wenn kein störungsfreier Betrieb durch häufiges Ansprechen der Kollisionsüberwachung möglich ist.</p>	<p>Für die Robotermodelle RV-S: 0,0,1 RH-S: 1,0,1</p>

Tab. 9-2: Übersicht der Bewegungsparameter (8)

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Ansprechschwelle der Kollisionsüberwachung Dieser Parameter ist ab Software-Version J2 verfügbar. Der Parameter kann mit folgenden Robotermodellen verwendet werden: RV-S/RH-S	COLLVL	Ganze Zahl 8	Einstellung der Ansprechschwelle im Programmbetrieb für jede Achse Bei Einstellung eines Wertes außerhalb des zulässigen Einstellbereiches wird der nächste zulässige Wert gesetzt. Einstellbereich: 1 bis 500 % Einheit: %	200,200,200,200, 200,200,200,200
Ansprechschwelle der Kollisionsüberwachung im JOG-Betrieb Dieser Parameter ist ab Software-Version J2 verfügbar. Der Parameter kann mit folgenden Robotermodellen verwendet werden: RV-S/RH-S	COLLVLJG	Reelle Zahl 8	Einstellung der Ansprechschwelle im JOG-Betrieb für jede Achse Zur Erhöhung der Empfindlichkeit ist der numerische Wert zu verringern. Spricht die Kollisionsüberwachung im JOG-Betrieb auch ohne einen Zusammenstoß an, erhöhen Sie den Wert. Einstellbereich: 1 bis 500 % Einheit: %	Der Standardwert ist 200,200,200,200, 200,200,200,200 und modellabhängig

Tab. 9-2: Übersicht der Bewegungsparameter (9)

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Koordinatensystem für den Handgelenkdrehwinkel (Achse A)	RCD	Ganze Zahl 1	<p>Auswahl des Steuer- und Anzeigemodus des Handgelenkdrehwinkels (Achse A im XYZ-Koordinatensystem) eines 5-achsigen Roboters</p> <p>2: allgemeiner Winkelmodus</p> <p>Die Achse A wird so gesteuert, dass die Stellung der Hand aufrecht erhalten wird, wenn der Wert der Achse A vor einer Bewegung mit dem Wert nach der Bewegung übereinstimmt. Beachten Sie, dass die Stellung der Hand in Abhängigkeit der Handgelenkneigung (Achse B im XYZ-Koordinatensystem) nicht immer aufrecht erhalten werden kann. Im Normalfall kann die Werkseinstellung beibehalten werden.</p> <p>0/1/3: allgemeiner Winkelmodus der E-Serie/Gelenkwinkelmodus/alter allgemeiner Winkelmodus</p> <p>Diese Einstellungen dienen der Kompatibilität von Roboterprogrammen (Positionsdaten) und älteren Robotermodellen (z. B. RV-E3J, RV-E5NJ). Um Programme (Positionsdaten), die für ältere Robotermodelle entworfen wurden, weiterzuverwenden, setzen Sie den Parameter RCD auf den für das ältere Robotermodell festgelegten Wert.</p> <p>Die Modi sind untereinander nicht kompatibel. Bei zwei unterschiedlichen Einstellungen des Parameters können die Handstellungen während einer Roboterbewegung voneinander abweichen, auch wenn die gleichen Positionsdaten angefahren werden. Wählen Sie den Modus so, wie Sie die Positionsdaten bei Ausführung des Programms verwenden möchten.</p>	2 (allgemeiner Winkelmodus)
Freigabe des Warmlaufbetriebs Dieser Parameter ist ab Software-Version G8 verfügbar.	WUPENA	Ganze Zahl 1	<p>Warmlaufbetrieb freigeben oder sperren</p> <p>0: gesperrt 1: freigegeben</p> <p>HINWEISE: Bei Einstellung von anderen als den oben genannten Werten ist der Warmlaufbetrieb gesperrt.</p> <p>Bei mehreren Mechanismen muss der Modus für jeden Mechanismus eingestellt werden.</p>	0 (gesperrt)

Tab. 9-2: Übersicht der Bewegungsparameter (10)

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Auswahl der Achse für den Warmlaufbetrieb Dieser Parameter ist ab Software-Version G8 verfügbar.	WUPAXIS	Ganze Zahl 1	<p>Auswahl der Achse für den Warmlaufbetrieb durch Ein- und Ausschalten der Bits im Hexadezimalcode (J1, J2 ... mit dem niedrigsten Bit beginnend)</p> <p>Bit EIN: Achse auswählen Bit AUS: Achse nicht auswählen</p> <p>Wählen Sie die Achsen aus, die bei niedrigen Temperaturen große Positionsabweichungen verursachen.</p> <p>HINWEISE: Durch Setzen eines Bits, zu dem keine Achse existiert, wird keine Achse ausgewählt.</p> <p>Wird keine Achse für einen Warmlaufbetrieb festgelegt, bleibt der Warmlaufbetrieb deaktiviert.</p> <p>Bei mehreren Mechanismen müssen die Achsen für jeden Mechanismus ausgewählt werden.</p>	<p>Für die Robotermodelle RV-6S/12S: 00111000 (J4-, J5- und J6-Achse)</p> <p>RV-3SB: 00001110</p> <p>RV-3SJB: 00000110</p> <p>für alle anderen Robotermodelle: 0</p>
Dauer des Warmlaufbetriebs Dieser Parameter ist ab Software-Version G8 verfügbar.	WUPTIME	Reelle Zahl 2	<p>Einstellung der Zeit für den Warmlaufbetrieb (Gesamtdauer, Wiederholtschwelle) in Minuten</p> <p>Gesamtdauer: Legen Sie die Dauer fest, in der die Achse im Warmlaufbetrieb mit reduzierter Geschwindigkeit verfahren wird. (Einstellbereich 0 bis 60)</p> <p>Wiederholtschwelle: Legen Sie die Zeit fest, nach der nach Beendigung eines Warmlaufbetriebs und einer kontinuierlichen Betriebspause einer Achse erneut ein Warmlaufbetrieb aktiviert wird (Einstellbereich: 1 bis 1440)</p> <p>HINWEISE: Bei Einstellung eines Wertes außerhalb des Einstellbereichs, wird der nächstliegende Wert innerhalb des Einstellbereichs verwendet.</p> <p>Ist eine Gesamtdauer von „0“ eingestellt, wird der Warmlaufbetrieb deaktiviert.</p> <p>Bei mehreren Mechanismen müssen die Werte für jeden Mechanismus eingestellt werden.</p>	1, 60

Tab. 9-2: Übersicht der Bewegungsparameter (11)

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Übersteuerung im Warmlaufbetrieb Dieser Parameter ist ab Software-Version G8 verfügbar.	WUPOVRD	Ganze Zahl 2	Einstellung der Geschwindigkeit im Warmlaufbetrieb (Startwert, Faktor für den Bereich konstanter Geschwindigkeit) in Prozent Startwert: Startwert der Geschwindigkeitsübersteuerung im Warmlaufbetrieb (Einstellbereich 50 bis 100) Faktor für den Bereich konstanter Geschwindigkeit: Stellen Sie den Bereich der konstanten Geschwindigkeit in Bezug zur Gesamtdauer des Warmlaufbetriebs ein (Einstellbereich 0 bis 50) Folgende Abbildung zeigt den Einfluss der eingestellten Werte auf die Geschwindigkeit.	70, 50
			HINWEISE: Bei Einstellung eines Wertes außerhalb des Einstellbereichs, wird der nächstliegende Wert innerhalb des Einstellbereichs verwendet Ist für die Übersteuerung ein Startwert von 100 % eingestellt, bleibt der Warmlaufbetrieb deaktiviert. Bei mehreren Mechanismen müssen die Werte für jeden Mechanismus eingestellt werden.	

Tab. 9-2: Übersicht der Bewegungsparameter (12)

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Freigabe von Fehlermeldungen im Betrieb mit aktivierter Achsenweichheit Dieser Parameter ist ab Software-Version H6 verfügbar.	CMPERR	Ganze Zahl 1	<p>Mit diesem Parameter kann die Generierung der Fehler 2710 bis 2740, die bei aktivierter Achsenweichheit auftreten können, gesperrt werden.</p> <p>1: Fehlergenerierung freigeben 0: Fehlergenerierung sperren</p> <p>Die Fehlermeldungen haben folgende Bedeutung:</p> <p>2710: Die Abweichung von der Sollwertposition ist zu groß.</p> <p>2720: Der zulässige Gelenkwinkel bei Ausführung des CMP-Befehls wurde überschritten.</p> <p>2730: Die zulässige Geschwindigkeit bei Ausführung des CMP-Befehls wurde überschritten.</p> <p>2740: Fehlerhafte Konvertierung von Koordinaten bei Ausführung des CMP-Befehls</p> <p>Tritt einer dieser Fehler auf, liegt eine Fehlfunktion im Betrieb mit aktivierter Achsenweichheit vor. Zur Korrektur des Fehlers ist es notwendig, die geteachten Positionen und den Programminhalt zu überprüfen. Setzen Sie den Parameter nur auf „0“ (Fehlergenerierung sperren), wenn Sie sicher sind, dass durch einen Betrieb ohne die Generierung entsprechender Fehler keine Komplikationen auftreten können.</p>	1 (Fehlergenerierung freigeben)

Tab. 9-2: Übersicht der Bewegungsparameter (13)

9.3 Signalparameter

Diese Parameter dienen der Einstellung von Größen zur Beeinflussung von Signalen.

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Spezielle E/A-Signale			Eine detaillierte Beschreibung der speziellen E/A-Signale finden Sie in Abschn. 10.2.1.	
Stopp-Eingangs-Signalbearbeitung	INB	Ganze Zahl 1	Definition des Stopp-Eingangs als Standard oder Drahtbrucherkennung (Standard/Drahtbrucherkennung = 0/1)	0 (Standard)
Einlesen der Programmnummer vom numerischen Eingang bei Eingabe des Startsignals	PST	Ganze Zahl 1	Die Programmauswahl über das normale externe Eingangssignal erfolgt durch Anlegen der Programmnummer als Eingangssignal (IODETA), durch Auswahl des Programms über das PRGSEL-Signal und durch Starten des Programms über das Startsignal. Durch Freigabe der Funktion über den Parameter PST kann die Programmauswahl über das Signal PRGSEL entfallen. Nach Einschalten des Startsignals wird die Programmnummer über den numerischen Eingang (IODETA) eingelesen. (gesperrt/freigegeben = 0/1)	0 (gesperrt)
CC-Link-Fehleraufhebung Dieser Parameter ist ab Software-Version H7 verfügbar.	E7730	Ganze Zahl 1	Ist im Steuergerät eine CC-Link-Schnittstelle installiert, das Steuergerät aber nicht an ein CC-Link-Netzwerk angeschlossen, erfolgt die Fehlermeldung 7730. Das Steuergerät ist nicht mehr betriebsbereit. Der Fehler kann in der Regel nicht zurückgesetzt werden. Mit Hilfe des Parameters ist ein temporäres Zurücksetzen des Fehlers möglich. (temporäres Zurücksetzen des Fehlers freigeben/sperrern = 1/0) Der Parameter ist direkt nach seiner Einstellung über die Teaching Box oder die PC-Software wirksam. Ein Aus- und Wiedereinschalten der Spannungsversorgung ist nicht notwendig. Beim Aus- und Wiedereinschalten der Spannungsversorgung wird der Parameter wieder auf „0“ gesetzt (der Fehler kann nicht mehr zurückgesetzt werden), da der eingestellte Wert nicht gespeichert ist.	0 (temporäres Zurücksetzen des Fehlers sperren)

Tab. 9-3: Übersicht der Signalparameter (1)

Parameter	Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Ausgangsbitmuster beim Rücksetzen (siehe auch Abschn. 9.15)		Ausgangsbitmuster beim Rücksetzen über den CLR-Befehl oder den Eingang OUTRESET Das voreingestellte Ausgangsbitmuster wird auch beim Einschalten der Versorgungsspannung ausgegeben. Wird in Einheiten von 32 Bit mit den folgenden Parametern gesetzt (AUS/EIN/HALTEN = 0/1/*)	
	ORST0	Zeichenkette 4	Setzen der Ausgangsbits 0–31
	ORST32 : ORST8016	Zeichenkette 4	Setzen der Ausgangsbits 32–63 : Setzen der Ausgangsbits 8016–8047
Ausgang beim RESET rücksetzen	SLRSTIO	Ganze Zahl 1	Zurücksetzen der Ausgänge, wenn das Programm zurückgesetzt wird (gesperrt/freigegeben = 0/1)
			00000000,00000000, 00000000,00000000
			00000000,00000000, 00000000,00000000 :

Tab. 9-3: Übersicht der Signalparameter (2)

9.4 Betriebsparameter

Diese Parameter dienen der Einstellung von Größen zur Beeinflussung des Steuergerätes und des Teaching-Box-Betriebs usw.

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Summer EIN/AUS	BZR	Ganze Zahl 1	Schaltet den Signalton des Steuergerätes bei einem Fehler EIN oder AUS (AUS/EIN = 0/1)	1 (EIN)
Betriebsrechte zum Rücksetzen	PRSTENA	Ganze Zahl 1	Festlegung, ob zum Rücksetzen des Programms Betriebsrechte erforderlich sind (erforderlich/nicht erforderlich = 0/1) Sind die Betriebsrechte zum Rücksetzen nicht erforderlich, kann das Programm über ein beliebiges externes Gerät zurückgesetzt werden.	0 (erforderlich)
Rücksetzen des Programms über den [MODE]-Schalter des Steuergerätes	MDRST	Ganze Zahl 1	Ein unterbrochenes Programm wird durch Betätigung des [MODE]-Schalters fortgesetzt. (gesperrt/freigegeben = 0/1)	0 (gesperrt)
Anzeige auf dem Steuergerät Dieser Parameter ist ab Software-Version J1 verfügbar	OPDISP	Ganze Zahl 1	Festlegung der LED-Anzeige auf dem Steuergerät bei Betätigung des [MODE]-Schalters 0: Anzeige der Geschwindigkeitsübersteuerung 1: aktuelle Anzeige beibehalten	
Betriebsrechte zur Programmwahl	OPPSL	Ganze Zahl 1	Festlegung der Betriebsrechte zur Programmwahl für den Automatikbetrieb (OP) (extern/OP = 0/1)	1 (OP)
	RMTPSL	Ganze Zahl 1	Festlegung der Betriebsrechte zur Programmwahl für den Automatikbetrieb (Ext.) (extern/OP = 0/1)	0 (Ext.)
Betriebsrechte der Teaching Box zur Einstellung des Übersteuerungswertes	OVRDTB	Ganze Zahl 1	Festlegung, ob zur Änderung des Übersteuerungswertes über die Teaching Box Betriebsrechte erforderlich sind (nicht erforderlich/erforderlich = 0/1)	0 (nicht erforderlich)

Tab. 9-4: Übersicht der Betriebsparameter (1)

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Betriebsart-abhängige Geschwindigkeit	OVRDMD	Ganze Zahl 2	Der Übersteuerungswert wird automatisch beim Betriebsartenwechsel geändert. Über das erste Element ist die Übersteuerung festgelegt, wenn die Betriebsart über die Teaching Box geändert wird. Über das zweite Element ist die Übersteuerung festgelegt, wenn die Betriebsart von „Auto“ auf „Teach“ geändert wird. Ist die Einstellung „0“, wird der aktuelle Wert beibehalten.	0,0
Betriebsgeschwindigkeit für Automatikbetrieb	SPI		Legt die Grundgeschwindigkeit für den Automatikbetrieb fest	
Übersteuerungswert für Automatikbetrieb	EOV		Legt den Übersteuerungswert für den Automatikbetrieb fest (externe Übersteuerung, Programmübersteuerung)	
Betriebsrechte zur Einstellung des Übersteuerungswertes	OVRDENA	Ganze Zahl 1	Festlegung, ob zur Änderung des Übersteuerungswertes Betriebsrechte erforderlich sind. (nicht erforderlich/erforderlich = 0/1) Ist die Einstellung „0“, kann der Übersteuerungswert über alle Eingabemöglichkeiten verändert werden.	0 (nicht erforderlich)
ROM-Modus (siehe auch Abschn. 9.19) Dieser Parameter ist ab Software-Version H7 verfügbar.	ROMDRV	Ganze Zahl 1	Der Zugriff auf Programme kann zwischen RAM und ROM umgeschaltet werden. 0: RAM-Modus (Standard-Modus) 1: ROM-Modus (Spezial-Modus) 2: Highspeed-RAM-Modus (Als Speicher wird ein DRAM verwendet. Die Funktion steht ab Software-Version J1 zur Verfügung.)	0 (RAM-Zugriff)
Kopiert die Daten des RAMs in das ROM (siehe auch Abschn. 9.19) Dieser Parameter ist ab Software-Version H7 verfügbar.	BACKUP	Zeichenkette 1	Kopiert Programme, Parameter, globale Variablen und Fehler-Logfiles vom RAM in das ROM. Dieser Parameter darf nicht geändert werden.	SRAM → FLROM (nicht einstellbar)
Schreibt die Daten des ROMs in das RAM zurück (siehe auch Abschn. 9.19) Dieser Parameter ist ab Software-Version H7 verfügbar.	RESTORE	Zeichenkette 1	Schreibt Programme, Parameter, globale Variablen und Fehler-Logfiles vom ROM in das RAM zurück. Dieser Parameter darf nicht geändert werden.	FLROM → SRAM (nicht einstellbar)

Tab. 9-4: Übersicht der Betriebsparameter (2)

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Wartungsintervalle überwachen Dieser Parameter ist ab Software-Version J1 verfügbar.	MFENA	Ganze Zahl 1	Gibt die Überwachung der Wartungsintervalle frei 1: freigegeben 0: gesperrt Die Funktion steht nur bei den Robotermodellen RV-S/RH-S zur Verfügung. Bei allen anderen Modellen hat die Einstellung dieses Parameters keine Auswirkung.	Für die Robotermodelle RV-S/RH-S: 1, für alle anderen Robotermodelle: 0
Datenerfassung für Wartungsintervall	MFINTVL	Ganze Zahl 2	Festlegung der Stufe und der Abtastrate der Datenerfassung zur Bestimmung des Wartungsintervalls 1. Element: Datenerfassungsstufe 1 (niedrigste) bis 8 (höchste) 2. Element: Abtastrate für die Datenerfassung (Einheit: Stunden)	1 (niedrigste), 6 (Stunden)
Wartungsmeldung	MFEPRO	Ganze Zahl 2	Festlegung der Methode zur Meldung eines abgelaufenen Wartungsintervalls 1. Element: 1 (Warnmeldung) 0 (keine Warnmeldung) 2. Element: 1 (Ausgabe eines speziellen Signals) 0 (keine Ausgabe eines speziellen Signals)	0 (keine Warnmeldung) 0 (keine Ausgabe eines speziellen Signals)
Wartungsdaten zurücksetzen Geben Sie vor dem Einlesen dieser Parameter über die Teaching Box alle Parameternamen ein und starten Sie dann den Lesevorgang.	MFGRST	Ganze Zahl 1	Zurücksetzen aller Wartungsdaten, die auf Schmiermittel bezogen sind Erfolgt für eine Achse die Ausgabe der Fehlermeldung 7530, ist das Schmiermittel zu erneuern. Danach sind die auf Schmiermittel bezogenen Wartungsdaten zurückzusetzen. In der Regel werden die Daten über die Programmier-Software (ab Version E1) zurückgesetzt. Es ist jedoch auch ein Zurücksetzen über die Teaching Box durch Einstellung dieses Parameters möglich.	0: alle Achsen zurücksetzen 1 bis 8: festgelegte Achse zurücksetzen
	MFBRST	Ganze Zahl 1	Zurücksetzen aller Wartungsdaten, die auf die Zahnriemen bezogen sind Erfolgt für eine Achse die Ausgabe der Fehlermeldung 7530, ist der Zahnriemen zu erneuern. Danach sind die auf den Zahnriemen bezogenen Wartungsdaten zurückzusetzen. In der Regel werden die Daten über die Programmier-Software (ab Version E1) zurückgesetzt. Es ist jedoch auch ein Zurücksetzen über die Teaching Box durch Einstellung dieses Parameters möglich.	0: alle Achsen zurücksetzen 1 bis 8: festgelegte Achse zurücksetzen

Tab. 9-4: Übersicht der Betriebsparameter (3)

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Wiederherstellung von Positionsdaten Dieser Parameter ist ab Software-Version J2 verfügbar.	DJNT	Reelle Zahl 8	Festlegung der Nullpunktkorrekturdaten für die Funktion zur Wiederherstellung von Positionsdaten Die Daten dürfen ausschließlich über die Funktion zur Wiederherstellung von Positionsdaten geändert werden. Ein Zugriff auf den Parameter ist nur über ein spezielles Parametermenü in der Programmier-Software möglich.	Modellabhängig
	MEXDTL	Reelle Zahl 6	Festlegung der Standardwerte der Nullpunktkorrekturdaten für die Funktion zur Wiederherstellung von Positionsdaten Die Daten dürfen ausschließlich über die Funktion zur Wiederherstellung von Positionsdaten geändert werden.	(X, Y, Z, A, B, C) = 0.0,0.0,0.0,0.0,0.0,0.0
	MEXDTL1	Reelle Zahl 6	Festlegung der Nullpunktkorrekturdaten des Werkzeugs 1 für die Funktion zur Wiederherstellung von Positionsdaten Die Daten dürfen ausschließlich über die Funktion zur Wiederherstellung von Positionsdaten geändert werden.	(X, Y, Z, A, B, C) = 0.0,0.0,0.0,0.0,0.0,0.0
	MEXDTL2	Reelle Zahl 6	Festlegung der Nullpunktkorrekturdaten des Werkzeugs 2 für die Funktion zur Wiederherstellung von Positionsdaten Die Daten dürfen ausschließlich über die Funktion zur Wiederherstellung von Positionsdaten geändert werden.	(X, Y, Z, A, B, C) = 0.0,0.0,0.0,0.0,0.0,0.0
	MEXDTL3	Reelle Zahl 6	Festlegung der Nullpunktkorrekturdaten des Werkzeugs 3 für die Funktion zur Wiederherstellung von Positionsdaten Die Daten dürfen ausschließlich über die Funktion zur Wiederherstellung von Positionsdaten geändert werden.	(X, Y, Z, A, B, C) = 0.0,0.0,0.0,0.0,0.0,0.0
	MEXDTL4	Reelle Zahl 6	Festlegung der Nullpunktkorrekturdaten des Werkzeugs 4 für die Funktion zur Wiederherstellung von Positionsdaten Die Daten dürfen ausschließlich über die Funktion zur Wiederherstellung von Positionsdaten geändert werden.	(X, Y, Z, A, B, C) = 0.0,0.0,0.0,0.0,0.0,0.0
	MEXDBS	Reelle Zahl 6	Festlegung der Nullpunktkorrekturdaten der Basis für die Funktion zur Wiederherstellung von Positionsdaten Die Daten dürfen ausschließlich über die Funktion zur Wiederherstellung von Positionsdaten geändert werden.	(X, Y, Z, A, B, C) = 0.0,0.0,0.0,0.0,0.0,0.0

Tab. 9-4: Übersicht der Betriebsparameter (4)

9.5 Befehlsparameter

Diese Parameter dienen der Einstellung von Größen zur Beeinflussung der Programmausführung.

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Anzahl der Programmplätze	TASKMAX	Ganze Zahl 1	Festlegung der maximalen Anzahl der Programmplätze für eine parallele Ausführung (Multitasking)	8
Programmplatzliste	SLT : SLT32	Zeichenkette 4	<p>Festlegung der Einstellungen (Programmname, Ausführungsformat, Startbedingung und Priorität) jedes Programmplatzes bei der Initialisierung</p> <p>Programmname: Einstellwert Bei Verwendung von Buchstaben sollten nur Großbuchstaben verwendet werden. Kleinbuchstaben werden nicht erkannt.</p> <p>Ausführungsformat: kontinuierlich/zyklisch = REP/CYC REP: Das Programm wird wiederholt ausgeführt. CYC: Das Programm endet nach Ausführung eines Zyklus. (Das Programm endet nicht bei Ausführung einer Endlosschleife über die GOTO-Anweisung.)</p> <p>Startbedingung: Normal/Fehler/Ständig = START/ERROR/ALWAYS START: Die Programmausführung erfolgt über die START-Taste des Steuergerätes oder über das Startsignal ALWAYS: Das Programm wird nach dem Einschalten der Spannungsversorgung ausgeführt. Das Programm hat keinen Einfluss auf die Einschalt routine. Zur Editierung eines Programmes mit der Startbedingung „ALWAYS“ muss zuerst das ALWAYS-Attribut zurückgesetzt werden. Ändern Sie die Startbedingung „ALWAYS“ auf „START“ und schalten Sie anschließend die Spannungsversorgung aus und wieder ein, um die ständige Ausführung des Programms zu unterbrechen. ERROR: Das Programm wird nach Auftreten eines Fehlers ausgeführt. Das Programm hat keinen Einfluss auf die Einschalt routine.</p> <p>In Programmen mit der Startbedingung ALWAYS oder ERROR können folgende Befehle nicht ausgeführt werden: MOV, MVS, MVR, MVR2, MVR3, MVC, MVA, GETM, RELM, JRC</p> <p>Priorität: 1 bis 31 Die Einstellung legt die Anzahl der auszuführenden Zeilen für einen Durchgang fest. Die Funktion entspricht der PRIORITY-Anweisung. Ist die Priorität z. B. für SLT1 auf „1“, für SLT 2 auf „2“ gesetzt, werden nach Abarbeitung von 1 Zeile des Programms in SLT1 2 Zeilen des Programms in SLT2 ausgeführt. Es werden also mehr Programmteile in SLT 2 ausgeführt, d. h. deren Priorität ist höher.</p>	"" , REP, START, 1

Tab. 9-5: Übersicht der Befehlsparameter (1)

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Programmwahl speichern	SLOTON	Ganze Zahl 1	<p>Dieser Parameter legt fest, ob der Programmname bei Auswahl des Programms im Programmplatzparameter SLT1 gespeichert werden soll und ob das Programm nach Beendigung des Zyklus weiterhin ausgewählt bleibt</p> <ul style="list-style-type: none"> ● Speicherung des Programmnamens bei Auswahl des Programms (Bit 0, Speicherung aktiviert/deaktiviert = 1/0) Aktiviert: Bei Auswahl des Programms wird der Programmname im Programmplatzparameter SLT1 gespeichert. Nach Einschalten der Spannungsversorgung wird das im Programmplatzparameter SLT1 gespeicherte Programm ausgewählt. Deaktiviert: Bei Auswahl des Programms wird der Programmname nicht im Programmplatzparameter SLT1 gespeichert. Nach Einschalten der Spannungsversorgung wird das im Programmplatzparameter SLT1 gespeicherte Programm ausgewählt. ● Programmaufrechterhaltung nach Beendigung des Zyklusbetriebs (Bit 1, Programm aufrechterhalten/nicht aufrechterhalten = 1/0) Aufrechterhalten: Das Programm bleibt auch weiterhin ausgewählt, auch wenn im Zyklusbetrieb ein Zyklus beendet wurde. Der Parameter wird nicht auf „P.0000“ gesetzt. Nicht aufrechterhalten: Die Programmauswahl wird zurückgesetzt, wenn das Programm zurückgesetzt wird. Der Parameter wird auf „P.0000“ gesetzt. <p>Einstellwerte und Funktionen: 0: Speicherung deaktiviert/ Programm nicht aufrechterhalten 1: Speicherung aktiviert/ Programm nicht aufrechterhalten (Standardwert) 2: Speicherung deaktiviert/ Programm aufrechterhalten 3: Speicherung aktiviert/ Programm aufrechterhalten</p>	1 (aktiviert)
X□□- und SERVO-Befehle in Programmplätzen mit der Startbedingung ALWAYS zulassen (siehe auch Abschn. 9.12)	ALWENA	Ganze Zahl 1	<p>Die Befehle XRUN, XLOAD, XSTP, XRST, SERVO und RESET ERR werden für Programmplätze freigegeben, deren Startbedingung in den Programmplatzparametern SLT□ auf „ALWAYS“ gesetzt ist. Ab Software-Version J1 kann der Befehl XRUN direkt über die Teaching Box oder die Programmier-Software ausgeführt werden. freigeben/sperrern = 7/0</p>	0 (gesperrt)

Tab. 9-5: Übersicht der Befehlsparameter (2)

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Benutzerbasisprogramm (siehe auch Abschn. 5.1.10)	PRGUSR	Zeichenkette 1	Ein Benutzerbasisprogramm wird zur Deklaration von benutzerdefinierten externen Variablen verwendet. Die Variablen werden über die Befehle DEF oder DIM deklariert. Sollen im Benutzerbasisprogramm Feldvariablen erstellt und als externe Variablen eingesetzt werden, so ist eine zweite Deklaration über die DIM-Anweisung in dem Programm erforderlich, in dem sie verwendet werden. Einzelne Variablen erfordern keine erneute Deklaration.	—
Programm fortsetzen	CTN	Ganze Zahl 1	<p>Für den Programmplatz 1 wird nach dem Ausschalten der Spannungsversorgung die aktuelle Position innerhalb der Anwendung gespeichert. Nach dem nächsten Einschalten der Spannungsversorgung startet die Anwendung von dieser gespeicherten Position. Gespeichert wird die Übersteuerung, die Programmzeile, Programmvariable und der Zustand des Ausgangssignals.</p> <p>(freigegeben/gesperrt = 1/0)</p> <ul style="list-style-type: none"> Die Funktion CTN greift beim Ausschalten der Spannungsversorgung auf den Programmspeicherbereich zurück. Die Funktion benötigt 100 kB freien Speicherplatz. Bei CTN = 1 kann die Position über die Software gespeichert und die Parameter verändert werden. Nach dem Aus- und Wiedereinschalten der Spannungsversorgung werden die geänderten Parameter über die Software eingelesen und gespeichert. <p>Beachten Sie die folgenden Hinweise bei Verwendung der Funktion. Bei Nichtbeachtung kann das Programm zerstört werden.</p> <ul style="list-style-type: none"> Sichern Sie alle Programme auf dem PC und löschen Sie alle Programme im Steuergerät. Setzen Sie Parameter CNT auf „1“ und schalten Sie die Spannungsversorgung aus und wieder ein. Laden Sie nur die notwendigen Programme in das Steuergerät. Bei zu geringem Speicherplatz kann die Funktion nicht verwendet werden. Setzen Sie in diesem Fall die Parameter wieder zurück. 	0 (gesperrt)

Tab. 9-5: Übersicht der Befehlsparameter (3)

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
			<ul style="list-style-type: none"> ● Ändern Sie keine Parameter, während das Programm läuft, da das Programm zerstört werden kann. Ist CTN = 1 gesetzt, wird die freie Speicherkapazität des Standard-speichers von 210 kB auf 110 kB herabgesetzt. Wird der Programmspeicher in eine Anzahl von Adressen konvertiert, wird die Kapazität um 1100 Adressen herabgesetzt. Bei einer Konvertierung in Schritte wird die Kapazität um 2200 Schritte herabgesetzt. Mit den Standardeinstellungen reduziert sich der Speicherplatz um 56 %. Adressenanzahl = 2500 → 1400 Anzahl der Schritte = 5000 → 2800 ● Bei Robotern, die über Achsen ohne Bremse verfügen (z. B. J4- und J6-Achse beim RV-1A/2AJ oder J4-Achse beim RH-5AH), kann es zu einem Heruntersinken des Armes aufgrund der Erdanziehungskraft oder zu Rotationen kommen, wenn die Spannungsversorgung ausgeschaltet wird. Treffen Sie daher geeignete Vorsichtsmaßnahmen (z. B. Unterstützen des Roboterarms), wenn Sie die Funktion verwenden. ● Nur das Programm im Programmplatz 1 kann aus dem „Standby-Modus“ gestartet werden. Die Programme aus den anderen Programmplätzen werden aus dem „Reset-Modus“ gestartet. ● Die Parameter SLT□, SLOTON und TASKMAX können nach Freigabe der Funktion nicht mehr verändert werden. Ändern Sie die Parameter vor einer Freigabe der Funktion. ● Änderungen der Programmplatzparameter SLT□ nach Freigabe der Funktion werden nicht übernommen. Deaktivieren Sie die Funktion, schalten Sie die Spannungsversorgung aus und wieder ein und ändern Sie dann die Einstellwerte der Parameter. 	

Tab. 9-5: Übersicht der Befehlsparameter (4)

Parameter	Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung	
JRC-Befehl (Multirotationsfunktion der Achsen)		Festlegung des Ausführungsstatus des JRC-Befehls		
JRCXEXE	Ganze Zahl 1	Freigabe des JRC-Befehls (freigegeben/gesperrt = 1/0)	0 (Ausführung gesperrt)	
JRCQTT	Reelle Zahl 8	Im Parameter JRCQTT werden die Werte festgelegt, um die benutzerdefinierte Achsen verschoben werden. Die Achsen werden in der Reihenfolge J1, J2, J3 bis J8 angegeben. J7 und J8 sind zusätzliche Roboterachsen. Die Einheit der Werte aus JRCQTT ist im Parameter AXUNT festgelegt.	JRC freigegeben: 0,0,0,0,0,360,0,0 oder 0,0,0,360,0,0,0,0 JRC gesperrt: 0,0,0,0,0,0,0,0	
JRCORG	Reelle Zahl 8	Festlegung der Grundposition für JRC = 0 Diese Einstellung ist nur bei benutzerdefinierten Achsen möglich. Die Einheit der Werte aus JRCORG ist im Parameter AXUNT festgelegt.	0,0,0,0,0,0,0,0	
Einstellung zusätzlicher Achsen	AXUNT	Reelle Zahl 16	Festlegung der Einheit (Winkel (°)/Länge (mm) = 0/1)	0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
Benutzerdefinierter Fehler	UER1 : UER20	Ganze Zahl 1, Zeichenkette 3	Legt eine Fehlermeldung, die Fehlerursache und die Fehlerbehebung für einen benutzerdefinierten Fehler fest. Es können maximal 20 benutzerdefinierte Fehler festgelegt werden. Das erste Element legt eine Fehlernummer zwischen 9000 und 9299 fest. Der Standardwert 9900 kann nicht festgelegt werden. Im zweiten Element wird die Fehlermeldung definiert. Im dritten Element wird die Fehlerursache definiert. Im vierten Element wird die Fehlerbehebung definiert. Enthält die Fehlermeldung ein Leerzeichen, ist die ganze Meldung in Anführungszeichen zu setzen (""). Beispiel: 9000,"Zeitüberschreitung", "Kein Signal", "Taste betätigen"	9900,"Fehlermeldung", "Fehlerursache", "Fehlerbehebung"
Einheit für drehende Positionselemente	PRGMDEG	Ganze Zahl 1	Festlegung der Einheit für drehende Elemente von Positionsdaten 0: RAD 1: Grad Beispiel: M1 = P1.A Die Einheit ist definiert. Für den direkten Zugriff auf Positionskomponenten ist die Standardeinheit Radiant. Die Standardeinheit für Positionskonstanten (P1 = (100,0,300,0,180,0,180)(7,0)) ist Grad. Der Parameter ist unwirksam.	0 (RAD)

Tab. 9-5: Übersicht der Befehlsparameter (5)

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Verzögerungszeit für die Befehle GC, GO und Bewegungsbefehle Dieser Parameter ist nur bei der MOVEMASTER-COMMAND-Programmierungsmethode einsetzbar. Dieser Parameter ist ab Software-Version H7 verfügbar.	HNDDLY	Ganze Zahl 1	Die Verzögerungszeit zum Öffnen und Schließen der Greifhand wird in der MOVEMASTER-COMMAND-Programmierungsmethode über den GP-Befehl festgelegt (Standardwert: 0,3 s) -1: Bei der motorbetriebenen Greifhand ist die durch den GP-Befehl festgelegte Verzögerungszeit wirksam, wenn sich der Handstatus ändert. Bei der pneumatischen Greifhand wird die im GP-Befehl festgelegte Verzögerungszeit in diesem Parameter gespeichert, wenn die Greifhand sich öffnet oder schließt, auch wenn der Handstatus sich nicht ändert. 0: keine Verzögerung Wert: Bei einer Änderung des Handstatus ist der eingestellte Wert in ms wirksam. Die Einheit der Verzögerungszeit ist beim Befehl GP 1/10 s und beim Parameter HANDDLY 1/1000 s (= ms).	-1
Programmierungsmethode	RLNG	Ganze Zahl 1	Auswahl der Programmierungsmethode 1: MELFA-BASIC IV 0: MOVEMASTER COMMAND Die Funktion ist nur bei bestimmten Robotermodellen verfügbar (z. B. RV-1A). Prüfen Sie vor Einstellung des Parameters im Technischen Handbuch, ob der von Ihnen verwendete Roboter über diese Funktion verfügt.	1
Landessprache	LNG	Zeichenkette 1	Auswahl der angezeigten Landessprache JPN = Japanisch ENG = Englisch Der Parameter beeinflusst folgende Funktionen: ● Sprache auf der LCD-Anzeige der Teaching Box ● Fehlermeldungen, die über Datenkommunikation abgelesen werden (Standardschnittstelle RS232C, zusätzliche serielle Schnittstelle, Ethernet-Schnittstelle)	Die japanische Version wird durch die Zeichenkette "JPN", die englische durch die Zeichenkette "ENG" angezeigt.
Erweiterung von externen Variablen	PRGGBL	—	Zur Nutzung des erweiterten Bereichs ist Parameter PRGGBL von „0 (Standard/Grundeinstellung)“ auf „1 (erweiterter Bereich)“ zu ändern und die Spannungsversorgung aus- und wieder einzuschalten. Wurde für eine programmexterne und eine benutzerdefinierte externe Variable zweimal derselbe Name vergeben, erfolgt beim Einschalten der Spannungsversorgung eine Fehlermeldung und eine Erweiterung des Bereichs ist nicht möglich. Ändern Sie in diesem Fall den Namen der benutzerdefinierten Variablen.	0

Tab. 9-5: Übersicht der Befehlsparameter (6)

9.6 Kommunikationsparameter

Diese Parameter dienen der Einstellung von Größen zur Beeinflussung der Kommunikation.

Parameter	Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
Kommunikationseinstellungen (siehe auch Abschn. 9.16)		Die Kommunikationseinstellungen beziehen sich auf die RS232C-Schnittstelle vorne am Steuergerät. Da die Schnittstelle in Kombination mit der Roboter-Programmier-Software verwendet wird, entfällt hier in der Regel eine Einstellung. Verwenden Sie zum Anschluss optischer Sensoren usw. die optionale serielle Erweiterungsschnittstelle.	
COMDEV	Zeichenkette 8	<p>Der Parameter dient bei Verwendung des OPEN-Befehls in MELFA-BASIC IV der Zuweisung der Kommunikationsleitungen an die Schnittstellen COM1 und COM2. Der Parameter muss für eine Datenverbindung durch den OPEN-Befehl gesetzt werden.</p> <p>Der Parameter legt also fest, welches Gerät über welche Kommunikationleitung COMn ($1 \leq n \leq 8$) angesprochen wird. Die einzelnen Elemente sind von links beginnend wie folgt angeordnet: COM1, COM2, ..., COM8</p> <p>Soll eine serielle Erweiterungsschnittstelle als COM verwendet werden, ist der Parameter wie folgt einzustellen:</p> <ul style="list-style-type: none"> ● bei Montage von CH1 in Steckplatz 1: "OPT11", ● bei Montage von CH2 (RS232) in Steckplatz 1: "OPT12", ● bei Montage von CH2 (RS422) in Steckplatz 1: "OPT13", ● bei Montage von CH1 in Steckplatz 2: "OPT21", ● bei Montage von CH2 (RS232) in Steckplatz 2: "OPT22", ● bei Montage von CH2 (RS422) in Steckplatz 2: "OPT23". <p>Wird z. B. CH1 in Steckplatz 1 montiert und COM2 zugewiesen, ist der Parameter auf "RS232", "OPT11", , , einzustellen.</p> <p>Wird z. B. CH2 in Steckplatz 2 montiert und COM3 zugewiesen, ist der Parameter auf "RS232", , "OPT22", , einzustellen.</p>	"RS232", , , , , , ,

Tab. 9-6: Übersicht der Kommunikationsparameter (1)

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
			<p>Soll die Ethernet-Schnittstelle in Steckplatz 1 montiert und als COM verwendet werden, ist der Parameter wie folgt einzustellen:</p> <ul style="list-style-type: none"> ● NETPORT1: "OPT11", ● NETPORT2: "OPT12", ● NETPORT3: "OPT13", ● NETPORT4: "OPT14", ● NETPORT5: "OPT15", ● NETPORT6: "OPT16", ● NETPORT7: "OPT17", ● NETPORT8: "OPT18", ● NETPORT9: "OPT19". <p>COM1 ist bereits die Standard-schnittstelle RS232 an der Vorderseite des Steuergerätes zugewiesen. Bei Installation optionaler Zusatz-schnittstellen gelten folgende Zuordnungsbedingungen:</p> <p>Steckplatz 1: Zusatzachse, serielle Erweiterungsschnittstelle, Ethernet Steckplatz 2: Zusatzachse, serielle Erweiterungsschnittstelle, CC-Link Steckplatz 3: Zusatzachse</p> <p>Schnittstellenkarten für Zusatzachsen können nur in den Steuergeräten CR1, CR2A und CR2B verwendet werden.</p> <p>Eine detaillierte Beschreibung der optionalen Schnittstellenkarten finden Sie im jeweiligen Handbuch der Schnittstelle.</p>	
	CBAU232	Ganze Zahl 1	<p>Festlegung der Übertragungsrates (9600, 19200)</p> <p>Informationen zu den optionalen zusätzlichen seriellen Schnittstellenkarten finden Sie in den Handbüchern der Schnittstellen.</p>	9600
	CLEN232	Ganze Zahl 1	<p>Festlegung der Datenlänge</p> <p>Informationen zu den optionalen zusätzlichen seriellen Schnittstellenkarten finden Sie in den Handbüchern der Schnittstellen.</p>	0
	CPRTY232	Ganze Zahl 1	<p>Festlegung der Parität (0: keine, 1: ungerade, 2: gerade)</p> <p>Informationen zu den optionalen zusätzlichen seriellen Schnittstellenkarten finden Sie in den Handbüchern der Schnittstellen.</p>	2 (gerade)
	CSTOP232	Ganze Zahl 1	<p>Festlegung des Stoppbits (1, 2)</p> <p>Informationen zu den optionalen zusätzlichen seriellen Schnittstellenkarten finden Sie in den Handbüchern der Schnittstellen.</p>	2
	CTERM232	Ganze Zahl 1	<p>Festlegung des Endezeichens (0: CR, 1: CR + LF)</p> <p>Informationen zu den optionalen zusätzlichen seriellen Schnittstellenkarten finden Sie in den Handbüchern der Schnittstellen.</p>	0 (CR)

Tab. 9-6: Übersicht der Kommunikationsparameter (2)

Parameter		Anzahl der Felder/ Zeichen	Beschreibung	Werkseinstellung
	CPRC232	Ganze Zahl 1	<p>Kommunikationsart (Protokoll)</p> <p>0: für PC-Support-Software oder Roboter-Software COSIROP (kein Protokoll) Wird mit dieser Einstellung eine Datenübertragung ausgeführt (Befehle OPEN, PRINT und INPUT), muss das externe Gerät die Zeichen „PRN“ an den Anfang der zu übertragenen Daten setzen.</p> <p>1: für PC-Support-Software oder Roboter-Software COSIROP (Protokoll) Die Einstellung muss auch PC-seitig erfolgen.</p> <p>2: für Datenverbindung mit dem Roboterprogramm (z. B. bei Kommunikation mit optischen Sensoren) Bei dieser Einstellung ist keine Verbindung mit der Roboter-Software möglich. Verwenden Sie eine optionale zusätzliche serielle Schnittstelle. Informationen zu den optionalen zusätzlichen seriellen Schnittstellenkarten finden Sie in den Handbüchern der Schnittstellen.</p>	0

Tab. 9-6: Übersicht der Kommunikationsparameter (3)

9.7 Standard-Werkzeugkoordinaten

Die Einstellung der Werkzeugdaten ist bei einer Verschiebung des Werkzeugmittelpunkts (Tool Centre Point) erforderlich. Dies ist z. B. nach der Montage eines Handgreifers der Fall. Bei Werkzeuggestellung sind die Werkzeugkoordinaten auf Null gesetzt, d. h. der Werkzeugmittelpunkt liegt mittig im Handflansch.

Die Einstellung kann auf drei Arten erfolgen:

- Einstellung der Werkzeugdaten über den Parameter MEXTL
- Einstellung der Werkzeugdaten im Roboterprogramm über den Befehl TOOL
- Einstellung der Werkzeugnummer über die Variable M_TOOL (ab Software-Version J1)
Die Werkzeugdaten sind dabei in den Parametern MEXTL1 bis 4 festgelegt (siehe auch Abschn. 7.2.44).

Auf den folgenden Seiten finden Sie Einstellbeispiele für die verschiedenen Robotermodelle.

9.7.1 Aufbau der Werkzeugdaten

Die Werkzeugdaten sind wie folgt aufgebaut: X, Y, Z, A, B, C.

X-, Y-, Z-Achse: Verschiebung des Werkzeugmittelpunkts bezogen auf den Handflansch im Werkzeugkoordinatensystem

A-Achse: Drehung um die X-Achse im Werkzeugkoordinatensystem

B-Achse: Drehung um die Y-Achse im Werkzeugkoordinatensystem

C-Achse: Drehung um die Z-Achse im Werkzeugkoordinatensystem

Beispiele ▾

Die folgenden Beispiele zeigen die verschiedenen Einstellmethoden an unterschiedlichen Robotermodellen.

5-achsiger vertikaler Knickarmroboter

- Einstellung über Parameter
Parameter MEXTL: 0,0,95,0,0,0
- Einstellung über TOOL-Befehl
10 TOOL (0,0,95,0,0,0)

Beim 5-achsigen Roboter ist aufgrund des Bewegungsbereiches nur die Einstellung der Z-Achsen-Komponente wirksam. Die Einstellung anderer Achsen wird ignoriert.

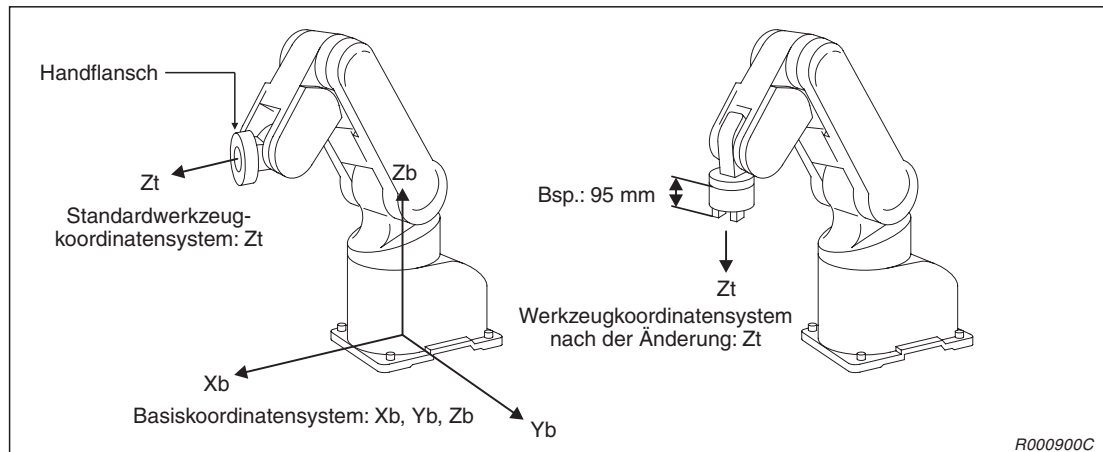


Abb. 9-1: Werkzeugkoordinaten beim 5-achsigen Knickarmroboter

6-achsiger vertikaler Knickarmroboter

- Einstellung über Parameter
Parameter MEXTL: 0,0,95,0,0,0
- Einstellung über TOOL-Befehl
10 TOOL (0,0,95,0,0,0)

Beim 6-achsigen Roboter sind innerhalb des Bewegungsbereiches verschiedene Stellungen möglich.

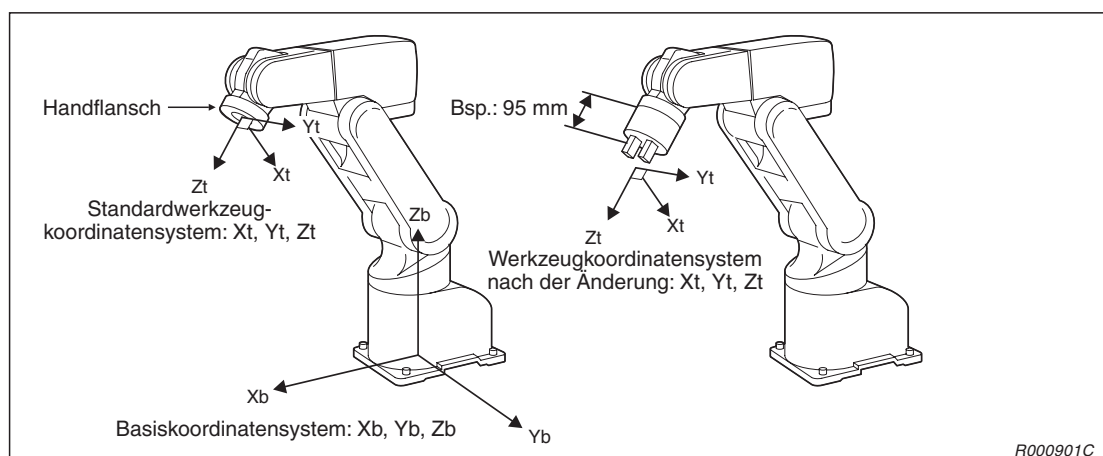


Abb. 9-2: Werkzeugkoordinaten beim 6-achsigen Knickarmroboter

4-achsiger SCARA-Roboter

- Einstellung über Parameter
Parameter MEXTL: 0,0,-10,0,0,0
- Einstellung über TOOL-Befehl
10 TOOL (0,0,-10,0,0,0)

Beim 4-achsigen SCARA-Roboter erfolgt die Einstellung des Werkzeugmittelpunktes durch Parallelverschiebung. Die Einstellung der Orientierung des Werkzeugkoordinatensystems unterscheidet sich von der des vertikalen Knickarmroboters.

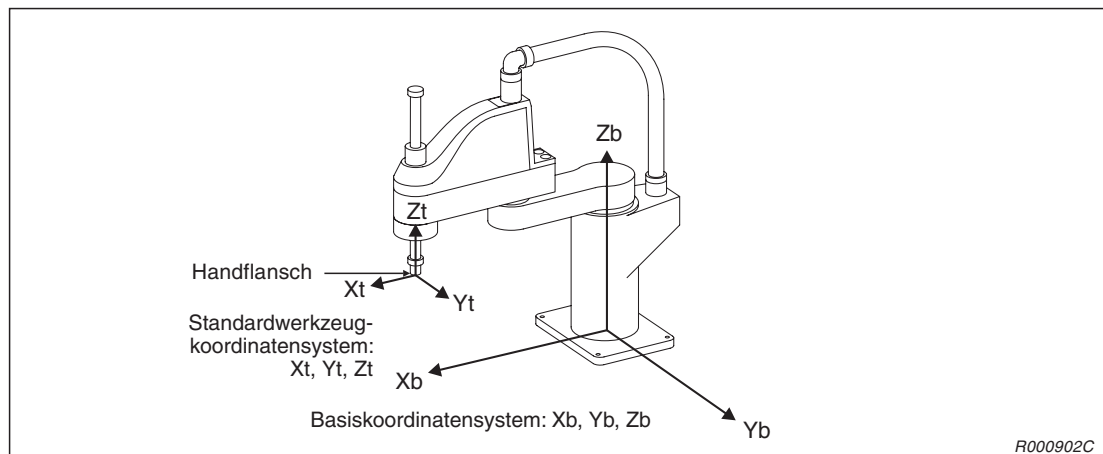


Abb. 9-3: Werkzeugkoordinaten beim 4-achsigen SCARA-Roboter

Doppelarm-SCARA-Roboter

- Einstellung über Parameter
Parameter MEXTL: 0,0,-10,0,0,0
- Einstellung über TOOL-Befehl
10 TOOL (0,0,-10,0,0,0)

Das Werkzeugkoordinatensystem des Doppelarm-SCARA-Roboters entspricht dem des 4-achsigen SCARA-Roboters. Die Einstellung der Orientierung des Werkzeugkoordinatensystems unterscheidet sich von der des vertikalen Knickarmroboters.

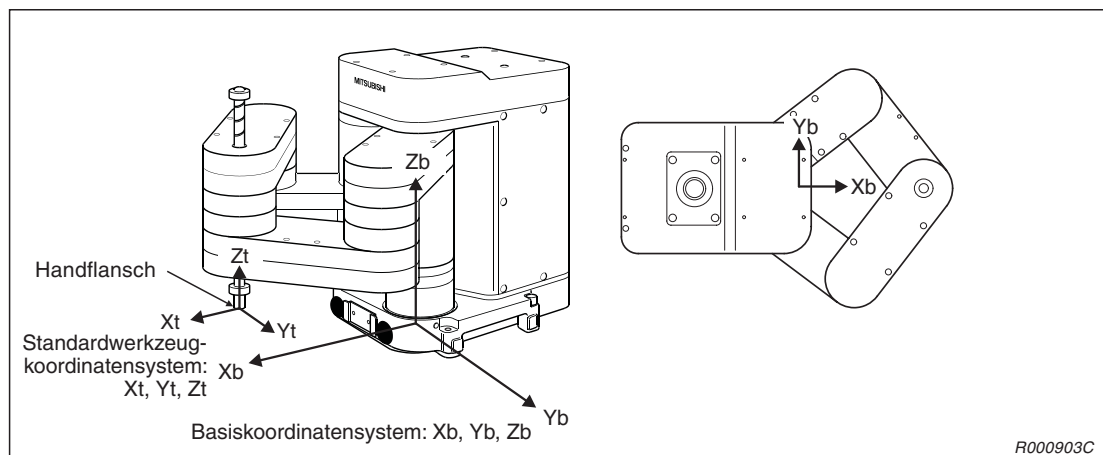


Abb. 9-4: Werkzeugkoordinaten beim Doppelarm-SCARA-Roboter



Die für die Werkzeug-Konvertierungsdaten relevanten Achsen sind vom Robotermodell abhängig. Folgende Tabelle zeigt den Zusammenhang:

Roboter	Anzahl der Achsen	Relevante Achsen					
		X	Y	Z	A	B	C
RP-1AH/3AH/5AH	4						
RH-5AH/10AH/15AH RH-6SH/12SH	4		✓			0	✓
RV-2AJ, RV-3AJ, RV-5AJ, RV-4AJL, RV-3SJB	5		✓			0	✓
RV-1A, RV-2A, RV-4A, RV-3AL, RV-3SB, RV-6S/6SL/12S/12SL	6				✓		

Tab. 9-7: Werkzeug-Konvertierungsdaten und relevante Achsen

- ✓ : für die Werkzeug-Konvertierungsdaten gültige Achse
- 0 : der Wert ist auf „0“ festgelegt (Eine andere Einstellung kann sich nachteilig auf den Betrieb auswirken.)

9.8 Standard-Basiskoordinaten

Soll der Nullpunkt des Roboters nicht der Mittelpunkt der J1-Achse sein, kann das Standard-Basiskoordinatensystem verschoben werden. Die Koordinaten geteachter Positionen beziehen sich auf das verschobene Basiskoordinatensystem. Bei Werkseinstellung ist die Position des Basiskoordinatensystem auf Null gesetzt, d. h. der Robotermittelpunkt liegt im Nullpunkt des Weltkoordinatensystems.

Die Einstellung kann auf zwei Arten erfolgen:

- Verschiebung des Basiskoordinatensystems über den Parameter MEXBS
- Verschiebung des Basiskoordinatensystems im Roboterprogramm über den Befehl BASE

9.8.1 Aufbau der Basiskoordinatendaten

Die Basiskoordinatendaten sind wie folgt aufgebaut: X, Y, Z, A, B, C.

X-, Y-, Z-Achse: Verschiebung des Basiskoordinatensystems bezogen auf das Weltkoordinatensystem

A-Achse: Drehung um die X-Achse im Basiskoordinatensystem

B-Achse: Drehung um die Y-Achse im Basiskoordinatensystem

C-Achse: Drehung um die Z-Achse im Basiskoordinatensystem

Beispiel ▾

- Einstellung über Parameter
Parameter MEXBS: 100,150,0,0,0,-30
- Einstellung über TOOL-Befehl
10 TOOL (100,150,0,0,0,-30)

Eine Änderung des Basiskoordinatensystems ist im Normalfall nicht nötig. Beachten Sie, dass die Änderung des Basiskoordinatensystems über den BASE-Befehl innerhalb eines Programms zu unvorhersehbaren Bewegungen führen kann.

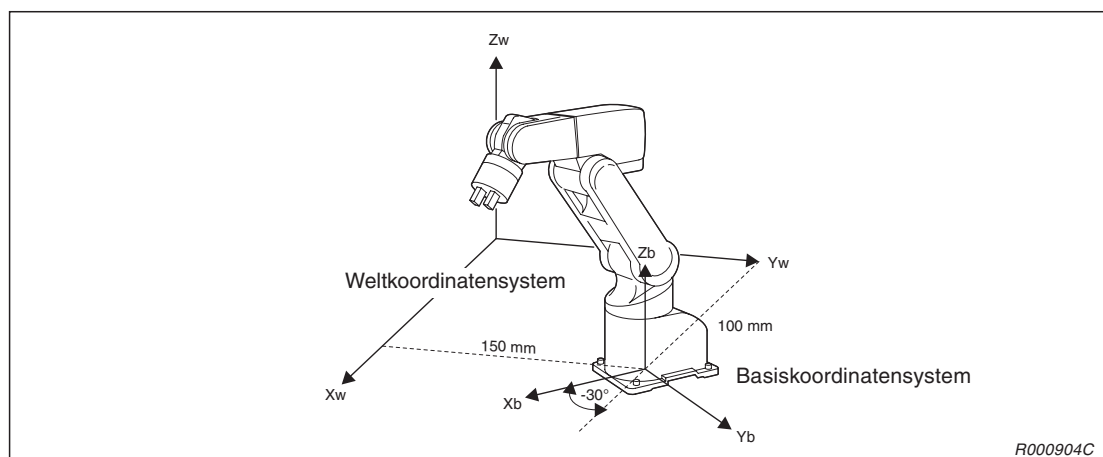


Abb. 9-5: Basiskoordinatensystem



Die für die Standard-Basis-Konvertierungsdaten relevanten Achsen sind vom Robotermodell abhängig. Folgende Tabelle zeigt den Zusammenhang:

Roboter	Anzahl der Achsen	Relevante Achsen					
		X	Y	Z	A	B	C
RP-1AH/3AH/5AH	4						
RH-5AH/10AH/15AH RH-6SH/12SH	4		✓		0		✓
RV-2AJ, RV-3AJ, RV-5AJ, RV-4AJL, RV-3SJB	5		✓		0		✓
RV-1A, RV-2A, RV-4A, RV-3AL, RV-3SB, RV-6S/6SL/12S/12SL	6				✓		

Tab. 9-8: Werkzeug-Konvertierungsdaten und relevante Achsen

- ✓ : für die Standard-Basis-Konvertierungsdaten gültige Achse
- 0 : der Wert ist auf „0“ festgelegt (Eine andere Einstellung kann sich nachteilig auf den Betrieb auswirken.)

9.9 Benutzerdefinierter Bereich

Oft arbeitet ein Roboter mit anderen Maschinen so zusammen, dass die Arbeitsbereiche geteilt werden müssen. In solchen Fällen ist es wünschenswert, dass eine Maschine der anderen mitteilt, wann sie sich innerhalb des gemeinsamen Arbeitsbereichs aufhält. Der Roboter kann so eingestellt werden, dass bei Eindringen in einen über Parameter definierten Bereich die Ausgabe eines Signals erfolgt.

In folgendem Beispiel soll die Parametereinstellung so erfolgen, dass bei Eindringen des Roboters in den Bereich ① das Signal 10 und bei Eindringen in den Bereich ② das Signal 11 ausgegeben wird.

Das gleiche Ergebnis kann auch über ein Programm unter Verwendung der Variablen M_UAR erzielt werden (siehe auch Abschn. 7.2.45).

Beispiel ▾

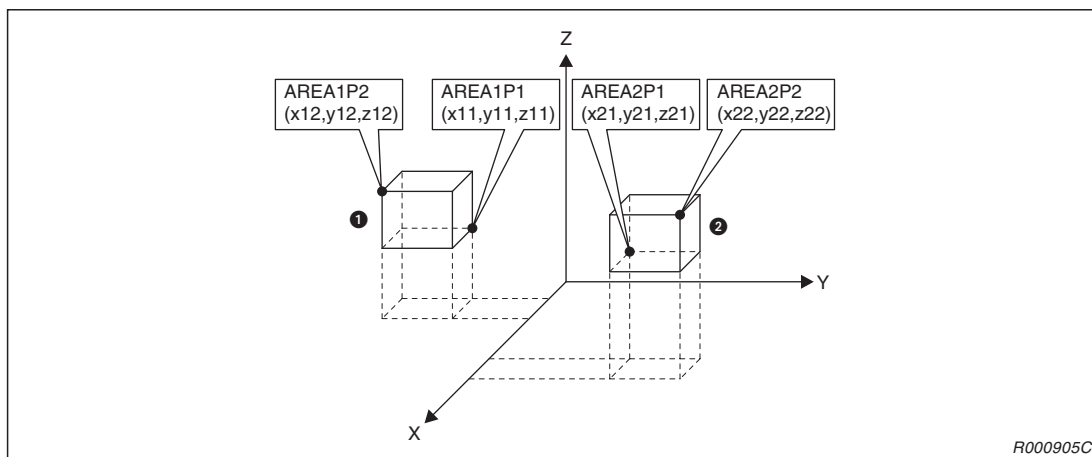


Abb. 9-6: Benutzerdefinierter Bereich

Parameter	Bedeutung	Einstellung
AREA1P1	Positionsdaten des 1. Punktes: X, Y, Z, A, B, C, L1, L2	x11, y11, z11, -360, -360, -360, 0, 0
AREA1P2	Positionsdaten des 2. Punktes: X, Y, Z, A, B, C, L1, L2	x12, y12, z12, 360, 360, 360, 0, 0
AREA1ME	Mechanismusnummer: standardmäßig 1	1
AREA1AT	Gesperrt/Signalausgabe/Fehler = 0/1/2	1
AREA2P1	Positionsdaten des 1. Punktes: X, Y, Z, A, B, C, L1, L2	x11, y11, z11, -360, -360, -360, 0, 0
AREA2P2	Positionsdaten des 2. Punktes: X, Y, Z, A, B, C, L1, L2	x12, y12, z12, 360, 360, 360, 0, 0
AREA2ME	Mechanismusnummer: standardmäßig 1	1
AREA2AT	Gesperrt/Signalausgabe/Fehler = 0/1/2	1
USRAREA	Ausgangssignal: Startnummer, Endnummer	10, 11 Befindet sich der Roboter im Bereich AREA1, wird das Signal 10, befindet sich der Roboter im Bereich AREA2, das Signal 11 ausgegeben. Verwenden Sie nur einen Bereich, setzen Sie die Werte: 10, 10

Tab. 9-9: Parametereinstellungen



HINWEISE

Geben Sie die x-, y- und z-Werte der Koordinaten x11 bis z22 ein.

Im obigen Beispiel werden die Stellungsdaten A, B und C ignoriert, d. h. es wird von der Stellung unabhängig ein Signal ausgegeben. Setzen Sie zur Festlegung der Stellungsdaten die Werte von AREA*P1 bis AREA*P2. (Beispiel: Sind die Werte $-10 \rightarrow +30$, wird ein Wert von -10 bis $+30$ als innerhalb des Bereichs gewertet; sind die Werte $+30 \rightarrow -10$, wird ein Wert von $+30$ bis -10 als innerhalb des Bereichs gewertet.)

Für die Stellungsdaten nicht existierender Achsen (z. B. A- oder B-Achse beim SCARA-Roboter) muss AREA*P1 auf „ -360° “ und AREA*P2 auf „ $+360^\circ$ “ gesetzt werden.

Der Parameter AREA*ME legt fest, für welchen Mechanismus die Bereichsprüfung durchgeführt werden soll. Standardmäßig, d. h. bei Anschluss einer Einheit, ist der Wert auf „1“ gesetzt.

Über den Parameter AREA*AT wird die Art der Prüfmethode festgelegt.

0 = keine Prüfung

1 = Ausgabe eines Signals bei Eindringen in den Bereich

2 = Generiert einen Fehler bei Eindringen in den Bereich

Bei der Einstellung „2“ werden die Stellungsdaten L1 und L2 ignoriert.

Definieren Sie bei Verwendung von Zusatzachsen jeweils die Bereiche für L1 und L2.

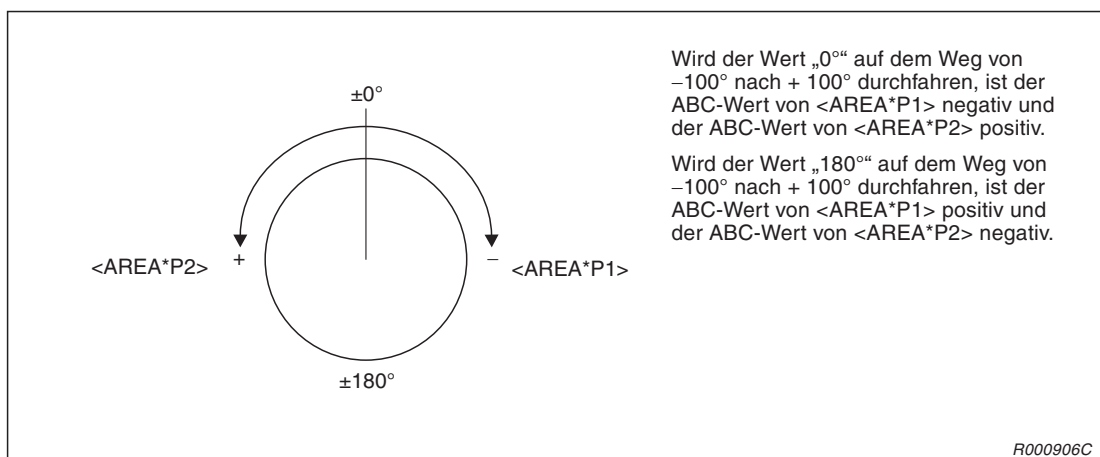


Abb. 9-7: Prüfung der Stellungsdaten

9.10 Verfahrwegbegrenzungsebene

Die Verfahrwegsgrenzen werden über eine Ebene im Basiskoordinatensystem definiert. Bei Überschreitung dieser Bereichsgrenzen erfolgt eine Fehlermeldung.

Die Definition der Ebene erfolgt über die drei Punkte P1, P2 und P3. Dabei wird festgelegt, ob der Bereich vor – d. h. auf der Seite des Roboter-Nullpunkts – oder hinter der Ebene als Arbeitsbereich bewertet wird.

Die Funktion soll Zusammenstöße mit der Arbeitfläche oder mit umliegenden Einheiten vermeiden. Es können maximal 8 Ebenen definiert werden. Die Ebenen selbst unterliegen keiner Beschränkung.

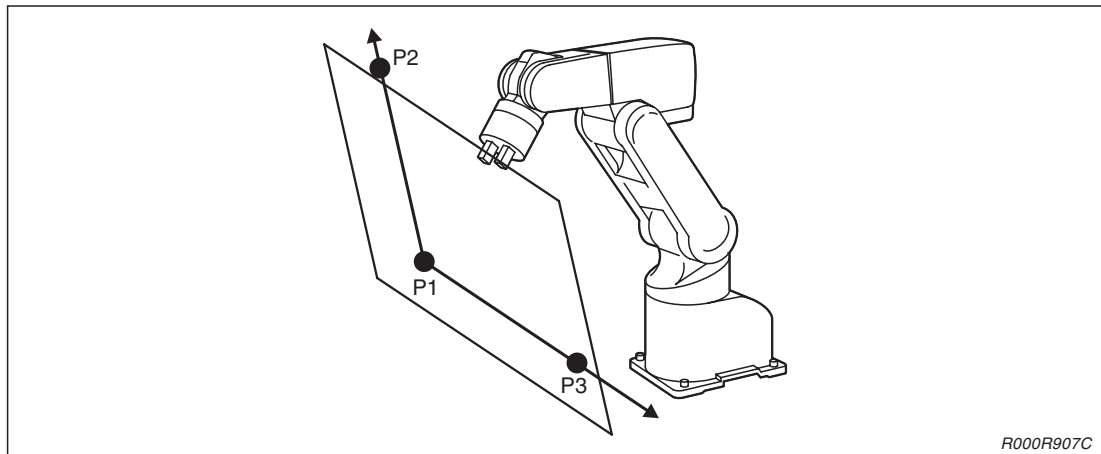


Abb. 9-8: Verfahrwegbegrenzungsebene

Parametereinstellung	Beschreibung
SFCnP ($1 \leq n \leq 8$)	Festlegung der 3 Punkte zur Definition der Ebene P1-Koordinaten X1, Y1 und Z1: Nullpunkt der Ebene P2-Koordinaten X2, Y2 und Z2: Position auf der X-Achse in der Ebene P3-Koordinaten X3, Y3 und Z3: Position in positiver Richtung auf der Y-Achse in der Ebene
SFCnME ($1 \leq n \leq 8$)	Festlegung der Mechanismusnummer, dem die Begrenzungsebene zugeordnet ist. Standardmäßig ist der Wert auf „1“ gesetzt. Beim Betrieb mehrerer Mechanismen muss die entsprechende Mechanismusnummer gesetzt werden.
SFCnAT ($1 \leq n \leq 8$)	Freigabe der Begrenzungsebenen: 0: gesperrt 1: freigegeben (Der zugelassene Arbeitsbereich liegt auf der Seite, auf der auch der Nullpunkt des Basiskoordinatensystems liegt.) -1: freigegeben (Der zugelassene Arbeitsbereich liegt auf der Seite, auf der der Nullpunkt des Basiskoordinatensystems nicht liegt. Die Einstellung „-1“ ist ab Software-Version J1 verfügbar.)

Tab. 9-10: Parametereinstellung

HINWEIS

Die Änderung der Parametereinstellungen wird erst nach Aus- und Wiedereinschalten der Spannungsversorgung des Steuergerätes aktiv.

9.11 Automatische Rückkehr

Die Funktion bewirkt nach einer Unterbrechung des Automatik- oder des Schrittbetriebs, bei der der Roboter im JOG-Betrieb über die Teaching Box zu einer anderen Position bewegt wurde, dass die Fortsetzung des Betriebs von der Position aus erfolgt, die bei der Unterbrechung aktuell war.

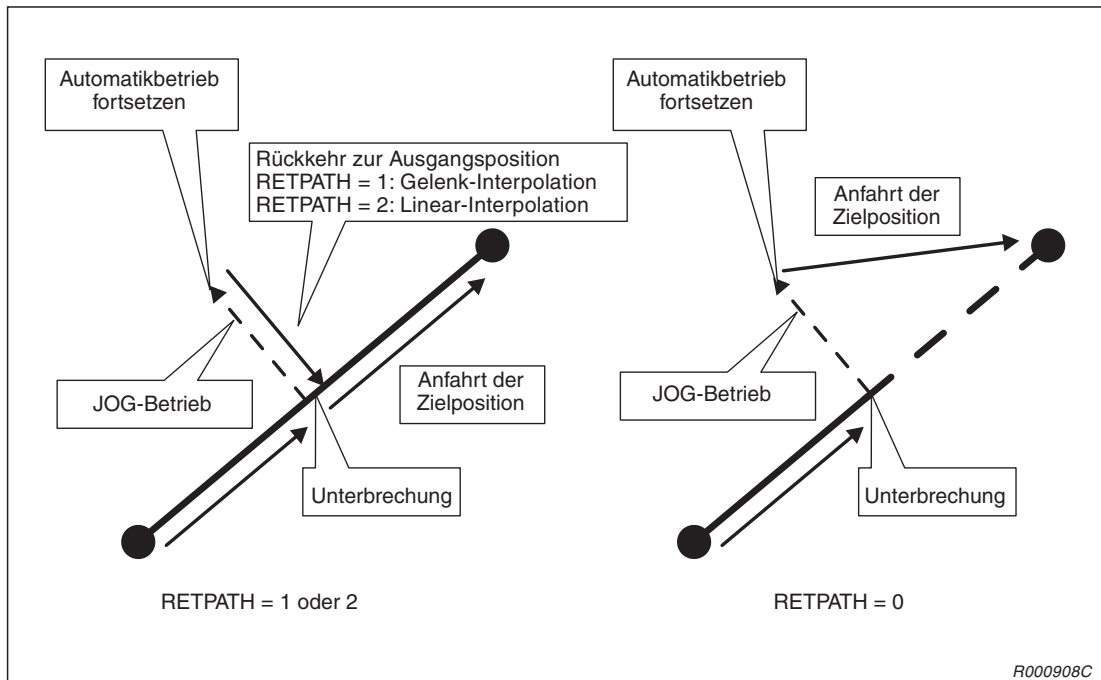


Abb. 9-9: Automatische Rückkehr nach einer Unterbrechung

Parametereinstellung	Beschreibung
RETPATH = 1 (Werkseinstellung)	<ul style="list-style-type: none"> ● Rückkehr mittels Gelenk-Interpolation zu der zum Zeitpunkt der Unterbrechung aktuellen Position ● Fortsetzung des Betriebs in der zum Zeitpunkt der Unterbrechung aktuellen Zeile
RETPATH = 0	Fortsetzung des Betriebs in der zum Zeitpunkt der Unterbrechung aktuellen Zeile von der im JOG-Betrieb angefahrenen Position Die Verfahrbewegung wird mit der in der Befehlszeile festgelegten Interpolationsart von der aktuellen zur Zielposition ausgeführt.
RETPATH = 2 ^①	<ul style="list-style-type: none"> ● Rückkehr mittels Linear-Interpolation zu der zum Zeitpunkt der Unterbrechung aktuellen Position ● Fortsetzung des Betriebs in der zum Zeitpunkt der Unterbrechung aktuellen Zeile

Tab. 9-11: Parametereinstellung

^① Die Einstellung „2“ ist für Steuergeräte ab Software-Version H4 verfügbar.

Die möglichen Einstellwerte des Parameters RETPATH unterscheiden sich in Abhängigkeit von der Interpolationsart und der Software-Version des Steuergerätes:

Interpolationsbefehl	Vor Software-Version H4 ^①	Ab Software-Version H4
MOV MVS	0 1	0 1 2
MVC MVR MVR2 MVR3	0 1	0 (wie bei Einstellung „1“) ^② 1 2
MVA	0 (wie bei Einstellung „1“) ^③ 1	0 (wie bei Einstellung „1“) ^③ 1 2

Tab. 9-12: Mögliche Einstellwerte des Parameters RETPATH

- ① Ist die Software-Version des Steuergerätes älter als Version H4, wird bei einer Einstellung des Parameters RETPATH auf „2“ die gleiche Funktion ausgeführt wie bei der Einstellung „1“ (Rückkehr mittels Gelenk-Interpolation).
- ② Bei einer Einstellung des Parameters RETPATH auf „0“ hängt die Funktion bei der Kreis-Interpolation (MVC, MVR, MVR2 und MVR3) von der Software-Version des Steuergerätes ab. Ab Version H4 wird dieselbe Funktion ausgeführt, wie bei einer Einstellung des Parameters auf „1“ (Rückkehr mittels Gelenk-Interpolation).
- ③ Bei der Bogen-Interpolation entspricht die Funktion bei der Einstellung „0“ der Funktion bei der Einstellung „1“ (Rückkehr mittels Gelenk-Interpolation).



GEFAHR:

Auch wenn bei einer Einstellung des Parameters RETPATH auf „1“ der JOG-Betrieb in einer anderen Interpolationsart als der Gelenk-Interpolation (XYZ-, Werkzeug-, Kreis-JOG-Betrieb usw.) ausgeführt wurde, erfolgt die Rückkehr zu der zum Zeitpunkt der Unterbrechung aktuellen Position mittels Gelenk-Interpolation. Achten Sie daher darauf, dass es zu keinen Zusammenstößen mit umliegenden Einrichtungen kommen kann.

HINWEISE

Ist Parameter RETPATH auf „2“ eingestellt, und der Roboter wird nach einer Unterbrechung im JOG-Betrieb zu einer Position bewegt, in der die Stellungs- oder Multirotationsdaten von denen der Ursprungsposition abweichen, kann es vorkommen, dass der Roboter die Position der Unterbrechung nicht mehr anfahren kann. Fahren Sie in diesem Fall die Position der Unterbrechung an und setzen sie den Betrieb fort.

Ist Parameter RETPATH bei aktivierter CNT-Einstellung auf „1“ oder „2“ eingestellt, kehrt der Roboter nicht zu der Position zurück, die zum Zeitpunkt der Unterbrechung aktuell war, sondern zu einem Punkt auf der geradlinigen Verbindung zwischen den Positionen (siehe folgende Abbildung). Ist der Parameter auf „0“ eingestellt, erfolgt die Anfahrt der Zielposition direkt von der aktuellen Position aus.

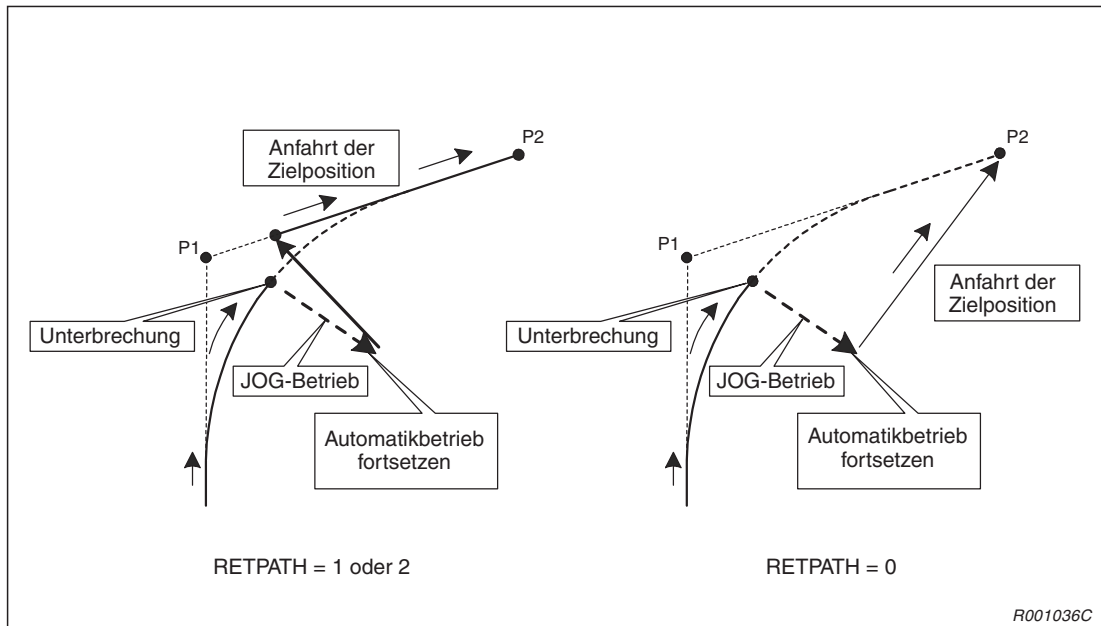


Abb. 9-10: Automatische Rückkehr bei aktivierter CNT-Einstellung

9.12 Automatischer Programmstart nach dem Einschalten

Die Funktion ermöglicht den automatischen Start eines Roboterprogramms nach dem Einschalten der Spannungsversorgung.



GEFAHR:

Beachten Sie, dass der Roboterbetrieb sofort nach dem Einschalten der Spannungsversorgung startet. Verwenden Sie die Funktion daher nur nach sorgfältiger Prüfung aller betriebsrelevanten Umstände.

Parametereinstellung	Beschreibung
SLT□	Beispiel: SLT2 = 2, ALWAYS, REP Festlegung von Programmname, Startbedingung und Ausführungsformat In diesem Abschnitt wird die Startbedingung erläutert.
ALWENA	0 oder 7 Der Parameter ermöglicht in Programmen mit der Startbedingung „ALWAYS“ die Ausführung von Multitask-Befehlen wie XRUN und XLOAD und des Befehls SERVO.

Tab. 9-13: Parametereinstellung

- ① Erstellen Sie zuerst das ständig auszuführende Programm (Startbedingung: ALWAYS) und das Betriebsprogramm.

Programm #2, Programm mit Startbedingung ALWAYS

```

100 'Automatisch startendes Beispielprogramm
110 '
120 'Führt Programm #1 aus, wenn der MODE-Umschalter auf AUTO (Ext.) geschaltet ist
130 'Stoppt das Programm und springt in die erste Programmzeile zurück, wenn der
    'MODE-Umschalter nicht auf AUTO (Ext.) geschaltet ist
140 '
150 IF M_MODE <> 3 AND (M_RUN(2) = 1 OR MWAI(2) = 1) THEN GOSUB *MTSTOP
160 IF M_MODE = 3 AND M_RUN(2) = 0 AND MWAI(2) = 0 THEN GOSUB *MTSTART
165 IF M_MODE = 2 THEN HLT 'Zum Debuggen
170 END
180 '
190 *MTSTART
200 XRUN 1,"1"
210 RETURN
220 '
230 *MTSTOP
240 XSTP 1
250 XRST 1
250 RETURN

```

Programm #1, Betriebsprogramm (beliebiges Programm)

```

100 'Hauptprogramm (Positionsdaten für RV-2AJ)
110 SERVO ON
120 M_OUT(8) = 0
130 MOV P1
140 M_OUT(8) = 1
150 MOV P2
160 END

P1 = (+300.00,-200.00,+200.00,+0.00,+180.00,+0.00)(6,0)
P2 = (+300.00,+200.00,+200.00,+0.00,+180.00,+0.00)(6,0)

```

- ② Stellen Sie anschließend die Parameter entsprechend den Werten in der folgenden Tabelle ein.

Parametereinstellung	Beschreibung
SLT2	SLT2 = 2, ALWAYS, REP 'Ständige Ausführung des Programms #2
ALWENA	7 In Programmen mit der Startbedingung „ALWAYS“ ist die Ausführung von Multitask-Befehlen wie XRUN und XLOAD und des Befehls SERVO möglich.

Tab. 9-14: Einstellung der Parameter

Schalten Sie nach der Einstellung der Parameter die Spannungsversorgung des Steuergerätes aus.

- ③ Im oben beschriebenen Beispielprogramm wird nach Einschalten der Spannungsversorgung das Programm #1 ausgeführt und der Roboterbetrieb gestartet, wenn sich der MODE-Umschalter in der Stellung AUTO (Ext.) befindet.

9.13 Handgreifer

In der Werkseinstellung ist ein Handgreifer mit Doppelmagnetspule voreingestellt. Bei Einsatz einer Einfachmagnetspule oder bei Steuerung des Roboters durch die Handsensorsignale über allgemeine Eingänge muss der Parameter HANDTYPE wie im Folgenden beschrieben eingestellt werden.

In der Werkseinstellung ist der Parameter auf folgende Werte gesetzt:

Parameter	Einstellung
HANDTYPE	D900,D902,D904,D906, , , ,

Tab. 9-15: Parametereinstellung

Die Werte entsprechen von links beginnend: Hand #1, #2 usw. Die Werkseinstellung ist wie folgt:

Hand 1 = Zugriff auf die Eingangssignale #900 und #901

Hand 2 = Zugriff auf die Eingangssignale #902 und #903

Hand 3 = Zugriff auf die Eingangssignale #904 und #905

Hand 4 = Zugriff auf die Eingangssignale #906 und #907

Die Handnummern 1 bis 4 (oder 8) werden als Argument in den Befehlen zum Öffnen und Schließen der Hände (HOPEN und HCLOSE) verwendet.

Einstellung

Bei Einsatz eines Handgreifers mit Doppelmagnetspule muss vor der Signalnummer ein „D“ eingefügt werden. Für die Handnummern gilt: $1 \leq \text{Handnummer} \leq 4$.

Bei Einsatz eines Handgreifers mit Einfachmagnetspule muss vor der Signalnummer ein „S“ eingefügt werden. Für die Handnummern gilt: $1 \leq \text{Handnummer} \leq 8$.

Beispiele ▾

Die Zuweisung von zwei Handgreifern mit Doppelmagnetspule an das allgemeine Eingangssignal #10 erfolgt über: HANDTYPE = D10,D12, , , , ,

Die Zuweisung von drei Handgreifern mit Einfachmagnetspule an das allgemeine Eingangssignal #10 erfolgt über: HANDTYPE = S10,S11,S12, , , , ,

Die Zuweisung der Hand 1 mit Doppelmagnetspule an das allgemeine Eingangssignal #10 und der Hand 2 mit Einfachmagnetspule an das allgemeine Eingangssignal #12 erfolgt über: HANDTYPE = D10,S12, , , , ,

△

9.14 Handgreiferzustand nach Initialisierung

Die folgende Tabelle zeigt die Werkseinstellung des Handgreiferzustands:

Handausführung	Handgreiferzustand	Ausgangssignalzustand		
		Mechanismus #1	Mechanismus #2	Mechanismus #3
Bei installierter Schnittstellenkarte zur Steuerung der pneumatischen Greifhand (Doppelmagnetspule)	Hand 1 = geöffnet	900 = 1	910 = 1	920 = 1
		901 = 0	911 = 0	921 = 0
	Hand 2 = geöffnet	902 = 1	912 = 1	922 = 1
		903 = 0	913 = 0	923 = 0
	Hand 3 = geöffnet	904 = 1	914 = 1	924 = 1
		905 = 0	915 = 0	925 = 0
	Hand 4 = geöffnet	906 = 1	916 = 1	926 = 1
		907 = 0	917 = 0	927 = 0
Bei installierter Schnittstellenkarte zur Steuerung der motorbetriebenen Greifhand	Hand geöffnet	M_OUT (9*0) bis M_OUT (9*7) werden vom System verwendet und können daher vom Anwender nicht eingesetzt werden. Ein korrektes Öffnen und Schließen des Handgreifers ist über die Signale nicht möglich.		
Schnittstelle ohne installierte Schnittstellenkarte	—	M_OUT (9*0) bis M_OUT (9*7) sind ohne Funktion		

Tab. 9-16: Handgreiferzustand und Handsensorsignale

Ein Steuergerät kann mehrere Mechanismen steuern. Wird für jeden Mechanismus eine Schnittstellenkarte zur Steuerung der pneumatischen Greifhand eingesetzt, erfolgt die Zuweisung der Handsteuersignale in folgender Form:

Mechanismus #1 = #900 bis #907 (Standardanschluss mit einem Roboter an einer Steuereinheit)

Mechanismus #2 = #910 bis #917

Mechanismus #3 = #920 bis #927

Wird für den Roboter eine Schnittstellenkarte zur Steuerung einer motorbetriebenen Greifhand eingesetzt, verwendet das System die 900er Signale als festgelegte Steuersignale. Der Anwender hat keine Zugriffsmöglichkeit mehr auf diese Signale. Verwenden Sie daher die Handsteuerbefehle oder die Tasten zur Handsteuerung auf der Teaching Box.

In der Grundeinstellung sind alle Handgreifer nach dem Einschalten der Spannungsversorgung geöffnet.

Parameter	Signalnummer	Einstellung
HANDNIT	900, 901, 902, 903, 904, 905, 906, 907	1, 0, 1, 0, 1, 0, 1, 0

Tab. 9-17: Grundeinstellung des Parameters HANDNIT

Die Tabelle gilt für den Standardanschluss mit einem Roboter an einer Steuereinheit. Beim Anschluss mehrerer Mechanismen an eine Steuereinheit muss eine Zuordnung von Mechanismus und Ausgangssignal über den Parameter HANDNIT erfolgen.

Beispiel ▾

Soll nur die Hand 1 nach dem Einschalten der Spannungsversorgung geschlossen sein, muss der Parameter HANDNIT wie folgt eingestellt werden. Analog dazu wird eine motorbetriebene Greifhand (Handnummer = 1) mit der gezeigten Einstellung geschlossen.

Parameter	Signalnummer	Einstellung
HANDNIT	900, 901, 902, 903, 904, 905, 906, 907	0, 1, 1, 0, 1, 0, 1, 0

Tab. 9-18: Beispielleinstellung



HINWEIS

Der Parameter HANDNIT legt den Handgreiferzustand nach der Initialisierung über die 900er Signale fest. Soll die Einstellung des Handgreiferzustandes nach der Initialisierung über allgemeine E/A (andere als die 900er Signale) oder CC-Link (ab 6000) (im Parameter HANDTYPE sind andere Signale als die 900er festgelegt) erfolgen, verwenden Sie zur Festlegung der Zustände nicht den Parameter HANDNIT, sondern die Parameter ORST*. Die über die Parameter ORST* gesetzten Ausgangsbitmuster entsprechen dem Initialisierungszustand der Signale nach dem Einschalten der Spannungsversorgung.



GEFAHR:

Beachten Sie, dass ein Werkstück aus einer Hand, die nach dem Einschalten der Spannungsversorgung geöffnet wird, herunterfallen kann. Es besteht Verletzungsgefahr!

9.15 Ausgangsbitmuster

In der Werkseinstellung werden alle allgemeinen Ausgangssignale nach dem Einschalten ausgeschaltet, d. h. auf „0“ gesetzt. Die Signalzustände können über Parameter geändert werden. Beachten Sie, dass eine Änderung des Parameters auch die Ausgangssignalmuster beim Zurücksetzen über einen Eingang oder über die CLR-Anweisung beeinflusst.

	Parameter	Einstellung
Dezentrale E/A	ORST0	Signalnummer 0-----7 8-----15 16-----23 24-----31 00000000, 00000000, 00000000, 00000000
	ORST32	32-----39 40-----48 49-----56 57-----65 00000000, 00000000, 00000000, 00000000
	ORST64	00000000, 00000000, 00000000, 00000000
	ORST96	00000000, 00000000, 00000000, 00000000
	ORST128	00000000, 00000000, 00000000, 00000000
	ORST160	00000000, 00000000, 00000000, 00000000
	ORST192	00000000, 00000000, 00000000, 00000000
PROFIBUS-Schnittstellenkarte	ORST224	00000000, 00000000, 00000000, 00000000
	ORST2000	00000000, 00000000, 00000000, 00000000
	ORST2032	00000000, 00000000, 00000000, 00000000
	ORST2064	00000000, 00000000, 00000000, 00000000
	ORST2096	00000000, 00000000, 00000000, 00000000
	ORST2128	00000000, 00000000, 00000000, 00000000
	ORST2160	00000000, 00000000, 00000000, 00000000
	ORST2192	00000000, 00000000, 00000000, 00000000
	ORST2224	00000000, 00000000, 00000000, 00000000
	ORST2256	00000000, 00000000, 00000000, 00000000
	ORST2288	00000000, 00000000, 00000000, 00000000
	:	:
	:	:
:	:	
ORST4984	00000000, 00000000, 00000000, 00000000	
ORST5016	00000000, 00000000, 00000000, 00000000	

Tab. 9-19: Ausgangsbitmuster (1)

	Parameter	Einstellung
CC-Link-Schnittstellenkarte	ORST6000	00000000, 00000000, 00000000, 00000000
	ORST6032	00000000, 00000000, 00000000, 00000000
	ORST6064	00000000, 00000000, 00000000, 00000000
	ORST6096	00000000, 00000000, 00000000, 00000000
	ORST6128	00000000, 00000000, 00000000, 00000000
	ORST6160	00000000, 00000000, 00000000, 00000000
	ORST6192	00000000, 00000000, 00000000, 00000000
	ORST6224	00000000, 00000000, 00000000, 00000000
	ORST6256	00000000, 00000000, 00000000, 00000000
	ORST6288	00000000, 00000000, 00000000, 00000000
	:	:
	:	:
	:	:
	ORST7984	00000000, 00000000, 00000000, 00000000
ORST8016	00000000, 00000000, 00000000, 00000000	

Tab. 9-19: Ausgangsbitmuster (2)

Folgende Einstellungen sind möglich:

0 = AUS

1 = EIN

* = Status beibehalten (wird beim Einschalten ausgeschaltet)

Beispiel ▾

Die allgemeinen Ausgangssignale 10, 11, und 12 der Standardschnittstelle und 32, 33 und 40 der zusätzlichen Schnittstelle sollen nach Einschalten der Spannungsversorgung eingeschaltet werden.

Parameter	Einstellung
ORST0	00000000, 00111000, 00000000, 00000000
ORST32	11000000, 10000000, 00000000, 00000000

Tab. 9-20: Beispielerstellung der Parameter

Sollen zusätzlich die Signale 20, 21 und 22 beim Zurücksetzen der allgemeinen Ausgangssignale ihren vorherigen Zustand beibehalten, sind folgende Einstellungen vorzunehmen:

Parameter	Einstellung
ORST0	00000000, 00111000, 0000** *0, 00000000
ORST32	11000000, 10000000, 00000000, 00000000

Tab. 9-21: Zusätzliche Parametereinstellung

Nach Einstellung der Parameter werden die Signale 20, 21 und 22 beim Einschalten der Spannungsversorgung auf „0“ (AUS) gesetzt. Es ist nicht möglich, die Einstellung so vorzunehmen, dass die Signale nach dem Einschalten der Spannungsversorgung den Zustand „1“ (EIN) annehmen und diesen Zustand dann nach Zurücksetzen der allgemeinen Ausgangssignale beibehalten.



HINWEIS

Geben Sie bei der Einstellung der Parameter die richtige Anzahl von Nullen ein, da ansonsten beim Einschalten der Spannungsversorgung eine Fehlermeldung erfolgt.

9.16 Kommunikationseinstellungen

9.16.1 Allgemeine Beschreibung

Die Steuergeräte CR1, CR2, CR2A, CR2B und CR3 unterstützen eine RS232C-Standard-schnittstelle und zwei optionale, serielle Erweiterungsschnittstellenkarten (2 Anschlüsse pro Karte). Von den beiden Anschlüssen der Erweiterungsschnittstellenkarten entspricht einer dem RS232C-Standard, beim anderen Anschluss kann zwischen den Standards RS232C und RS422 gewählt werden. Der Anschluss von maximal zwei zusätzlichen seriellen Schnittstellenkarten ist möglich.

Die RS232C-Standard-schnittstelle dient in der Regel dem Anschluss eines PCs. Mit Hilfe der PC-Support-Software oder COSIROP können Roboterprogramme übertragen und getestet werden. Eine Vielzahl zusätzlicher Optionskarten ermöglicht die Datenkommunikation z. B. mit optischen Sensoren und weiteren externen Geräten.

Kommunikationsvorgänge werden durch Kommunikationsbefehle (OPEN, CLOSE, PRINT und INPUT usw.) im Roboterprogramm gesteuert. Das Steuergerät kann nicht durch externe Geräte wie z. B. einen PC (mit Ausnahme des Automatikbetriebs und der Statusanzeige) gesteuert werden. Sollte eine Steuerung durch externe Geräte erforderlich sein, wenden Sie sich an Ihren Vertriebspartner.

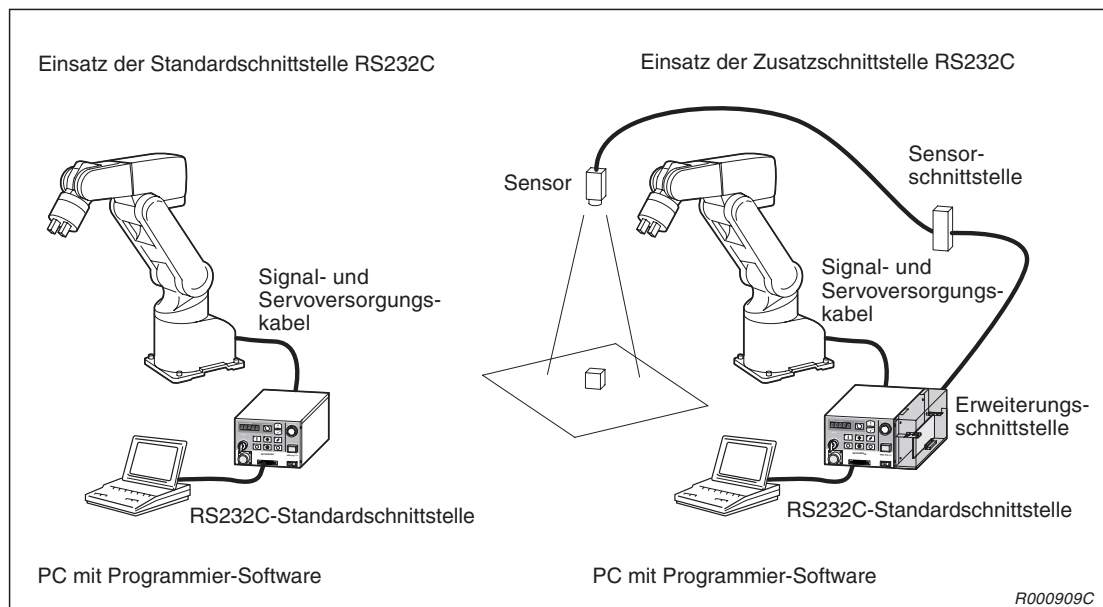


Abb. 9-11: Beispiele für eine Schnittstellenbelegung

HINWEIS

Die in der Abbildung oben gezeigte Konfiguration gilt für das Steuergerät CR1. Bei den Steuergeräten CR2, CR2A, CR2B und CR3 wird die Erweiterungsschnittstelle intern montiert.

9.16.2 Datenübertragung über die RS232C-Schnittstelle

Die serielle RS232C-Standardschnittstelle dient zum Korrigieren des Roboterprogramms und zur Signalüberwachung während der Initialisierungsphase. Sie ermöglicht zwar auch den Datenaustausch zwischen dem Steuergerät und externen Geräten, allerdings empfiehlt sich zu diesem Zweck die Verwendung einer optionalen Erweiterungsschnittstelle. Folgendes Beispielprogramm zeigt den Einsatz einer seriellen Erweiterungsschnittstelle.

Parameter	Werkseinstellung	Einstellung	Beschreibung
COMDEV	RS232, , , , , ,	RS232, OPT11, , , , ,	Bei Verwendung von CH1 in Steckplatz 1
CBAUE11	9600	9600	Übertragungsrate = 9600 Bit/s
CPRTYE11	2	2	Parität = gerade
CSTOPE11	2	2	Stoppbits = 2
CTERME11	0 (CR)	1 (CRLF)	Endezeichen = CRLF (Carriage Return + LineFeed)
CPRCE11	0	2	Umschaltung auf Datenverbindung Diese Einstellung wird zur Kommunikation mit externen Geräten über die Befehle OPEN, PRINT, INPUT und CLOSE im Roboterprogramm verwendet.
	Kommunikations- beispiel	OPEN "COM2:" AS #1 PRINT #1, "ABC" M1 = 10 PRINT #1, M1 INPUT #1, C1\$ INPUT #1, M1	Roboter Externes Gerät "ABC"CRLF → "10"CRLF → ← "RUN"CRLF ← "20"CRLF

Tab. 9-22: Parametereinstellung bei Einsatz einer seriellen Erweiterungsschnittstelle

Eine detaillierte Beschreibung der seriellen Erweiterungsschnittstelle finden Sie im Handbuch der Schnittstelle.

Beispielprogramm

Im folgenden Beispielprogramm fährt der Roboter die Position der Kamera an, erhält Korrekturdaten über die RS232C-Schnittstelle, fährt eine Position oberhalb der korrigierten Position an, fährt die Zielposition an, schließt den Handgreifer und nimmt das Werkstück auf.

10	OPEN "COM2:" AS #1	'Öffnet den RS232C-Kommunikationskanal als Datei Nr. 1
20	MOV P1,-50	'Fährt die Position oberhalb der Kameraposition an
30	M_OUT(10) = 1 DLY 0.5	'Ausgabe des Kamera-Startsignals (Sensor starten)
40	PX = P1	'Überträgt den Nullpunkt der Kamera in die Position PX
50	INPUT #M1,MX,MY,MC	'Liest die Korrekturdaten ein
60	PX.X = PX.X + MX	'Korrektur in X-Richtung
70	PX.Y = PX.Y + MY	'Korrektur in Y-Richtung
80	PX.C = PX.C + MC	'Korrektur der C-Achsendaten
90	MOV PX,-50	'Fährt eine Position oberhalb des Werkstücks an
100	OVRD 30	'Verlangsamt die Verfahrbewegung
110	MVS PX	'Fährt die Zielposition an
120	DLY 0.3	'Positioniervorgang über Timer abschließen
130	HCLOSE	'Schließt den Handgreifer
140	DLY 0.3	'Greifvorgang über Timer abschließen
150	MVS P1,-50	'Fährt die Position oberhalb der Kameraposition mittels Linearinterpolation an
	:	:

In dem gezeigten Programmbeispiel muss der Nullpunkt der Kamera vor der Programmausführung als eine geteachte Position gespeichert werden. Das Aufnehmen des Werkstücks in Richtung der Z-Achse muss möglich sein.

Kommunikationsdaten

(Zeichenkette, die von einem externen Gerät, z. B. einem Sensor, gesendet wird.)

Die Daten werden in der Reihenfolge X, Y und C zum Roboter gesendet. Das Abschlusszeichen der Daten ist ein „Carriage Return“ (CR = hexadezimal 0D). Die Einheiten sind mm, mm und Grad.

Beispiel ▾

Folgende Daten werden als Zeichenkette gesendet:
X = 12.34 mm, Y = 0.39 mm und C = 56.78°

Die Zeichenkette ist:
"12.34, 0.39, 56.78(CR)"

△

Parallele Nutzung von Programmier-Software und Roboter-Programmen mit der Standardschnittstelle

Wird die serielle Standardschnittstelle an der Vorderseite des Steuergerätes während der Einschalt routine zum Datenaustausch mit der Programmier-Software eines PCs und während des Betriebs zum Datenaustausch mit externen Geräten verwendet, muss jeder Datenkette, die von externen Geräten an das Robotersteuergerät übertragen wird, die Zeichenkette „PRN“ vorangestellt werden.

Beispiel ▾

"PRN12.34, 0.39, 56.78(CR)"



Die Zeichenkette dient als Identifizierungsmerkmal der Datenquelle. Die Daten müssen ohne Leerzeichen auf die Zeichen „PRN“ folgen.

Ist es nicht möglich, das Kommunikationsprotokoll des externen Gerätes in dieser Weise zu ändern, kann es nicht an der Standardschnittstelle betrieben werden. Auch wenn die Einstellung des Parameters CPRC232 eine Datenübertragung ohne die vorangestellte Zeichenkette „PRN“ erlaubt, ist nach der Änderung des Parameters kein Betrieb mehr über die Programmier-Software möglich. Können die Zeichen „PRN“ nicht jeder Datenkette vorangestellt werden, verwenden Sie zur Kommunikation mit externen Geräten die serielle Erweiterungsschnittstelle.

Der optische Sensor AS50VS von MITSUBISHI ELECTRIC sendet standardmäßig vor jeder Datenkette die Zeichen „PRN“ und kann somit an der seriellen Standardschnittstelle betrieben werden. Ein gleichzeitiger Betrieb von Programmier-Software und Sensor ist jedoch an einem Schnittstellenanschluss rein physikalisch nicht möglich.

Stabile Kommunikation zwischen Programmier-Software und Steuergerät

In Abhängigkeit des PC-Modells, der Kommunikationseinstellungen der Programmier-Software und des Steuergerätes können bei der Übertragung großer Datenmengen (z. B. bei der Erstellung von Backups oder beim Upload/Download) Instabilitäten auftreten. Ändern Sie die Kommunikationseinstellungen der Programmier-Software und des Steuergerätes wie folgt, um eine stabile Kommunikation zu gewährleisten. Eine Kommunikation ist nur möglich, wenn die Kommunikationseinstellungen des Steuergerätes mit denen der Programmier-Software übereinstimmen.

	Parameter	Werkseinstellung	Einstellung
Steuergerät	Kommunikationsprotokoll (CPRC232)	0 (kein Protokoll)	1 (Protokoll)
Programmier-Software	Protokoll	Kein Protokoll	Protokoll

Tab. 9-23: Kommunikationseinstellungen

HINWEIS

Soll die Kommunikation mit dem PC über eine RS232C-Erweiterungsschnittstelle erfolgen, ist der Parameter der Erweiterungsschnittstelle zu ändern.

9.17 Hand- und Werkstückbedingung

9.17.1 Optimale Beschleunigung/Abbremsung

Die Funktion erlaubt die Einstellung der optimalen Beschleunigungs-/Abbremszeit in Abhängigkeit der Last an der Handspitze. Folgende Parameter müssen zur Nutzung der optimalen Beschleunigung/Abbremsung eingestellt werden.

Bei den Robotermodellen RV-S/RH-S sind die Parameter auch bei Verwendung der Kollisionsüberwachung einzustellen. Dabei sind die Parameter HNDDAT0 und WRKDAT0 bei aktivierter Kollisionsüberwachung im JOG-Betrieb wirksam.

	Parameter	Einstellung
Hand- bedingungen	HNDDAT0	Der Parameter kann mit den Robotermodellen RV-S/RH-S verwendet werden und ist vom jeweiligen Roboter abhängig.
	HNDDAT1	Standardlast, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	HNDDAT2	Standardlast, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	HNDDAT3	Standardlast, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	HNDDAT4	Standardlast, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	HNDDAT5	Standardlast, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	HNDDAT6	Standardlast, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	HNDDAT7	Standardlast, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	HNDDAT8	Standardlast, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
Werkstück- bedingungen	WRKDAT0	Der Parameter kann mit den Robotermodellen RV-S/RH-S verwendet werden und ist vom jeweiligen Roboter abhängig.
	WRKDAT1	0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	WRKDAT2	0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	WRKDAT3	0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	WRKDAT4	0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	WRKDAT5	0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	WRKDAT6	0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	WRKDAT7	0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0
	WRKDAT8	0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0

Tab. 9-24: Werkseinstellung der Parameter

Die Parameter haben folgende Struktur:

Gewicht, Größe X, Größe Y, Größe Z, Schwerpunkt X, Y, und Z

Eine Einstellung von bis zu 8 Hand- und Werkstückbedingungen ist möglich. Verwenden Sie zur Bestimmung der Handmaße einen gedachten Quader, der die Hand völlig umschließt. Die Werte für die optimale Beschleunigung/Abbremsung basieren auf den Hand- und Werkstückbedingungen, die über den LOADSET-Befehl ausgewählt werden.

9.17.2 Handgreiferzustand

Über einen weiteren Parameter kann der Handgreiferzustand bei Ausführung der Handsteuerbefehle HOPEN (HCLOSE) festgelegt werden. In Abhängigkeit des Handgreiferstatus wird die optimale Beschleunigung für den Handgreifer ohne Werkstück oder für den Handgreifer mit Werkstück berechnet.

Parameter	Einstellung
HNDHOLD1	0, 1
HNDHOLD2	0, 1
HNDHOLD3	0, 1
HNDHOLD4	0, 1
HNDHOLD5	0, 1
HNDHOLD6	0, 1
HNDHOLD7	0, 1
HNDHOLD8	0, 1

Tab. 9-25: Werkseinstellung der Parameter

0 = nicht schließen

1 = schließen

9.17.3 Definition der Koordinatensysteme für die Hand- und Werkstückbedingungen

Folgende Abbildungen zeigen die Koordinatensysteme zur Einstellung der Hand- und Werkstückbedingungen für die verschiedenen Robotermodelle. Die Koordinatensysteme für die Hand- und für die Werkstückbedingung sind identisch. Alle Größenangaben sind positiv.

Robotermodelle RP-1AH/3AH/5AH

Der Nullpunkt des Koordinatensystems liegt im unteren Ende der Kugelumlaufspindel.

Z-Achse: Die Aufwärtsrichtung wird positiv gezählt.

X-Achse: Die Bewegungsrichtung bei Armstreckung wird positiv gezählt.

Y-Achse: Die Y-Achse entspricht dem Rechtssystem.

Folgende Elemente müssen eingestellt werden:

Schwerpunkt in X- und Y-Richtung

Größe in X- und Y-Richtung

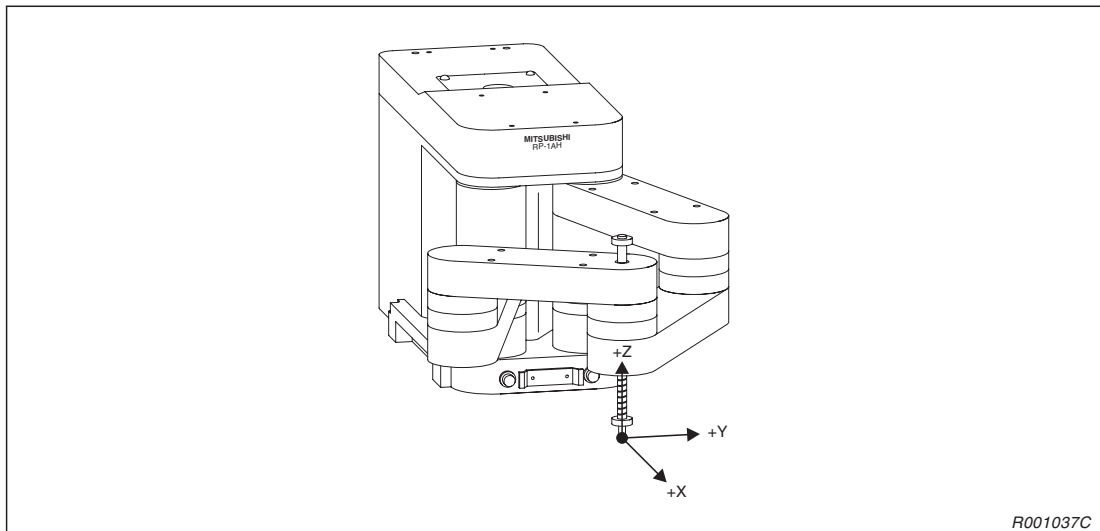


Abb. 9-12: Koordinatensystem für die Hand- und Werkstückbedingungen für die Robotermodelle RP-1AH/3AH/5AH

Robotermodelle RV-A/RV-S

Das Koordinatensystem entspricht dem Werkzeugkoordinatensystem.

Folgende Elemente müssen eingestellt werden:

Schwerpunkt in X-, Y- und Z-Richtung

Größe in X-, Y- und Z-Richtung

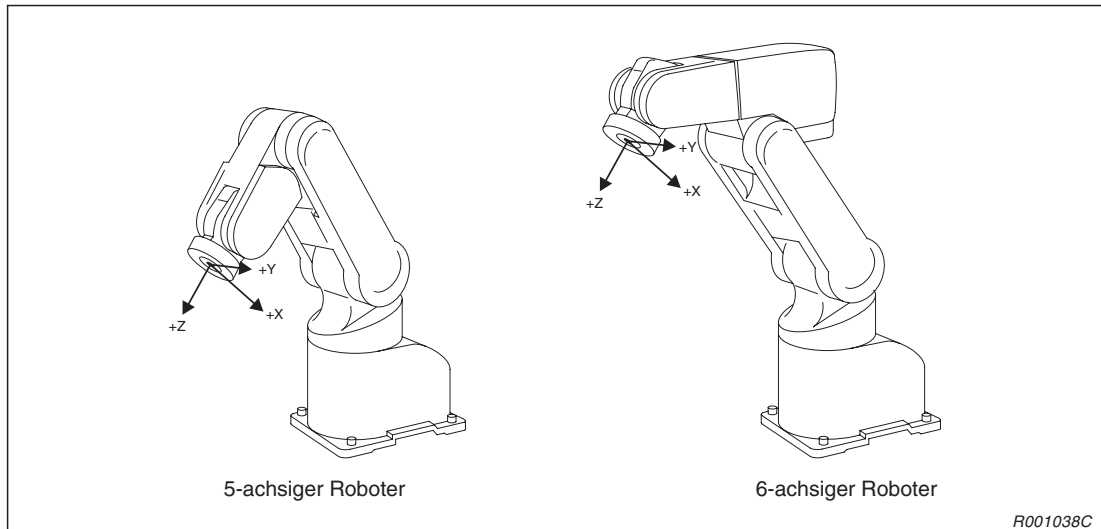


Abb. 9-13: Koordinatensystem für die Hand- und Werkstückbedingungen für die Robotermodelle RV-A/RV-S

Robotermodelle RH-5AH/10AH/15AH, RH-6SH/12SH

Der Nullpunkt des Koordinatensystems liegt im unteren Ende der Kugelumlaufspindel.

Z-Achse: Die Aufwärtsrichtung wird positiv gezählt.

X-Achse: Die Bewegungsrichtung bei Armstreckung wird positiv gezählt.

Y-Achse: Die Y-Achse entspricht dem Rechtssystem.

Folgende Elemente müssen eingestellt werden:

Schwerpunkt in X-Richtung

Größe in X- und Y-Richtung

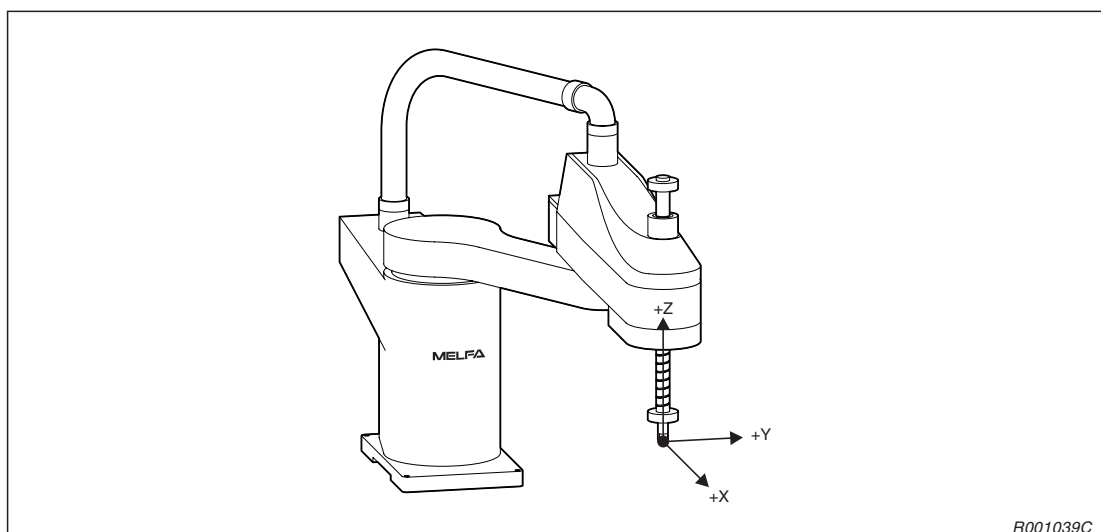


Abb. 9-14: Koordinatensystem für die Hand- und Werkstückbedingungen für die Robotermodelle RH-5AH/10AH/15AH, RH-6SH/12SH

9.18 Fehlermeldung bei Erreichen des singulären Punktes

Wird der Roboter über die Teaching Box verfahren, erfolgt bei Erreichen eines singulären Punktes eine Fehlermeldung. Der Roboter kann trotz Fehlermeldung noch weiter verfahren werden, bis er einen Bereich erreicht, in dem kein Betrieb mehr möglich ist. Es erfolgt automatisch ein Rücksetzen der Fehlermeldung, wenn sich der Roboter vom singulären Punkt entfernt. Die Funktion ist ab der Software-Version G9 gültig.

Aktivierung der Fehlermeldung

Die Fehlermeldung bei Erreichen des singulären Punktes erfolgt bei Ausführung folgender Funktionen über die Teaching Box:

- JOG-Betrieb (außer im Gelenk-JOG-Modus)
- Schrittbetrieb in Vorwärts- und Rückwärtsrichtung
- Verfahrbewegungen mit Linear-Interpolation
- direkte Befehlsausführung

Erreicht der Roboter einen singulären Punkt, ertönt ein durchgehender Signalton vom Summer im Steuergerät. Die Anzeige „STATUS.NUMBER“ des Steuergeräts ändert sich nicht. Im Falle des JOG-Betriebs erscheint parallel zum Signalton folgende Warnung auf der Anzeige der Teaching Box:

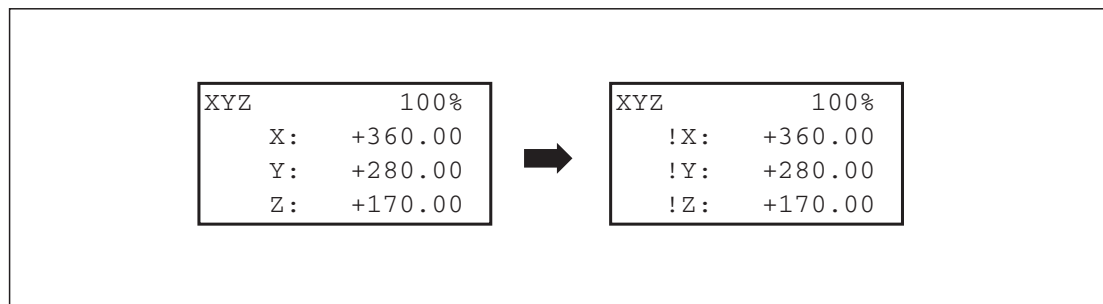


Abb. 9-15: Warnung im JOG-Betrieb auf der Anzeige der Teaching Box

Bei folgenden Funktionen erfolgt auch bei Erreichen eines singulären Punktes keine Fehlermeldung:

- JOG-Betrieb einer Zusatzachse im Gelenk-JOG-Modus über die Teaching Box
- Ausführung eines Befehls für Gelenk-Interpolation über die Teaching Box (z. B. MOV-Befehl)
- Automatikbetrieb eines Programmes
- JOG-Betrieb über spezielle Eingänge (z. B. JOGENA oder JOGM)
- Bewegung des Roboters mit gelösten Bremsen durch äußere Krafteinwirkung
- unbewegter Roboter
- TYPE-Befehl

9.19 ROM- und Highspeed-RAM-Modus

Im ROM- und Highspeed-RAM-Modus sind einige Einschränkungen beim Programmbetrieb und der Datensicherung zu beachten.

9.19.1 Übersicht

In der Werkseinstellung werden die Roboterprogramme in einem RAM (SRAM) mit Batterie-pufferung gespeichert.

Dabei können jedoch Datenverluste durch eine leere Backup-Batterie und Programmfehler durch Spannungseinbrüche (z. B. kurzzeitiger Netzausfall) während eines Datenzugriffs auftreten bzw. Programme und Positionsdaten ungewollt geändert oder gelöscht werden.

Mit Hilfe von Parametern kann der Speicherort für den Datenzugriff zwischen ROM und RAM umgeschaltet werden.

Die Funktion steht ab Software-Version H7 des Steuergerätes zur Verfügung. Ab Software-Version J1 besteht zusätzlich die Möglichkeit zur Auswahl eines Highspeed-RAM-Modus durch Umschaltung auf einen DRAM-Speicher.

Parameter	Einstellung
ROMDRV	Der Zugriff auf Programme kann zwischen RAM und ROM umgeschaltet werden. 0: RAM-Modus (Werkseinstellung) 1: ROM-Modus 2: Highspeed-RAM-Modus (Ab Software-Version J1 besteht die Möglichkeit zur Auswahl des Highspeed-RAM-Modus. Dabei erfolgt der Datenzugriff über einen DRAM-Speicher.)
BACKUP	Kopiert Programme, Parameter, globale Variablen und Fehler-Logfiles vom RAM in das ROM. SRAM → FLROM (Dieser Parameter darf nicht geändert werden.) Bei einer Unterbrechung des Kopiervorgangs erfolgt die Anzeige „CANCEL“.
RESTORE	Schreibt Programme, Parameter, globale Variablen und Fehler-Logfiles vom ROM in das RAM zurück. FLROM → SRAM (Dieser Parameter darf nicht geändert werden.) Bei einer Unterbrechung des Kopiervorgangs erfolgt die Anzeige „CANCEL“.

Tab. 9-26: Parameter zur Speicherauswahl

Der Speicherzugriff und die somit ausführbaren Funktionen hängen von der Einstellung des Parameters ROMDRV ab. Folgende Tabelle zeigt den Zusammenhang.

Speicher	Beschreibung	Parameter ROMDRV		
		0 (RAM-Modus)	1 (ROM-Modus)	2 (Highspeed-RAM-Modus)
DRAM	Highspeed-Modus möglich Programme können ausgeführt werden, gehen aber beim Ausschalten der Versorgungsspannung verloren.		Ausführung von Programmen (Programmergebnisse gehen verloren)	Ausführung von Programmen (Programmergebnisse gehen verloren)
SRAM	Beim Ausschalten der Versorgungsspannung bleiben die Programme erhalten. Die Programme gehen erst bei entladener Batterie verloren. Es ist ein Schreib- und Lesezugriff möglich.	Ausführung von Programmen (Programmergebnisse werden gespeichert) Programmverwaltungsfunktionen Systemdaten lesen/schreiben Globale Variablen lesen/schreiben Programme lesen/schreiben	Systemdaten lesen/schreiben	Programmverwaltungsfunktionen Systemdaten lesen/schreiben Globale Variablen lesen/schreiben Programme lesen/schreiben
ROM	Die Programme bleiben sowohl beim Ausschalten der Versorgungsspannung als auch bei entladener Batterie erhalten. Es nur ein Lesezugriff möglich.		(Programmverwaltungsfunktionen gesperrt) Globale Variablen lesen/schreiben Programme lesen/schreiben	

Tab. 9-27: Speicherzugriff und Parameter ROMDRV

HINWEISE

Speichern Sie zur Sicherung der Systemdaten alle Parameter und Fehler-Logfiles. Das gilt auch für die Dateien, auf die ein Schreib- bzw. Lesezugriff durch die Programme (Befehle OPEN, PRINT und INPUT) erfolgt.

Unter Programmverwaltungsfunktionen versteht u. a. das Kopieren, Löschen oder Umbenennen von Programmen im Steuergerät mit Hilfe der Teaching Box, der PC-Support-Software oder COSIROP.

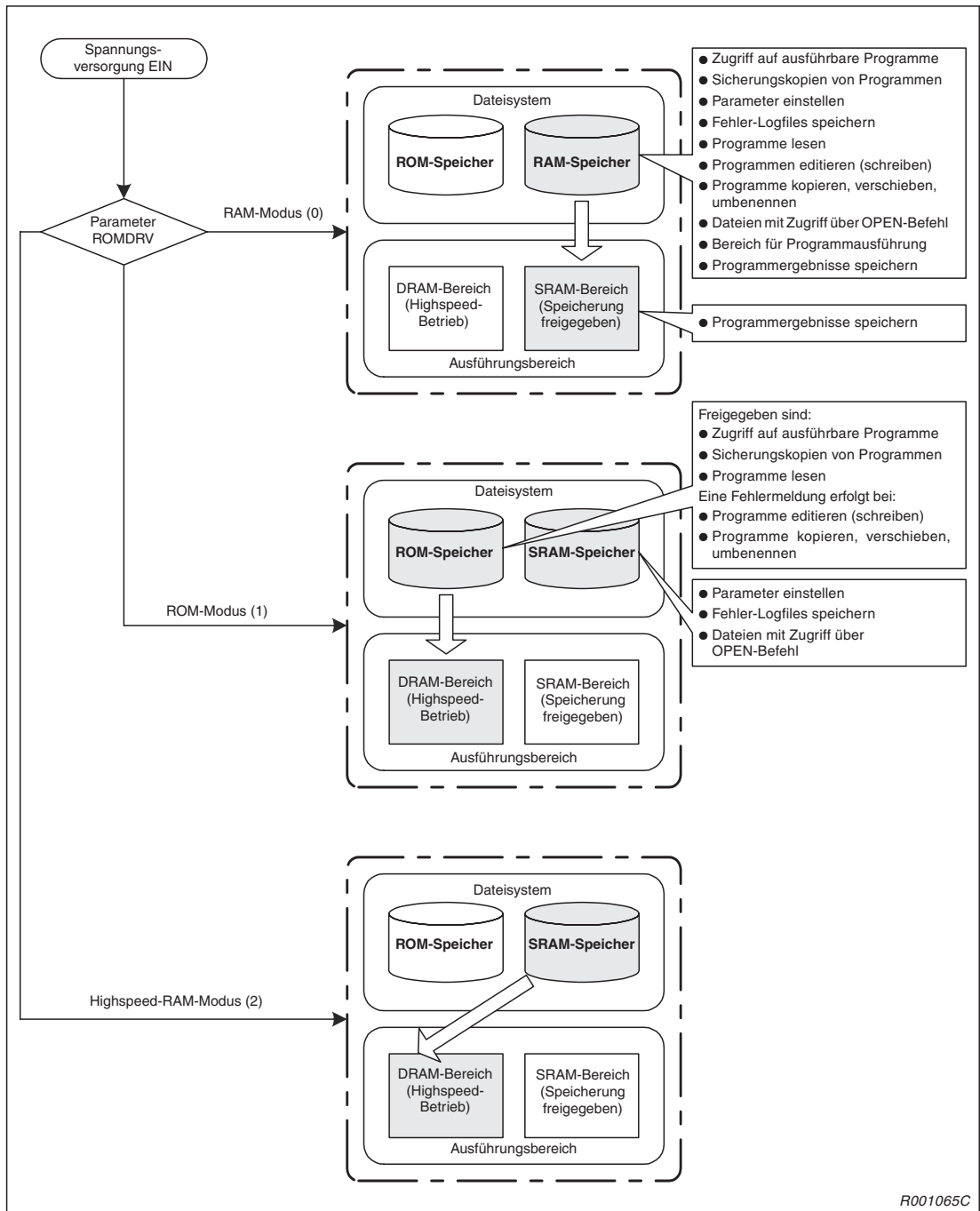


Abb. 9-16: Speicherbereiche

Der DRAM-Bereich dient im ROM- und Highspeed-RAM-Modus als Ausführungsspeicher. Die Verarbeitungsgeschwindigkeit ist etwa 1,2-mal so hoch wie die im SRAM-Bereich, der im einfachen RAM-Modus als Ausführungsspeicher dient. Die Verarbeitungsgeschwindigkeit hängt auch vom Programminhalt ab.

Im ROM- und Highspeed-RAM-Modus können alle Funktionen wie z. B. Programmausführung und Schrittbetrieb wie im RAM-Modus ausgeführt werden. Folgende Einschränkungen sind jedoch zu beachten:

- **Variablen**
Im ROM-Modus und Highspeed-RAM-Modus können Variablen durch ausgeführte Programme geändert werden. Die geänderten Werte werden jedoch nach Ausschalten der Spannungsversorgung nicht gespeichert.

Variablen ^①	RAM-Modus	ROM-Modus	Highspeed-RAM-Modus
Lokale Variablen	Die Variablenwerte bleiben auch nach Ausschalten der Spannungsversorgung gespeichert. ^②	Während der Programmausführung werden die Werte lokaler Variablen gespeichert. Bei Umschaltung der Programme über das Bedienfeld oder externe E/A-Signale und beim Ausschalten der Spannungsversorgung werden die Werte gelöscht. Die Variablenwerte in einem Programm, das über die CALLP-Anweisung aufgerufen wurde, werden beim Rücksprung in das aufrufende Programm gelöscht.	Variablenwerte in einem Programm, das beim Ausschalten der Spannungsversorgung ausgeführt wurde, werden gelöscht. Sie werden bei Auswahl eines Programms und nach Ausführung der CALLP-Anweisung gespeichert.
Programmexterne Variablen	Die Variablenwerte bleiben auch nach Ausschalten der Spannungsversorgung gespeichert.	Die Variablenwerte bleiben bis zum Ausschalten der Spannungsversorgung gespeichert. (Sie werden bei Umschaltung der Programme nicht gelöscht. Änderungen der Variablenwerte gehen beim Ausschalten der Spannungsversorgung verloren.)	Die Variablenwerte bleiben auch nach Ausschalten der Spannungsversorgung gespeichert.
Benutzerdefinierte externe Variablen			Änderungen der Variablenwerte gehen beim Ausschalten der Spannungsversorgung verloren.

Tab. 9-28: Variablen im RAM-, ROM und Highspeed-RAM-Modus

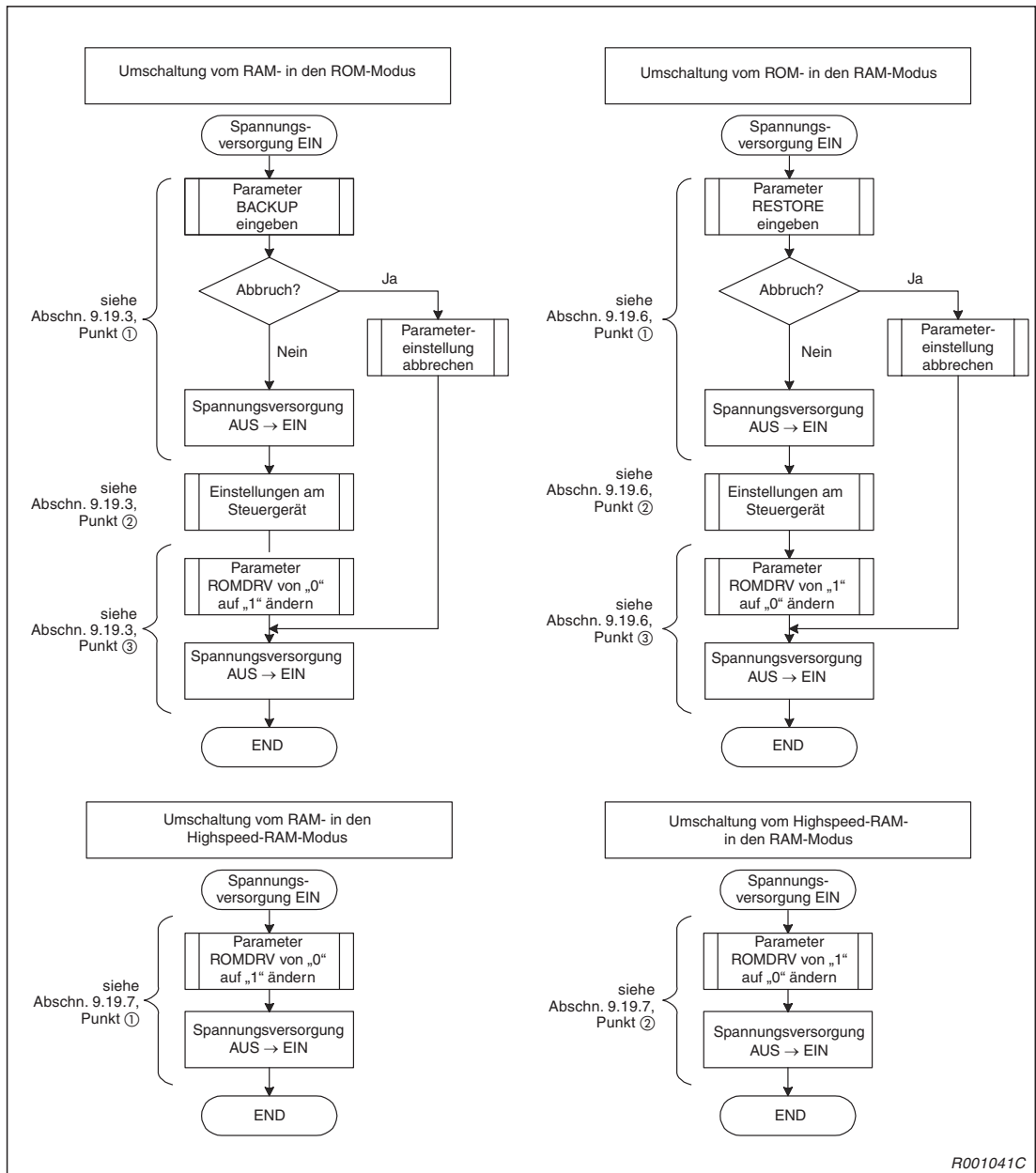
- ^① Die Variablentypen sind: numerische Variablen, Zeichenkettenvariablen, Positions- und Gelenkvariablen.
- ^② Beim Überschreiben eines Programms mit Hilfe der Programmier-Software werden die im Programm verwendeten Werte aller lokalen Variablen gelöscht.

- **Änderung von Variablen während der Programmausführung**
Werden Variablenwerte beim Abbruch eines Programms im ROM-Modus mit Hilfe der Teaching Box im Menü „Monitor-Funktion für Variable“ (siehe auch Abschn. 3.12.3) geändert, ist keine Fortsetzung des Programms möglich. Auch wenn die STOP-LED am Steuergerät leuchtet, startet das Programm bei einer erneuten Ausführung in der ersten Zeile. Achten Sie daher darauf, dass es zu keinen Zusammenstößen mit umliegenden Einrichtungen kommen kann.
Im ROM-Modus können die Variablen nicht über die „Monitor-Funktion für Variable“ geändert werden. Eine Änderung von Variablenwerten in Programmplätzen, die nicht editiert werden, kann über die Programmier-Software erfolgen.
- **Programme**
Die Editierung von Programmen erfolgt im ROM-Bereich. Die im Steuergerät gespeicherten Programme sind geschützt (PROTECT ON) und können im ROM-Modus nicht gelöscht werden. Bei Umschaltung in den RAM-Modus nimmt das Programm wieder den Status an, den es im RAM-Modus vor der Umschaltung hatte. Programme können im ROM-Betrieb gelesen aber nicht geschrieben werden. Dementsprechend kann ein Programm nicht kopiert oder umbenannt werden.

- **Parameter**
Parameter und Fehler-Logfiles werden unabhängig vom Modus immer im RAM-Bereich gespeichert. Eine Änderung des Parameters RLNG (siehe auch Seite 9-26) zur Umschaltung der Programmiermethode im ROM-Betrieb ist nicht möglich. (Prüfen Sie vor Einstellung des Parameters im Technischen Handbuch, ob der von Ihnen verwendete Roboter über diese Funktion verfügt.)
- **Backup**
Im ROM-Modus werden Programme aus dem ROM-Speicher und Parameter und Fehler-Logfiles aus dem RAM-Bereich gesichert.
- **Direkte Befehlsausführung**
Im ROM-Modus können lokale Variable nicht durch direkte Befehlsausführung geschrieben werden.
- **CTN-Funktion**
Im ROM-Modus ist die CTN-Funktion „Programm fortsetzen“ unwirksam, auch wenn sie aktiviert ist.
- **Erweiterungsspeicher**
Wird im ROM-Modus ein Erweiterungsspeicher installiert, erfolgt eine Fehlermeldung. Die Installation eines Erweiterungsspeichers ist nur im RAM-Modus möglich.
- **Einschaltzeit**
Die Einschaltzeit und die verbleibende Lebensdauer der Batterie werden unabhängig von der Umschaltung zwischen ROM- und RAM-Modus gezählt.
- **Betriebsdaten**
Die Überwachung der Betriebsdaten (Programmzähler, Zykluszeit usw.) durch die Programmier-Software werden im ROM-Modus nicht aktualisiert.

9.19.2 Umschaltung zwischen ROM- und RAM-Modus

Folgendes Flussdiagramm zeigt die Vorgehensweise zum Umschalten vom ROM- in den RAM- und den Highspeed-RAM-Modus und umgekehrt.



R001041C

Abb. 9-17: Umschaltung zwischen ROM- und RAM-Modus

9.19.3 Umschaltung in den ROM-Modus

Gehen Sie zur Umschaltung in den ROM-Modus wie in den Schritten ① bis ③ beschrieben vor.

- ① Bereiten Sie den Kopiervorgang zur Übertragung der Daten vom RAM- in den ROM-Speicher vor.
Die Programme, die vor der Umschaltung vom RAM- in den ROM-Modus erstellt wurden, sind im RAM-Speicher des Dateisystems im Steuergerät gespeichert. Nachdem die Programme im ROM-Speicher gelöscht sind, werden die Programme aus dem RAM-Speicher übertragen.

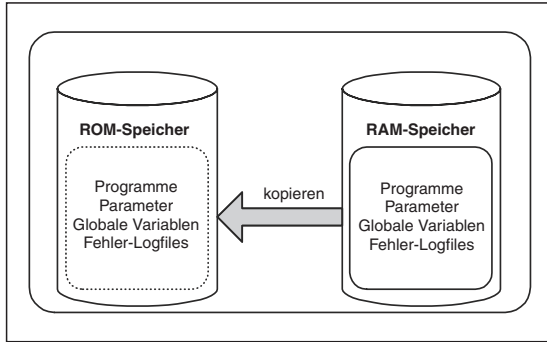


Abb. 9-18:
Dateisystem im Steuergerät

R001042C

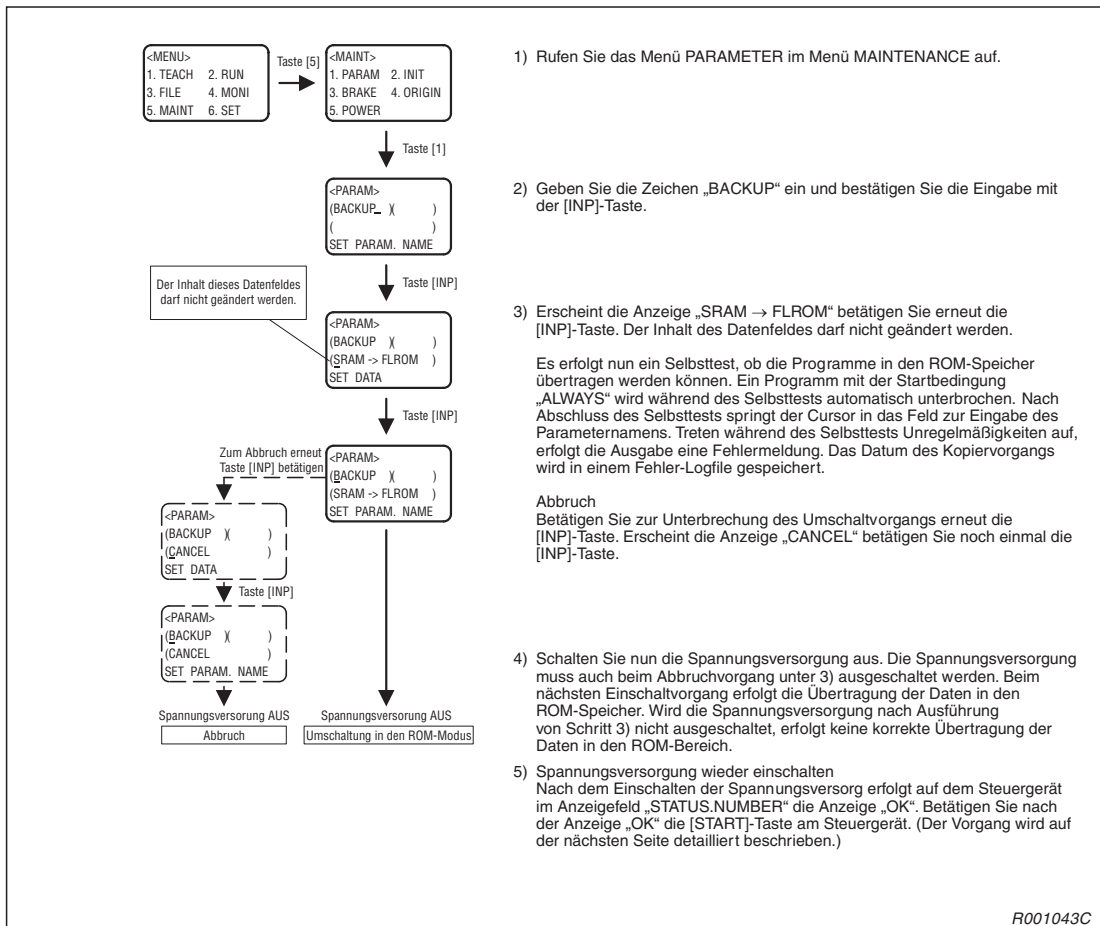


Abb. 9-19: Umschaltung in den ROM-Modus

② Ausführung des Kopiervorgangs über das Steuergerät

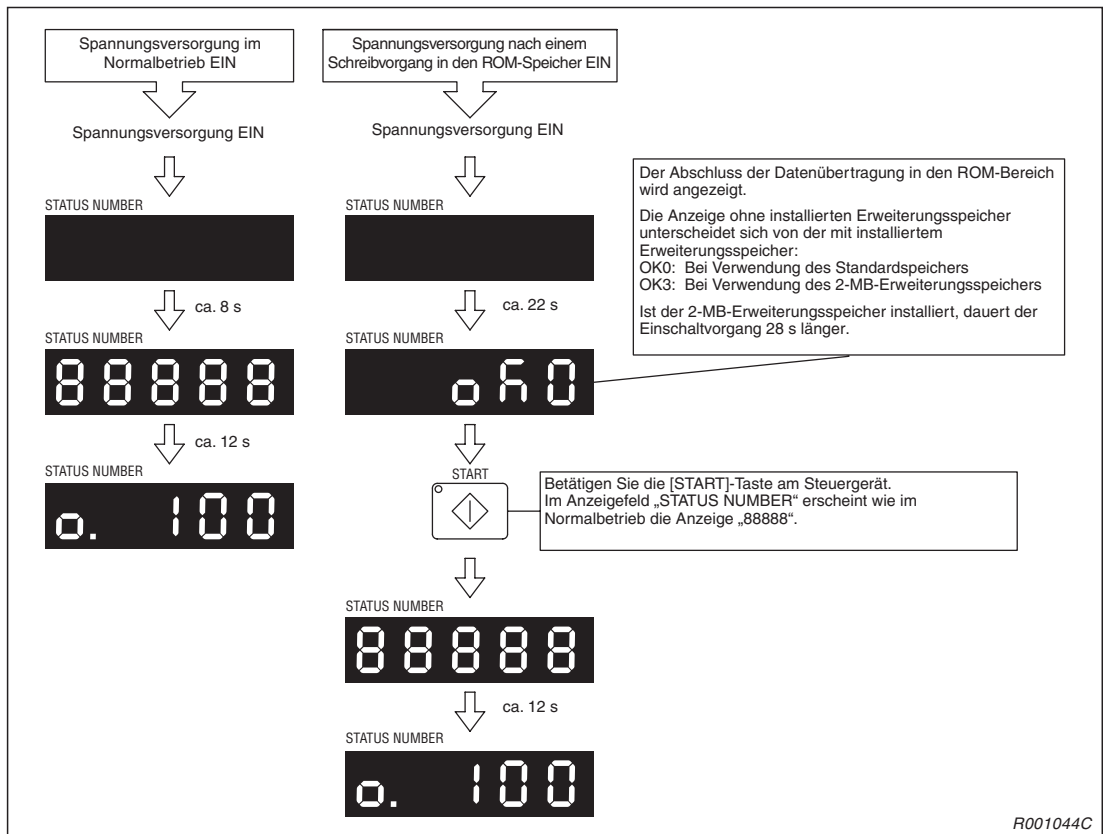


Abb. 9-20: Ausführung des Kopiervorgangs

HINWEIS

Die Dauer des Einschaltvorgangs in der Abbildung oben gilt für die Software-Version H7. Sie ist von der Software-Version und dem verwendeten Speicher abhängig.



ACHTUNG:

Erfolgt die Umschaltung vom RAM- in den ROM-Modus, ohne dass ein Programm in den ROM-Speicher kopiert wird, ist entweder kein Programm zur Ausführung im Speicher oder ein Programm, dessen Änderungen nicht geprüft sind wird ausgeführt. Führen Sie daher unbedingt den oben beschriebenen Kopiervorgang aus.

③ Änderung des Parameterwertes und Umschaltung in den ROM-Modus

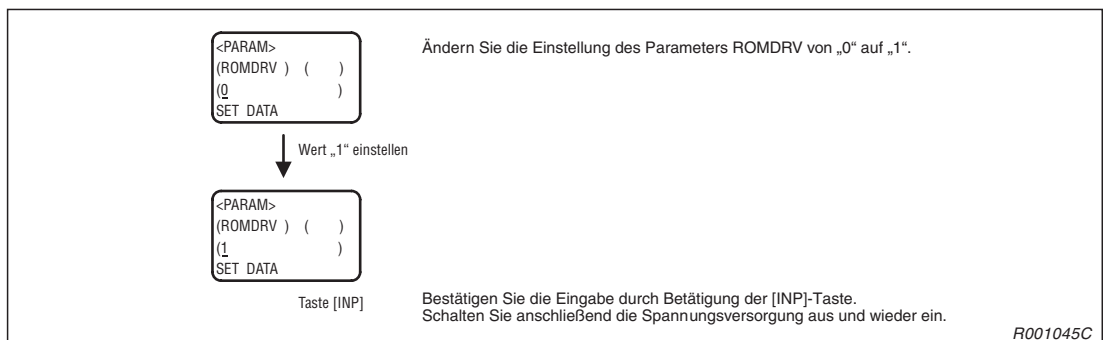


Abb. 9-21: Parameter ROMDRV einstellen

9.19.4 Anzeigen im ROM-Modus

Im ROM-Modus leuchtet in der rechten unteren Ecke der Anzeige „STATUS NUMBER“ ein Punkt.

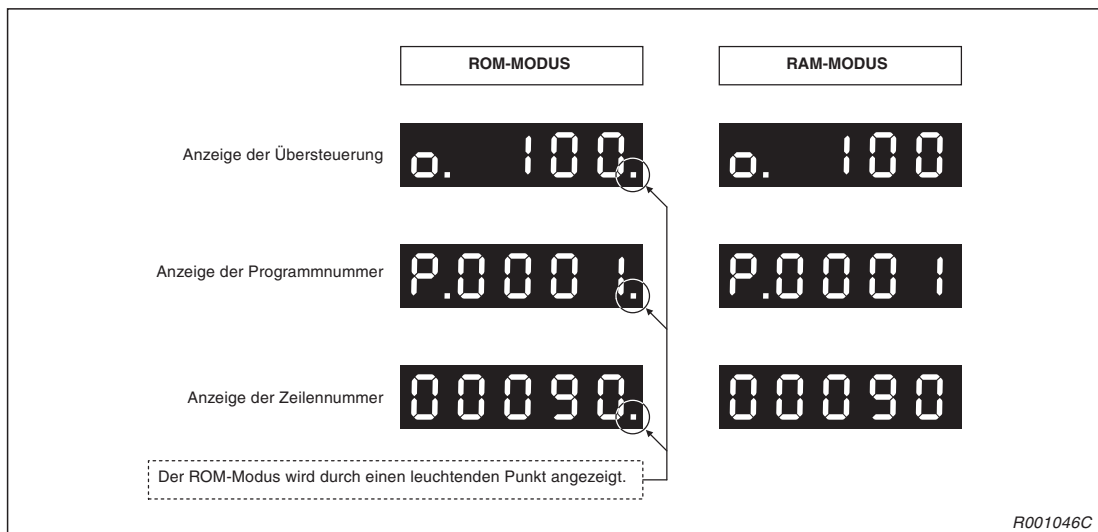


Abb. 9-22: Anzeigen im ROM- und RAM-Modus

9.19.5 Programmeditierung im ROM-Modus

Während des Betriebs im ROM-Modus können Programme im Steuergerät gelesen werden. Bei Überschreiben eines Programmes erfolgt eine Fehlermeldung. Zur Programmeditierung ist zuerst der ROM-Modus zu unterbrechen (Änderung des Parameters ROMDRV von „1“ auf „0“ und Aus- und Wiedereinschalten der Spannungsversorgung). Dann kann das Programm editiert werden. Soll nach Abschluss der Programmeditierung zurück in den ROM-Modus geschaltet werden, stellen Sie sicher, dass Sie die Programme wieder in den ROM-Speicher kopieren.

9.19.6 Umschaltung in den RAM-Modus

Gehen Sie zur Umschaltung in den RAM-Modus wie in den Schritten ① bis ③ beschrieben vor.

- ① Bereiten Sie den Kopiervorgang zur Übertragung der Daten vom ROM- in den RAM-Speicher vor.

Die Programme und Parameter, die bei der Umschaltung vom RAM- in den ROM-Modus in den ROM-Speicher kopiert wurden, werden zurück in den RAM-Speicher übertragen. Werden die Daten (Programme, Parameter, globale Variablen und Fehler-Logfiles) im RAM-Speicher gelöscht, erfolgt die Wiederherstellung aus dem ROM-Speicher.

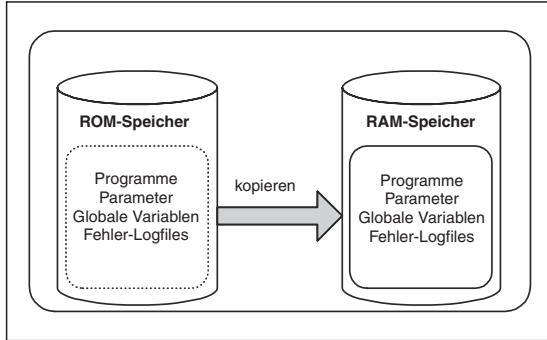


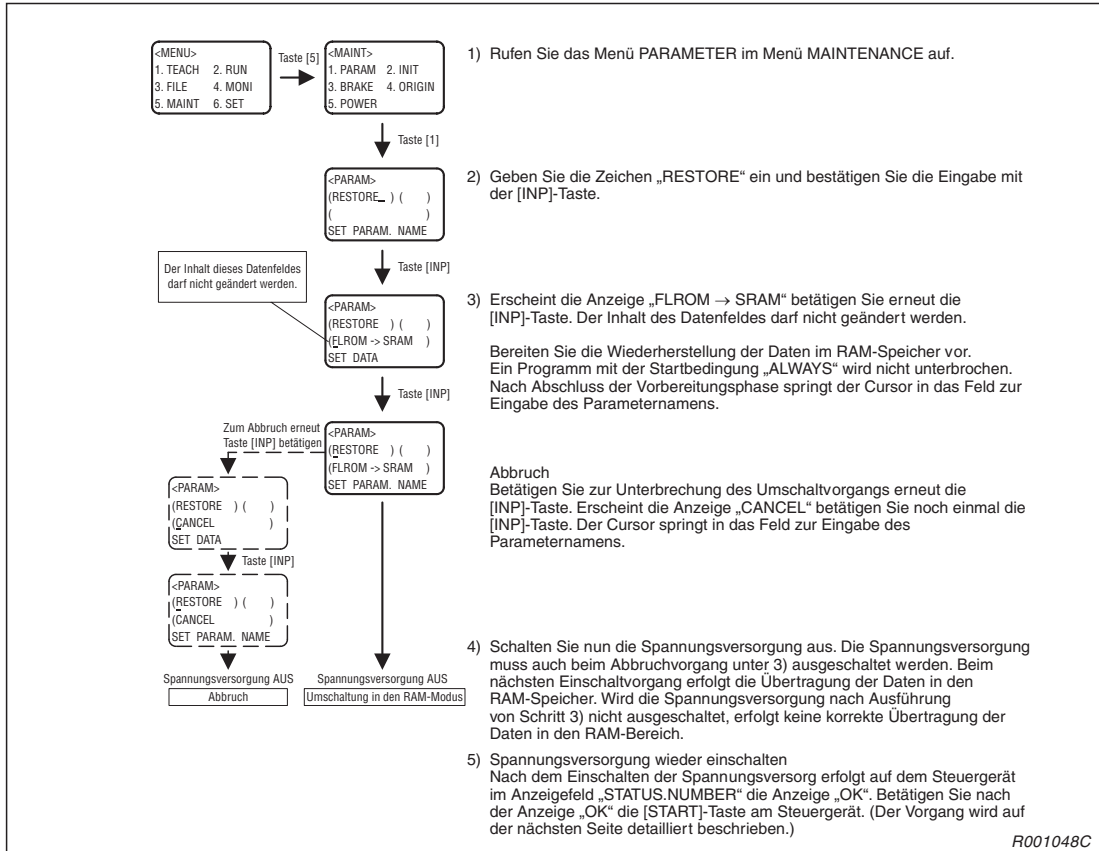
Abb. 9-23:
Dateisystem im Steuergerät

R001047C



ACHTUNG:

Bei Wiederherstellung von Daten des RAM-Speichers im ROM-Bereich werden alle im ROM-Modus geänderten Werte der Parameter, globalen Variablen und Fehler-Logfiles überschrieben. Parameter, die im ROM-Modus geändert wurden, müssen vor der Umschaltung in den RAM-Modus erneut eingestellt werden.



R001048C

Abb. 9-24: Umschaltung in den RAM-Modus

② Wiederherstellung der Daten über das Steuergerät

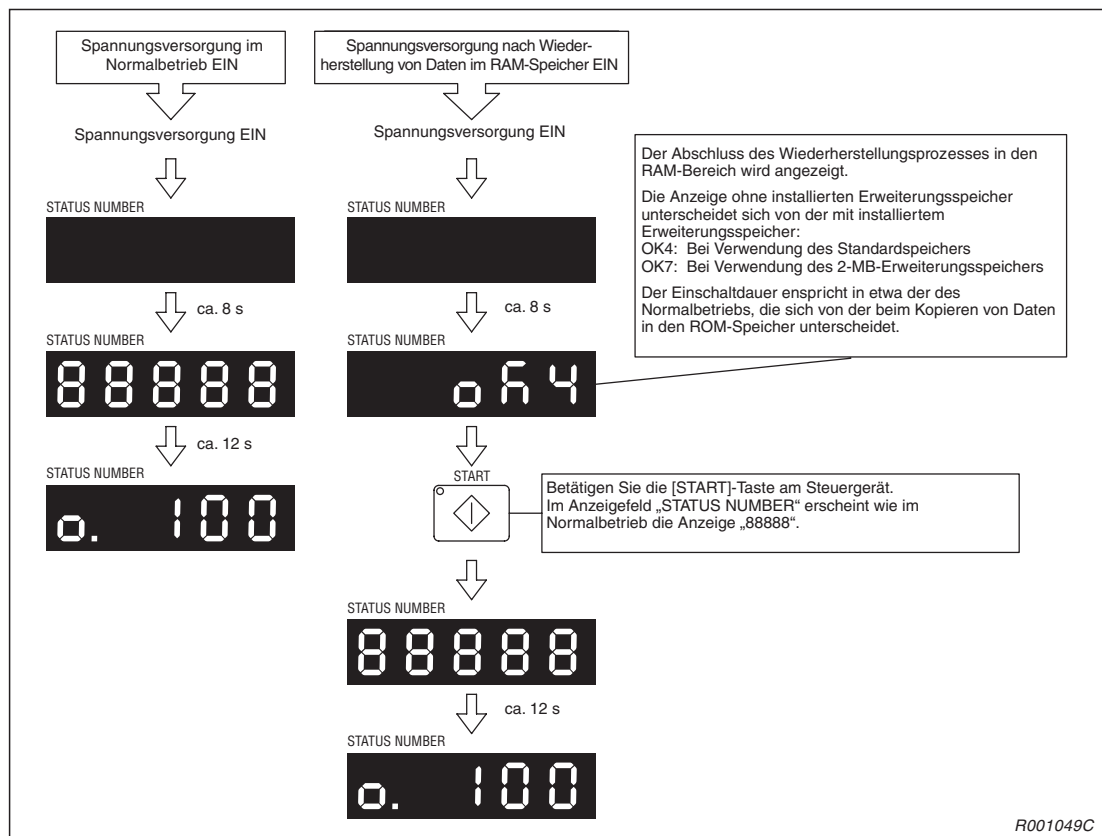


Abb. 9-25: Wiederherstellung der Daten

③ Änderung des Parameterwertes und Umschaltung in den RAM-Modus

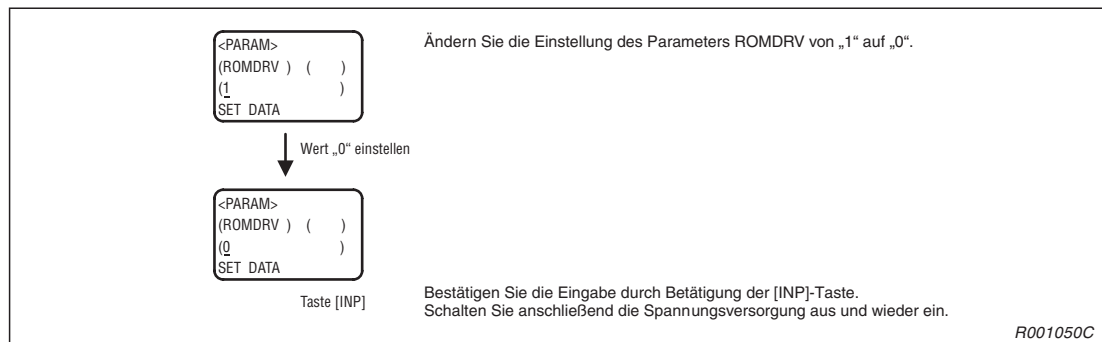


Abb. 9-26: Parameter ROMDRV einstellen

9.19.7 Umschaltung in den Highspeed-RAM-Modus

Gehen Sie zur Umschaltung in den Highspeed-RAM-Modus und zurück in den RAM-Modus wie folgt vor.

- ① Änderung des Parameterwertes und Umschaltung in den Highspeed-RAM-Modus (DRAM)

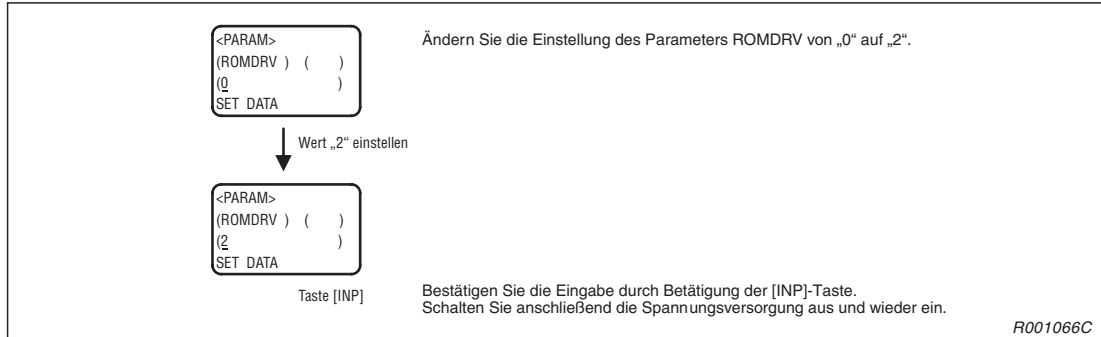


Abb. 9-27: Parameter ROMDRV einstellen

- ② Änderung des Parameterwertes und Umschaltung zurück in den RAM-Modus

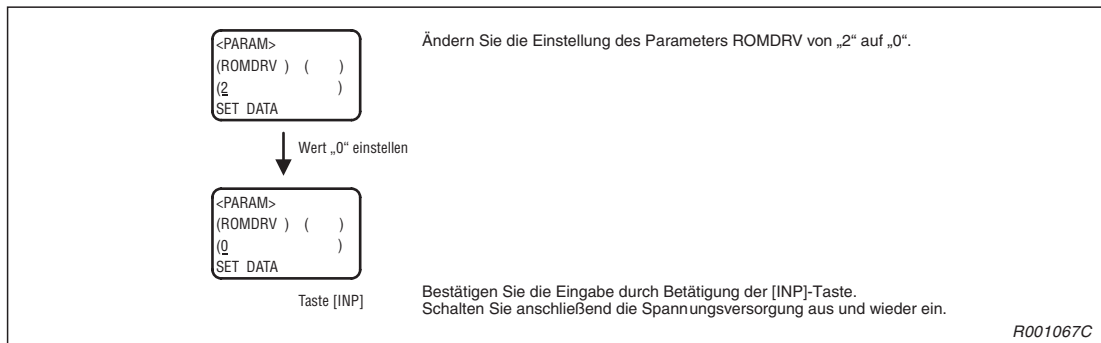


Abb. 9-28: Parameter ROMDRV einstellen

9.20 Warmlaufbetrieb

9.20.1 Funktionsbeschreibung

Das Beschleunigungs- und Bremsverhalten sowie das Servosystem der MITSUBISHI-Roboter sind so ausgelegt, dass der Roboter in Umgebungen mit normalen Temperaturbedingungen die bestmöglichen Betriebswerte erreicht. Niedrige Temperaturen oder längere Betriebspausen beeinflussen die Viskosität der verwendeten Schmiermittel. Dadurch kann die Präzision abnehmen und es können Servofehler auftreten, die zu Positionsabweichungen führen. In einem solchen Fall empfiehlt es sich, den Roboter erst nach einer Anwärmphase mit niedriger Geschwindigkeit in den endgültigen, zeitoptimierten Produktionsprozess zu integrieren. Dazu wäre jedoch ein separates Programm für den Betrieb in der Anwärmphase erforderlich.

Im Warmlaufbetrieb wird der Roboter direkt nach dem Einschalten des Steuergerätes mit verminderter Geschwindigkeit verfahren. Anschließend erfolgt eine kontinuierliche Anhebung der Geschwindigkeit, die nach Ablauf der für den Warmlaufbetrieb festgelegten Zeit ihren Endwert erreicht. Der Warmlaufbetrieb erlaubt die Ausführung einer Anwärmphase, ohne dass ein separates Programm erstellt werden muss. Treten beim Einsatz des Roboters in Umgebungen mit niedrigen Temperaturen oder nach längeren Betriebspausen Positionsabweichungen auf, aktivieren Sie den Warmlaufbetrieb.

Die Funktion steht ab der Software-Version J8 des Steuergerätes zur Verfügung.

9.20.2 Aktivierung des Warmlaufbetriebs

Setzen Sie den Parameter WUPENA auf „1“, um den Warmlaufbetrieb freizugeben. Schalten Sie anschließend die Spannungsversorgung des Steuergerätes aus und wieder ein.

HINWEIS

Um die Funktion des Warmlaufbetriebs mit einem Roboter zu verwenden, der nicht zur RV-S-Serie gehört, muss – anders als bei der Aktivierung über den Parameter WUPENA – mit Hilfe des Parameters WUPAXIS eine Achse für den Warmlaufbetrieb festgelegt werden. Eine detaillierte Beschreibung finden Sie auf Seite 9-73.

9.20.3 Aktivierter Warmlaufbetrieb

Ist der Warmlaufbetrieb freigegeben, wird er durch Ein- und Wiederausschalten der Spannungsversorgung des Steuergerätes aktiviert. (Die Geschwindigkeit wird automatisch reduziert.)

Im Warmlaufbetrieb bewegt der Roboter sich mit einer Geschwindigkeit, die kleiner als die festgelegte Betriebsgeschwindigkeit ist. Anschließend steigt die Geschwindigkeit wieder, bis sie nach Ablauf der für die Achse festgelegten Zeit für den Warmlaufbetrieb ihren Endwert erreicht.

Die Steilheit der Geschwindigkeitsänderung wird über die Übersteuerung im Warmlaufbetrieb festgelegt. Bei einem Wert von 100 % entspricht die Geschwindigkeit des Roboters der festgelegten Betriebsgeschwindigkeit. In der werksseitigen Parametereinstellung ändert sich die Übersteuerung in der für die Achse festgelegten Zeit für den Warmlaufbetrieb (siehe folgende Abbildung).

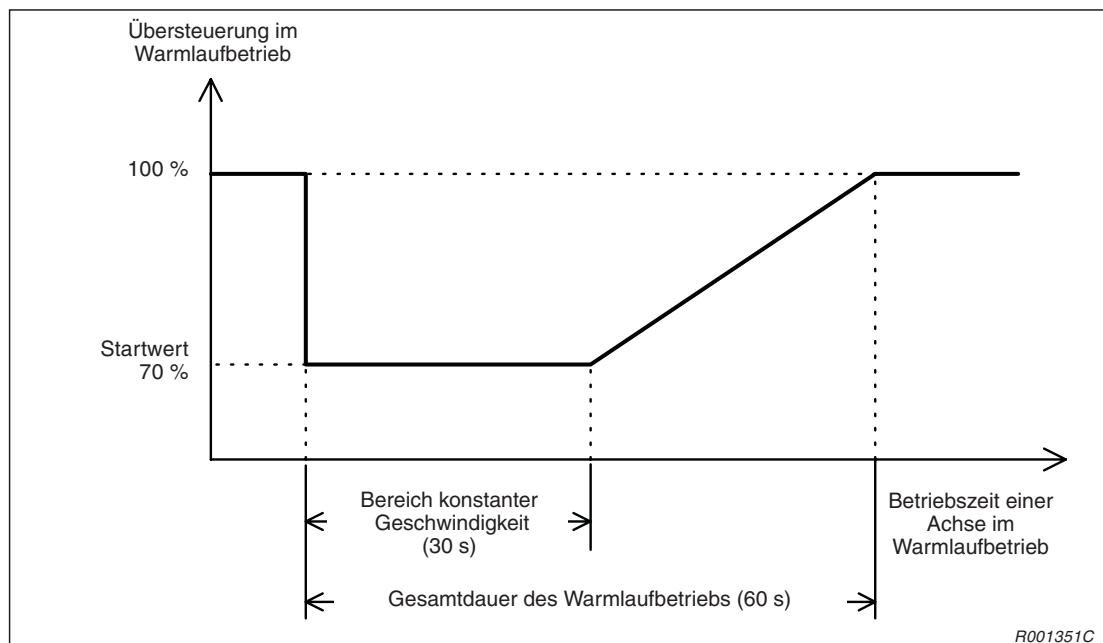


Abb. 9-29: Änderungen während des Warmlaufbetriebs



ACHTUNG:

- Ist der [MODE]-Umschalter des Steuergerätes auf „TEACH“ eingestellt, im JOG-Betrieb oder im externen Echtzeit-Steuerungs-Betrieb (MXT-Befehl) bewegt sich der Roboter auch bei aktiviertem Warmlaufbetrieb nicht mit der reduzierten Geschwindigkeit sondern mit der festgelegten Betriebsgeschwindigkeit.
- Da sich der Roboter im Warmlaufbetrieb mit einer kleineren Geschwindigkeit als der festgelegten Betriebsgeschwindigkeit bewegt, ist eine Synchronisierung mit umliegenden Einheiten vorzusehen.
- Bei niedriger Betriebsbeanspruchung der für den Warmlaufbetrieb festgelegten Achse, können auch bei aktiviertem Warmlaufbetrieb Servofehler wie z. B. Positionsabweichungen auftreten. Ändern Sie in diesem Fall das Programm und vermindern Sie sowohl die Beschleunigungs-/Bremszeiten als auch die Geschwindigkeit.

Der Warmlaufbetrieb wird bei Anzeige des Übersteuerungswerts auf der STATUS.NUMBER-Anzeige des Steuergeräts durch einen Unterstrich (_) an der zweiten Stelle gekennzeichnet.

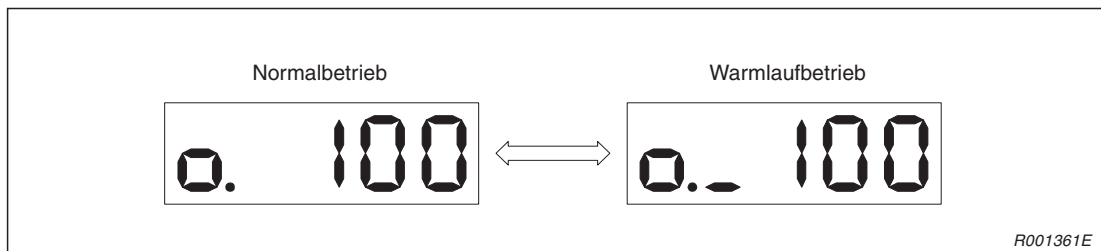


Abb. 9-30: STATUS.NUMBER-Anzeige im Warmlaufbetrieb

Bei Abbruch des Warmlaufbetriebs wird die für den Warmlaufbetrieb festgelegte Achse wieder mit der Betriebsgeschwindigkeit verfahren. Bei niedrigen Temperaturen kühlt sich der Gelenkbereich der Achse nach dem Abbruch des Warmlaufbetriebs in einer Betriebspause wieder ab. Nach längeren Betriebspausen der für den Warmlaufbetrieb festgelegten Achse (Werkseinstellung: 60 min) sollte der Warmlaufbetrieb erneut aktiviert und der Roboter mit einer reduzierten Geschwindigkeit verfahren werden.

HINWEISE

Dauert der ausgeschaltete Zustand beim Aus- und Wiedereinschalten des Steuergerätes nur kurze Zeit an, kühlt das Gelenk des Roboters auch nur wenig ab. Daher wird der Roboter nach Beendigung des Warmlaufbetriebs und einer kurzen Ausschaltzeit mit der Betriebsgeschwindigkeit und nicht im Warmlaufbetrieb verfahren.

Die Festlegung der Achse, die im Warmlaufbetrieb verfahren werden soll, erfolgt im Parameter WUPAXIS.

9.20.4 Parameter, spezielle Ein- und Ausgänge und Statusvariablen im Warmlaufbetrieb

Folgende Parameter, spezielle Ein- und Ausgänge und Statusvariablen sind im Warmlaufbetrieb zusätzlich gültig:

Parameter	Beschreibung
WUPENA	Freigabe des Warmlaufbetriebs 0: gesperrt/1: freigegeben
WUPAXIS	Auswahl der Achse für den Warmlaufbetrieb durch Ein- und Ausschalten der Bits im Hexadezimalcode (J1, J2 ... mit dem niedrigsten Bit beginnend) Bit EIN: Achse auswählen / Bit AUS: Achse nicht auswählen
WUPTIME	Dauer des Warmlaufbetriebs in Minuten Legen Sie die Gesamtdauer im ersten Element und die Wiederholsschwelle im zweiten Element ein. Gesamtdauer: Legen Sie die Dauer fest, in der die Achse im Warmlaufbetrieb mit reduzierter Geschwindigkeit verfahren wird. (Einstellbereich 0 bis 60) Wiederholsschwelle: Legen Sie die Zeit fest, nachdem nach Beendigung eines Warmlaufbetriebs und einer kontinuierlichen Betriebspause einer Achse erneut ein Warmlaufbetrieb aktiviert wird (Einstellbereich: 1 bis 1440)
WUPOVRD	Einstellung der Geschwindigkeit im Warmlaufbetrieb Legen Sie den Startwert im ersten Element und den Bereich der konstanten Geschwindigkeit im zweiten Element in Prozent fest. Startwert: Startwert der Geschwindigkeitsübersteuerung im Warmlaufbetrieb (Einstellbereich 50 bis 100) Faktor für den Bereich konstanter Geschwindigkeit: Stellen Sie den Bereich der konstanten Geschwindigkeit in Bezug zur Gesamtdauer des Warmlaufbetriebs ein (Einstellbereich 0 bis 50)

Tab. 9-29: Parameter für den Warmlaufbetrieb

Parameter	Ein-/Ausgang	Beschreibung
MnWUPENA (n = bis 3) (Betriebsrecht erforderlich)	Eingang	Freigabe des Warmlaufbetriebs eines Mechanismus (n: Mechanismusnummer)
	Ausgang	Ausgabe bei freigegebenem Warmlaufbetrieb (n: Mechanismusnummer)
MnWUPMD (n = 1 bis 3)	Ausgang	Ausgabe bei aktiviertem Warmlaufbetrieb, in dem der Roboter mit reduzierter Geschwindigkeit verfahren wird (n: Mechanismusnummer)

Tab. 9-30: Spezielle Ein- und Ausgänge im Warmlaufbetrieb

Parameter	Beschreibung
M_WUPOV	Enthält den Übersteuerungswert im Warmlaufbetrieb, der den Geschwindigkeits-sollwert auf die Geschwindigkeit reduziert, mit der der Roboter im Warmlaufbetrieb verfahren wird.
M_WUPRT	Enthält die Restzeit, in der eine Achse im Warmlaufbetrieb verfahren wird, bis der Warmlaufbetrieb beendet ist.
M_WUPST	Enthält die Zeit bis nach Beendigung eines Warmlaufbetriebs ein erneuter Warmlaufbetrieb aktiviert wird.

Tab. 9-31: Statusvariablen für den Warmlaufbetrieb

9.20.5 Ausführung des Warmlaufbetriebs

Der Warmlaufbetrieb kann über Parameter oder ein spezielles Eingangssignal aktiviert werden.

- Aktivierung über Parameter

Aktivieren Sie den Warmlaufbetrieb, indem Sie den Parameter WUPENA auf „1“ setzen. Schalten Sie anschließend das Steuergerät aus und wieder ein. In folgenden Fällen wird der Warmlaufbetrieb nicht aktiv, auch wenn er über den Parameter WUPENA freigegeben ist:

- Wenn der Parameter WUPAXIS auf „0“ gesetzt ist. (Es ist keine Achse für den Warmlaufbetrieb festgelegt.)
- Wenn das erste Element des Parameters WUPTIME auf „0“ gesetzt ist. (Die Gesamtdauer des Warmlaufbetriebs ist auf „0“ gesetzt.)
- Wenn das erste Element des Parameters WUPOVRD auf „100“ gesetzt ist. (Die Geschwindigkeit wird im Warmlaufbetrieb nicht reduziert.)

Setzen Sie die Parameter zur Aktivierung des Warmlaufbetriebs auf die entsprechenden Werte.

HINWEIS

Bei allen Robotern, die nicht zur RV-S-Serie gehören, ist der Parameter WUPAXIS werksseitig auf „0“ gesetzt. Zur Ausführung der Funktion, ist eine Achse für den Warmlaufbetrieb festzulegen. (Zielachse, die im Warmlaufbetrieb gesteuert wird; z. B. eine Achse, die bei niedrigen Temperaturen zu Positionsabweichungen führt.)

- Aktivierung über spezielles Eingangssignal

Wird der Warmlaufbetrieb über das spezielle Eingangssignal MnWUPENA (n = 1 bis 3: Mechanismusnummer) freigegeben, muss das Steuergerät nicht aus- und wieder eingeschaltet werden. Das Ausgangssignal MnWUPENA (n = 1 bis 3: Mechanismusnummer) zeigt an, ob der Warmlaufbetrieb freigegeben ist.

HINWEISE

Um den Warmlaufbetrieb über das spezielle Eingangssignal freizugeben, müssen die zuvor beschriebenen Parameter entsprechend eingestellt werden.

Das spezielle Eingangssignal benötigt die Freigabe der Betriebsrechte für externe Signale. Während des Betriebs oder im JOG-Betrieb wird eine Eingabe des Signals nicht akzeptiert.

Der über das Eingangssignal festgelegte Status „freigegeben/gesperrt“ wird auch nach Entzug der Betriebsrechte weiter aufrechterhalten.

9.20.6 Wenn der Warmlaufbetrieb freigegeben ist

Ist der Warmlaufbetrieb freigegeben, wird er durch Aus- und Einschalten der Spannungsversorgung des Steuergerätes aktiviert.

Im aktivierten Warmlaufbetrieb wird der Roboter mit einer reduzierten Geschwindigkeit verfahren, die durch den Übersteuerungswert für den Warmlaufbetrieb festgelegt ist. Bis zum Ablauf der Gesamtdauer für den Warmlaufbetrieb erfolgt eine kontinuierliche Erhöhung der Geschwindigkeit bis auf den Wert der Betriebsgeschwindigkeit. Nach Beendigung des Warmlaufbetriebs wird der Roboter wieder mit der Betriebsgeschwindigkeit verfahren.

Zustand nach Einschalten der Spannungsversorgung des Steuergerätes

Ist der Warmlaufbetrieb freigegeben, wird er durch Aus- und Einschalten der Spannungsversorgung des Steuergerätes aktiviert.

Wird das Steuergerät nach Beendigung des Warmlaufbetriebs nur kurz aus- und dann wieder eingeschaltet, startet der Roboter im Normalbetrieb und nicht im Warmlaufbetrieb, da das Gelenk des Roboters während der kurzen ausgeschalteten Phase nicht stark abkühlen konnte. Genauer gesagt, startet der Roboter im Normalbetrieb, wenn folgende Bedingung erfüllt ist:

Nach Beendigung des Warmlaufbetriebs startet der Roboter dann im Normalbetrieb, wenn die Betriebspause der festgelegten Achse bis zum Einschalten des Steuergerätes kürzer als die im zweiten Element des Parameters WUPTIME eingestellten Zeit ist (Wiederholtschwelle).

HINWEIS

Erfolgt die Freigabe des Warmlaufbetriebs über das spezielle Eingangssignal MnWUPENA (n = 1 bis 3: Mechanismusnummer), ist der Warmlaufbetrieb immer aktiviert.

Zustand des Warmlaufbetriebs

Ob sich der Roboter im Warmlauf- oder Normalbetrieb befindet, kann mit den folgenden drei Methoden überprüft werden:

- Abfrage über STATUS.NUMBER-Anzeige auf dem Steuergerät

Der aktuelle Status wird auf der STATUS.NUMBER-Anzeige des Steuergerätes parallel zum Übersteuerungswert angezeigt. Der Warmlaufbetrieb ist durch einen Unterstrich (_) an der zweiten Stelle der Anzeige gekennzeichnet.

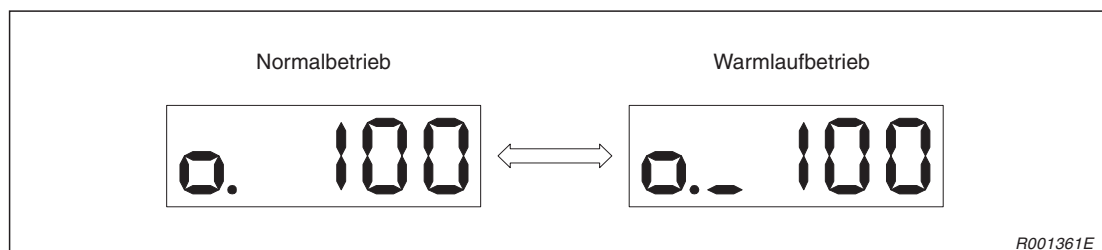


Abb. 9-31: STATUS.NUMBER-Anzeige im Warmlaufbetrieb

- Abfrage über Statusvariable

Der aktuelle Status kann über den Wert der Statusvariablen M_WUPOV abgefragt werden (Übersteuerungswert). Im Normalbetrieb ist der Wert der Statusvariablen 100 %, im Warmlaufbetrieb liegt er darunter.

- Abfrage über spezielles Ausgangssignal

Im Warmlaufbetrieb wird das spezielle Ausgangssignal MnWUPMD (n = 1 bis 3: Mechanismusnummer) ausgegeben.

9.20.7 Umschaltung zwischen Normal- und Warmlaufbetrieb

Überschreitet die Betriebszeit einer Achse im Warmlaufbetrieb die Gesamtdauer für den Warmlaufbetrieb, wird der Warmlaufbetrieb beendet und es erfolgt eine Umschaltung auf den Normalbetrieb. Kühlt die Achse in einer längeren Betriebspause, deren Dauer den eingestellten Schwellwert für einen erneuten Warmlaufbetrieb überschreitet, wieder ab, so erfolgt eine Umschaltung vom Normal- in den Warmlaufbetrieb.

Beendigung des Warmlaufbetriebs

Überschreitet die Betriebszeit einer Achse im Warmlaufbetrieb die Gesamtdauer für den Warmlaufbetrieb, wird der Warmlaufbetrieb beendet und es erfolgt eine Umschaltung auf den Normalbetrieb. Legen Sie die Gesamtdauer für den Warmlaufbetrieb im ersten Element des Parameters WUPTIME fest. (Die Werkseinstellung ist 1 min.)

Werden mehrere Achsen im Warmlaufbetrieb verfahren, erfolgt die Beendigung des Warmlaufbetriebs, wenn die Betriebsdauer aller festgelegten Achsen die eingestellte Gesamtbetriebsdauer für den Warmlaufbetrieb überschreitet. Mit Hilfe der Statusvariablen M_WUPRT kann die Dauer einer gesteuerten Achse überprüft werden, in der die Achse noch im Warmlaufbetrieb verfahren wird.

Umschaltung vom Normalbetrieb in den Warmmeldungsbetrieb

Überschreitet die Betriebspause einer für den Warmlaufbetrieb festgelegten Achse den für einen erneuten Warmlaufbetrieb eingestellten Schwellwert, erfolgt eine Umschaltung vom Normalbetrieb in den Warmlaufbetrieb. Legen Sie den Schwellwert für einen erneuten Warmlaufbetrieb im zweiten Element des Parameters WUPTIME fest. (Die Werkseinstellung ist 6 min.)

Sind mehrere Achsen für den Warmlaufbetrieb festgelegt, startet der Warmlaufbetrieb, wenn eine der Achsen den Schwellwert für einen erneuten Warmlaufbetrieb überschreitet. Mit Hilfe der Statusvariablen M_WUPST kann die Dauer einer gestoppten Achse bis zur Umschaltung in den Warmlaufbetrieb überprüft werden.

HINWEIS

Wird eine für den Warmlaufbetrieb festgelegte Achse während des Roboterbetriebs nicht verfahren, so wird die Achse als gestoppt bewertet.

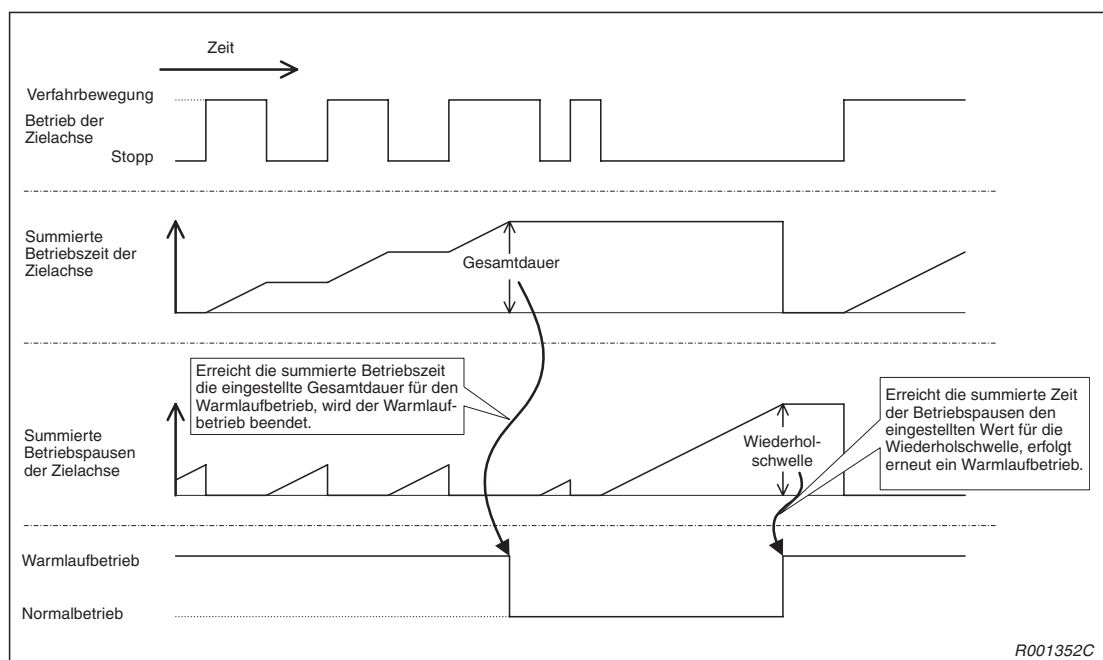


Abb. 9-32: Beispiel für eine Umschaltung vom Normal- in den Warmlaufbetrieb

Übersteuerungswert für den Warmlaufbetrieb

Die reduzierte Geschwindigkeit im Warmlaufbetrieb wird durch den Übersteuerungswert für den Warmlaufbetrieb bestimmt. Der Übersteuerungswert ändert sich in Abhängigkeit der Betriebsdauer einer Achse und beeinflusst direkt die Verfahrgeschwindigkeit des Roboters.

Legen Sie mit Hilfe des Parameters WUPOVRD den Startwert der Übersteuerung für den Warmlaufbetrieb und die Zeit in Bezug zur Gesamtdauer des Warmlaufbetriebs fest, in der die Übersteuerung konstant bleiben soll. (Die Werkseinstellungen sind 70 % für den Startwert und 50 % (= 30 s) für den Bereich der konstanten Geschwindigkeit.)

Mit Hilfe der Statusvariablen M_WUPOV kann die Übersteuerung überprüft werden.

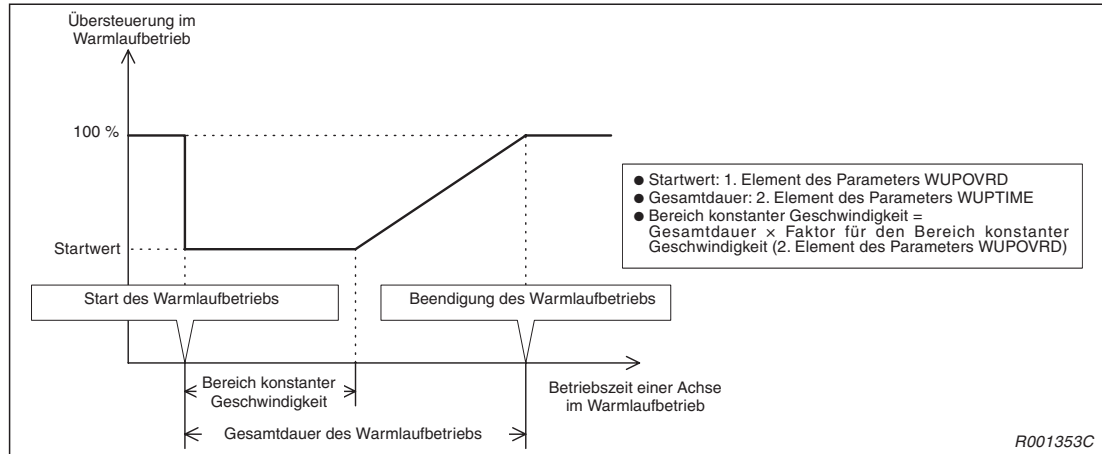


Abb. 9-33: Übersteuerung im Warmlaufbetrieb

Die aktuelle Übersteuerung im Warmlaufbetrieb ergibt sich wie folgt:

$$\text{Gelenk-Interpolation} = \text{Übersteuerungswert der T/B oder des Steuergeräts} \times \text{Einstellwert des OVRD-Befehls} \times \text{Einstellwert des JOVRD-Befehls} \times \text{Übersteuerungswert für den Warmlaufbetrieb}$$

$$\text{Linear-Interpolation} = \text{Übersteuerungswert der T/B oder des Steuergeräts} \times \text{Einstellwert des OVRD-Befehls} \times \text{Einstellwert des SPD-Befehls} \times \text{Übersteuerungswert für den Warmlaufbetrieb}$$

HINWEISE

Ist der [MODE]-Umschalter des Steuergeräts auf „TEACH“ eingestellt, im JOG-Betrieb oder im externen Echtzeit-Steuerungs-Betrieb (MXT-Befehl) ist der Übersteuerungswert für den Warmlaufbetrieb unwirksam und der Roboter wird auch bei aktiviertem Warmlaufbetrieb nicht mit der reduzierten Geschwindigkeit verfahren, sondern mit der festgelegten Betriebsgeschwindigkeit.

Da sich der Roboter im Warmlaufbetrieb mit einer kleineren Geschwindigkeit als der festgelegten Betriebsgeschwindigkeit bewegt, ist eine Synchronisierung mit umliegenden Einheiten vorzusehen.

Sind mehrere Achsen für den Warmlaufbetrieb festgelegt, erfolgt die Festlegung der Übersteuerung durch die Achse mit der kleinsten Betriebsdauer im Warmlaufbetrieb. Wird eine bestimmte, für den Warmlaufbetrieb festgelegte, Achse nicht verfahren und der Wert der Statusvariablen M_WUPRT ändert sich nicht, ändert sich – unabhängig von den anderen Achsen, die verfahren werden – auch der Wert der Übersteuerung nicht. In Abhängigkeit davon, ob eine Achse gesteuert oder gestoppt ist, kann sich der Übersteuerungswert wieder auf den Startwert ändern, bevor er 100 % erreicht hat. Übersteigt z. B. der Übersteuerungswert für den Warmlaufbetrieb den Startwert und eine bestimmte Achse schaltet vom Normalbetrieb in den Warmlaufbetrieb, so ist diese Achse die mit der geringsten Betriebsdauer (Betriebsdauer = 0 s). Der Übersteuerungswert für den Warmlaufbetrieb wird wieder auf den Startwert gesetzt.

9.20.8 Alarmer im Warmlaufbetrieb

Trotz Warmlaufbetrieb treten Positionsabweichungen auf

- Tritt der Fehler auf, wenn der Startwert im Warmlaufbetrieb wirksam ist, verkleinern Sie den Startwert (1. Element des Parameters WUPOVRD).
- Tritt der Fehler auf, wenn die Übersteuerung im Warmlaufbetrieb auf 100 % ansteigt, ist eventuell die Gesamtbetriebszeit für den Warmlaufbetrieb oder der Bereich der konstanten Geschwindigkeit zu klein. Vergrößern Sie in diesem Fall das erste Element (Gesamtbetriebszeit für den Warmlaufbetrieb) oder das zweite Element (Faktor für den Bereich konstanter Geschwindigkeit) des Parameters WUPOVRD.
- Kann der Fehler durch die oben genannten Maßnahmen nicht behoben werden, ändern Sie das Programm, verringern Sie die Geschwindigkeiten und die Beschleunigung-/Abbremsung.

Nach Beendigung des Warmlaufbetriebs treten Positionsabweichungen auf

- Vergrößern Sie den Wert des ersten Elements im Parameter WUPTIME und verlängern Sie die Gesamtbetriebsdauer des Warmlaufbetriebs.
- Prüfen Sie, ob die Last des Roboters und die Umgebungstemperatur innerhalb der zulässigen Bereiche liegen.
- Prüfen Sie, ob eine Achse nach Beendigung des Warmlaufbetriebs für längere Zeit gestoppt ist. Erhöhen Sie in einem solchen Fall das zweite Element des Parameters WUPTIME und verkürzen Sie die Zeit bis zu einer erneuten Ausführung des Warmlaufbetriebs.
- Kann der Fehler durch die oben genannten Maßnahmen nicht behoben werden, ändern Sie das Programm, verringern Sie die Geschwindigkeiten und die Beschleunigung-/Abbremsung.

Der Warmlaufbetrieb wird nicht beendet

- Prüfen Sie den Wert des Parameters WUPAXIS, um festzustellen, ob eine Achse, die nicht gesteuert wird, für den Warmlaufbetrieb festgelegt wurde.
- Prüfen Sie, ob die Betriebspause einer für den Warmlaufbetrieb festgelegten Achse den Wert für die Wiederholschwelle (2. Element des Parameters WUPTIME) übersteigt.
- Prüfen Sie, ob der Roboter mit einer extrem niedrigen Geschwindigkeit verfahren wird (3 % bis 5 % Übersteuerung bei Gelenkinterpolation). Bei dieser niedrigen Geschwindigkeit ist kein Warmlaufbetrieb nötig. Deaktivieren Sie den Warmlaufbetrieb.

9.21 Durchfahren eines singulären Punktes

MITSUBISHI-Robotersysteme berechnen linear interpolierte Bewegungen und speichern geteachte Positionen als Positionsdaten im XYZ-Koordinatensystem. Für einen 6-achsigen Roboter können Positionen über die Koordinaten X, Y, Z, A, B, und C festgelegt werden. Es gibt jedoch eine Vielzahl von Positionen mit den gleichen Positionsdaten, aber mit unterschiedlichen Roboterstellungen (Stellung der Robotergelenke). Diese unterschiedlichen Roboterstellungen werden über sogenannte Stellungsmerker eindeutig festgelegt. Jedes einzelne Robotergelenk kann jedoch eine unendliche Anzahl von Winkeln einnehmen.

Auch unter Zuhilfenahme des Stellungsmerkers ist es an den Stellen, an denen der Stellungsmerker umschaltet, nicht immer möglich, den Roboter mit der gewünschten Position und Stellung zu steuern. (Bei einem 6-achsigen Knickarmroboter z. B. sind die Achsen J4 und J6 nicht eindeutig definiert, wenn der Winkel der J5-Achse 0° ist.) Diese Positionen werden singuläre Punkte genannt. Sie können im XYZ-JOG-Betrieb und mit Linear-Interpolation nicht erreicht werden. Früher wurde das Problem umgangen, indem man bei der Planung der Bewegungsabläufe singuläre Punkte vermied oder der Roboter bei einer unvermeidbaren Durchfahrt eines singulären Punktes mit Gelenk-Interpolation verfahren wurde.

Die Funktion zur Durchfahrt singulärer Punkte erlaubt ein Durchfahren singulärer Punkte im XYZ-JOG-Betrieb und mit Linear-Interpolation. Dadurch wird eine freiere Planung der Bewegungsabläufe und eine Vergrößerung des Arbeitsraumes durch die Nutzung der Linear-Interpolation möglich.

9.21.1 Positionen singulärer Punkte, die durchfahren werden können

Folgende Positionen können mit der Funktion zur Durchfahrt singulärer Punkte durchfahren werden.

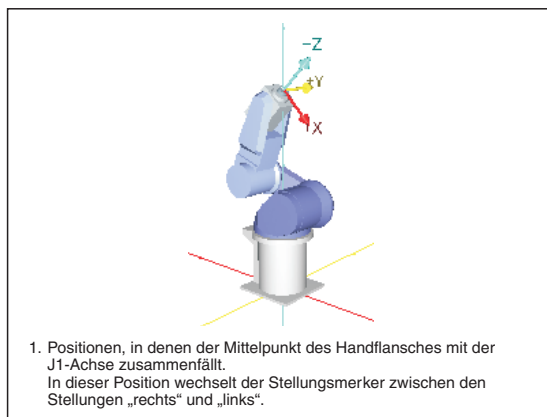
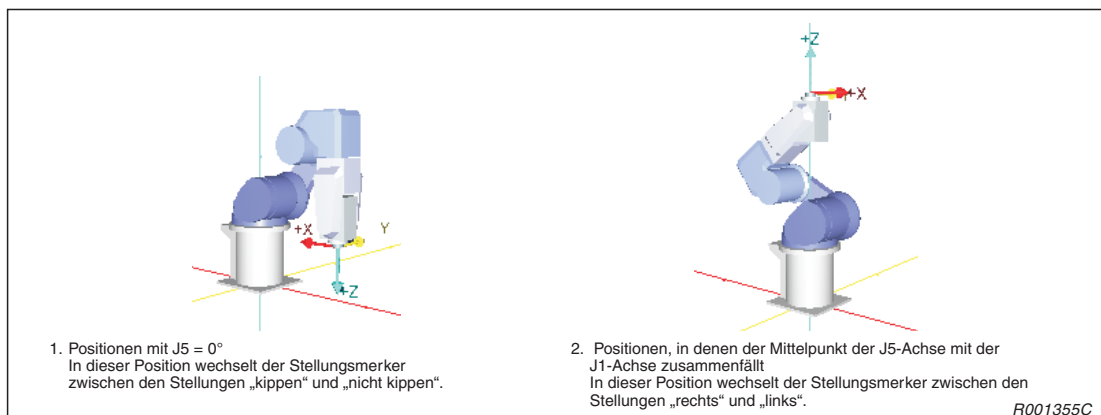


Abb. 9-34

Bei fünfachsigen Knickarmrobotern

R001354C



R001355C

Abb. 9-35: Bei sechsachsigen Knickarmrobotern

9.21.2 Betrieb, mit aktivierter Funktion zum Durchfahren singulärer Punkte

Ist die Funktion zum Durchfahren singulärer Punkte aktiviert, kann der Roboter im XYZ-JOG-Betrieb oder mittels Linear-Interpolation von der Position A über die Position B (singulärer Punkt) zur Position C und umgekehrt verfahren werden. Der Stellungsmerker wechselt den Status vor und nach Durchfahren des Punktes B.

Ist die Funktion zum Durchfahren singulärer Punkte deaktiviert (oder sie wird nicht unterstützt), stoppt der Roboter vor der Ausführung der Bewegung von A nach B und es erfolgt die Ausgabe einer Fehlermeldung. Im XYZ-JOG-Betrieb stoppt der Roboter unmittelbar vor der Position B.

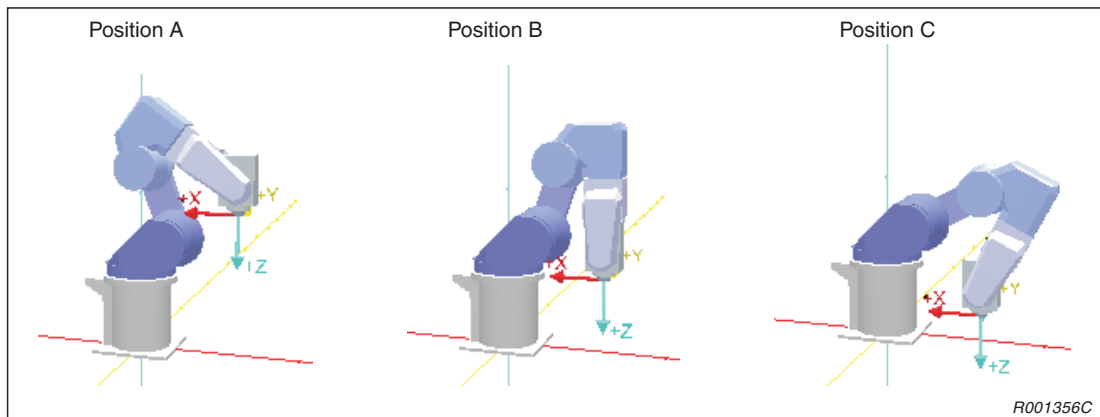


Abb. 9-36: Durchfahren eines singulären Punktes

Der Roboter kann einen singulären Punkt durchfahren, wenn der Verfahrensweg durch den singulären Punkt verläuft. Verläuft der Verfahrensweg nicht durch den singulären Punkt (sondern in der Nähe des singulären Punktes) bewegt sich der Roboter, ohne dass ein Zustandswechsel des Stellungsmerkers erfolgt.

In der folgenden Abbildung verläuft der Verfahrensweg beim Durchfahren der Positionen D -> E -> F durch einen singulären Punkt. (Der Stellungsmerker wechselt den Status vor und nach Durchfahren des Punktes E.)

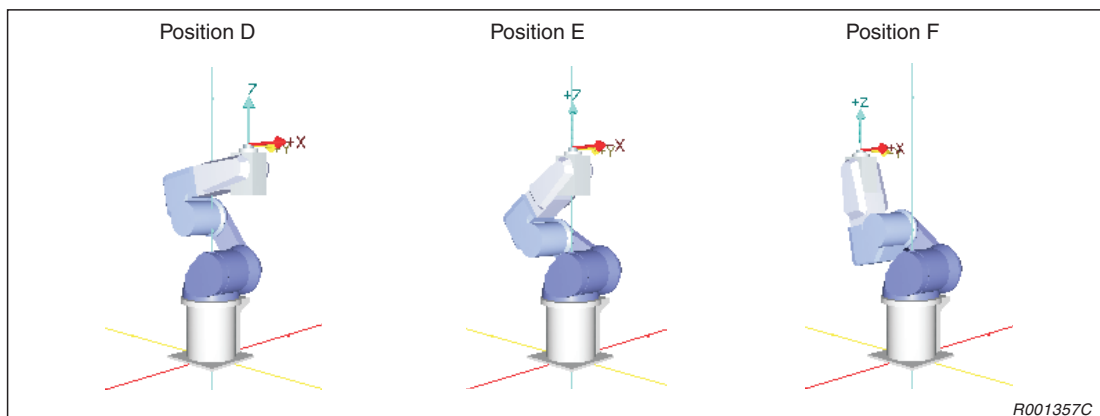


Abb. 9-37: Durchfahren eines singulären Punktes

In der folgenden Abbildung verläuft der Verfahrensweg beim Durchfahren der Positionen G -> H -> I in der Nähe des singulären Punktes (Der Stellungsmerker wechselt den Zustand nicht.)

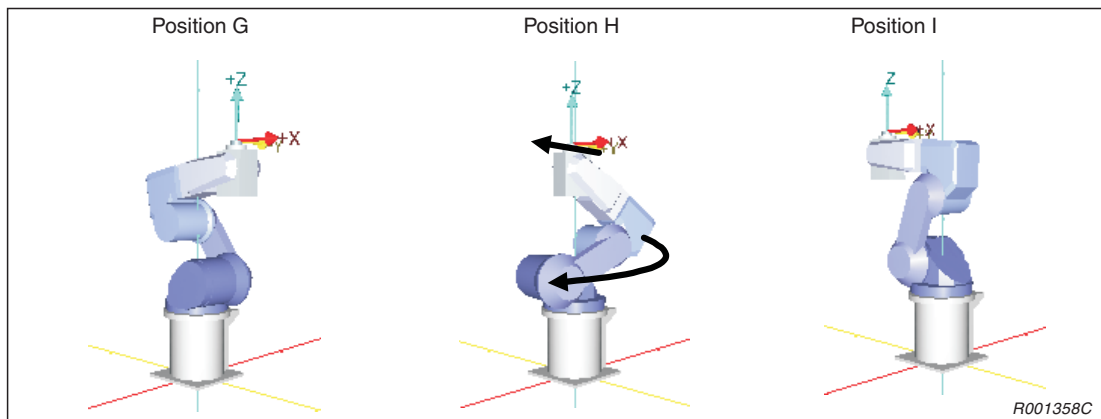


Abb. 9-38: Verfahrensweg verläuft in der Nähe eines singulären Punktes



ACHTUNG:

Verläuft der Verfahrensweg in der Nähe eines singulären Punktes, kann der Roboter in einem weiten Kreisbogen rotieren, wie in der Abbildung oben (Position H) gezeigt. Beobachten Sie daher die Roboterbewegungen genau und halten Sie sich nicht im Aktionsradius des Roboters auf wie z. B. beim Teachen von Positionen.

9.21.3 Aktivierung der Funktion zum Durchfahren singulärer Punkte

Möchten Sie die Funktion zum Durchfahren singulärer Punkte für den JOG-Betrieb aktivieren, setzen Sie Parameter FSPJOGMD auf „1“. Schalten Sie anschließend die Spannungsversorgung des Steuergerätes aus und wieder ein. Aktivieren Sie die Funktion für den Automatikbetrieb, indem Sie die Konstante 2 in der TYPE-Festlegung im jeweiligen Interpolationsbefehl auf „2“ setzen.

HINWEIS

Die Funktion zum Durchfahren singulärer Punkte ist ab Software-Version K1 verfügbar und mit allen Knickarmrobotern der S-Serie verwendet werden. Wird die Funktion bei einem anderen Modell aktiviert, erfolgt bei Ausführung des JOG-Betriebs ein Normalbetrieb und bei Ausführung des Automatikbetriebs eine Fehlermeldung.

Bei der Nutzung der Funktion zum Durchfahren singulärer Punkte gelten folgende Einschränkungen:

- Die Funktion kann beim Einsatz von Zusatzachsen für einen Betrieb mehrerer Mechanismen nicht verwendet werden.
- Die Funktion kann bei einer Synchronsteuerung von Zusatzachsen nicht verwendet werden.
- Die Funktion kann bei aktivierter Achsenweichheit nicht verwendet werden.
- Die Funktion kann bei aktivierter Kollisionsüberwachung nicht verwendet werden.
- Die Datenerfassungsstufe zur Überwachung der Wartungsintervalle muss auf „1“ gesetzt sein (Werkseinstellung).
- Die Programmiermethode MELFA-BASIC IV verfügt im Gegensatz zu MOVEMASTER COMMAND über Anweisungen, die in Beziehung zur Funktion zum Durchfahren singulärer Punkte stehen. Verwenden Sie die Funktion zum Durchfahren singulärer Punkte daher mit der MELFA-BASIC-IV-Programmiermethode.

9.21.4 Funktion zum Durchfahren singulärer Punkte im JOG-Betrieb

Im JOG-Betrieb wird die Funktion zum Durchfahren singulärer Punkte durch Einstellung des Parameters FSPJOGMD auf „1“ aktiviert und durch Einstellung auf „0“ deaktiviert.

FSPJOGMD	XYZ-JOG	TOOL-JOG	3-Achsen-XYZ-JOG	Kreis-JOG	Gelenk-JOG
0 (Werkseinstellung)	Wie vorher	Wie vorher	Wie vorher	Wie vorher	Wie vorher
1	Durchfahren singulärer Punkte im XYZ-JOG-Betrieb	Durchfahren singulärer Punkte im TOOL-JOG-Betrieb	Wie vorher	Wie vorher	Wie vorher

Tab. 9-32: Einstellung des Parameters FSPJOGMD

- Bei Robotern, die nicht zur Nutzung der Funktion zum Durchfahren singulärer Punkte geeignet sind, hat eine Einstellung des Parameters FSPJOGMD keine Wirkung. Der vorhergehende Betrieb wird nicht beeinflusst. (Unterstützt wird die Funktion zum Durchfahren singulärer Punkte von allen Knickarmrobotern der S-Serie.)
- Es ist nicht möglich mehrere Achsen gleichzeitig beim Durchfahren singulärer Punkte im JOG-Betrieb zu verfahren. Der Versuch während einer Achsenbewegung eine zweite Achse zu verfahren, wird ignoriert.
- Wird eine Achse im JOG-Betrieb mittels der Teaching Box in die Nähe eines singulären Punktes verfahren, erfolgt die Ausgabe einer Fehlermeldung (siehe auch Abschn. 9.18).
- Der Wert des Parameters FSPJOGMD ist auch bei Festlegung des JOG-Betriebs über spezielle Eingangssignale wirksam.

9.21.5 Funktion zum Durchfahren singulärer Punkte beim Anfahren definierter Positionen

Der Wert des Parameters FSPJOGMD ist auch beim Anfahren definierter Positionen wirksam.

FSPJOGMD	MOV-Bewegungsbefehl	MVS-Bewegungsbefehl
0 (Werkseinstellung)	Wie vorher	Wie vorher
1	Wie vorher	Durchfahren singulärer Punkte

Tab. 9-33: Durchfahren singulärer Punkte beim Anfahren definierter Positionen



ACHTUNG:

Wird ein Interpolationsbefehl (z. B. MVS P1) bei einer Einstellung des Parameters auf „1“ (aktiv) direkt über die Teaching Box ausgeführt, ist die Funktion zum Durchfahren singulärer Punkte aktiv, auch wenn sie nicht über die TYPE-Festlegung eingestellt wurde.

9.21.6 Funktion zum Durchfahren singulärer Punkte im Automatikbetrieb

Aktivieren Sie die Funktion zum Durchfahren singulärer Punkte im Automatikbetrieb über die TYPE-Festlegung des jeweiligen Interpolationsbefehls.

9.21.7 TYPE (Type)

Funktion: Durchfahren des singulären Punktes festlegen

Definieren Sie die Durchfahrt durch den singulären Punkt in der TYPE-Festlegung des Interpolationsbefehls. Die Festlegung ist bei folgenden Interpolationsbefehlen möglich: Linear-Interpolation (MVS), Kreis-Interpolation (MVR, MVR2 und MVR) und Kreis-Interpolation (MVC). Die Funktion steht ab Software-Version K1 zur Verfügung.

Eingabeformat

TYPE <Konstante 1>,<Konstante 2>

<Konstante 1> Direkte/indirekte Anfahrt = 0/1

<Konstante 2> Drehung/3-Achsen-XYZ/Durchfahren des singulären Punktes = 0/1/2

Programmbeispiel

10 MVS P1 TYPE 0,2 Position P1 mittels Linear-Interpolation und aktivierter Funktion zum Durchfahren singulärer Punkte anfahren

20 MVR P1,P2,P3 TYPE 0,2 Position P3 mittels Kreis-Interpolation und aktivierter Funktion zum Durchfahren singulärer Punkte von P1 aus anfahren

Beschreibung

- Wird die Konstante 2 bei einem Roboter, der die Funktion zum Durchfahren singulärer Punkte nicht unterstützt, auf „2“ gesetzt, tritt ein Laufzeitfehler auf.
- Bei aktivierter Funktion zum Durchfahren singulärer Punkte wird der Stellungsmerker zwischen dem Start- und Endpunkt nicht geprüft. Da der Zustand des Stellungsmerkers an der Zielposition nicht bewertet wird, erfolgt auch vor der Ausführung der Bewegung keine Prüfung des Bewegungsbereiches hinsichtlich der Zielposition und der Zwischenpositionen.
- Wurde über den Befehl SPD eine Geschwindigkeit festgelegt, so entspricht diese Geschwindigkeit dem Maximalwert. In der Nähe eines singulären Punktes wird die Geschwindigkeit automatisch auf einen Wert verringert, bei dem kein Geschwindigkeitsfehler auftritt.
- Interpolationsbefehle, für die die Funktion zum Durchfahren singulärer Punkte aktiviert ist, können nicht mit optimaler Beschleunigung/Abbremsung ausgeführt werden. Die Beschleunigung/Abbremsung ist fest vorgegeben. Entspricht die Beschleunigungszeit aufgrund der Festlegung im Befehl ACCEL nicht der Bremszeit, wird der größere Wert sowohl für die Beschleunigung als auch für die Abbremsung verwendet.
- Eine freigegebene CNT-Einstellung zur Steuerung kontinuierlicher Bewegungen ist bei aktivierter Funktion zum Durchfahren singulärer Punkte unwirksam. Der Roboter bewegt sich mit der festgelegten Beschleunigung/Abbremsung.
- Entspricht die aktuelle Position vor der Ausführung einer Kreis-Interpolation nicht der Startposition, erfolgt die Anfahrt des Startpunktes mittels 3-Achsen-XYZ-Linear-Interpolation, auch wenn die Funktion zum Durchfahren singulärer Punkte in der TYPE-Festlegung aktiviert wurde.
- Wird die Ausführung eines Interpolationsbefehls mit aktivierter Funktion zum Durchfahren singulärer Punkte unterbrochen und nach anschließendem JOG-Betrieb wieder fortgesetzt, kehrt der Roboter entsprechend der Einstellung im Parameter RETPATH zu der Position der Unterbrechung zurück und führt die Verfahrbewegung fort. Ist der Parameter RETPATH auf „0“ (deaktiviert) gesetzt, erfolgt keine Rückkehr zur Position der Unterbrechung. Der Stellungsmerker wechselt seinen Zustand nicht, solange nach der Fortsetzung des Verfahrweges kein singulärer Punkt durchfahren wird (siehe folgende Abbildung). Daher kann die Stellung des Roboters nach Abschluss des Interpolationsvorgangs von der Stellung ohne Unterbrechung des Interpolationsvorgangs abweichen.

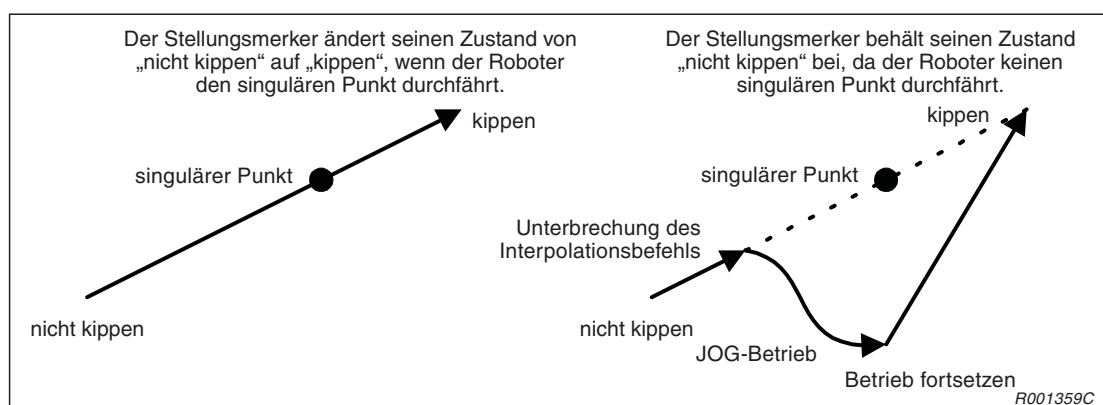


Abb. 9-39: Unterbrechung des Interpolationsbefehls

- Ist die Funktion zum Durchfahren singulärer Punkte aktiviert, kann die Geschwindigkeit geringer sein, als bei einer normalen Interpolation. Da die Funktion komplizierte Berechnungsprozesse erfordert, kann die gesamte Programmverarbeitungszeit ansteigen. Verwenden Sie daher die Funktion zum Durchfahren singulärer Punkte nur dann, wenn sie wirklich nötig ist.

10 Externe Ein-/Ausgänge

10.1 Einteilung

Die externen Ein-/Ausgänge sind in 3 Gruppen aufgeteilt:

- **Spezielle Ein-/Ausgänge**
Die Ein-/Ausgänge dienen zur Steuerung und Statusanzeige des Roboters. Häufig verwendete Funktionen sind dabei vordefiniert. Der Anwender hat die Möglichkeit, neue Funktionen hinzuzufügen und bestehende Funktionen zu modifizieren.
- **Allgemeine Ein-/Ausgänge**
Die Ein-/Ausgänge dienen zur Kommunikation mit Peripheriegeräten über das Roboterprogramm und können frei programmiert werden. Sie ermöglichen die Überwachung der Roboterposition und der Übertragung von Positioniersignalen von externen Geräten (z. B. SPS).
- **Ein-/Ausgänge für Greifhand**
Die Ein-/Ausgänge können zur Unterstützung von Handfunktionen programmiert werden. Sie dienen z. B. zum Öffnen und Schließen der Hand oder zum Einlesen von Handsensordaten. Dazu benötigen Sie das optionale Steuermodul für die pneumatische/elektrische Greifhand. Die Steuerung und Überwachung der Eingänge erfolgt über das Roboterprogramm.



ACHTUNG:

Sie können die Spezial-Eingänge während der Programmausführung in allgemeine Eingänge umdefinieren. Das ist aus Sicherheitsgründen nur für die numerischen Dateneingänge zu empfehlen. Dagegen können Sie die Spezialausgänge nicht als allgemeine Ausgänge im Programm benutzen. Bei einem Versuch löst der Roboter Alarm aus.

10.1.1 Allgemeine Übersicht der Ein- und Ausgänge

	E/A-Signalnummer	Zugriff
Allgemeine E/A	0–31 (15) ^①	Über die Variablen M_IN, M_INB, M_INW, M_OUT, M_OUTB oder M_OUTW Beispiel: IF M_IN(0) = 1 THEN M_OUT(0) = 1
Maximale E/A	255 (240) ^① (zusätzliche E/A = 240)	Siehe oben
E/A für die Greifhand	900–907	Siehe oben Können durch die Befehle HOPEN und HCLOSE ersetzt werden Beispiel: IF M_IN(900) THEN M_OUT(900) = 1 HOPEN 1, HCLOSE 1
CC-Link-Bits ^②	Bei Belegung von 1 Station: 6000–6031, bei Belegung von 4 Stationen: 6000–6127 (Entspricht der Signalnummer für Station 1. Die letzten beiden Bits können nicht verwendet werden.)	Über die Variablen M_IN, M_INB, M_INW, M_OUT, M_OUTB oder M_OUTW Beispiel: IF M_IN(6000) = 1 THEN M_OUT(6000) = 1
CC-Link-Register ^②	Bei Belegung von 1 Station: 6000–6003, bei Belegung von 4 Stationen: 6000–6015 (Entspricht der Signalnummer für Station 1.)	Über die Variablen M_DIN oder M_DOUT Beispiel: IF M_DIN(6000) = 1000 THEN M_DOUT(6000) = 200

Tab. 10-1: Übersicht der Ein- und Ausgänge

- ^① Die in Klammern angegebenen Werte gelten für das Steuergerät CR1.
- ^② Eine detaillierte Beschreibung der CC-Link-Funktionen finden Sie Handbuch der CC-Link-Schnittstellenkarte.

10.2 Parallele Ein-/Ausgangsschnittstelle

Standardmäßig verfügt das Steuergerät über eine parallele Ein-/Ausgangsschnittstelle. Die Ein-/Ausgangskapazität kann bei den Steuergeräten CR2 und CR2A durch Anschluss von weiteren sieben externen, parallelen Ein-/Ausgangs-Schnittstellenmodulen auf 256 und beim Steuergerät CR1 auf 240 Ein- und Ausgänge (inkl. Standardschnittstellenmodul) erweitert werden. Die parallele Ein-/Ausgangsschnittstelle (Standard) ist mit einem 50-poligen Centronics-Steckeranschluss ausgerüstet. Wenn Sie externe Geräteeinheiten an einen Roboter anschließen möchten, benötigen Sie ein spezielles Ein-/Ausgangskabel RV-E-E/A (Option).

Die 24-V-DC-Spannungsversorgung für die externen Ein-/Ausgangsschnittstellen und die 12- bis 24-V-DC-Spannungsversorgung für die Ein- und Ausgangskreise müssen vom Anwender bereitgestellt werden.

Eine Übersicht der Standard-Pin-Belegung finden Sie in Tab. 10-2 und Tab. 10-3. Die Pin-Belegung für positive Logik des Steuergerätes CR1 finden Sie in Tab. 10-4.

Eine detaillierte Beschreibung der Ein- und Ausgangsschnittstellen und eine Auflistung der technischen Daten der Ein- und Ausgangskreise finden Sie im Technischen Handbuch des jeweiligen Roboters.

HINWEIS

Das Steuergerät CR1 verfügt standardmäßig über 16 Eingangsadressen und 16 Ausgangsadressen. Die Steuergeräte CR2, CR2A, CR2B und CR3 verfügen standardmäßig über 32 Eingangsadressen und 32 Ausgangsadressen.

Übersicht der Pinbelegung für Anschluss CN100 (Kabel RV-E-E/A)

Pin-Nr.	Aderfarbe	Funktion	
		Allgemeine Verwendung	Spezial-Versorgungsspannung / Bezugspunkt
1	Weiß		FG
2	Braun		0 V für Pins 4–7
3	Grün		+12 V/+24 V für Pins 4–7
4	Gelb	Ausgang 0	Betrieb
5	Grau	Ausgang 1	Servo EIN
6	Rosa	Ausgang 2	Fehler
7	Blau	Ausgang 3	Betriebsrechte
8	Rot		0 V für Pins 10–13
9	Schwarz		+12 V/+24 V für Pins 10–13
10	Violett	Ausgang 8	
11	Grau-rosa	Ausgang 9	
12	Rot-blau	Ausgang 10	
13	Weiß-grün	Ausgang 11	
14	Braun-grün		COM0 (12 V/24 V): Bezugspunkt für Pins 15–22
15	Weiß-gelb	Eingang 0	Stopp (für alle Anwendungen) ^①
16	Gelb-braun	Eingang 1	Servo AUS
17	Weiß-grau	Eingang 2	Fehler quittieren
18	Grau-braun	Eingang 3	Start
19	Weiß-rosa	Eingang 4	Servo EIN
20	Rosa-braun	Eingang 5	Betriebsrechte

Tab. 10-2: Übersicht der Pinbelegung des Standard-Ein/Ausgangsmoduls CN100 (1)

Pin-Nr.	Aderfarbe	Funktion	
		Allgemeine Verwendung	Spezial-Versorgungsspannung / Bezugspunkt
21	Weiß-blau	Eingang 6	
22	Braun-blau	Eingang 7	
23	Weiß-rot		
24	Braun-rot		
25	Weiß-schwarz		
26	Braun-schwarz		FG
27	Grau-grün		0 V für Pins 29–32
28	Gelb-grau		+12 V/+24 V für Pins 29–32
29	Rosa-grün	Ausgang 4	
30	Gelb-rosa	Ausgang 5	
31	Grün-blau	Ausgang 6	
32	Gelb-blau	Ausgang 7	
33	Grün-rot		0 V für Pins 35–38
34	Gelb-rot		+12 V/+24 V für Pins 35–38
35	Grün-schwarz	Ausgang 12	
36	Gelb-schwarz	Ausgang 13	
37	Grau-blau	Ausgang 14	
38	Rosa-blau	Ausgang 15	
39	Grau-rot		COM1 (12 V/24 V): Bezugspunkt für Pins 40–47
40	Rosa-rot	Eingang 8	
41	Grau-schwarz	Eingang 9	
42	Rosa-schwarz	Eingang 10	
43	Blau-schwarz	Eingang 11	
44	Rot-schwarz	Eingang 12	
45	Weiß-braun-schwarz	Eingang 13	
46	Gelb-grün-schwarz	Eingang 14	
47	Grau-rosa-schwarz	Eingang 15	
48	Blau-rot-schwarz		
49	Weiß-grün-schwarz		
50	Grün-braun-schwarz		

Tab. 10-2: Übersicht der Pinbelegung des Standard-Ein/Ausgangsmoduls CN100 (2)

- ① Der Stopp-Eingang ist fest mit dem Eingangsbit „0“ verbunden und kann nicht gelöscht und einem anderen Eingangsbit zugewiesen werden.

HINWEIS

Bei Verwendung des CR1-Steuergerätes müssen die Pins 2, 8, 27 und 33 mit 0 V verbunden werden sowie die Pins 3, 9, 28 und 34 mit der +12 V/24 V Spannungsversorgung.

Übersicht der Pinbelegung für Anschluss CN300 (Kabel RV-E-E/A)

Pin-Nr.	Aderfarbe	Funktion	
		Allgemeine Verwendung	Spezial-Versorgungsspannung / Bezugspunkt
1	Weiß		FG
2	Braun		0 V für Pins 4–7
3	Grün		+12 V/+24 V für Pins 4–7
4	Gelb	Ausgang 16	
5	Grau	Ausgang 17	
6	Rosa	Ausgang 18	
7	Blau	Ausgang 19	
8	Rot		0 V für Pins 10–13
9	Schwarz		+12 V/+24 V für Pins 10–13
10	Violett	Ausgang 24	
11	Grau-rosa	Ausgang 25	
12	Rot-blau	Ausgang 26	
13	Weiß-grün	Ausgang 27	
14	Braun-grün		COM0 (12 V/24 V): Bezugspunkt für Pins 15–22
15	Weiß-gelb	Eingang 16	
16	Gelb-braun	Eingang 17	
17	Weiß-grau	Eingang 18	
18	Grau-braun	Eingang 19	
19	Weiß-rosa	Eingang 20	
20	Rosa-braun	Eingang 21	
21	Weiß-blau	Eingang 22	
22	Braun-blau	Eingang 23	
23	Weiß-rot		
24	Braun-rot		
25	Weiß-schwarz		
26	Braun-schwarz		FG
27	Grau-grün		0 V für Pins 29–32
28	Gelb-grau		+12 V/+24 V für Pins 29–32
29	Rosa-grün	Ausgang 20	
30	Gelb-rosa	Ausgang 21	
31	Grün-blau	Ausgang 22	
32	Gelb-blau	Ausgang 23	
33	Grün-rot		0 V für Pins 35–38
34	Gelb-rot		+12 V/+24 V für Pins 35–38
35	Grün-schwarz	Ausgang 28	
36	Gelb-schwarz	Ausgang 29	
37	Grau-blau	Ausgang 30	
38	Rosa-blau	Ausgang 31	
39	Grau-rot		COM1 (12 V/24 V): Bezugspunkt für Pins 40–47
40	Rosa-rot	Eingang 24	

Tab. 10-3: Übersicht der Pinbelegung des Standard-Ein/Ausgangsmoduls CN300 (1)

Pin-Nr.	Aderfarbe	Funktion	
		Allgemeine Verwendung	Spezial-Versorgungsspannung / Bezugspunkt
41	Grau-schwarz	Eingang 25	
42	Rosa-schwarz	Eingang 26	
43	Blau-schwarz	Eingang 27	
44	Rot-schwarz	Eingang 28	
45	Weiß-braun-schwarz	Eingang 29	
46	Gelb-grün-schwarz	Eingang 30	
47	Grau-rosa-schwarz	Eingang 31	
48	Blau-rot-schwarz		
49	Weiß-grün-schwarz		
50	Grün-braun-schwarz		

Tab. 10-3: Übersicht der Pinbelegung des Standard-Ein/Ausgangsmoduls CN300 (2)

Übersicht der Pinbelegung für Anschluss CN100 für das Steuergerät CR1 in positiver Logik (Kabel RV-E-E/A)

Pin-Nr.	Aderfarbe	Funktion	
		Allgemeine Verwendung	Spezial-Versorgungsspannung / Bezugspunkt
1	Weiß		FG
2	Braun		0 V für Pins 4–7, 10–13
3	Grün		+12 V/+24 V für Pins 4–7, 10–13
4	Gelb	Ausgang 0	Betrieb
5	Grau	Ausgang 1	Servo EIN
6	Rosa	Ausgang 2	Fehler
7	Blau	Ausgang 3	Betriebsrechte
8	Rot		Reserviert
9	Schwarz		Reserviert
10	Violett	Ausgang 8	
11	Grau-rosa	Ausgang 9	
12	Rot-blau	Ausgang 10	
13	Weiß-grün	Ausgang 11	
14	Braun-grün		COM0 (12 V/24 V): Bezugspunkt für Pins 15–22
15	Weiß-gelb	Eingang 0	Stopp (für alle Anwendungen) ^①
16	Gelb-braun	Eingang 1	Servo AUS
17	Weiß-grau	Eingang 2	Fehler quittieren
18	Grau-braun	Eingang 3	Start
19	Weiß-rosa	Eingang 4	Servo EIN
20	Rosa-braun	Eingang 5	Betriebsrechte
21	Weiß-blau	Eingang 6	
22	Braun-blau	Eingang 7	
23	Weiß-rot		Reserviert
24	Braun-rot		Reserviert
25	Weiß-schwarz		Reserviert
26	Braun-schwarz		FG
27	Grau-grün		0 V für Pins 29–32, 35–38
28	Gelb-grau		+12 V/+24 V für Pins 29–32, 35–38
29	Rosa-grün	Ausgang 4	
30	Gelb-rosa	Ausgang 5	
31	Grün-blau	Ausgang 6	
32	Gelb-blau	Ausgang 7	
33	Grün-rot		Reserviert
34	Gelb-rot		Reserviert
35	Grün-schwarz	Ausgang 12	
36	Gelb-schwarz	Ausgang 13	
37	Grau-blau	Ausgang 14	
38	Rosa-blau	Ausgang 15	
39	Grau-rot		COM1 (12 V/24 V): Bezugspunkt für Pins 40–47
40	Rosa-rot	Eingang 8	

Tab. 10-4: Übersicht der Pinbelegung des Standard-Ein/Ausgangsmoduls CN100 für das Steuergerät CR1 in positiver Logik (1)

Pin-Nr.	Aderfarbe	Funktion	
		Allgemeine Verwendung	Spezial-Versorgungsspannung / Bezugspunkt
41	Grau-schwarz	Eingang 9	
42	Rosa-schwarz	Eingang 10	
43	Blau-schwarz	Eingang 11	
44	Rot-schwarz	Eingang 12	
45	Weiß-braun-schwarz	Eingang 13	
46	Gelb-grün-schwarz	Eingang 14	
47	Grau-rosa-schwarz	Eingang 15	
48	Blau-rot-schwarz		Reserviert
49	Weiß-grün-schwarz		Reserviert
50	Grün-braun-schwarz		Reserviert

Tab. 10-4: Übersicht der Pinbelegung des Standard-Ein/Ausgangsmoduls CN100 für das Steuergerät CR1 in positiver Logik (2)

- ① Der Stopp-Eingang ist fest mit dem Eingangsbit verbunden und kann nicht gelöscht und einem anderen Eingangsbit zugewiesen werden.

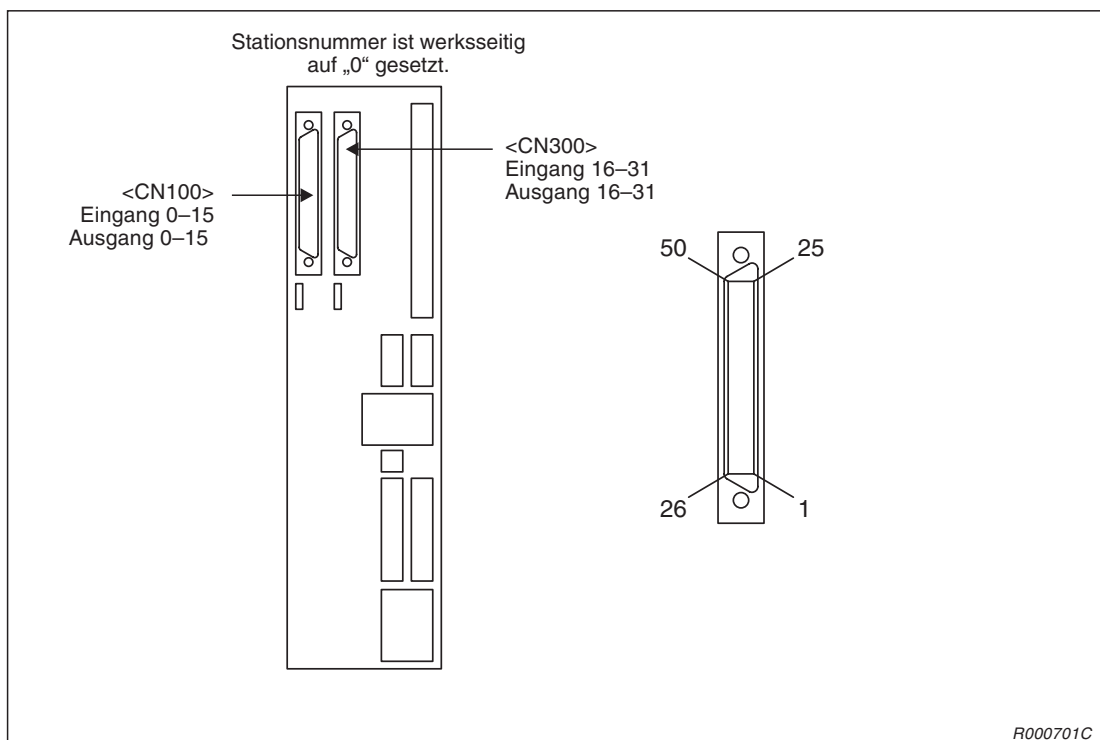


Abb. 10-1: Anschlussbelegung der parallelen Ein-/Ausgangsschnittstelle für das Steuergerät CR2



ACHTUNG:

Werksseitig ist die Stationsnummer auf „0“ gesetzt. Stellen Sie keine Nummer zwischen 8–F ein, da dieses zu undefinierten Aktivitäten führen kann. Die parallele Standardschnittstelle im Steuergerät CR1 besitzt keinen Schalter zur Einstellung der Stationsnummer.

10.2.1 Ein-/Ausgangsbelegung der parallelen Ein-/Ausgangsschnittstelle

In folgender Tabelle sind die Funktionen aufgelistet, die den Ein-/Ausgängen zugewiesen werden können. Die Parameter werden den Signalnummern in der Reihenfolge Eingangssignalnummer/Ausgangssignalnummer zugewiesen. Die genaue Vorgehensweise zur Einstellung von Parametern finden Sie im Abschn. 3.13.1. Die Parameter können mit der Teaching Box im Menü zur Einstellung von Parametern, die optionale PC-Support-Software oder COSIROP eingestellt werden. Eine Verwendung der Parameter ist nur dann möglich, wenn der MODE-Umschalter am Steuergerät auf „AUTO (Ext.)“ eingestellt ist.

Die Anzahl der verfügbaren Ein-/Ausgangssignale kann durch die optionalen parallelen Ein-/Ausgangsschnittstellen vergrößert werden.

Parameter	Zuordnung	Bezeichnung	Beschreibung	Signalpegel ^①	Werkseinstellung ^②
RCREADY	Eingang	—	—		-1 (unwirksam)
	Ausgang	Spannungsversorgung des Steuergerätes eingeschaltet	Zeigt an, dass die Spannungsversorgung des Steuergerätes eingeschaltet ist und externe Signale empfangen werden können		-1
ATEXTMD	Eingang	—	—		-1 (unwirksam)
	Ausgang	Ausgangssignal externer Betrieb	Zeigt an, dass der MODE-Umschalter am Steuergerät auf „AUTO (Ext.)“ eingestellt ist Das Signal muss eingeschaltet sein, bevor die Funktionen der E/A genutzt werden können.		-1
TEACHMD	Eingang	—	—		-1 (unwirksam)
	Ausgang	Ausgangssignal Teach-Modus	Zeigt an, dass der MODE-Umschalter am Steuergerät auf Teach-Betrieb eingestellt ist		1
ATTOPMD	Eingang	—	—		-1 (unwirksam)
	Ausgang	Ausgangssignal Automatikbetrieb	Zeigt an, dass der MODE-Umschalter am Steuergerät auf „AUTO (Op.)“ eingestellt ist		-1
IOENA	Eingang	Eingangssignal Betriebsrechte	Anforderung der Betriebsrechte für eine externe Steuerung	H	5
	Ausgang	Ausgangssignal Betriebsrechte	Zeigt an, dass der Betrieb über externe Signale freigegeben ist Die Betriebsrechte werden freigegeben, wenn das Eingangssignal eingeschaltet ist, der MODE-Umschalter auf „AUTO (Ext.)“ eingestellt ist und kein anderes Gerät momentan über die Betriebsrechte verfügt.		3

Tab. 10-5: Spezielle Parameter für Ein-/Ausgänge (1)

Parameter	Zuordnung	Bezeichnung	Beschreibung	Signal-pegel ^①	Werkseinstellung ^②
START (Betriebsrechte erforderlich)	Eingang	Startsignal	Startet die Programme im Multitasking-Betrieb Zum Starten eines bestimmten Programms muss das Programm über das Programmwahlsignal „PRGSEL“ und den numerischen Eingang „IOWDATA“ ausgewählt und anschließend über das Startsignal gestartet werden. Beachten Sie, dass die Programmnummer bei einer Freigabe des Parameters „PST“ über den numerischen Eingang „IOWDATA“ eingelesen und das entsprechende Programm unabhängig von der Programmwahl gestartet wird. Im Multitask-Betrieb werden alle Programme gestartet. Programmplätze, deren Startbedingung in den Programmplatzparametern „SLT***“ auf „ALWAYS“ oder „ERROR“ gesetzt ist, werden nicht ausgeführt.	↑	3
	Ausgang	Ausgangssignal Programm aktiv	Zeigt ein aktives Programm an Programmplätze, deren Startbedingung in den Programmplatzparametern „SLT***“ auf „ALWAYS“ oder „ERROR“ gesetzt ist, werden nicht ausgeführt.		0
STOP	Eingang	Stoppsignal	Stoppt die ausgeführten Programme (Programmplätze, deren Startbedingung auf „ALWAYS“ oder „ERROR“ gesetzt ist, werden nicht gestoppt). Die Eingangssignalnummer ist auf „0“ festgelegt. Im Multitask-Betrieb werden alle Programme gestoppt. Programmplätze, deren Startbedingung in den Programmplatzparametern „SLT***“ auf „ALWAYS“ oder „ERROR“ gesetzt ist, werden nicht gestoppt. Mit dem Parameter INB kann beim Eingang zwischen Standard und Drahtbruchererkennung gewählt werden. HINWEIS: Verwenden Sie für alle sicherheitsrelevanten Stopps den NOT-AUS-Eingang.	H	0 (keine Änderung möglich)
	Ausgang	Wartestatus aktiv	Zeigt an, dass die Abarbeitung des entsprechenden Programms vorübergehend unterbrochen worden ist Programmplätze, deren Startbedingung in den Programmplatzparametern „SLT***“ auf „ALWAYS“ oder „ERROR“ gesetzt ist, werden nicht angezeigt.		-1

Tab. 10-5: Spezielle Parameter für Ein-/Ausgänge (2)

Parameter	Zuordnung	Bezeichnung	Beschreibung	Signal- pegel ^①	Werkseinstellung ^②
STOP2	Eingang	Stoppsignal	Stoppt die ausgeführten Programme Die Funktion entspricht der des STOP-Parameters. Im Gegensatz zum STOP-Parameter können jedoch die Signalnummern geändert werden.	H	-1
	Ausgang	Wartestatus aktiv	Zeigt an, dass die Abarbeitung des entsprechenden Programms vorübergehend unterbrochen worden ist Die Funktion entspricht der des STOP-Parameters.		-1
STOPSTS	Eingang	—	—		-1 (unwirksam)
	Ausgang	Eingabe des Stoppsignals	Zeigt an, dass das Stoppsignal eingegeben wurde (logische Addition aller Geräte)		-1
SLOTINIT (Betriebsrechte erforderlich)	Eingang	Programme zurücksetzen	Setzt den Wartestatus der Programme und die Programme selbst zurück Das Zurücksetzen von Programmen ermöglicht eine Programmwahl. Im Multitask-Betrieb werden alle Programmplätze zurückgesetzt. Programmplätze, deren Startbedingung in den Programmplatzparametern „SLT**“ auf „ALWAYS“ oder „ERROR“ gesetzt ist, werden nicht zurückgesetzt.	↑	-1,
	Ausgang	Ausgangssignal Programmwahl freigegeben	Zeigt an, dass die Programmwahl freigegeben ist Das Signal ist eingeschaltet wenn das Programm weder ausgeführt wird noch im Wartezustand ist. Im Multitask-Betrieb ist das Signal eingeschaltet, wenn kein Programm ausgeführt wird oder sich im Wartezustand befindet. Programmplätze, deren Startbedingung in den Programmplatzparametern „SLT**“ auf „ALWAYS“ oder „ERROR“ gesetzt ist, werden nicht freigegeben.		-1
ERRRESET	Eingang	Fehler quittieren	Quittiert den aktuellen Fehler	↑	2
	Ausgang	Ausgangssignal Fehler	Zeigt an, dass ein Fehler aufgetreten ist		2
SRVON (Betriebsrechte erforderlich)	Eingang	Servoversorgung einschalten	Schaltet die Servoversorgung für alle Servos ein		4
	Ausgang	Servoversorgung eingeschaltet	Zeigt an, dass die Servoversorgung eingeschaltet ist		1
SRVOFF	Eingang	Servoversorgung abschalten	Schaltet die Servoversorgung ab, das Einschalten der Servos wird gesperrt	H	1
	Ausgang	Servos einschalten gesperrt	Zeigt an, dass das Einschalten der Servos gesperrt ist (Rückmeldung)		-1

Tab. 10-5: Spezielle Parameter für Ein-/Ausgänge (3)

Parameter	Zuordnung	Bezeichnung	Beschreibung	Signal- pegel ^①	Werkseinstellung ^②
AUTOENA	Eingang	Freigabe Automatikbetrieb	EIN: Automatikbetrieb freigegeben, AUS: Automatikbetrieb gesperrt Wird der Automatikbetrieb im gesperrten Zustand freigegeben, erfolgt die Fehlermeldung L5010. Die Funktion dient zur Verriegelung der E/A-Signale bei Bedienung über das Steuergerät.	H	-1
	Ausgang	Ausgangssignal Automatikbetrieb freigegeben	Zeigt an, dass der Automatikbetrieb freigegeben ist		-1
CYCLE	Eingang	Zyklischen Betrieb stoppen	Stoppt den zyklischen Betrieb	↑	-1
	Ausgang	Ausgangssignal zyklischer Betrieb gestoppt	Zeigt an, dass der zyklische Betrieb gestoppt ist		-1
MELOCK (Betriebs- rechte erforderlich)	Eingang	Verriegelungssignal	Ein- bzw. Auschalten des Verriegelungszustandes Der Signalpegel wird bei Programmauswahl auf „H“ gesetzt.	H	-1
	Ausgang	Ausgangssignal Verriegelung aktiv	Zeigt an, dass der Mechanismus im verriegelten Zustand ist		-1
SAFEPOS (Betriebs- rechte erforderlich)	Eingang	Eingangssignal Rückzugspunkt anfahren	Anfahren des Rückzugspunkts Der im Parameter „JSAFE“ festgelegte Rückzugspunkt wird angefahren. Die Geschwindigkeit wird durch den Übersteuerungswert festgelegt. Achten Sie darauf, Kollisionen mit umliegenden Einheiten zu vermeiden.	↑	-1
	Ausgang	Fährt den Rückzugspunkt an	Zeigt an, dass der Rückzugspunkt angefahren wird		-1
BATERR	Eingang	—	—		-1
	Ausgang	Batteriespannung niedrig	Zeigt an, dass die Batteriespannung abgesunken ist		-1
OUTRESET (Betriebs- rechte erforderlich)	Eingang	Allgemeine Ausgangssignale zurücksetzen	Zurücksetzen der allgemeinen Ausgangssignale	↑	-1
	Ausgang	—	—		-1 (unwirksam)
HLVLERR	Eingang	—	—		-1 (unwirksam)
	Ausgang	Schwerer Fehler	Zeigt an, dass ein schwerer Fehler aufgetreten ist		-1
LLVLERR	Eingang	—	—		-1 (unwirksam)
	Ausgang	Leichter Fehler	Zeigt an, dass ein leichter Fehler aufgetreten ist		-1
CLVLERR	Eingang	—	—		-1 (unwirksam)
	Ausgang	Warnung	Zeigt eine Warnung an		-1

Tab. 10-5: Spezielle Parameter für Ein-/Ausgänge (4)

Parameter	Zuordnung	Bezeichnung	Beschreibung	Signal-pegel ^①	Werkseinstellung ^②
EMGERR	Eingang	—	—		-1 (unwirksam)
	Ausgang	Ausgangssignal NOT-HALT	Zeigt den NOT-HALT-Status an		-1
SnSTART ($1 \leq n \leq 32$) (Betriebsrechte erforderlich)	Eingang	Starteingang Programmplatz n	Startet das entsprechende Programm	↑	-1
	Ausgang	Programm in Programmplatz n aktiv	Zeigt den aktiven Status jedes Programms an		-1
SnSTOP ($1 \leq n \leq 32$) (Betriebsrechte erforderlich)	Eingang	Stoppeingang für Programmplatz n	Stoppt das entsprechende Programm	H	-1
	Ausgang	Programm in Programmplatz n gestoppt	Zeigt an, dass das Programm vorübergehend gestoppt wurde		-1
MnSRVOFF ($1 \leq n \leq 3$)	Eingang	Eingangssignal Servo AUS für Mechanismus n	Schaltet die Servoversorgung jedes Mechanismus aus und sperrt das Einschalten	H	-1
	Ausgang	Ausgangssignal Servo EIN bei Mechanismus n gesperrt	Zeigt an, dass das Einschalten der Servoversorgung gesperrt ist (Rückmeldung)		-1
MnSRVON ($1 \leq n \leq 3$) (Betriebsrechte erforderlich)	Eingang	Eingangssignal Servo EIN für Mechanismus n	Schaltet die Servoversorgung jedes Mechanismus ein	↑	-1
	Ausgang	Ausgangssignal Servo EIN bei Mechanismus n	Zeigt an, dass die Servoversorgung eingeschaltet ist		-1
MnMELOCK ($1 \leq n \leq 3$) (Betriebsrechte erforderlich)	Eingang	Eingangssignal Mechanismus n verriegeln	Schaltet die Verriegelung jedes Roboters aus oder ein	↑	-1
	Ausgang	Ausgangssignal Mechanismus n verriegelt	Zeigt an, dass der Mechanismus im Verriegelungszustand ist		-1
PRGSEL (Betriebsrechte erforderlich)	Eingang	Programmwahlsignal	Einlesen der numerischen Eingabe zur Programmwahl Das Signal sollte etwa 30 ms nach Ausgabe der numerischen Eingangsdaten „IOWDATA“ für mindestens 30 ms an den Roboter ausgegeben werden.	↑	-1 ^①
	Ausgang	—	—		-1 (unwirksam)
OVRDSEL	Eingang	Geschwindigkeitsübersteuerung auswählen	Einlesen der numerischen Geschwindigkeitsübersteuerung Das Signal sollte etwa 30 ms nach Ausgabe der numerischen Eingangsdaten „IOWDATA“ für mindestens 30 ms an den Roboter ausgegeben werden.	↑	-1 ^①
	Ausgang	—	—		-1 (unwirksam)

Tab. 10-5: Spezielle Parameter für Ein-/Ausgänge (5)

Parameter	Zuordnung	Bezeichnung	Beschreibung	Signalpegel ^①	Werkseinstellung ^②
IODATA ^③	Eingang	Eingang für numerische Eingabe (Start-Nr., End-Nr.)	<p>Numerische Eingaben werden als binäre Werte eingelesen.</p> <ul style="list-style-type: none"> ● Programmnummer (Einlesen über PRGSEL) <p>Ist der Parameter PST freigegeben, erfolgt das Einlesen mit dem Startsignal.</p> <ul style="list-style-type: none"> ● Geschwindigkeitsübersteuerung (Einlesen über OVRDSEL) <p>Die Bitbreite kann beliebig gewählt werden. Die Genauigkeit sinkt bei Überschreitung des durch die Bitbreite möglichen Wertebereiches.</p> <p>Das Eingangssignal sollte 30 ms vor Eingabe des Signals PRGSEL oder anderer Einstellsignale anliegen.</p>	H	-1 (Startbit), -1 (Endbit)
	Ausgang	Ausgang für numerische Ausgabe (Start-Nr., End-Nr.)	<p>Numerische Eingaben werden als binäre Werte ausgegeben.</p> <ul style="list-style-type: none"> ● Programmnummer (Ausgabe über PRGOUT) ● Geschwindigkeitsübersteuerung (Ausgabe über OVRDOUT) ● Zeilennummer (Ausgabe über LINEOUT) ● Fehlernummer (Ausgabe über ERROUT) <p>Die Bitbreite kann beliebig gewählt werden. Die Genauigkeit sinkt bei Überschreitung des durch die Bitbreite möglichen Wertebereiches.</p> <p>Das Signal sollte 30 ms nach Eingabe der Programmnummer (PRGOUT) oder anderer Signale eingelesen werden.</p>		-1 (Startbit), -1 (Endbit)

Tab. 10-5: Spezielle Parameter für Ein-/Ausgänge (6)

Parameter	Zuordnung	Bezeichnung	Beschreibung	Signalpegel ^①	Werkseinstellung ^②
DIODATA	Eingang	Eingang für numerische Eingabe (Register-Nr.)	<p>Die festgelegten numerischen Eingaben werden eingelesen. Als numerische Werte können reele Zahlen mit 16 Bits eingelesen werden. Wie beim Parameter IODATA ist das Einlesen einer Programmnummer, einer Geschwindigkeitsübersteuerung und anderer Werte in ein Register möglich.</p> <p>Das Eingangssignal sollte 30 ms vor Eingabe des Signals PRGSEL oder OVRDSEL anliegen.</p> <p>HINWEIS: Dieser Parameter ist ausschließlich zum Einsatz in einem CC-Link-Netzwerk vorgesehen. Die im Parameter IODATA eingegebenen numerischen Werte besitzen eine höhere Priorität.</p>	H	-1
	Ausgang	Ausgang für numerische Ausgabe (Register-Nr.)	<p>Die festgelegten numerischen Werte werden ausgegeben. Als numerische Werte können reele Zahlen mit 16 Bits ausgegeben werden. Wie beim Parameter IODATA ist die Ausgabe einer Programmnummer, einer Geschwindigkeitsübersteuerung, einer Zeilennummer, einer Fehlernummer und anderer Werte an ein Register möglich. Bei Eingabe der Zeilennummer (LINEOUT) und der Fehlernummer (ERROUT), werden diese Werte an die festgelegten Register ausgegeben. Bis zum Laden der Werte sollten etwa 30 ms vergehen.</p> <p>HINWEIS: Dieser Parameter ist ausschließlich zum Einsatz in einem CC-Link-Netzwerk vorgesehen. Die im Parameter IODATA und in diesem Parameter eingegebenen numerischen Werte werden ausgegeben.</p>		-1

Tab. 10-5: Spezielle Parameter für Ein-/Ausgänge (7)

Parameter	Zuordnung	Bezeichnung	Beschreibung	Signal-pegel ^①	Werkseinstellung ^②
PRGOUT ^①	Eingang	Ausgabeanforderung Programmnummer	Anforderung zur Ausgabe der Programmnummer über den numerischen Ausgang (IOWDATA) Nach Eingabe des Signals müssen mindestens 30 ms bis zum Einlesen des numerischen Ausgangssignals (IOWDATA) vergangen sein.	↑	-1
	Ausgang	Ausgabe der Programmnummer	Zeigt an, dass die Programmnummer über den numerischen Ausgang ausgegeben wird		-1
LINEOUT ^①	Eingang	Ausgabeanforderung Zeilennummer	Anforderung zur Ausgabe der Zeilennummer über den numerischen Ausgang (IOWDATA) Nach Eingabe des Signals müssen mindestens 30 ms bis zum Einlesen des numerischen Ausgangssignals (IOWDATA) vergangen sein.	↑	-1
	Ausgang	Ausgabe der Zeilennummer	Zeigt an, dass die Zeilennummer über den numerischen Ausgang ausgegeben wird		-1
OVRDOUT ^①	Eingang	Ausgabeanforderung Geschwindigkeitsübersteuerung	Anforderung zur Ausgabe der Geschwindigkeitsübersteuerung über den numerischen Ausgang (IOWDATA) Nach Eingabe des Signals müssen mindestens 30 ms bis zum Einlesen des numerischen Ausgangssignals (IOWDATA) vergangen sein.	↑	-1
	Ausgang	Ausgabe der Geschwindigkeitsbeeinflussung	Zeigt an, dass die Geschwindigkeitsübersteuerung über den numerischen Ausgang ausgegeben wird		-1
ERRROUT ^①	Eingang	Ausgabeanforderung Fehlernummer	Anforderung zur Ausgabe der Fehlernummer über den numerischen Ausgang (IOWDATA) Nach Eingabe des Signals müssen mindestens 30 ms bis zum Einlesen des numerischen Ausgangssignals (IOWDATA) vergangen sein.	↑	-1
	Ausgang	Ausgabe der Fehlernummer	Zeigt an, dass die Fehlernummer über den numerischen Ausgang ausgegeben wird		-1
JOGENA (Betriebsrechte erforderlich)	Eingang	Freigabe JOG-Betrieb	Freigabe des JOG-Betriebs über externe Signale	H	-1
	Ausgang	Freigabe JOG-Betrieb	Zeigt an, dass der JOG-Betrieb über externe Signale freigegeben ist		-1
JOGM	Eingang	2-Bit-Eingabe des JOG-Betriebs (Start-Nr., End-Nr.)	Festlegung des JOG-Betriebs 0/1/2/3/4 = Gelenk-, XYZ-, Kreis-, 3-Achsen-XYZ-, Werkzeug-JOG-Betrieb	H	-1 (Startbit), -1 (Endbit)
	Ausgang	2-Bit-Ausgabe des JOG-Betriebs (Start-Nr., End-Nr.)	Ausgabe des aktuellen JOG-Betriebs		-1 (Startbit), -1 (Endbit)

Tab. 10-5: Spezielle Parameter für Ein-/Ausgänge (8)

Parameter	Zuordnung	Bezeichnung	Beschreibung	Signal- pegel ^①	Werksein- stellung ^②
JOG+ ^⑤	Eingang	JOG-Vorschub in positiver Richtung für 8 Achsen (Start-Nr., End-Nr.)	Festlegung des JOG-Betriebs in positiver Richtung beginnend mit der Startnummer Gelenk-JOG-Betrieb: J1, J2, J3, J4, J5, J6, J7, J8 XYZ-JOG-Betrieb: X, Y, Z, A, B, C, L1, L2 Kreis-JOG-Betrieb: X, Y, Z, A, B, C, L1, L2 3-Achsen-XYZ-JOG-Betrieb: X, Y, Z, J4, J5, J6 Werkzeug-JOG-Betrieb: X, Y, Z, A, B, C	H	-1, -1
	Ausgang	—	—		
JOG- ^⑤	Eingang	JOG-Vorschub in negativer Richtung für 8 Achsen (Start-Nr., End-Nr.)	Festlegung des JOG-Betriebs in negativer Richtung beginnend mit der Startnummer Gelenk-JOG-Betrieb: J1, J2, J3, J4, J5, J6, J7, J8 XYZ-JOG-Betrieb: X, Y, Z, A, B, C, L1, L2 Kreis-JOG-Betrieb: X, Y, Z, A, B, C, L1, L2 3-Achsen-XYZ-JOG-Betrieb: X, Y, Z, J4, J5, J6 Werkzeug-JOG-Betrieb: X, Y, Z, A, B, C	H	-1, -1
	Ausgang	—	—		
JOGNER (Betriebs- rechte erforderlich)	Eingang	Fehler im JOG-Betrieb temporär ignorieren	Eingangssignal zum temporären Ignorieren von Fehlern, die im JOG-Betrieb nicht zurückgesetzt werden können Der Eingang kann ausschließlich für Mechanismus 1 geschaltet werden und ist ab Software-Version J2 verfügbar.	H	-1
	Ausgang	Fehler im JOG-Betrieb temporär ignorieren	Zeigt das temporäre Ignorieren von Fehlern, die im JOG-Betrieb nicht zurückgesetzt werden können, an Der Ausgang kann ausschließlich von Mechanismus 1 geschaltet werden und ist ab Software-Version J2 verfügbar.		-1
HNDCTRLn (1 ≤ n ≤ 3)	Eingang	—	—		
	Ausgang	Handsteuersignal für Hand Mechanismus n (Start-Nr., End-Nr.)	Ausgabe der Signalzustände der Handausgänge (n = 1) 900 bis 907 Ausgabe der Signalzustände der Handausgänge (n = 2) 910 bis 917 Ausgabe der Signalzustände der Handausgänge (n = 3) 920 bis 927 Beispiel: Zur Ausgabe der 4 Signale 900 bis 903 als allgemeine Ausgangssignale 3, 4, 5 und 6 setzen Sie HNDCTRL1 auf (3, 6)		-1 (Startbit), -1 (Endbit)

Tab. 10-5: Spezielle Parameter für Ein-/Ausgänge (9)

Parameter	Zuordnung	Bezeichnung	Beschreibung	Signal-pegel ^①	Werkseinstellung ^②
HNDSTSn (1 ≤ n ≤ 3)	Eingang	—	—		
	Ausgang	Handsensorsignal für Hand Mechanismus <i>n</i> (Start-Nr., End-Nr.)	Ausgabe der Signalzustände der Handeingänge (n = 1) 900 bis 907 Ausgabe der Signalzustände der Handeingänge (n = 2) 910 bis 917 Ausgabe der Signalzustände der Handeingänge (n = 3) 920 bis 927 Beispiel: Zur Ausgabe der 4 Signale 900 bis 903 als allgemeine Ausgangssignale 3, 4, 5 und 6 setzen Sie HNDSTS1 auf (3, 6)		-1 (Startbit), -1 (Endbit)
HNDERRn (1 ≤ n ≤ 3))	Eingang	Eingangssignal Fehler Hand <i>n</i>	Abfrage auf Handfehler Ein leichter Fehler (Fehlernummer 30) wird generiert.	H	-1
	Ausgang	Ausgangssignal Fehler Hand <i>n</i>	Zeigt an, dass ein Handfehler aufgetreten ist Ein leichter Fehler (Fehlernummer 31) wird generiert.		-1
AIRERRn (1 ≤ n ≤ 3)	Eingang	Luftdruck im Pneumatiksystem <i>n</i> fehlerhaft	Abfrage auf Pneumatikfehler	H	-1
	Ausgang	Ausgabe Pneumatikfehler im System <i>n</i>	Zeigt an, dass ein Fehler im Pneumatiksystem aufgetreten ist		-1
USRAREA ^⑥ (siehe auch Abschn. 9.9)	Eingang	—	—		
	Ausgang	Über 8 Punkte festgelegter Arbeitsbereich (Start-Nr., End-Nr.)	Zeigt an, dass sich der Roboter im Arbeitsbereich befindet Die Ausgabe erfolgt der Reihe nach für die Bereiche 1, 2 und 3 beginnend mit der Startnummer. Die Festlegung des Bereiches erfolgt über die Parameter AREA1P1, AREA1P2 bis AREA8P1 und AREA8P2. Beispiele: für Bereich 1 → USRAREA: 8, 8 für Bereich 1 und 2 → USRAREA: 8, 9 Nicht erlaubt sind Einstellungen wie: USRAREA: -1, -1 (ungültig) USRAREA: 8, -1 (ungültig/ keine Fehlermeldung) USRAREA: -1, 8 (ungültig/ keine Fehlermeldung) USRAREA: 9, 8 (ungültig/ Fehlermeldung H6643)		-1 (Startbit), -1 (Endbit)

Tab. 10-5: Spezielle Parameter für Ein-/Ausgänge (10)

Parameter	Zuordnung	Bezeichnung	Beschreibung	Signal- pegel ^①	Werkseinstellung ^②
MnPTEXC ($1 \leq n \leq 3$)	Eingang	—	—		-1 (unwirksam)
	Ausgang	Warnmeldung Wartungsintervall abgelaufen	Zeigt an, dass das Wartungsintervall abgelaufen ist und Verschleißteile erneuert werden müssen	H	-1
MnWUPENA ($1 \leq n \leq 3$) (Betriebsrechte erforderlich)	Eingang	Freigabe des Warmlaufbetriebs für Mechanismus n	Gibt den Warmlaufbetrieb für den festgelegten Mechanismus frei. HINWEIS: Damit der Warmlaufbetrieb über dieses Eingangssignal freigegeben bzw. gesperrt werden kann, muss der Warmlaufbetrieb über den Parameter WUPENA freigegeben sein. Ansonsten bewirkt eine Eingabe des Signals keine Freigabe des Warmlaufbetriebs.	H	-1
	Ausgang	Warmlaufbetrieb für Mechanismus n freigegeben	Zeigt an, dass der Warmlaufbetrieb freigegeben ist		-1
MnWUPMD ($1 \leq n \leq 3$)	Eingang	—	—		-1 (unwirksam)
	Ausgang	Ausgangssignal Mechanismus n im Warmlaufbetrieb	Zeigt an, dass sich der Roboter im Warmlaufbetrieb befindet und mit reduzierter Geschwindigkeit verahren wird	H	-1

Tab. 10-5: Spezielle Parameter für Ein-/Ausgänge (11)

Beachten Sie auch die Fußnoten auf der nächsten Seite.

- ① Eingang: „H“ bedeutet, die Funktion ist aktiv, wenn das externe Signal eingeschaltet ist und inaktiv, wenn das externe Signal ausgeschaltet ist. Das Signal muss mindestens 30 ms eingeschaltet sein.
 Eingang: „↑“ bedeutet, die Funktion ist aktiv, wenn das externe Signal vom AUS- in den EIN-Zustand wechselt. Die aktivierte Funktion bleibt auch nach einem Wechsel des externen Signals in den AUS-Zustand erhalten.
 Bsp.: Bei einer Einstellung der Startbedingung im Programmplatzparameter auf START (CYC, ERROR usw.) wird das Programm mit steigender Flanke gestartet. Das Programm wird nicht mit steigender Flanke gestoppt.

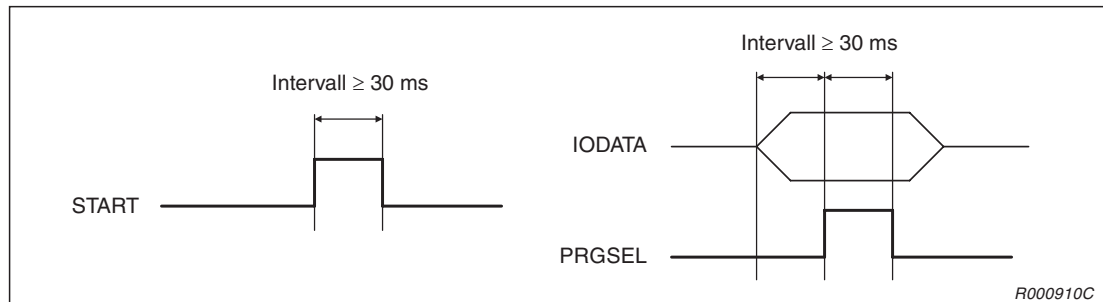


Abb. 10-2: Signaldauer

- ② Die Werkseinstellung „-1“ bedeutet, dass die Funktion nicht aktiviert ist.
 ③ Die Eingabe erfolgt in der Reihenfolge: Eingangsstartnummer, Eingangsendnummer, Ausgangsstartnummer, Ausgangsendnummer. Geben Sie bei einer Ein-/Ausgabe eines aktuellen Wertes die Start- und Endnummer als binären Wert an. Dabei entspricht die Startnummer dem niederwertigen, die Endnummer dem höherwertigen Bit. Setzen Sie nur die zur Einstellung notwendigen Werte. Stehen z. B. bei einer Programmwahl nur die Programme 1 bis 6 zur Auswahl, reichen zur Darstellung 3 Bits. Es können bis zu 16 Bits gesetzt werden.

Beispiele ▾

Die Zuweisung des Starteingangssignals an Eingang 16 und des Ausgangssignals „Programm aktiv“ an Ausgang 25 erfolgt über:
 Parameter START = [16, 25]

Die Zuweisung von 4 Bits der numerische Eingabe an die Eingänge 6 bis 9 und von 5 Bits der numerischen Ausgabe an die Ausgänge 6 bis 10 erfolgt über:
 Parameter IODATA = [6, 9, 6, 10]

- ④ Die Eingabe erfolgt in der Reihenfolge: Eingangsstartnummer, Eingangsendnummer, Ausgangsstartnummer, Ausgangsendnummer. Geben Sie bei Aktivierung des aktuellen JOG-Modus die Start- und Endnummer als binären Wert an. Dabei entspricht die Startnummer dem niederwertigen, die Endnummer dem höherwertigen Bit. Setzen Sie nur die zur Einstellung notwendigen Werte. △
- ⑤ Die Eingabe erfolgt in der Reihenfolge: Eingangsstartnummer, Eingangsendnummer. Über die Startnummer wird die Achse J1/X festgelegt und über die Endnummer können Achsen bis zu J8/L2 festgelegt werden.
- ⑥ Die Eingabe erfolgt in der Reihenfolge: Ausgangsstartnummer, Ausgangsendnummer. Über die Startnummer wird der Bereich 1, über die Endnummer maximal der Bereich 8 festgelegt. Die Festlegung zweier Benutzerbereiche erfolgt über zwei Bits. Es können maximal 8 Bits gesetzt werden.

Freigabe der zugewiesenen Eingangssignale

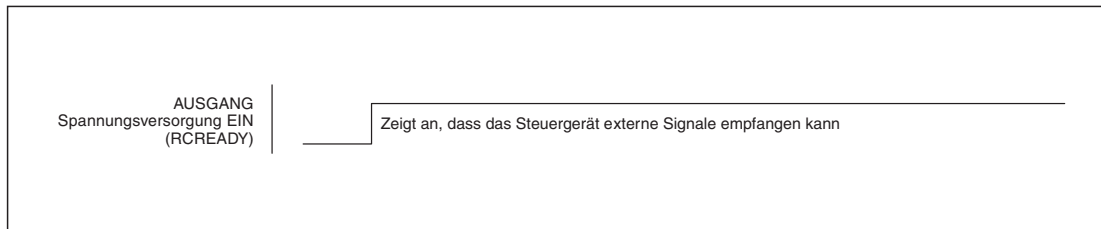
Die Gültigkeit eines anliegenden und zugewiesenen Eingangssignals hängt vom Betriebszustand des Roboters ab.

Parameter	Bezeichnung	Gültigkeit	
SLOTINIT	Programme zurücksetzen	Keine Funktion während des Betriebs (bei Ausgabe des START-Signals)	
SAFEPOS	Eingangssignal Ersatzposition anfahren		
OUTRESET	Allgemeine Ausgangssignale zurücksetzen		
PRGSEL	Programmwahlsignal		
MnWUPENA	Eingangssignal zur Freigabe des Warmlaufbetriebs für Mechanismus n		
START SnSTART (1 ≤ n ≤ 32)	Startsignal	Funktion nur bei Ausgabe des IOENA-Signals	
SLOTINIT	Programme zurücksetzen		
SRVON MnSRVON (1 ≤ n ≤ 3)	Servoversorgung einschalten		
MELOCK MnMELOCK (1 ≤ n ≤ 3)	Verriegelungssignal		
SAFEPOS	Eingangssignal Ersatzposition anfahren		
PRGSEL	Programmwahlsignal		
OVRDSEL	Geschwindigkeitsübersteuerung auswählen		
JOGENA	Freigabe JOG-Betrieb		
MnWUPENA	Eingangssignal zur Freigabe des Warmlaufbetriebs für Mechanismus n		
SLOTINIT	Programme zurücksetzen		Keine Funktion bei Eingabe des Stoppsignals (bei Ausgabe des STOPSTS-Signals)
SAFEPOS	Eingangssignal Ersatzposition anfahren		
JOGENA	Freigabe JOG-Betrieb		
SRVON	Servoversorgung einschalten		Keine Funktion bei eingeschaltetem SRVOFF-Signal
MELOCK	Verriegelungssignal	Funktion nur im Programmauswahlmodus (bei Ausgabe des SLOTINIT-Signals)	

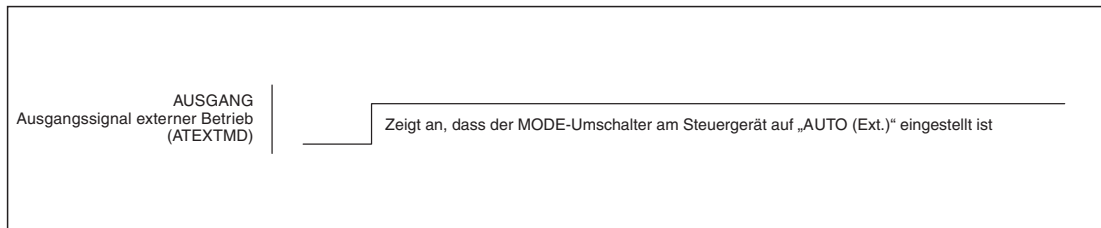
Tab. 10-6: Gültigkeit der Eingangssignale

Zeitablaufdiagramme externer Signale

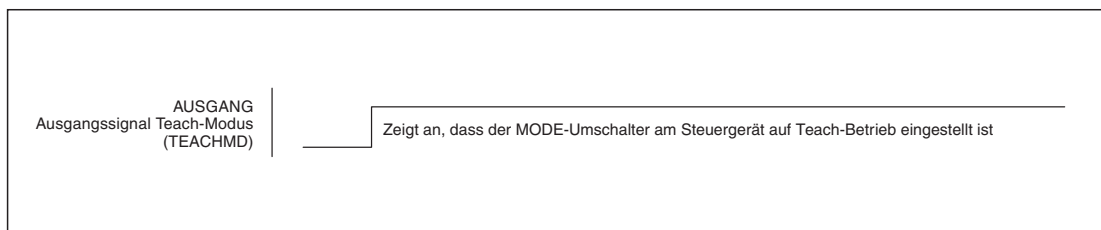
- RCREADY (Ausgang Spannungsversorgung des Steuergerätes eingeschaltet)



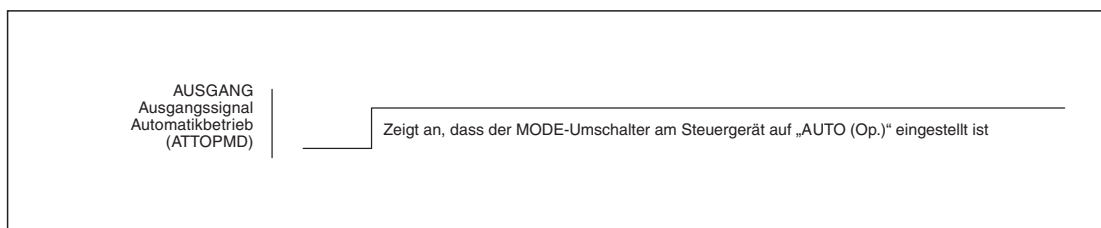
- ATEXTMD (Ausgang externer Betrieb)



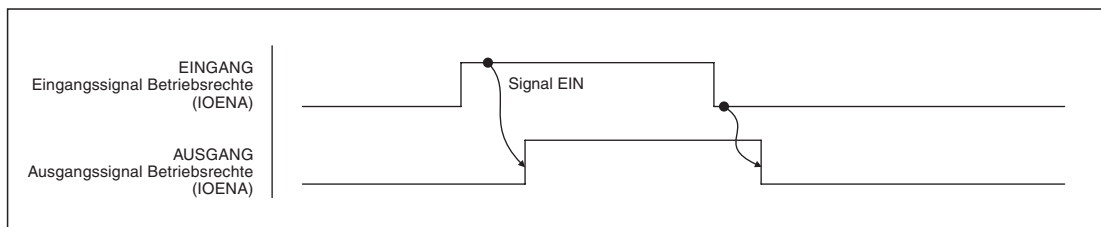
- TEACHMD (Ausgangssignal Teach-Modus)



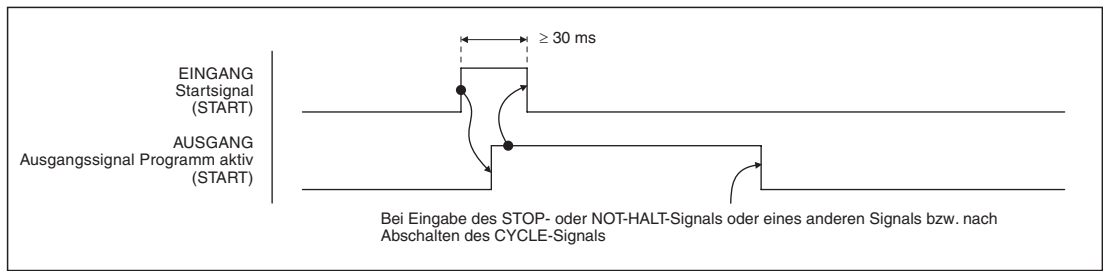
- ATTOPMD (Ausgang Automatikbetrieb)



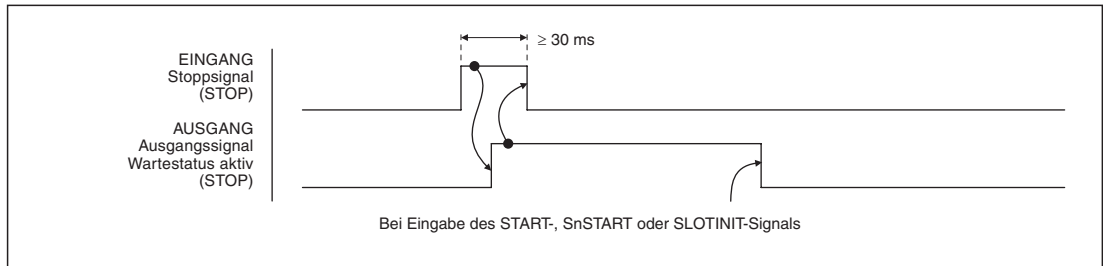
- IOENA (Eingang Betriebsrechte/Ausgang Betriebsrechte)



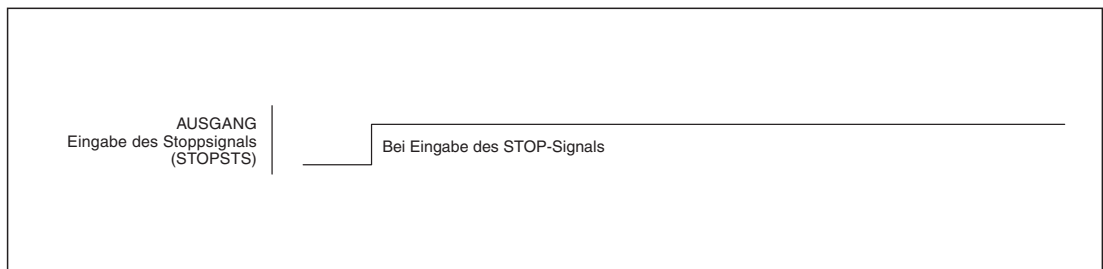
● **START (Eingang Startsignal/Ausgang Programm aktiv)**



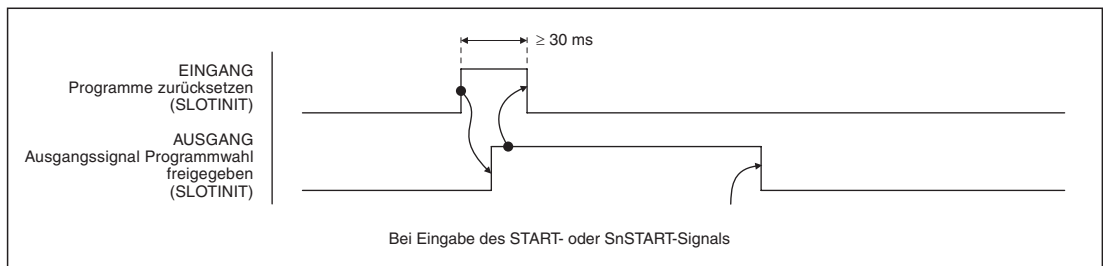
● **STOP (Eingang Stoppsignal/Ausgang Wartestatus aktiv)**



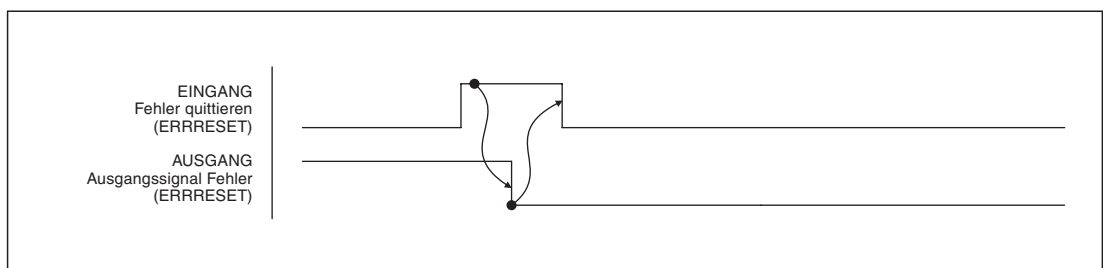
● **STOPSTS (Ausgang Eingabe des Stoppsignals)**



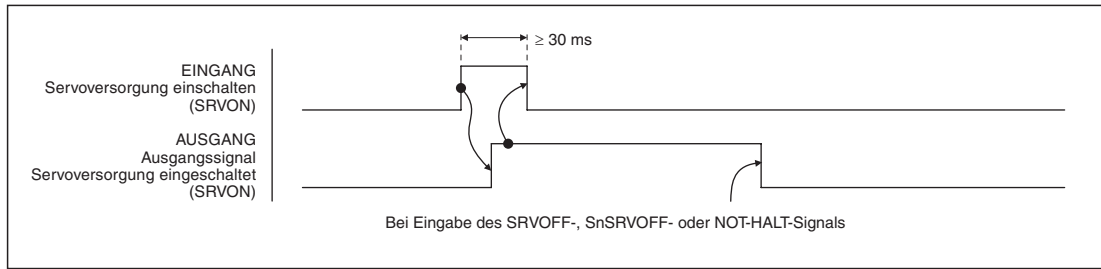
● **SLOTINIT (Eingang Programme zurücksetzen/Ausgang Programmwahl freigegeben)**



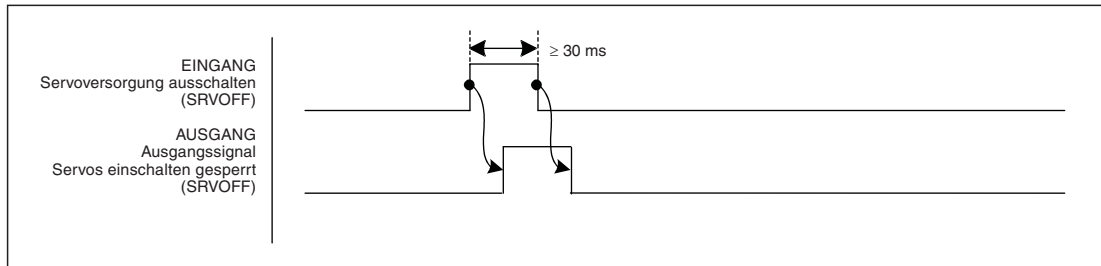
● **ERRRESET (Fehler quittieren/Ausgang Fehler)**



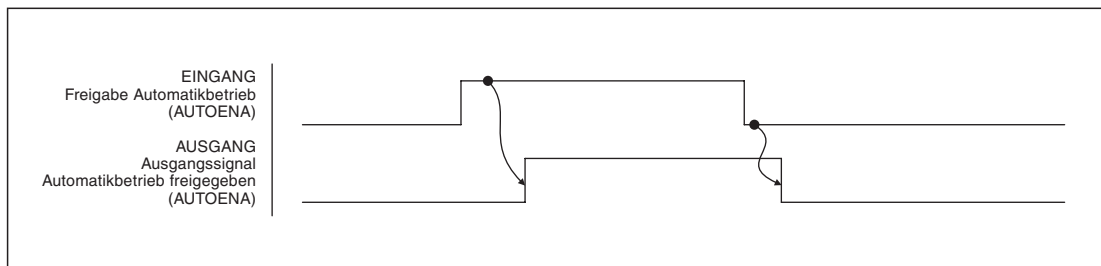
● SRVON (Eingang Servoversorgung einschalten/Ausgang Servoversorgung eingeschaltet)



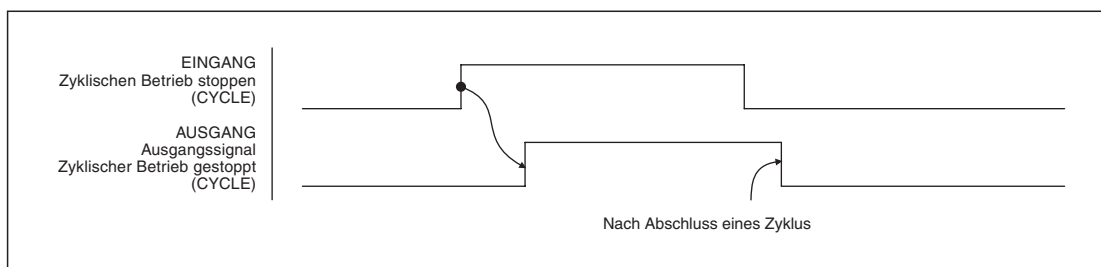
● SRVOFF (Eingang Servoversorgung ausschalten/Ausgang Servos einschalten gesperrt)



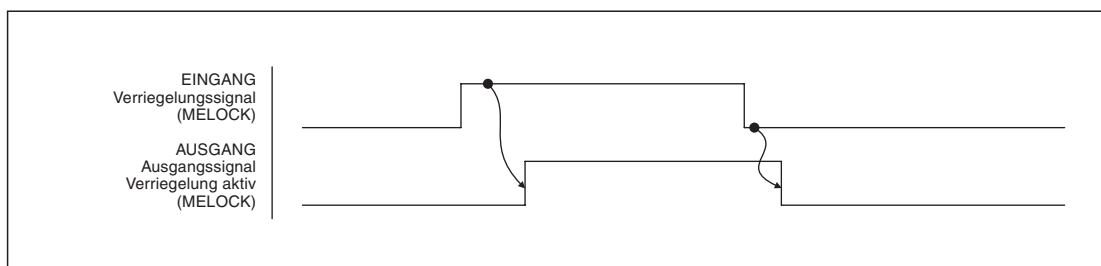
● AUTOENA (Eingang Freigabe Automatikbetrieb/Ausgang Automatikbetrieb freigegeben)



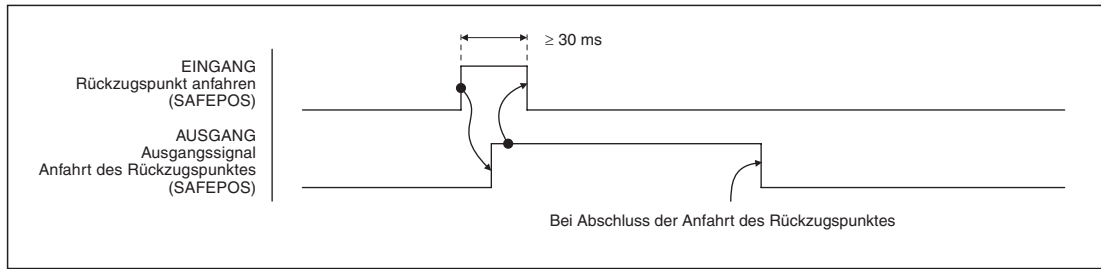
● CYCLE (Eingang zyklischen Betrieb stoppen/Ausgang zyklischer Betrieb gestoppt)



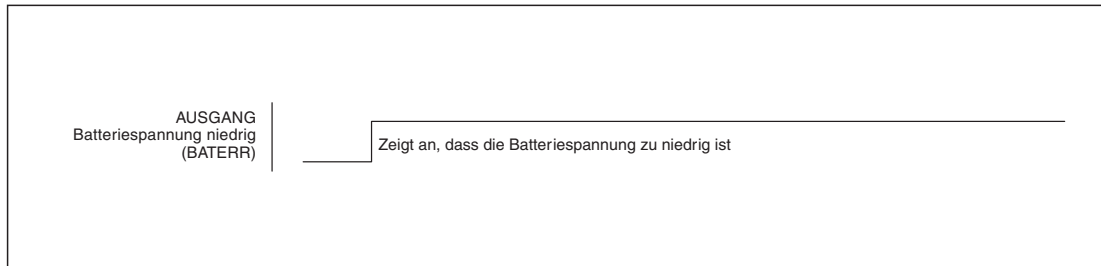
● MELOCK (Eingang Verriegelungssignal/Ausgang Verriegelung aktiv)



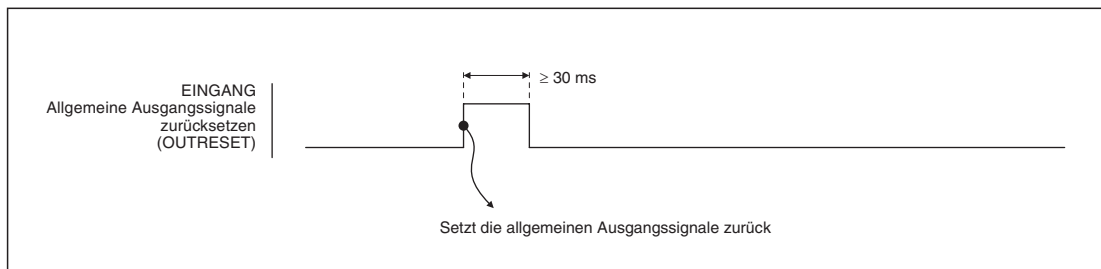
● **SAFEPOS (Eingang Rückzugspunkt anfahren/Ausgang Anfahrt des Rückzugspunktes)**



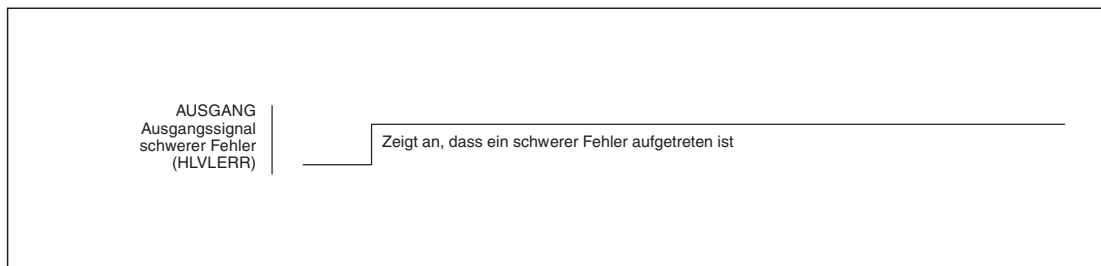
● **BATERR (Ausgang Batteriespannung niedrig)**



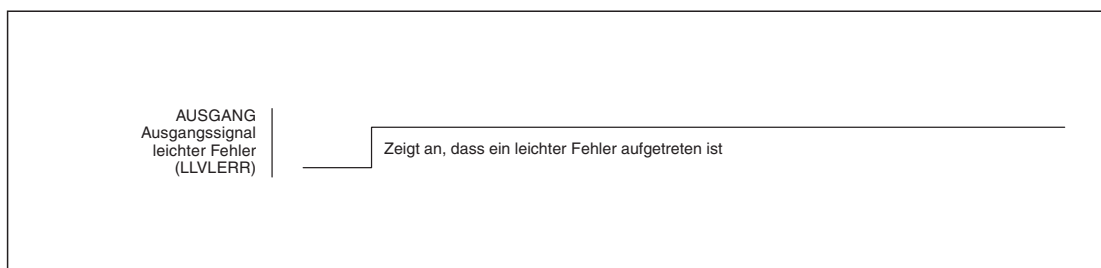
● **OUTRESET (Eingang zum Zurücksetzen der allgemeinen Ausgangssignale)**



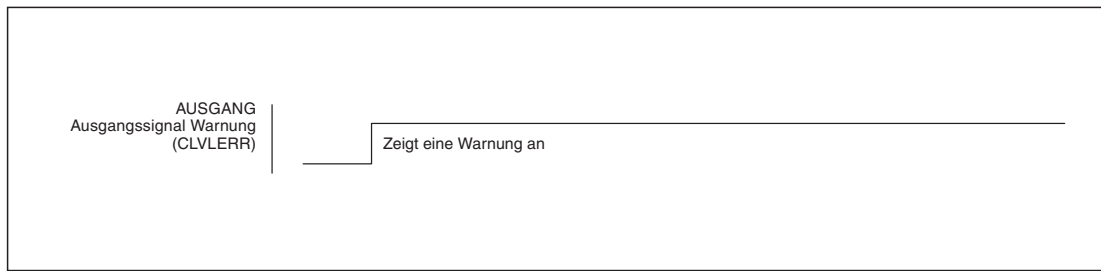
● **HLVLERR (Ausgang zur Anzeige schwerer Fehler)**



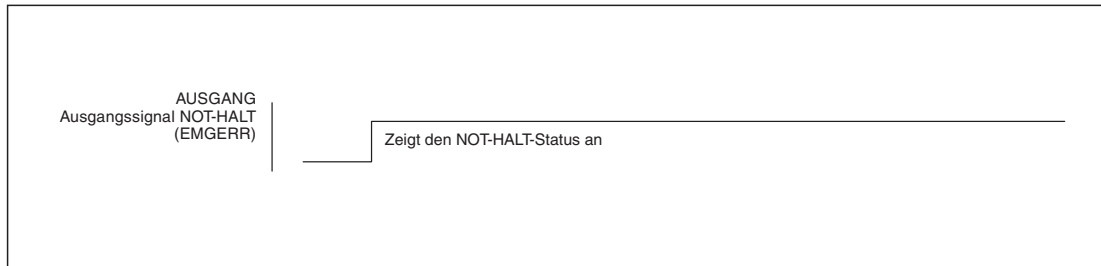
● **LLVLERR (Ausgang zur Anzeige leichter Fehler)**



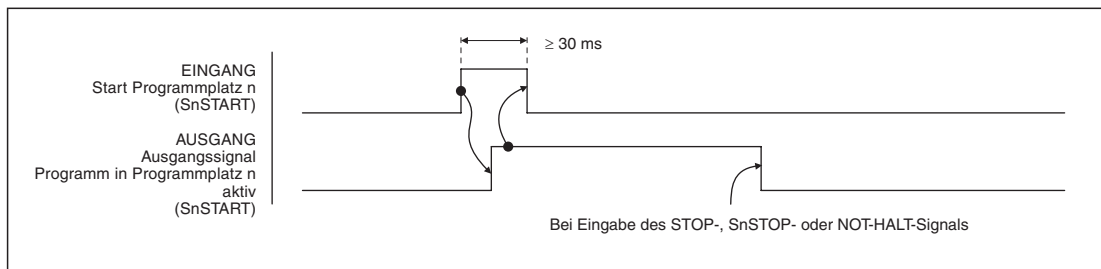
● CLVLERR (Ausgang zur Anzeige von Warnungen)



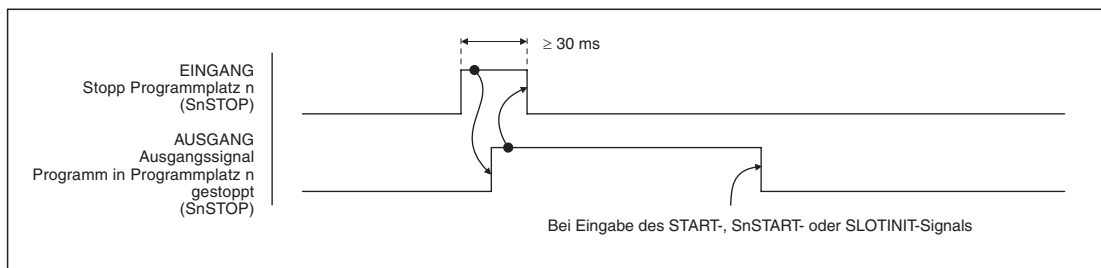
● EMGERR (Ausgang NOT-HALT)



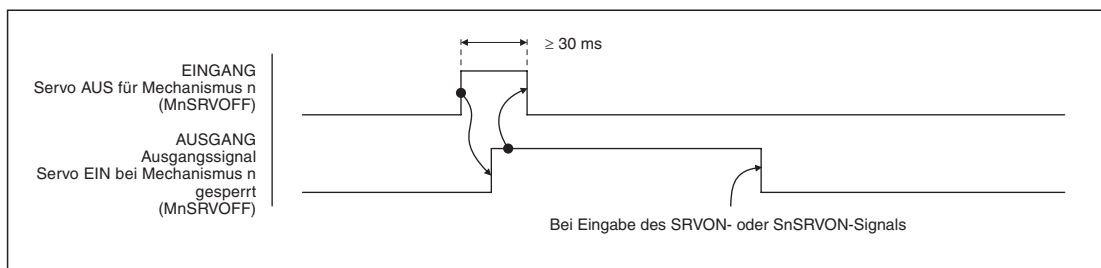
● SnSTART (Startergang Programmplatz n/Ausgang Programm in Programmplatz n aktiv)



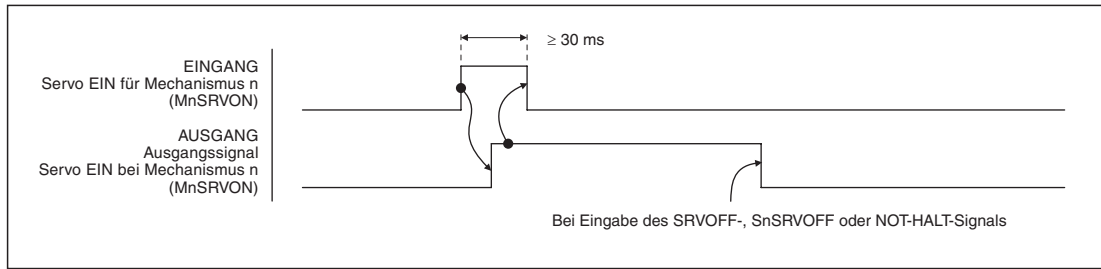
● SnSTOP (Stoppeingang Programmplatz n/Ausgang Programm in Programmplatz n gestoppt)



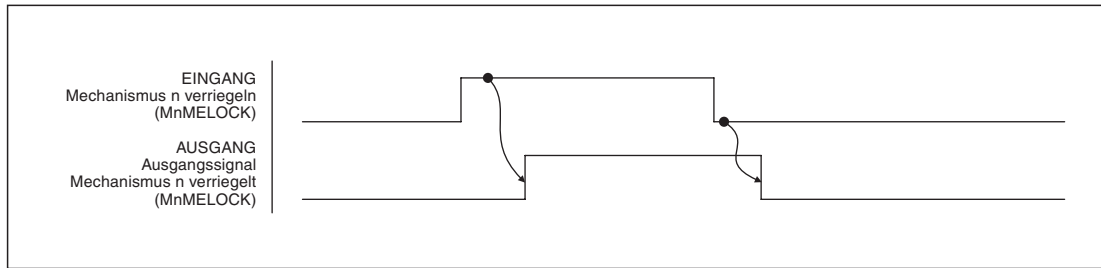
● MnSRVOFF (Eingang Servo AUS für Mechanismus n/Ausgang Servo EIN bei Mechanismus n gesperrt)



● MnSRVON (Eingang Servo EIN für Mechanismus n/Ausgang Servo EIN bei Mechanismus n)

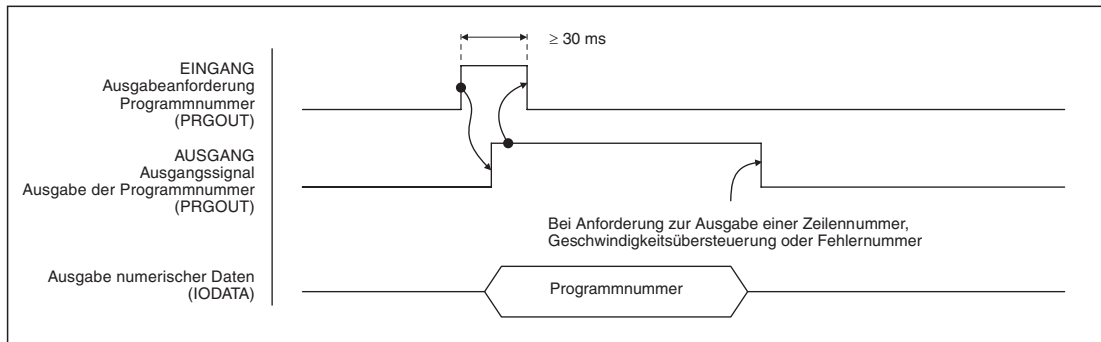


● MnMELOCK (Eingang Mechanismus n verriegeln/Ausgang Mechanismus n verriegelt)



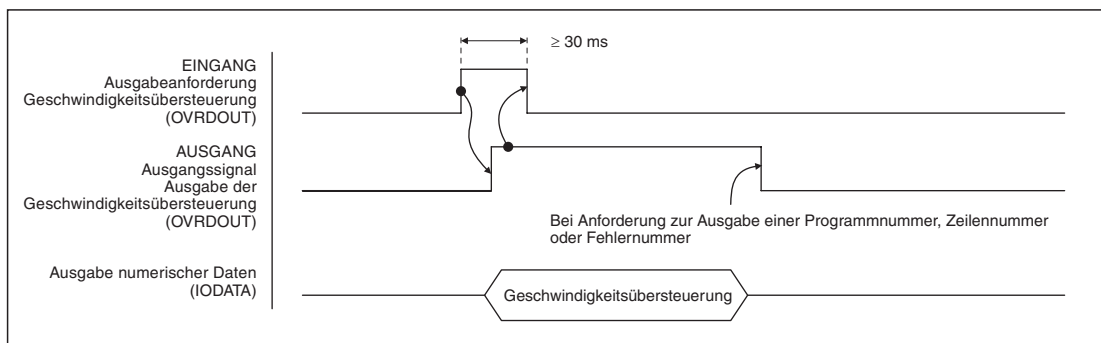
● PRGSEL (Eingang Programmwahlsignal)

Der Eingang wird zusammen mit dem Eingang für numerische Daten „IODETA“ verwendet.



● OVRDSEL (Eingang Geschwindigkeitsübersteuerung auswählen)

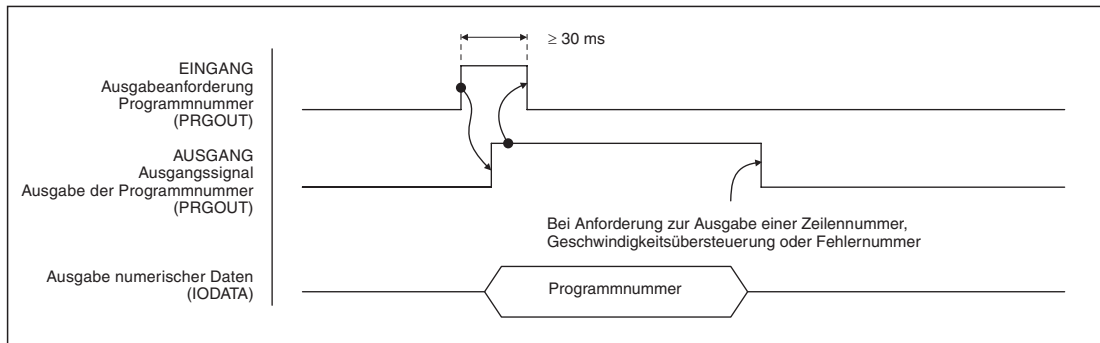
Der Eingang wird zusammen mit dem Eingang für numerische Daten „IODETA“ verwendet.



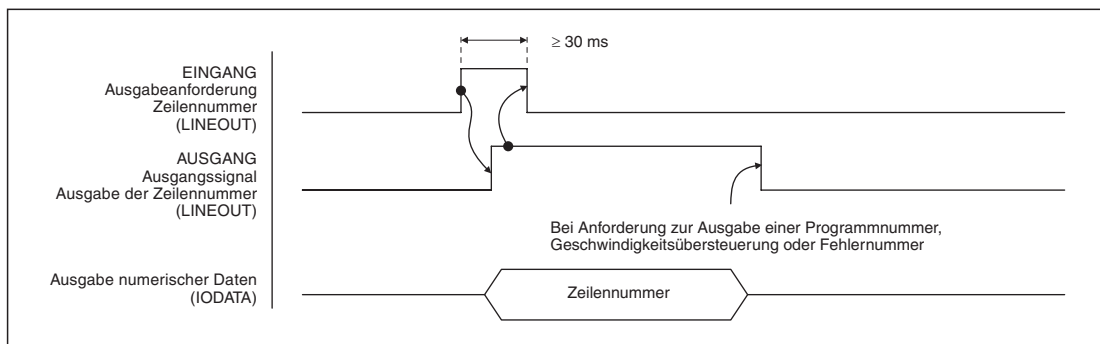
● IODETA (Eingang für numerische Eingabe/Ausgang für numerische Ausgabe)

IODETA wird zusammen mit dem Signal PRGSEL, OVRDSEL, PRGOUT, LINEOUT, OVRDOUT oder ERROUT verwendet.

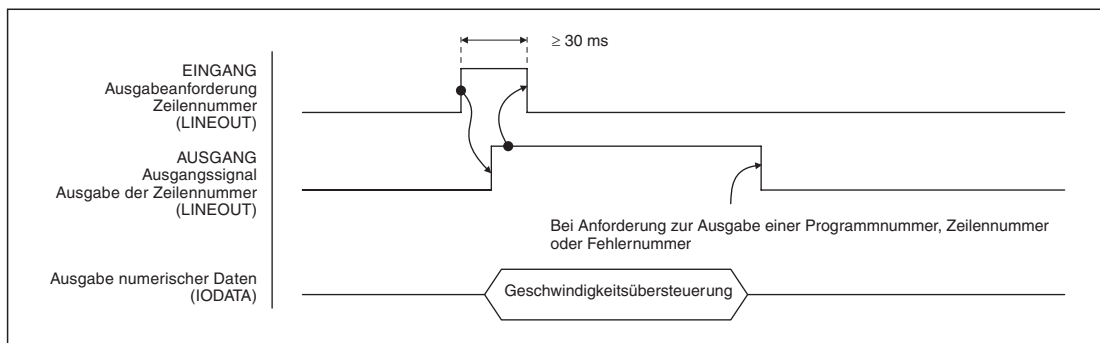
- PRGOUT (Eingang Ausgabeanforderung Programmnummer/Ausgabe der Programmnummer)
PRGOUT wird zusammen mit dem Ausgang für numerische Daten „IOWDATA“ verwendet.



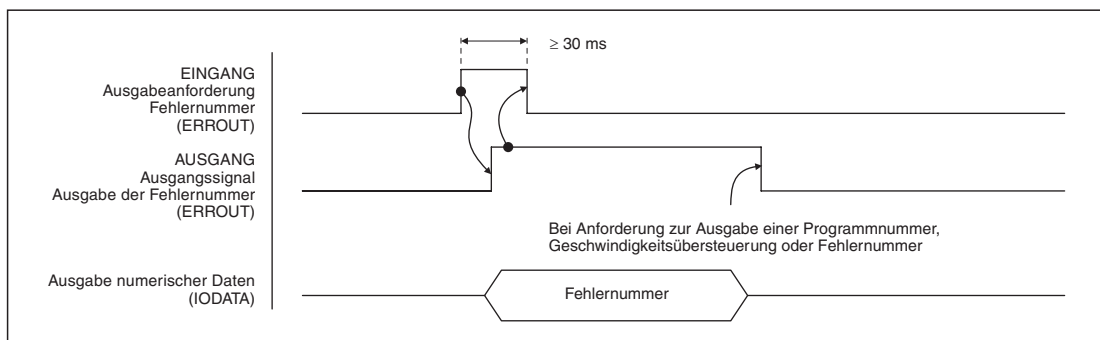
- LINEOUT (Eingang Ausgabeanforderung Zeilennummer/Ausgabe der Zeilennummer)
LINEOUT wird zusammen mit dem Ausgang für numerische Daten „IOWDATA“ verwendet.



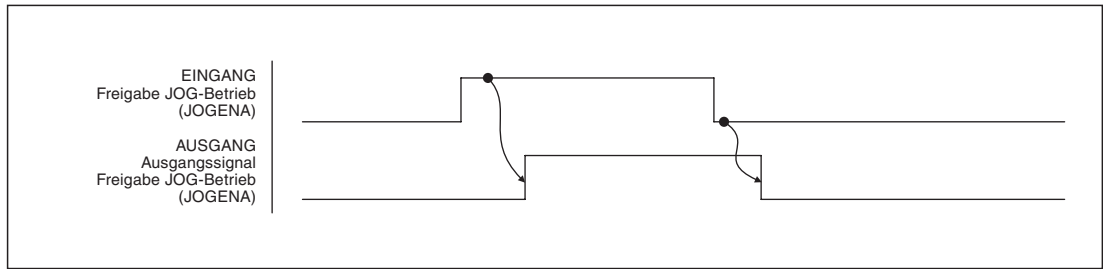
- OVRDOUT (Eingang Ausgabeanforderung Geschwindigkeitsübersteuerung/Ausgabe der Geschwindigkeitsübersteuerung)
OVRDOUT wird zusammen mit dem Ausgang für numerische Daten „IOWDATA“ verwendet.



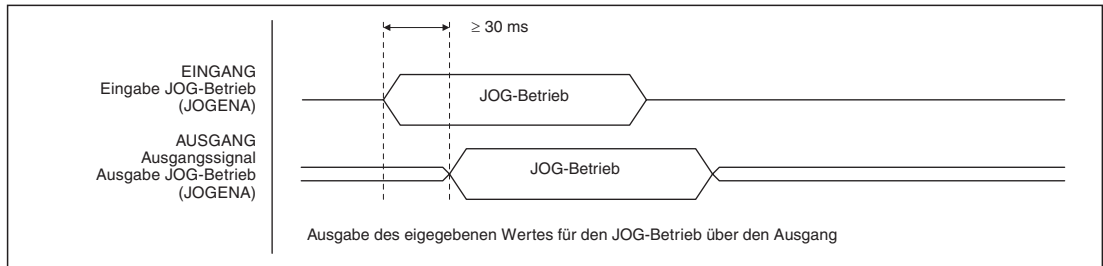
- ERROUT (Eingang Ausgabeanforderung Fehlernummer/Ausgabe der Fehlernummer)
ERROUT wird zusammen mit dem Ausgang für numerische Daten „IOWDATA“ verwendet.



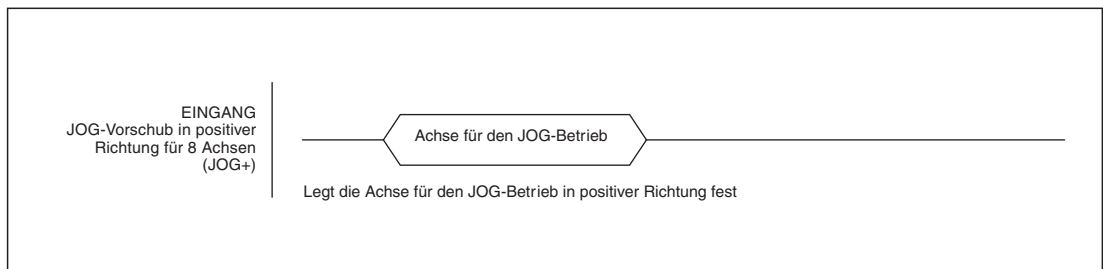
● JOGENA (Eingang Freigabe JOG-Betrieb/Freigabe JOG-Betrieb)



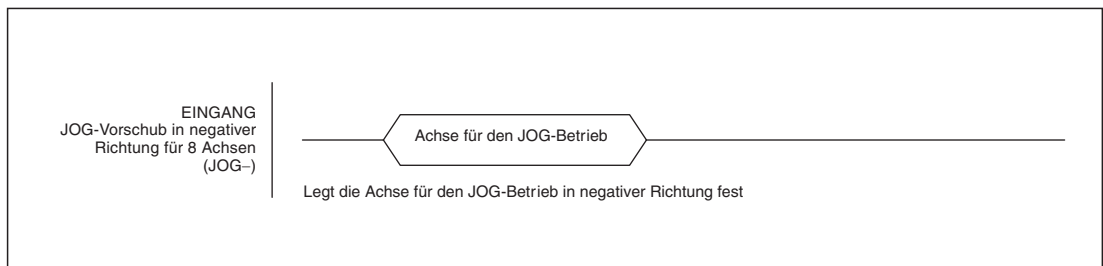
● JOGM (Eingang 2-Bit-Eingabe des JOG-Betriebs/2-Bit-Ausgabe des JOG-Betriebs)



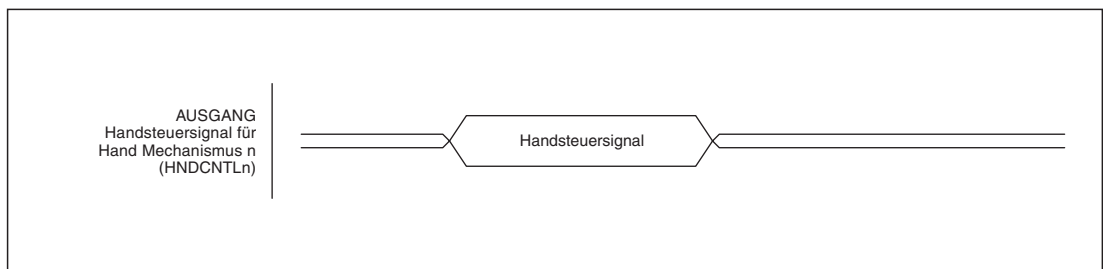
● JOG+ (Eingang JOG-Vorschub in positiver Richtung für 8 Achsen)



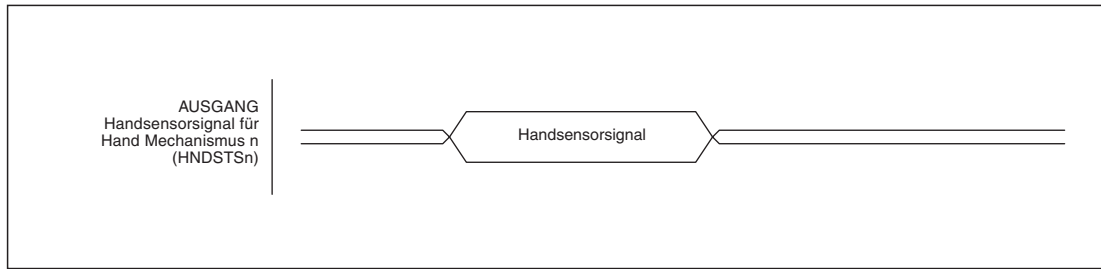
● JOG- (Eingang JOG-Vorschub in negativer Richtung für 8 Achsen)



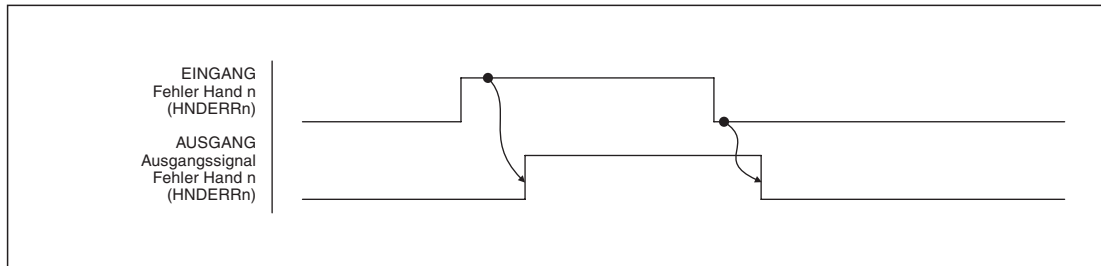
● HNDCTRLn (Handsteuersignal für Hand Mechanismus n)



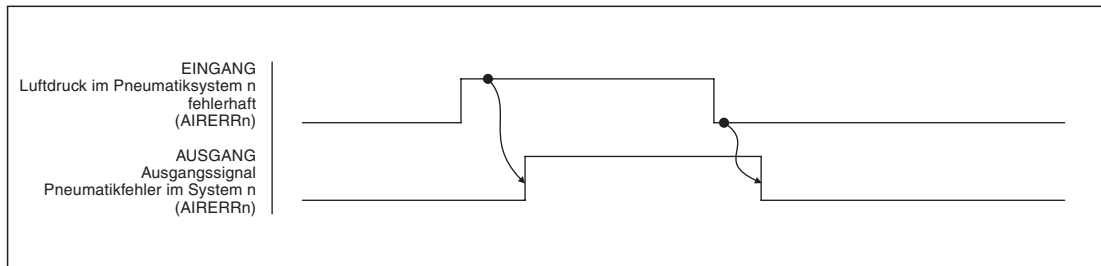
● HNDSTSn (Handsensorsignal für Hand Mechanismus n)



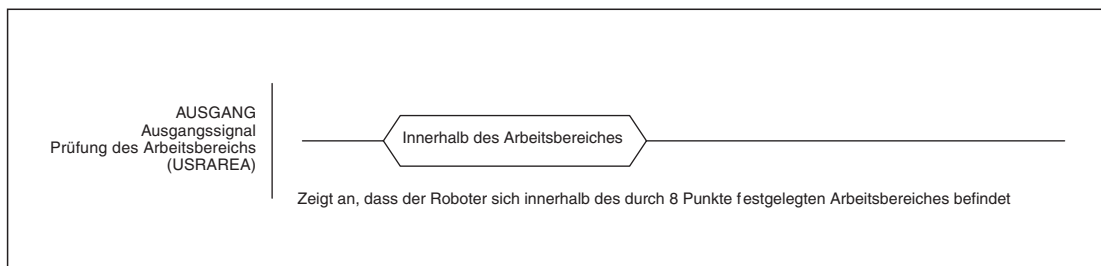
● HNDERRn (Eingang Fehler Hand n/Ausgang Fehler Hand n)



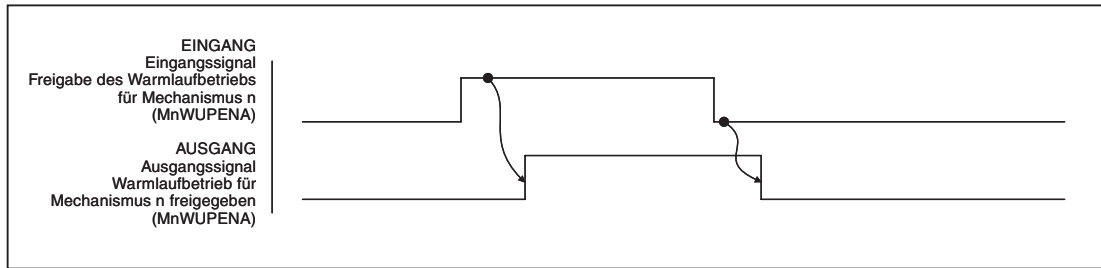
● AIRERRn (Eingang Luftdruck im Pneumatiksystem n fehlerhaft/Ausgang Pneumatikfehler im System n)



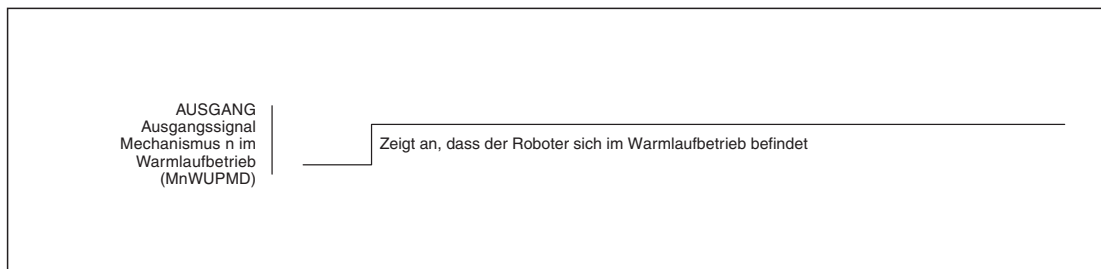
● USRAREA (Über 8 Punkte festgelegter Arbeitsbereich)



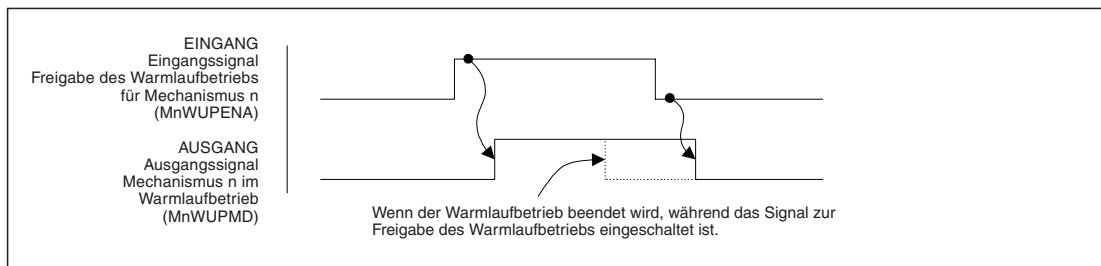
- MnWUPENA (Eingang Freigabe des Warmlaufbetriebs für Mechanismus n/Ausgang Warmlaufbetrieb für Mechanismus n freigegeben)



- MnWUPMD (Ausgangssignal Mechanismus n im Warmlaufbetrieb)



Folgende Abbildung zeigt das Zeitablaufdiagramm, wenn der Ausgang für die Signalausgabe zur Warmlaufbetriebsanzeige (MnWUPMD) und der Eingang zur Freigabe des Warmlaufbetriebs (MnWUPENA) gleichzeitig zugewiesen sind.



10.2.2 Programmsteuerung durch externe Signale

Zeitablaufdiagramme bei externer Steuerung

Folgende Abbildung zeigt das Zeitablaufdiagramm für die Steuerung der Funktionen „Programmwahl“, „Start“, „Stopp“ und „Neustart“ durch externe Signale:

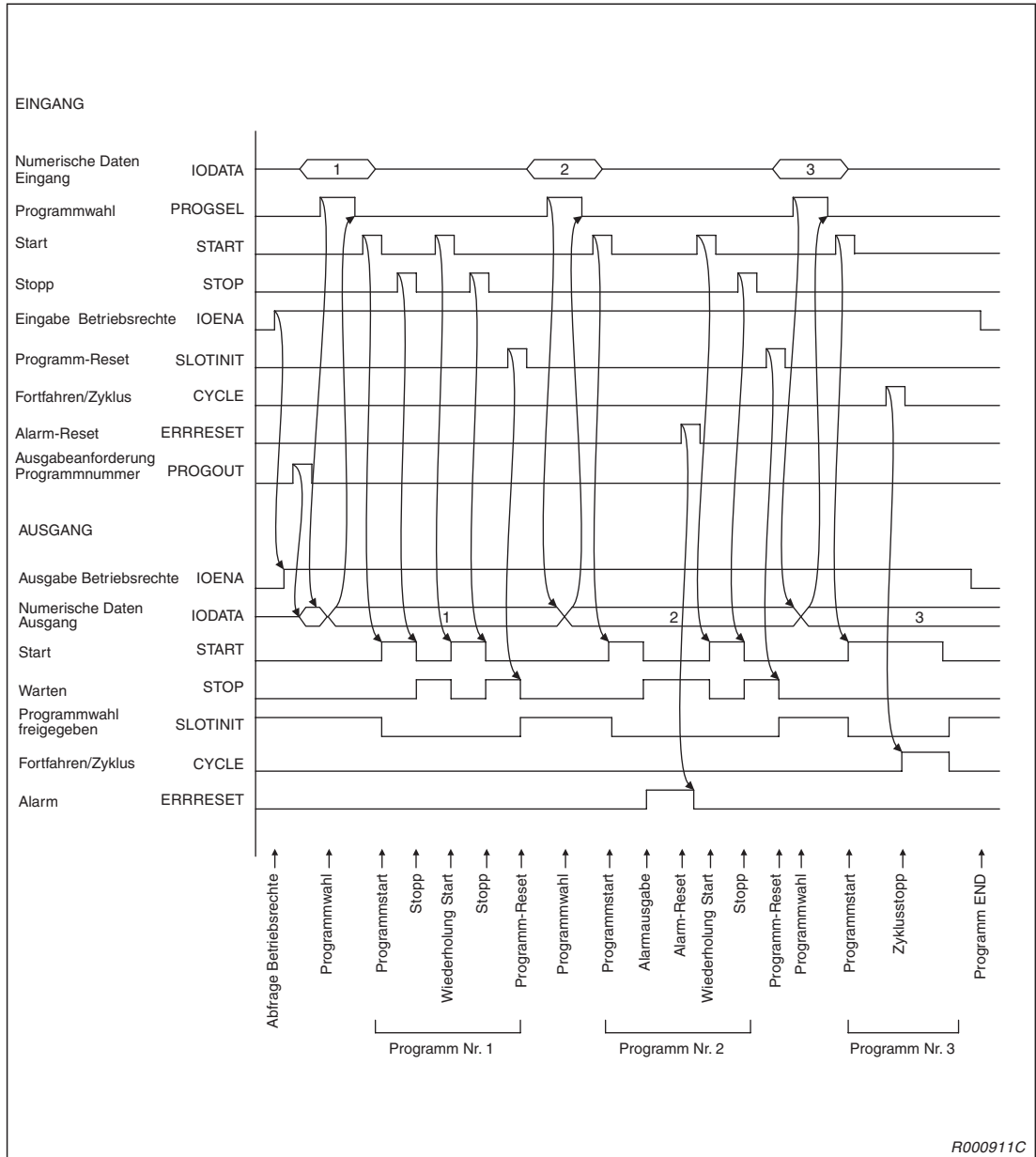


Abb. 10-3: Zeitablaufdiagramm 1 bei externer Steuerung

Folgende Abbildung zeigt das Zeitablaufdiagramm für die Steuerung der Funktionen „Servo EIN/AUS“, „Programmwahl“, „Auswahl des Geschwindigkeitsübersteuerungswertes“, „Start“, „Ausgabe der Zeilennummer“ usw. durch externe Signale:

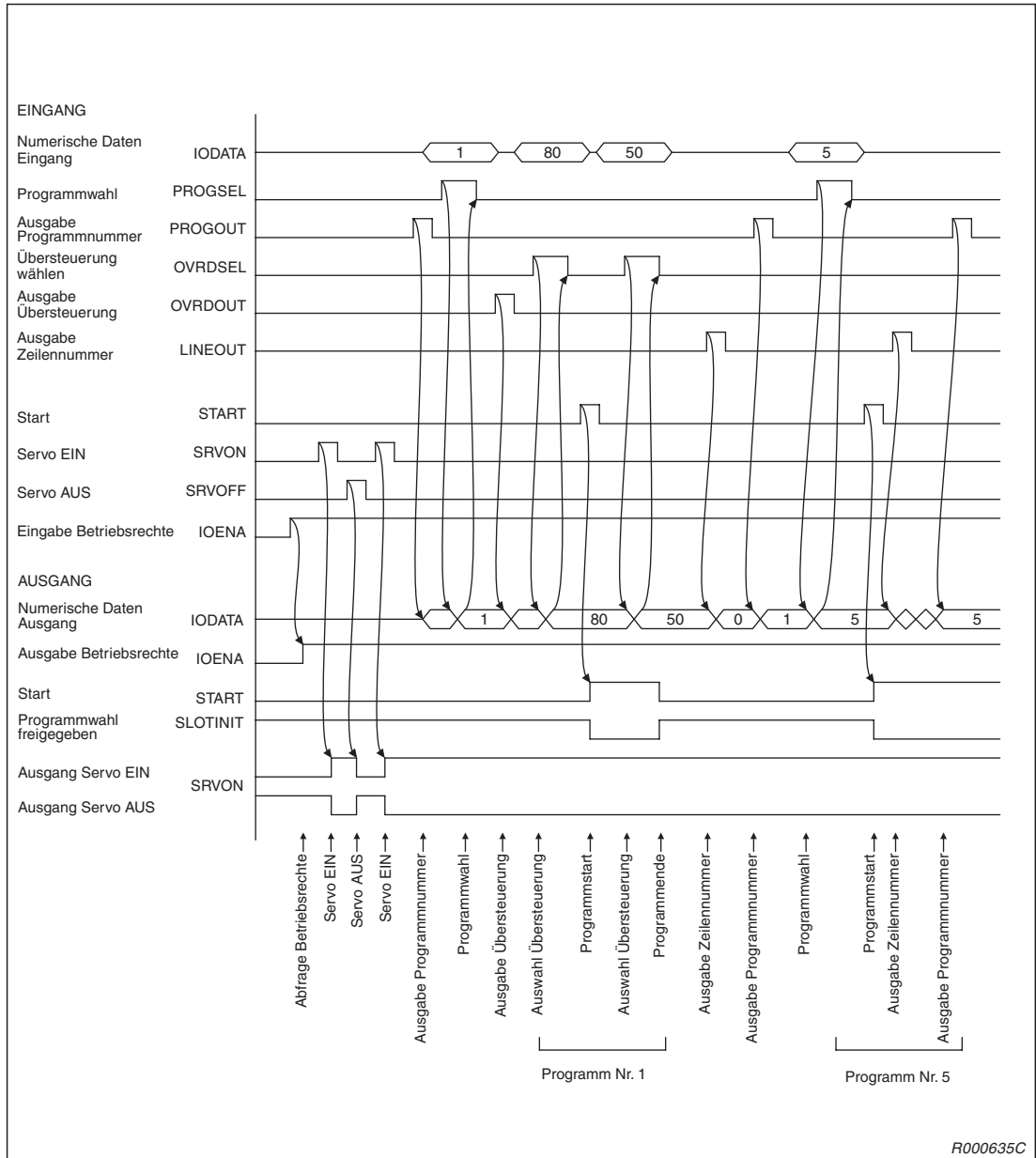


Abb. 10-4: Zeitablaufdiagramm 2 bei externer Steuerung

Folgende Abbildung zeigt das Zeitablaufdiagramm für die Steuerung der Funktionen „Fehler zurücksetzen“, „Allgemeinen Ausgang zurücksetzen“, „Programm zurücksetzen“ usw. durch externe Signale:

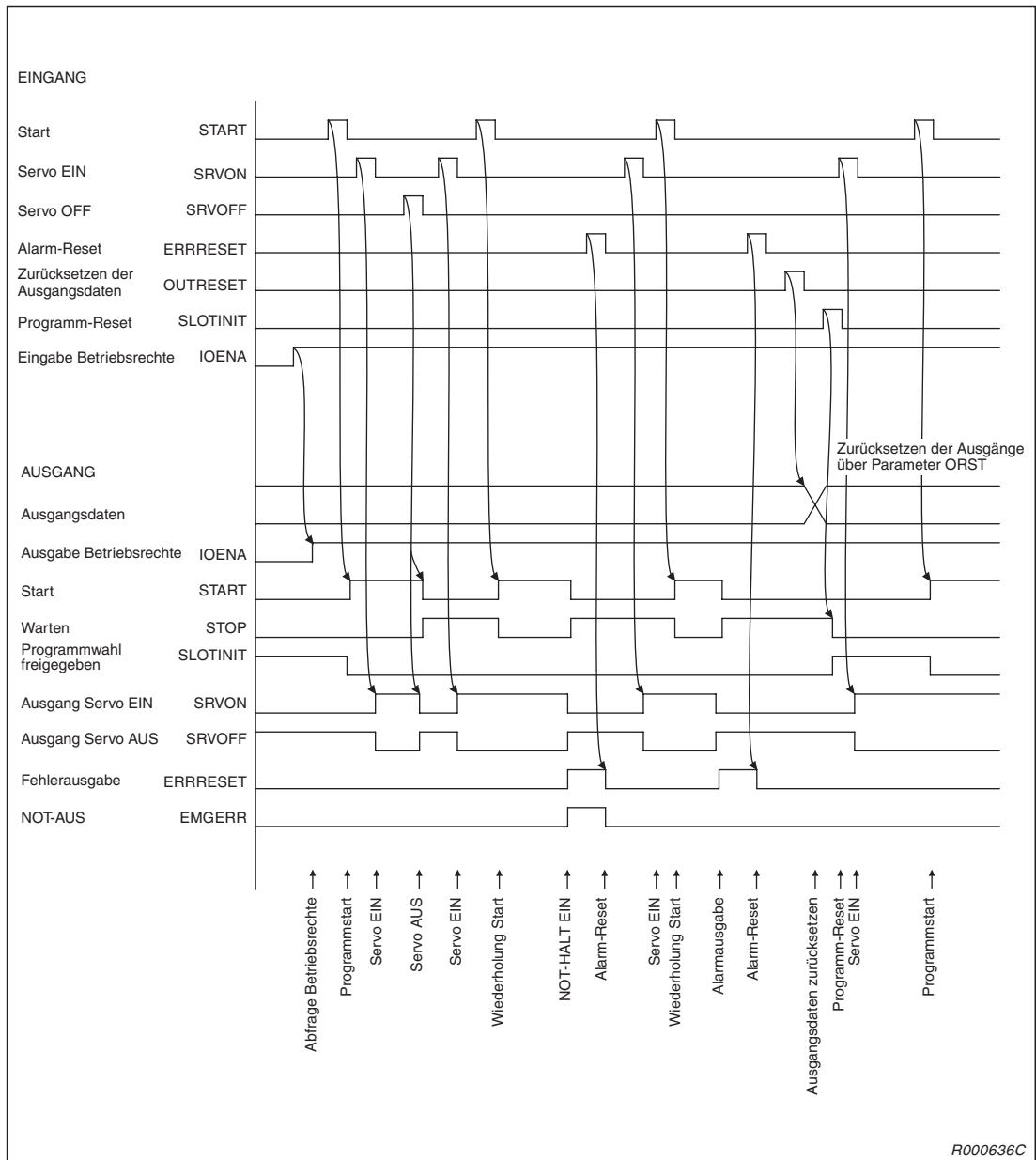


Abb. 10-5: Zeitablaufdiagramm 3 bei externer Steuerung

Folgende Abbildung zeigt das Zeitablaufdiagramm für die Steuerung der Funktionen „JOG-Betrieb“, „Anfahren der Ersatzposition“, „Programm zurücksetzen“ usw. durch externe Signale:

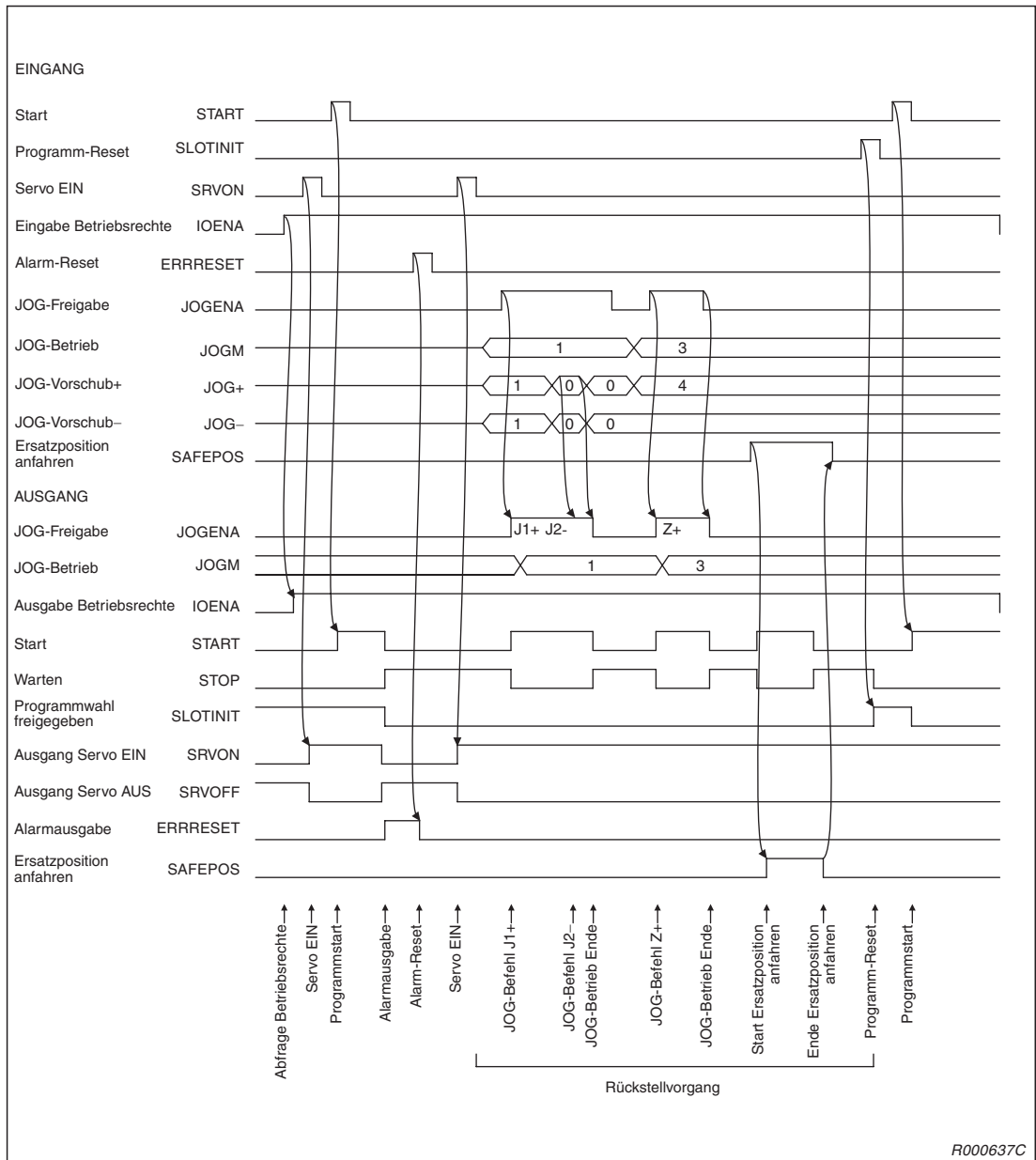


Abb. 10-6: Zeitablaufdiagramm 4 bei externer Steuerung

10.3.2 Steuergerät CR2

Der NOT-HALT-Stecker des Steuergerätes CR2 besitzt 5 Anschlussklemmen. An diese Anschlussklemmen kann ein externer NOT-HALT-Schalter angeschlossen werden. Der NOT-HALT-Schaltkreis des Steuergerätes ist redundant, so dass Sie für den externen NOT-HALT-Schalter 4 Anschlussklemmen verwenden müssen.

So integrieren Sie einen externen NOT-HALT-Schalter in den Roboterschaltkreis:

- ① Entfernen Sie die Kunststoffabdeckung des NOT-HALT-Steckers.
- ② Lösen Sie die Schrauben der entsprechenden Anschlussklemmen und entfernen die Drahtbrücken.
- ③ Nehmen Sie die Anschlussleitung des externen Schalters und entfernen 5 bis 7 mm der Leitungsisolierung.
- ④ Legen Sie die abisolierten Leitungsenden unter die Schraubenklemme. Der Anschluss des NOT-HALT-Schalters erfolgt wie in Abb. 10-7 über die Klemmen 1–2 und 3–4.
- ⑤ Drehen Sie die M3,5-Schrauben fest an.
- ⑥ Befestigen Sie die Kunststoffabdeckung des NOT-HALT-Steckers.

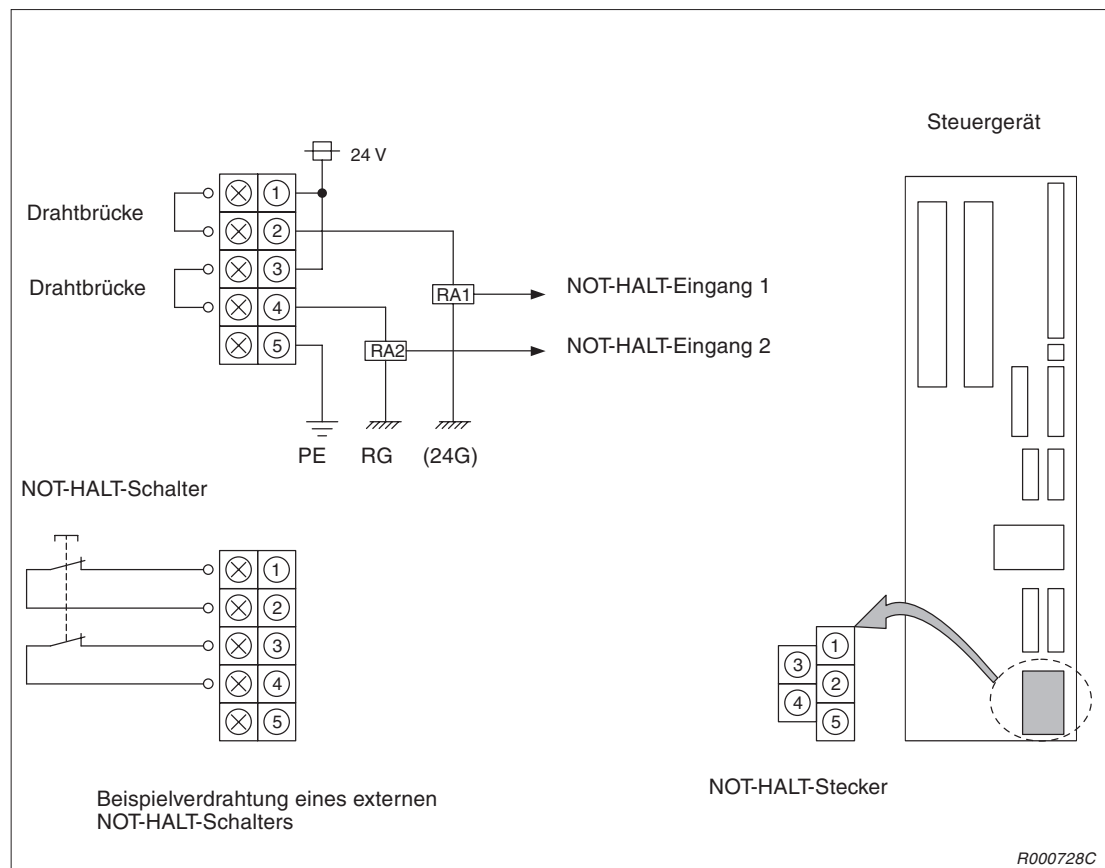


Abb. 10-8: Anschluss eines externen NOT-HALT-Schalters (Steuergerät CR2)

10.3.3 Steuergeräte CR2A und CR2B

Auf der Rückseite des Steuergerätes befinden sich die NOT-HALT-Stecker. Auf einem Stecker sind 6 Anschlussklemmen, je zwei um einen externen NOT-HALT-Schalter, einen Tür-Schließkontakt und eine Signallampe in den Schaltkreis des Roboters zu integrieren. Standardmäßig sind die Anschlussklemmen für den NOT-HALT-Schalter und den Tür-Schließkontakt mit jeweils einer Drahtbrücke kurzgeschlossen. Der NOT-HALT-Schaltkreis des Steuergerätes ist redundant, so dass Sie für den externen NOT-HALT-Schalter 4 Anschlussklemmen verwenden müssen. Der Roboter kann über den NOT-HALT-Schalter an der Vorderseite des Steuergerätes gestoppt werden.

Um einen externen NOT-HALT-Schalter oder Tür-Schließkontakt in den Roboterschaltkreis zu integrieren, gehen Sie wie folgt vor:

- ① Lösen Sie die Schrauben der entsprechenden Anschlussklemmen und entfernen die Drahtbrücke.
- ② Nehmen Sie die Anschlussleitung des externen Schalters, z. B. NOT-HALT-Schalter, und entfernen Sie 5 bis 7 mm der Leitungsisolierung.
- ③ Legen Sie das abisolierte Leitungsende unter die Schraubenklemme.
- ④ Drehen Sie die Schrauben fest an.

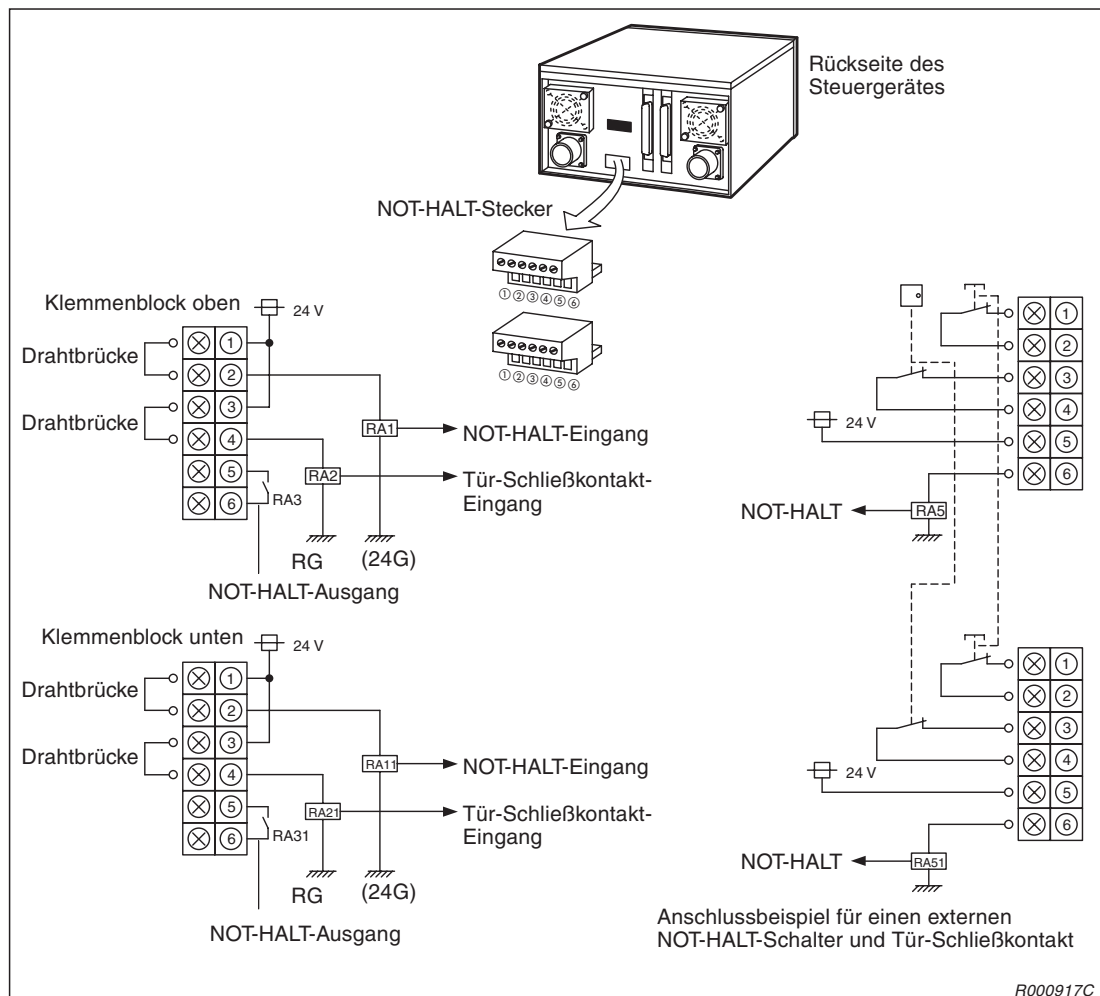


Abb. 10-9: Anschluss eines externen NOT-HALT-Schalters (Steuergerät CR2A)

10.3.5 Verhalten des Roboters bei Betätigung des NOT-AUS-Schalters

Eine Betätigung des NOT-AUS-Schalters während des Roboterbetriebes führt zur Abschaltung der Servoversorgungsspannung. Der weitere Verfahrweg der Handspitze und die Stopposition sind nach der Betätigung des NOT-AUS-Schalters nicht vorhersehbar. In Abhängigkeit der Geschwindigkeit und Last kann es zum Überschwingen des Roboterarms kommen.

11 Programmiertechniken

In diesem Kapitel finden Sie verschiedene Beispielprogramme mit unterschiedlichen Schwierigkeitsgraden. Das Kapitel ist in drei Hauptbereiche unterteilt:

- **Programmiertechniken für Einsteiger**
In diesem Abschnitt werden grundlegende Kenntnisse zur Erstellung eines Roboterprogramms vermittelt.
- **Programmiertechniken für fortgeschrittene Einsteiger**
In diesem Abschnitt werden spezielle Kenntnisse und gelegentlich benötigte Techniken zur Programmierung vermittelt.
- **Programmiertechniken für Fortgeschrittene**
In diesem Abschnitt werden seltener benötigte Techniken vermittelt, die jedoch nützlich bei der Erstellung komplexer Roboterprogramme sind.

Programmiertechnik		Seite
Einsteiger	Aufbau leicht verständlicher Programme	11-2
	Verwaltung von Programmversionen	11-8
	Änderung der Betriebsgeschwindigkeit in einem Programm	11-9
	Transportüberwachung des Werkstückes	11-10
	Positioniergenauigkeit	11-12
	Zeitverzögerte Signalprüfung	11-13
	Synchronisierung durch externe Eingangssignale	11-15
	Programmübergreifende Verwendung von Daten	11-17
	Überwachung der Abweichung von Soll- und Istposition	11-19
	Verkürzung der Zykluszeit	11-21
Fortgeschrittene Einsteiger	Schnelle Anpassung an unterschiedliche Werkstückgeometrien	11-23
	Vielseitige Anwendung der Pallettierungsfunktion	11-25
	Schreiben eines Kommunikationsprogramms	11-27
	Reduzierung von geteachten Positionen	11-30
	Verwendung einer P-Variablen in einem Zähler	11-32
Fortgeschrittene	Sensorgesteuerte Übertragung von Positionsdaten	11-33
	Einsatz eines Roboters als einfache SPS	11-35
	Implementierung einer Abbildungsfunktion	11-38
	Ausgabe ausgeführter Zeilen	11-41
	Status bei Auftreten eines Fehlers speichern	11-42

Tab. 11-1: Einteilung der Programmiertechniken

11.1 Programmiertechniken für Einsteiger

11.1.1 Aufbau leicht verständlicher Programme

Da die Roboterprogrammiersprache MELFA-BASIC IV auf der allgemeinen Programmiersprache BASIC basiert, ist sie vom Prinzip her leicht verständlich. Umfangreiche Programme, in denen der Zugriff auf Variablen von einem beliebigen Ort aus erfolgen kann und in denen keine Funktionen, die Argumente beziehungsweise Übergabewerte enthalten, definiert werden können, werden jedoch schnell unübersichtlich. Wie kann ein leicht verständliches Programm aufgebaut werden?

Methode

Beachten Sie die in folgender Abbildung aufgeführten Punkte, um in einer nicht strukturierten Programmiersprache wie BASIC ein leicht verständliches Programm zu entwerfen. Die in der Abbildung aufgeführten Regeln stellen dabei nur eine Empfehlung dar und können den Anforderungen Ihrer Anwendung entsprechend angepasst werden.

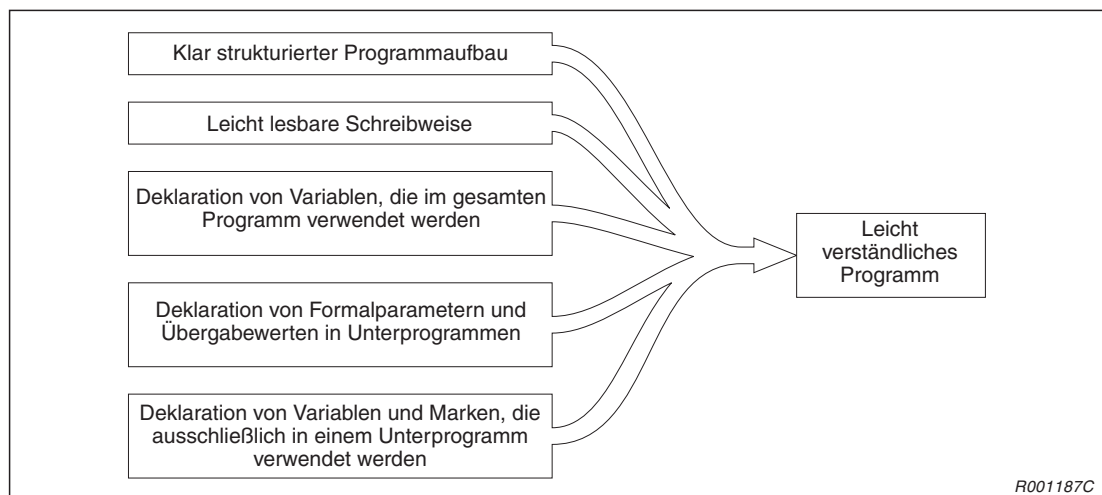


Abb. 11-1: Aufbau leicht verständlicher Programme

- Klar strukturierter Programmaufbau

Teilen Sie als ersten Schritt beim Entwurf eines Programmes das Programm in funktionsabhängige Blöcke auf. Ein derart strukturiertes Programm erleichtert nicht nur das Verständnis, sondern auch das Auffinden von Problemstellen im Fehlerfall. Weiterhin ermöglicht ein solcher Programmaufbau die Wiederverwendung einzelner Funktionsblöcke in anderen Programmen.

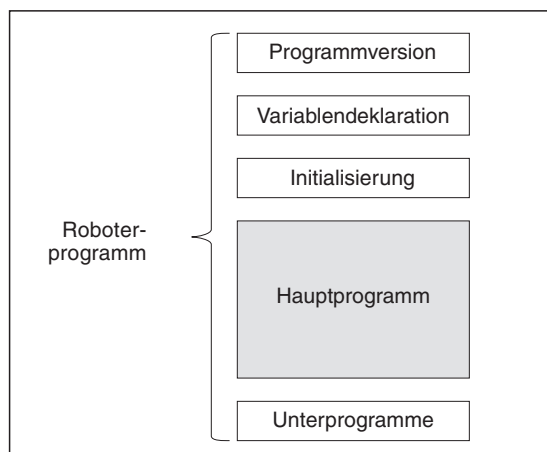


Abb. 11-2: Aufteilung eines Programmes in Blöcke

R001188C

Roboterprogramm	Programmblöcke
<pre> 1000 ' Werkstückaufnahme (Hauptprogramm) 1010 ' Datum:2004.04.01 VER 1.1 1020 ' (1)Änderung *** -> @@@ 1030 ' (2)Ergänzung \$\$\$ 1040 ' [Revisionen] 1050 ' 2004.03.01 VER 1.0 1060 ' 2004.02.01 VER 0.7 </pre>	<p><Programmversion> Dieser Block enthält die für die Verwaltung der Programmversionen nötigen Informationen. Es empfiehlt sich die Verwendung von Einzelbytes und alphanumerischen Zeichen, damit die Versionsinformationen auch auf der Teaching Box angezeigt werden können.</p>
<pre> 1070 ' === Variablendeklaration === 1080 DIM ML_DATA(3) 1090 DEF IO X1_REQ = BYTE,8,&H0F 1100 DEF IO Y1_ERR = BYTE,8,&H0F 1110 DEF POS PL_PFRAM 1120 DEF INTE ML_REQ </pre>	<p><Variablendeklaration> Dieser Block enthält z. B. die Deklarationen von Feldvariablen (DIM), Signalvariablen (DEF IO) und speziellen Variablen (DEF CHAR/INTE/POS usw.).</p>
<pre> 1130 ' === Initialisierung === 1140 PL_PFRAM = FRAM(PT001,PT002,PT003) 1150 DEF ACT 1,X1_REQ<>0,GOSUB *S50ACT1 1160 ACT 1 = M_ON 1170 OPEN "COM1:" AS #1 1180 ON COM(1) GOSUB *S60RECV 1190 COM(1) ON </pre>	<p><Initialisierung> Dieser Block enthält die Initialisierungsprozesse, die nur einmal beim Programmstart durchgeführt werden. Hier werden z. B. die Startwerte von Variablen, Interrupts und Kommunikationskanälen festgelegt.</p>
<pre> 1200 ' === Hauptprogramm === 1210 *MAIN 1220 IF ML_REQ = 1 THEN 1230 P00TMP = P_ZERO 1240 P00TMP.X = ML_DATA(1) 1250 P00TMP.Y = ML_DATA(2) 1260 P00TMP.Z = ML_DATA(3) 1270 MOV PL_FRAM * P00TMP 1280 ENDIF 1290 IF ML_REQ = &H0F THEN 1300 MX99ERNO = 9100 1310 GOSUB *S99ALARM 1320 YL_ERR = MY99RET 1330 IF M_CYS = M_ON THEN END 1340 GOTO MAIN </pre>	<p><Hauptprogramm> Das Hauptprogramm enthält die nachfolgende Sequenz, die beim Start des Programms zuerst ausgeführt wird. Die Ausführung hängt von einem Zyklusstopp in der IF-Zeile ab.</p> <pre> * MAIN : IF M_CYS = M_ON THEN END GOTO MAIN </pre> <p>Im Allgemeinen ist ein Programm leichter lesbar, wenn das Hauptprogramm durch Auslagerung von Funktionsblöcken in Unterprogrammen möglichst kurz gehalten wird.</p>
<pre> 1350 ' === Unterprogramm 50 === 1360 *S50ACT1 1370 ML_REQ = X1_REQ 1380 RETURN 0 1390 ' === Unterprogramm 60 === 1400 *S60RECV 1410 COM(1) STOP 1420 INPUT #1,M60X,M60Y,M60Z 1430 ML_DATA(1) = M60X 1440 ML_DATA(2) = M60Y 1450 ML_DATA(3) = M60Z 1460 COM(1) ON 1470 *L60END 1480 RETURN 0 1490 ' === Unterprogramm 99 === 1500 *S99ALARM 1510 ERROR MX99ERNO 1520 MY99RET = MX99ERNO 1530 *L99END 1540 RETURN </pre>	<p><Unterprogramme> Ein Unterprogramm beginnt mit einer Marke, die den Unterprogrammnamen enthält, und endet mit einer RETURN-Anweisung. Ein Unterprogramm umfasst einen Funktionsblock, der z. B. zur Ablage eines Werkstücks oder zum Einlesen von Kommunikationsanweisungen dient. Der Unterprogrammaufruf erfolgt aus dem Hauptprogramm oder einem Unterprogramm mit Hilfe der GOSUB-Anweisung. Der Rücksprung erfolgt in die Zeile nach der GOSUB-Anweisung. Beim Rücksprung aus einer Interrupt-Routine springt die Steuerung über die Anweisung RETURN 0/1 in die Zeile zurück, in der der Interrupt aufgetreten ist oder in die nächste Zeile.</p>

R001189C

Abb. 11-3: Aufteilung eines Beispielprogramms

● Leicht lesbare Schreibweise

Verwenden Sie für die Erstellung eines Programms eine leicht lesbare Schreibweise.

Merkmal	Beschreibung
Einzug	Verwenden Sie Einzüge zur Darstellung von Unterprogrammen, FOR-NEXT-Schleifen, IF-ENDIF-Blöcken usw.
<p>Fügen Sie nach der Zeilennummer eine leere Spalte und eine Symbolspalte ein. Verwenden Sie die Symbolspalte zur Eintragung von Zeichen, die z. B. zur Markierung einer Marke (*) oder eines Kommentars (') dienen. Sehen Sie auch einen Einzug von zwei Zeichenbreiten für IF- und FOR-Schleifen usw. vor. Beginnen Sie ein Programm mit der Zeilennummer 1000 und verwenden Sie in der ersten Zeile keinen Einzug.</p> <div style="border: 1px dashed black; padding: 10px; margin: 10px 0;"> <pre> 1000 * S50CALC 1010 MY50ANS = MX50CNT + 1 1020 RETURN </pre> </div> <p>— 2 Zeichen breiter Einzug — Symbolspalte — leere Spalte — Zeilennummer</p>	

Sprungmarke	Beschreibung
	<p>Führen Sie Programmsprünge mit Hilfe von Sprungmarken über die Anweisungen GOTO und GOSUB aus.</p> <p>Vermeiden Sie die Ausführung direkter Sprünge unter Angabe von Zeilennummern, da es die Lesbarkeit eines Programms verschlechtert. Führen Sie Sprünge mit Hilfe von Sprungmarken aus.</p> <p>Beispiel:</p> <pre> 1150 DEF ACT 1,X1_REQ <> 0,GOSUB *S50ACT1 1180 ON COM(1) GOSUB *S60RECV 1340 GOTO *MAIN </pre>

Kommentar	Beschreibung
	<p>Verwenden Sie Kommentare an Stellen, die ohne Erklärung schwer verständlich sind. Kommentieren Sie z. B. Programmabschnitte, Ein- und Ausgangssignale eines Unterprogramms und die Bedeutung komplexer Berechnungen.</p> <p>Beispiel:</p> <pre> 1330 IF M_CYS = M_ON THEN END ' Ende, falls Zyklusbetrieb aktiv 1500 ' = = = Unterprogramm 30 = = = 1510 ' Anfahrt der festgelegten Position 1520 ' Eingabe: MX30POS (Positionsnummer der Zielposition) 1530 ' Ausgabe: MY30RET (Ausführung fehlerfrei: 1, Ausführung fehlerhaft: 0) 1540 * S30MVPOS </pre>

● Deklaration von Variablen, die im gesamten Programm verwendet werden

Innerhalb eines Programms ist in MELFA-BASIC IV ein Zugriff auf die Werte der lokalen Variablen möglich. Lokale Variablen sind durch die nachfolgend aufgeführten Eigenschaften charakterisiert.

Merkmal	Beschreibung
Variablen, die innerhalb eines Programms verwendet werden	Die Variablen werden gemeinsam vom Hauptprogramm und den Unterprogrammen verwendet. Sie dienen zur Speicherung von Ergebnissen aus einem Anweisungsblock, von Ausgabeergebnissen einer Interruptroutine usw. Ein Zugriff kann von einem beliebigen Ort aus erfolgen.
	1. Zeichen: kennzeichnet den Variablentyp (M: numerische Variable, P: Positionsvariable, J: Gelenkvariable, C: Zeichenkettenvariable usw.) 2. Zeichen: kennzeichnet eine lokale Variable (L) 3. Zeichen: Unterstrich (<u> </u>) 4. bis 8. Zeichen: beliebige Zeichenkette (5 Zeichen) Beispiel: 1110 DEF POS PL_PFRAM 1140 PL_PFRAM = FRAM(PT001,PT002,PT003) 1270 MOV PL_FRAM * P00TMP
Geteachte Variablen	Diese Variablen dienen zur Speicherung von geteachten Werten oder von z. B. Offset-Daten. Ein Zugriff auf die Variablen kann von einem beliebigen Ort aus erfolgen, sie können jedoch von dort nicht überschrieben werden.
	1. Zeichen: kennzeichnet den Variablentyp (P, J) 2. Zeichen: kennzeichnet eine geteachte Variable (T oder D) 3. bis 8. Zeichen: beliebige Zeichenkette (6 Zeichen) Beispiel: PT001: Speicherung eines geteachten Wertes PD001: Offset zum geteachten Wert PT001
E/A-Variablen	Diese Variablen dienen zum Lesen und Schreiben von Variablen, die mit der DEF-IO-Anweisung deklariert wurden. Man unterscheidet Ein- und Ausgangsvariablen, wobei Eingangsvariablen nur gelesen und Ausgangsvariablen nur geschrieben werden können.
	1. Zeichen: kennzeichnet den Variablentyp (X: Eingangssignal, Y: Ausgangssignal) 2. Zeichen: kennzeichnet eine lokale Variable (L) 3. Zeichen: Unterstrich (<u> </u>) 4. bis 8. Zeichen: beliebige Zeichenkette (5 Zeichen) Beispiel: 1090 DEF IO X1_REQ = BYTE,8,&H0F 1100 DEF IO Y1_ERR = BYTE,8,&H0F

● Deklaration von Formalparametern und Übergabewerten in Unterprogrammen

Bei Ausführung der GOSUB-Anweisung erfolgt ein Sprung zu einer festgelegten Marke oder Zeilennummer. Der Rücksprung erfolgt über die RETURN-Anweisung. Dabei können im Unterprogramm Ein- und Ausgabewerte (Formalparameter und Übergabewerte) deklariert werden. Ein rekursiver Aufruf (d. h. ein Programm ruft sich selbst auf) ist nicht möglich.

Merkmal	Beschreibung
Unterprogrammnamen	Ein Unterprogramm ist durch eine eindeutige Zahl im Namen gekennzeichnet. Ergänzend wird ein Variablen- oder Markennamen hinzugefügt, der die Funktion des Unterprogramms wiedergibt.
	1. Zeichen: kennzeichnet ein Unterprogramm (S) 2. und 3. Zeichen: Nummer des Unterprogramms (01 bis 99) 4. bis 8. Zeichen: beliebige Zeichenkette (5 Zeichen) <ul style="list-style-type: none"> ● Vor einen Markennamen muss ein Asterisks (*) gesetzt werden. ● Die Nummer 00 darf nicht für ein Unterprogramm verwendet werden. Sie ist durch das Hauptprogramm belegt. ● Beschreiben Sie die Funktion und die E/A-Ausgangsvariablen jedes Unterprogramms Beispiel: 1360 *S50ACT1 1400 *S60RECV 1490 *S99ALARM
Formalparameter und Übergabewerte eines Unterprogramms	Diese Variablen dienen zum Lesen und Schreiben von Variablen, die mit der DEF-IO-Anweisung deklariert wurden. Man unterscheidet Ein- und Ausgangsvariablen, wobei Eingangsvariablen nur gelesen und Ausgangsvariablen nur geschrieben werden können.
	1. Zeichen: kennzeichnet den Variablentyp (M, P, J, C) 2. Zeichen: kennzeichnet eine Ein- oder Ausgangsvariable (Eingang: X, Ausgang: Y) 3. und 4. Zeichen: Nummer des Unterprogramms 5. bis 8. Zeichen: beliebige Zeichenkette (4 Zeichen) Beispiel: 1500 ' = = = Unterprogramm 30 = = = 1510 ' Erläuterung des Unterprogramms 1520 ' Eingang: MX30POS (Beschreibung der Eingangsvariablen) 1530 ' Ausgang: MY30RET (Beschreibung der Ausgangsvariablen) 1540 * S30MVPOS 1550 MOV PTPOS(MX30POS) 1560 MY30RET = MX30POS 1570 RETURN

- Deklaration von Variablen und Marken, die ausschließlich in einem Unterprogramm verwendet werden

In MELFA-BASIC IV ist ein Zugriff auf alle Variablen und Marken von einem beliebigen Ort aus möglich. Dadurch lässt sich jedoch oft nicht mehr erkennen, von welcher Programmstelle aus eine Variable geschrieben worden ist. Daher ist die Festlegung von Variablen und Marken sinnvoll, die ausschließlich innerhalb eines Unterprogrammes verwendet werden können. Auch wenn die Programmiersprache einen Zugriff auf alle Variablen und Marken von jedem Ort aus erlaubt, sollte diese Regel bei der Programmerstellung berücksichtigt werden.

Merkmal	Beschreibung
Marken, die ausschließlich in einem Unterprogramm verwendet werden	Diese Marken werden nur innerhalb eines Unterprogramms verwendet. Fehler durch mehrfach auftretende Markennamen werden bei Verwendung in unterschiedlich benannten Unterprogrammen vermieden.
1. Zeichen: kennzeichnet eine Marke (L) 2. und 3. Zeichen: Nummer des Unterprogramms (01 bis 99) 4. bis 8. Zeichen: beliebige Zeichenkette (5 Zeichen) Vor einen Markennamen muss ein Asterisks (*) gesetzt werden. Beispiel: 1470 *L60END 1530 *L99END	
Variablen, die ausschließlich in einem Unterprogramm verwendet werden	Diese Variablen werden nur innerhalb eines Unterprogramms verwendet. Fehler durch mehrfach auftretende Variablennamen werden bei Verwendung in unterschiedlich benannten Unterprogrammen vermieden.
1. Zeichen: kennzeichnet den Variablentyp (M, P, J, C) 2. und 3. Zeichen: Nummer des Unterprogramms (01 bis 99) 4. bis 8. Zeichen: beliebige Zeichenkette (5 Zeichen) Beispiel: 1230 P00TMP = P_ZERO 1420 INPUT #1,M60X,M60Y,M60Z	

11.1.2 Verwaltung von Programmversionen

Bei der Entwicklung und Pflege von Programmen entstehen immer neue Programmversionen. Wie können alle Programmversionen übersichtlich verwaltet werden?

Methode

Programme werden während der Entstehung und auch später im praktischen Einsatz häufig noch geändert, verbessert, angepasst, erweitert oder aus anderen Gründen modifiziert. Die ständige Überarbeitung von Programmen führt zu einer Vielzahl unterschiedlicher Programmversionen. Daher ist eine übersichtliche Verwaltung der Programmversionen erforderlich. Von den möglichen Programmverwaltungsmethoden werden hier zwei vorgestellt, die sich besonders beim Einsatz der Steuergeräte CR□-500 anbieten.

- **Version als Kommentar beschreiben**
 Der einfachste Weg zur Verwaltung von Programmversionen ist die Aufführung aller Revisionen als Kommentar. Damit die Informationen auch über die Teaching Box zu lesen sind, ist es sinnvoll, den Kommentar an den Anfang des Programms zu setzen. Der Kommentar sollte den Programmnamen, das Änderungsdatum, die Version und eine Revisionsliste umfassen (siehe folgende Abbildung).

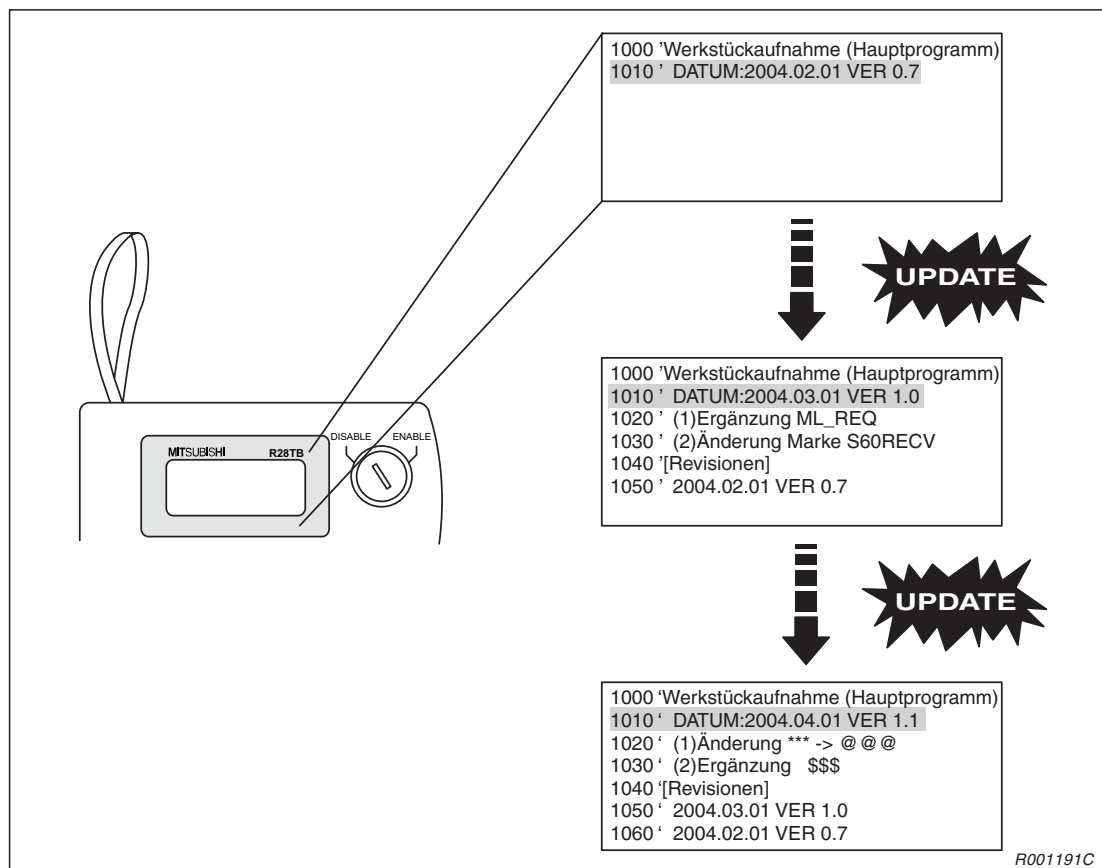


Abb. 11-4: Verwaltung von Programmversionen

- **Version im Parameter USERMSG speichern**
 Einige Robotersysteme werden mit unterschiedlichen Programmen gesteuert. Hier ist eine Angabe des Robotersystems zusätzlich zur Programmversion sinnvoll. Die Systemversion kann über die Teaching Box ausgelesen werden. Informationen wie der Systemname, die Version und das Erstellungsdatum können mit Hilfe der Teaching Box oder der PC-Software über maximal 64 Einzelbyte-Zeichen im Parameter USERMSG gespeichert werden.

11.1.3 Änderung der Betriebsgeschwindigkeit in einem Programm

Da manche Bewegungsabläufe, wie z. B. das Einsetzen eines Bolzens in eine Bohrung, in einem Roboterprogramm eine hohe Präzision erfordern, ist es sinnvoll, die Geschwindigkeiten für diese Bewegungen zu reduzieren. Bewegungsabläufe, die keine hohe Genauigkeit erfordern, können zur Reduzierung der Zykluszeiten hingegen mit höherer Geschwindigkeit ausgeführt werden.

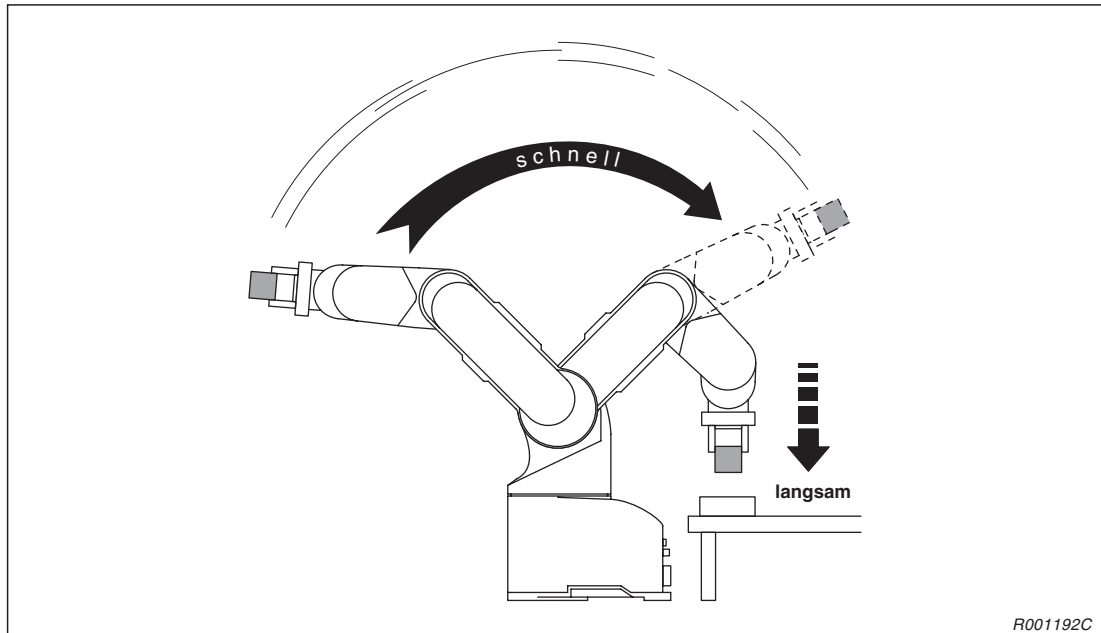


Abb. 11-5: Änderung der Betriebsgeschwindigkeit in einem Programm

Methode

Ändern Sie die Geschwindigkeit über die Anweisungen OVRD, JOVRD oder SPD.

OVRD: Die Anweisung OVRD ist unabhängig vom Interpolationstyp wirksam. Die Geschwindigkeitsanzeige auf dem Steuergerät ändert sich nicht.

JOVRD: Die Anweisung JOVRD ist nur bei Gelenk-Interpolation wirksam.

SPD: Die Anweisung JOVRD ist nur bei linearen und kreisförmigen Bewegungen wirksam.

11.1.4 Transportüberwachung des Werkstücks

Sinkt die Greifkraft durch ein Leck in einer Pneumtikleitung, ist das Werkstück beschädigt, kollidiert der Roboter mit umliegenden Einrichtungen o. Ä., kann ein Werkstück während des Transports herunterfallen. Wie können eine Transportüberwachung des Werkstücks und ein Stopp des Roboters im Fehlerfall durchgeführt werden?

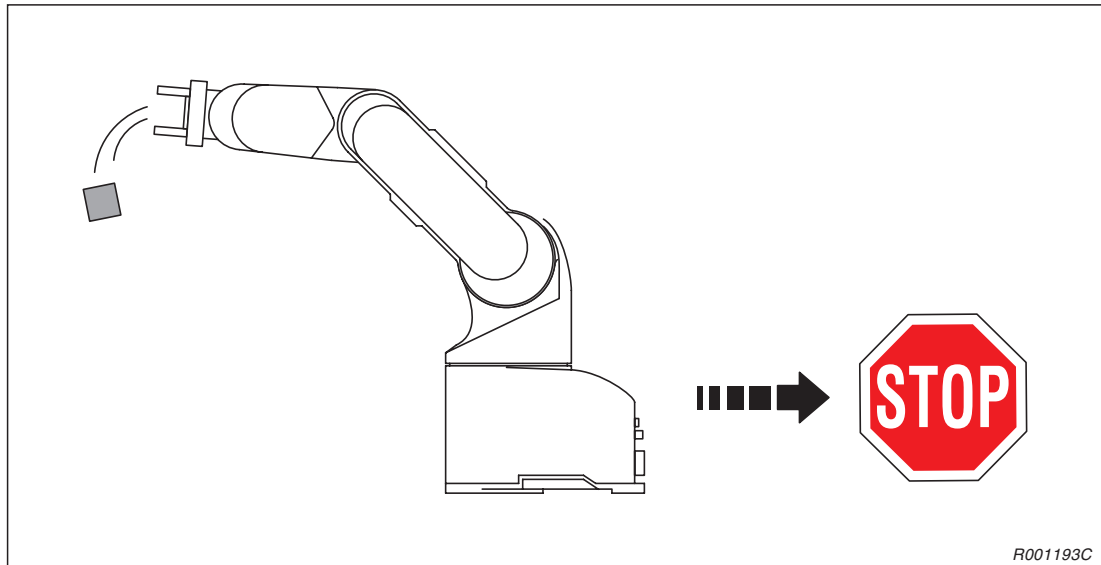


Abb. 11-6: Transportüberwachung des Werkstücks

Methode

Eine Methode, auf nicht vorhersehbare Ereignisse zu reagieren, ist die Verwendung von Interrupts. Ein Interrupt ermöglicht die Überwachung von Signalen oder Variablen. Tritt ein nicht vorhersehbares Ereignis auf, ändert sich der Signalzustand oder der Variablenwert. Es erfolgt eine Unterbrechungsanforderung und ein Interrupt-Prozess wird ausgeführt. Diese Methode ist für eine Vielzahl von Anwendungen einsetzbar und nicht nur auf die Transportüberwachung von Werkstücken beschränkt.

Im hier gezeigten Beispiel erfolgt die Überwachung von Signalen, die sich ändern, wenn das Werkstück herunterfällt. Sobald das Werkstück herunterfällt wird ein Interrupt eingeschaltet. Ist die Überwachung beendet, wird der Interrupt abgeschaltet.

Bei der Verwendung von Interrupts müssen die Interruptbedingung und das Sprungziel bei erfüllter Bedingung über die Anweisung DEF ACT definiert werden. Ist die Anzahl der verwendeten Interrupts nicht zu groß, sollten sie am Programmstart definiert werden.

Programmbeispiel

Ein Roboter transportiert Werkstücke mittels eines Sauggreifers. Sobald ein Werkstück verloren geht, stoppt der Roboter.

Positionen	E/A-Signale
PT01: Aufnahmeposition	M_IN(8): Aufnahme erlaubt
PT02: Vor der Aufnahmeposition	M_IN(9): Ablage erlaubt
PT03: Vor der Ablageposition	M_IN(901): Sensor zur Werkstück erfassung (bei EIN: halten)
PT04: Ablageposition	
PT99: Rückzugposition	

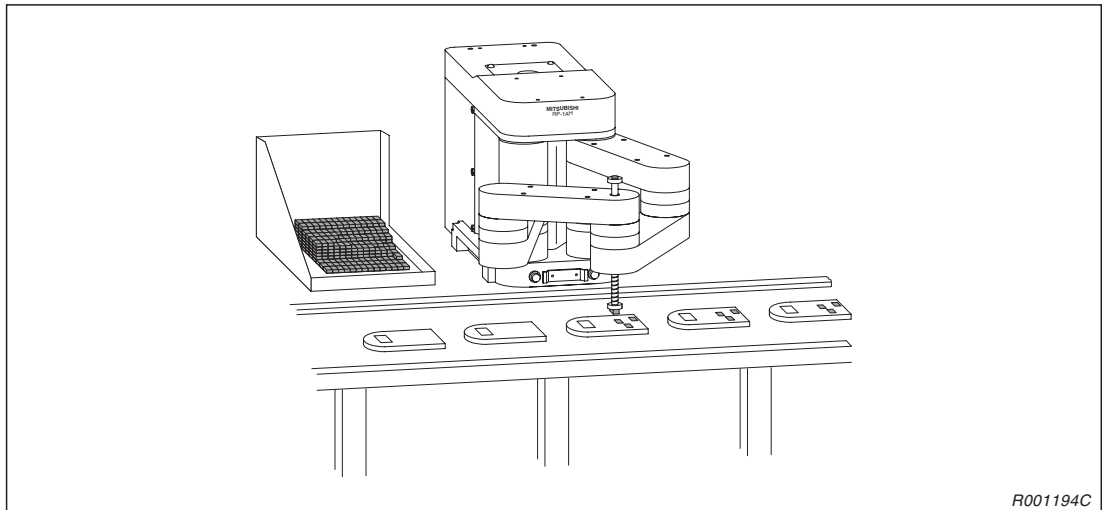
Tab. 11-2: Übersicht der Positionen und Signale

Programm	Erläuterung
1000 DEF ACT 1,M_IN(901) = M_OFF GOTO *S50FALL,S 1010 '	'Interrupt-Aufruf, bei Verlust des Werkstücks Durch das Anfügen der Stoppmethode „S“ stoppt der Roboter bei Verlust des Werkstückes in der kürzestmöglichen Zeit.
1020 PL_ZON1 = P99 - (10,10,10,1,1,1,10,10)(0,0) 1030 PL_ZON1 = P99 + (10,10,10,1,1,1,10,10)(0,0) 1040 ML_ZCHK = ZONE(P_CURR,PL_ZON1,PL_ZON2) 1050 IF ML_ZCHK = 0 THEN MOV PT99	'Prüfen, ob in Rückzugposition 'Falls nicht in Rückzugposition, Rückzugposition anfahren Die Berechnungen müssen zuerst durchgeführt werden, da während der Funktionsausführung keine Berechnung möglich ist.
1060 ' 1070 *MAIN 1080 OVRD 100 1090 MOV PT02 1100 *L00WAITG 1110 IF M_IN(8) = M_OFF THEN *L00WAITG 1120 OVRD 30 1130 MVS PT01 1140 HCLOSE 1 1150 DLY 0.1 1160 ACT 1 = M_ON 1170 MVS PT02 1180 OVRD 100 1190 MVS PT03 1200 *L00WAITP 1210 IF M_IN(9) = M_OFF THEN *L00WAITP 1220 OVRD 30 1230 MVS PT04 1240 ACT 1 = M_OFF 1250 HOPEN 1 1260 DLY 0.1 1270 OVRD 100 1280 MOV PT03 1290 GOTO *MAIN 1300 ' 1310 *S50FALL 1320 ERROR 9102 1330 END	'Position vor der Aufnahmeposition mit erhöhter Geschwindigkeit anfahren 'Warte, bis Aufnahme freigegeben 'Aufnahmeposition mit verringerter Geschwindigkeit anfahren 'Werkstück festhalten 'Wartezeit 'Werkstückverlust-Interrupt freigegeben 'Position vor der Aufnahmeposition anfahren 'Position vor der Ablageposition mit erhöhter Geschwindigkeit anfahren 'Warte, bis Ablage freigegeben 'Ablageposition mit verringerter Geschwindigkeit anfahren 'Werkstückverlust-Interrupt sperren 'Werkstück ablegen 'Wartezeit 'Position vor der Ablageposition mit erhöhter Geschwindigkeit anfahren 'Nächstes Werkstück holen Die Geschwindigkeit ist im Allgemeinen auf 100 % zu setzen, und nur wenn nötig zu verringern. Ab hier ist der Interrupt freigegeben. Der Interrupt wird vor der Ablage des Werkstücks gesperrt.
	'Fehlerausgabe

Tab. 11-3: Beispielprogramm zur Transportüberwachung

11.1.5 Positioniergenauigkeit

Eine der Hauptanforderungen an einen Roboter, ist die genaue Positionierung eines Werkstücks in einer Zielposition. Verfügt das System über eine Positioniereinheit, reicht eine ungefähre Positionierung durch den Roboter aus. Unter Berücksichtigung der Kosten, des Platzbedarfs und der Zykluszeiten muss der Roboter jedoch oft genaue Positionierungen ausführen. Wie kann der Roboter für genaue Positionieraufgaben eingesetzt werden?



R001194C

Abb. 11-7: Hohe Positioniergenauigkeit bei der Bestückung von SMD-Platinen

Methode

Eine exakte Positionierung von Werkstücken durch den Roboter kann mit Hilfe der Anweisung FINE zur Feinpositionierung erfolgen. Kommt es nicht auf eine möglichst kurze Zykluszeit an, kann zur Feinpositionierung nach dem Bewegungsbefehl auch eine DLY-Anweisung ausgeführt werden.

11.1.6 Zeitverzögerte Signalprüfung

In Abhängigkeit der Handgreiferausführung kann z. B. das Signal zur Rückmeldung des Handgreiferzustandes zeitverzögert zum Befehl „Hand schließen“ auftreten. Wie kann ein Signalzustand zeitverzögert überprüft werden?

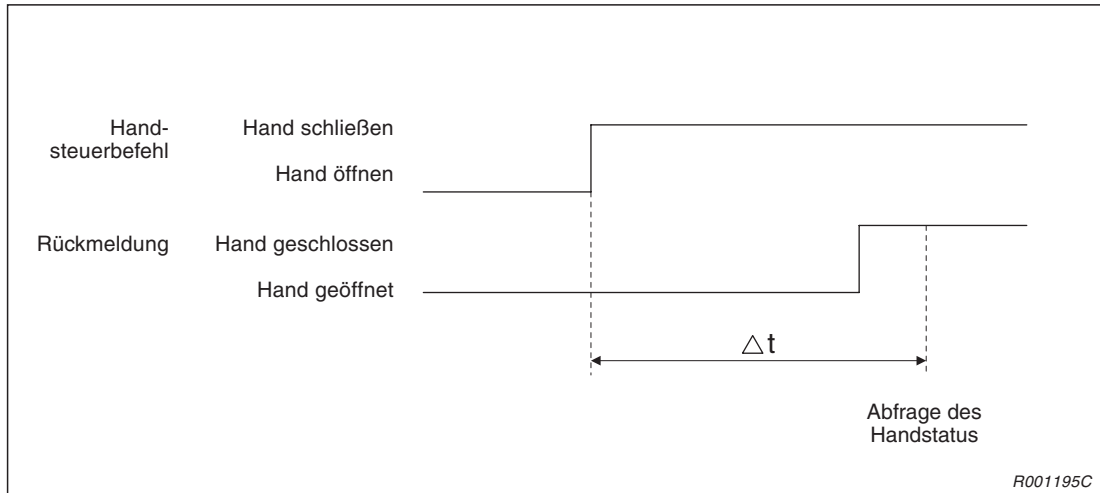


Abb. 11-8: Zeitverzögerte Signalprüfung

Methode

Da für diesen Zweck kein Befehl existiert, wird ein Unterprogramm mit der geforderten Funktionalität entworfen. Der Entwurf von Unterprogrammen zur Erfüllung allgemeiner Funktionalitäten führt aufgrund einer flexiblen Einsetzbarkeit zu einer effizienteren Programmierung und zu einem leicht verständlichen Programmaufbau.

Programmbeispiel

Dieses Programm dient zur Aufnahme und Ablage von Werkstücken, wobei die Abfrage des Rückmeldesignals für den Handgreiferzustand zeitverzögert zum Handsteuerbefehl für den Sauggreifer erfolgt.

Positionen	E/A-Signale
PT01: Aufnahmeposition	M_IN(901): Vakuum-Drucksensor (bei EIN: hält)
PT02: Vor der Aufnahmeposition	M_OUT(901): Handsteuerbefehl (bei EIN: halten)
PT03: Vor der Ablageposition	
PT04: Ablageposition	

Tab. 11-4: Übersicht der Positionen und Signale

Programm	Erläuterung
<pre> 1000 *MAIN 1010 OVRD 100 1020 MOV PT02 1030 MVS PT01 1040 M_OUT(901) = M_ON 1050 MX50SIG = 901 1060 MX50SEC = 3.0 1070 GOSUB *S50WON 1080 IF MY50SKP = 1 THEN GOTO *L40ALARM 1090 OVRD 10 1100 MVS PT02 1110 ' 1120 OVRD 100 1130 MOV PT03 1140 OVRD 10 1150 MOV PT04 1160 M_OUT(901) = M_OFF 1170 MX51SIG = 901 1180 MX51SEC = 3.0 1190 GOSUB *S51WOFF 1200 IF MY51SKP = 1 THEN GOTO *L40ALARM 1210 OVRD 100 1220 MOV PT03 1230 GOTO *MAIN 1240 ' 1250 *L40ALARM 1260 ERROR 9100 1270 END 1280 ' </pre>	<p>'Position vor der Aufnahme-position mit erhöhter Geschwindigkeit anfahren 'Aufnahme-position anfahren 'Ausgabe des Greifbefehls</p> <p>Parameter für Unterprogramme setzen</p> <p>'Warte bis Rückmeldesignal eingeschaltet 'Fehler bei Zeitüberschreitung 'Position vor der Aufnahme-position mit verringerter Geschwindigkeit anfahren</p> <p>'Position vor der Ablageposition 'mit erhöhter Geschwindigkeit anfahren 'Ablageposition mit verringerter Geschwindigkeit anfahren 'Werkstück ablegen</p> <p>Parameter für Unterprogramme setzen</p> <p>'Warte bis Rückmeldesignal ausgeschaltet 'Fehler bei Zeitüberschreitung 'Position vor der Ablageposition 'mit erhöhter Geschwindigkeit anfahren</p> <p>'Fehlermeldung</p>
<pre> 1290 ' == = Wartet eine festgelegte Zeit auf das Einschalten eines Signals == = 1300 ' IN:MX50SIG Überwachungssignal 1310 ' MX50SEC Überwachungszeit in Sekunden 1320 ' OUT:MY50SKP 0: fehlerfreie Ausführung, 1: Zeitüberschreitung 1330 *S50WON 1340 M_TIMER(1) = 0 1350 MY50SKP = 1 1360 M50SEC = MX50SEC * 1000 1370 *L50WON1 1380 IF M_TIMER(1) > M50SEC THEN *L50WON2 1390 IF M_IN(MX50SIG) = 1 THEN MY50SKP = 0 1400 IF MY50SKP = 0 THEN *L50WON2 1410 GOTO *L50WON1 1420 *L50WON2 1430 RETURN 1440 ' 1450 ' == = Wartet eine festgelegte Zeit auf das Ausschalten eines Signals == = 1460 ' IN:MX51SIG Überwachungssignal 1470 ' MX51SEC Überwachungszeit in Sekunden 1480 ' OUT:MY51SKP 0: fehlerfreie Ausführung, 1: Zeitüberschreitung 1490 *S51WOFF 1500 M_TIMER(1) = 0 1510 MY51SKP = 1 1520 M51SEC = MX51SEC * 1000 1530 *L51WOF1 1540 IF M_TIMER(1) > M51SEC THEN *L51WOF2 1550 IF M_IN(MX51SIG) = 0 THEN MY51SKP = 0 1560 IF MY51SKP = 0 THEN *L50WOF2 1570 GOTO *L51WOF1 1580 *L51WOF2 1590 RETURN </pre>	<p>Für den universellen Einsatz von Unterprogrammen ist es sinnvoll, einen allgemeinen Titel und allgemeine E/A-Definitionen zu verwenden.</p> <p>'Timer zurücksetzen 'Ausgangswert zurücksetzen 'Sekunden → Millisekunden</p> <p>'Ende bei Zeitüberschreitung 'Bei Einschalten des Signals 'Ausgangswert setzen 'Ende, wenn Signal EIN bestätigt</p> <p>'Timer zurücksetzen 'Ausgangswert zurücksetzen 'Sekunden → Millisekunden</p> <p>'Ende bei Zeitüberschreitung 'Bei Ausschalten des Signals 'Ausgangswert setzen 'Ende, wenn Signal AUS bestätigt</p>

Tab. 11-5: Beispielprogramm zur zeitverzögerten Signalprüfung

11.1.7 Synchronisierung durch externe Eingangssignale

Der Roboter ist meist Teil eines Gesamtsystems und wird in Abhängigkeit von anderen Komponenten genutzt. Wie kann eine Synchronisierung des Roboters mit weiteren Komponenten des Systems erfolgen?

Methode

Eine Synchronisierung des Gesamtsystems kann über die Ein- und Ausgabe externer Signale durchgeführt werden. Schalten Sie E/A-Signale über die Variablen M_IN und M_OUT. Verwenden Sie zur Definition von Ein- und Ausgangsvariablen die Anweisung DEF IO. Sie ermöglicht die Definition unterschiedlicher Variablen und die Vergabe von Variablennamen.

Programmbeispiel

Folgende Abbildung zeigt ein System, in dem ein Roboter jeweils 4 Werkstücke von einer Ladestation in einen Montagerahmen einsetzt, wenn sich in der Entladestation ein Montagerahmen befindet. Nach der vollständigen Bestückung eines Montagerahmens gibt der Roboter ein Signal aus und der Montagerahmen wird aus der Entladestation entnommen.

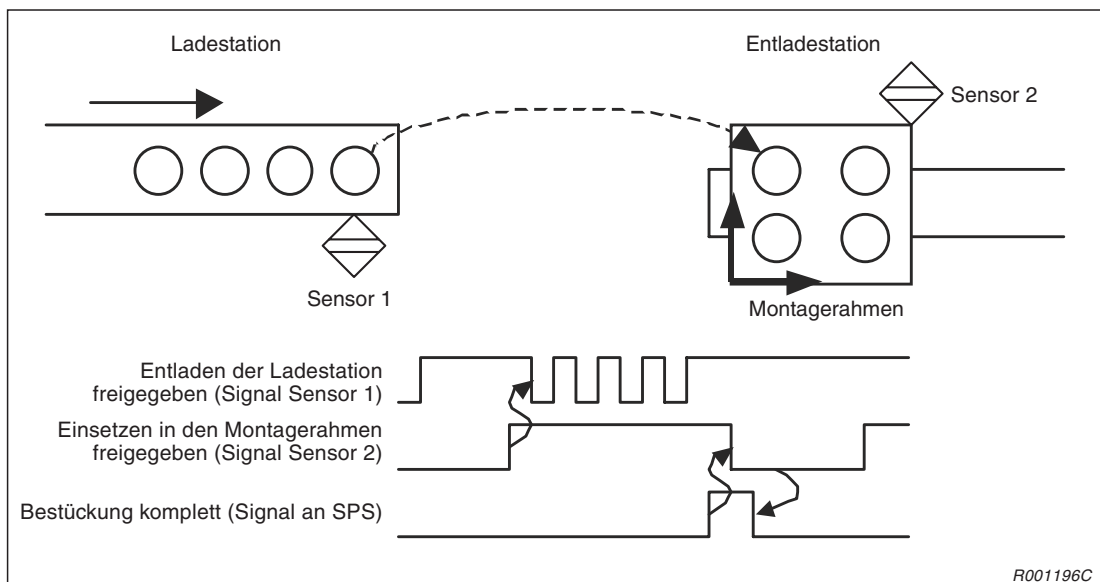


Abb. 11-9: Synchronisierung eines Gesamtsystems

Positionen	E/A-Signale
PTGET: Aufnahmeposition	M_IN(8): Sensor 1 (Entladen der Ladestation freigegeben)
PTPUT: Berechnete Ablageposition	M_IN(9): Sensor 2 (Einsetzen in den Montagerahmen freigegeben)
PT99: Rückzugsposition	M_OUT(8): Bestückung komplett
PL_FRM: Rahmenursprung	
PL_POS(4): Ablageposition im Montagerahmen (relative Koordinaten)	

Tab. 11-6: Übersicht der Positionen und Signale

Programm	Erläuterung
1000 DIM PL_POS(4)	
1010 DEF IO XL_GET = BIT,8	'Signal zur Freigabe der Aufnahme
1020 DEF IO XL_PUT = BIT,9	'Signal zur Freigabe der Ablage
1030 DEF IO YL_OUT = BIT,8	'Signal „Bestückung komplett“
1040 PL_FRM = FRAM(PTO,PTX,PTY)	
1050 MOV PT99	'Rückzugsposition anfahren
1060 HOPEN 1	'Hand öffnen
1070 '	
1080 *MAIN	
1090 IF XL_PUT = M_OFF THEN *MAIN	'Warte bis Entladestation bereit
1100 M00NUM = 0	'Setze Werkstückzähler auf „0“
1110 WHILE M00NUM < 4	'Schleife bis zur Bestückung mit 4 Werkstücken
1120 MOV PTGET,50	'Position über der Aufnahme position anfahren
1130 *L00WAIT1	
1140 IF XL_GET = M_OFF THEN *L00WAIT1	'Warte bis Ladestation bereit
1150 MOV PTGET	'Aufnahmeposition anfahren
1160 HCLOSE 1	'Werkstück aufnehmen
1170 MOV PTGET,50	'Position über der Aufnahme position anfahren
1180 PTPUT = PL_FRM * PL_POS(M00NUM + 1)	'Ablageposition berechnen
1190 MOV PTPUT,50	'Position über der Ablageposition anfahren
1200 MOV PTPUT	'Ablageposition anfahren
1210 HOPEN 1	'Werkstück ablegen
1220 MOV PTPUT,50	'Position über der Ablageposition anfahren
1230 M00NUM = M00NUM + 1	'Anzahl der abgelegten Werkstücke + 1
1240 WEND	'Nächstes Werkstück anfahren
1250 YL_OUT = M_OUT	'Nach Ablage von 4 Werkstücken 'Signal „Bestückung komplett“ ausgeben
1260 *L00WAIT2	
1270 IF XL_PUT = M_ON THEN *L00WAIT2	'Warte bis der bestückte Montagerahmen 'entfernt ist
1280 YL_OUT = M_OFF	'Signal „Bestückung komplett“ ausschalten
1290 GOTO MAIN	

Tab. 11-7: Beispielprogramm zur Synchronisierung eines Systems

11.1.8 Programmübergreifende Verwendung von Daten

Die programmübergreifende Verwendung von Daten ist z. B. notwendig, wenn Rechenergebnisse oder Positionsdaten eines Programmes in einem anderen Programm weiterverarbeitet werden sollen. Wie können Daten programmübergreifend verwendet werden?

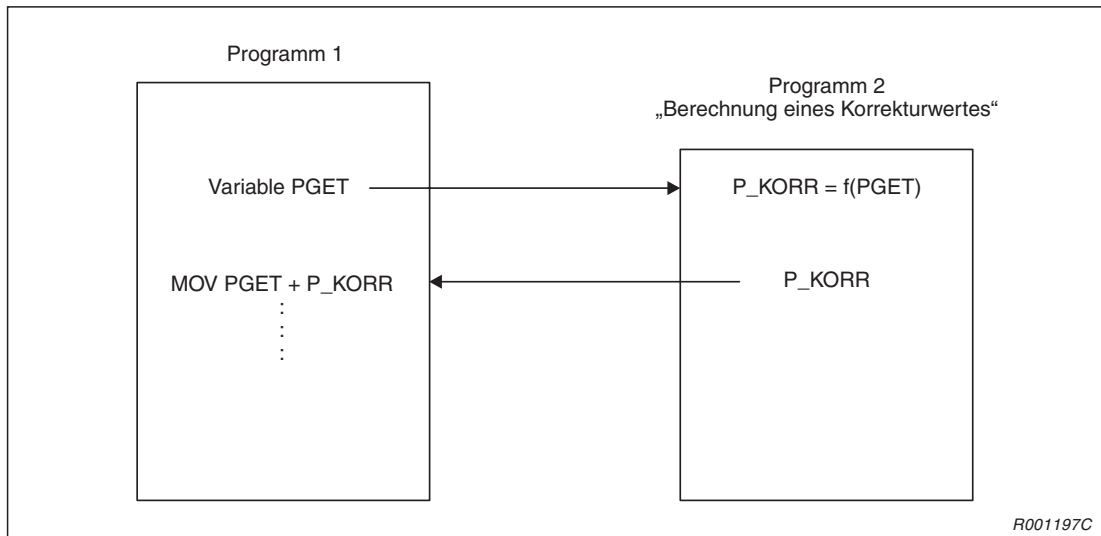


Abb. 11-10: Übergabe von Daten beim Aufruf eines Berechnungsprogramms

Methode

Die programmübergreifende Verwendung von Daten kann nicht durch den direkten Zugriff eines Programmes auf Daten eines anderen Programmes erfolgen. Die Übergabe von Werten ist über programmexterne Variablen oder benutzerdefinierte externe Variablen möglich (siehe Abschn. 5.1.10).

Programmbeispiel

Diese Beispiel zeigt die Übergabe der Daten PT001() von einem Unterprogramm an das Hauptprogramm. Die Übergabe erfolgt über programmexterne Variablen oder benutzerdefinierte Variablen.

Programm	Erläuterung
<p><Beispiel für eine Übergabe über programmexterne Variablen></p> <p><u>Hauptprogramm</u> 1010 CALLP "SUB1" 'Führe das Unterprogramm (SUB1.PRG) aus 1020 MOV P_100(1) 1030 MOV P_100(2) : 1010 CALLP "SUB2" 'Führe das Unterprogramm (SUB2.PRG) aus 1020 MOV P_100(1) 1030 MOV P_100(2) :</p> <p><u>Unterprogramm (SUB*.PRG)</u> 1000 DIM PT001(5) 1010 FOR M01 = 1 TO 5 1020 P_100(M01) = PT001(M01) 'Kopiere Daten in eine externe Variable 1030 NEXT 1040 END</p>	<p>Die Werte der Positionsdaten sind vom aufgerufenen Unterprogramm abhängig.</p> <p>Es wird dasselbe Programm mit unterschiedlichen Positionsdaten verwendet.</p>
<p><Beispiel für eine Übergabe über benutzerdefinierte externe Variablen></p> <p><u>Benutzerbasisprogramm (BASE.PRG)</u> 1000 DIM POS P_POS(5)</p> <p><u>Hauptprogramm</u> 1000 DIM POS P_POS(5) 1010 CALLP "SUB1" 'Führe das Unterprogramm (SUB1.PRG) aus 1020 MOV P_POS(1) 1030 MOV P_POS(2) : 1010 CALLP "SUB2" 'Führe das Unterprogramm (SUB2.PRG) aus 1020 MOV P_POS(1) 1030 MOV P_POS(2) :</p> <p><u>Unterprogramm (SUB*.PRG)</u> 1000 DIM POS P_POS(5) 1010 DIM PT001(5) 1020 FOR M01 = 1 TO 5 1030 P_POS(M01) = PT001(M01) 'Kopiere Daten in eine externe Variable 1040 NEXT 1050 END</p>	<p>Schreiben Sie nach der Installation „BASE.PRG“ in den Parameter PRGUSR. Schalten Sie anschließend die Versorgungsspannung aus und wieder ein.</p>

Tab. 11-8: Beispielprogramm für die Übergabe von Daten

11.1.9 Überwachung der Abweichung von Soll- und Istposition

Einige Anwendungen erfordern eine Überwachung, ob sich der Roboter in der gewünschten Position befindet. So kann es notwendig sein, dass der Roboter sich beim Start in einer bestimmten Position befindet, dass eine Meldung bei Erreichen der Zielposition erfolgt oder dass der Verfahrweg zum Rückzugspunkt in Abhängigkeit der Startposition geändert wird. Wie ist ein solches Verhalten zu erreichen?

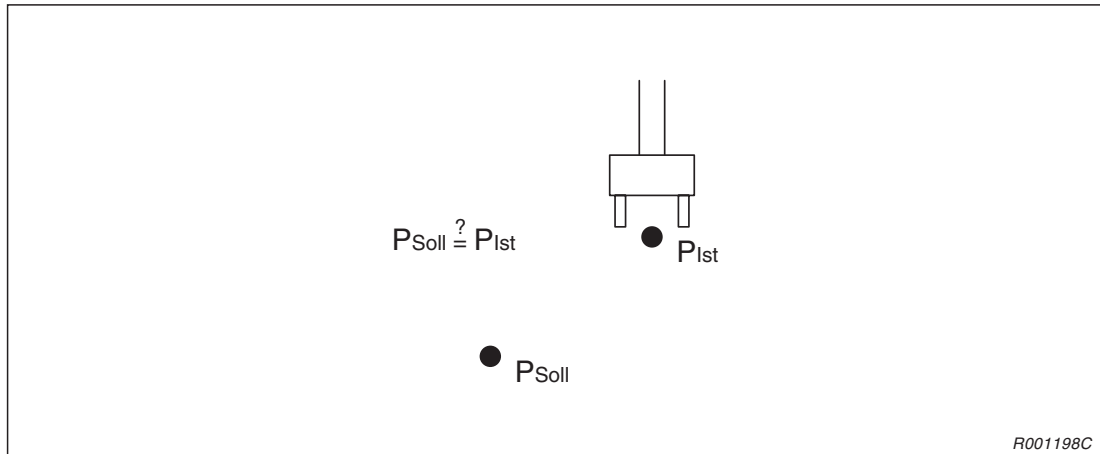


Abb. 11-11: Überwachung der Abweichung von Soll- und Istposition

Methode

Es gibt mehrere Möglichkeiten die Abweichung von Soll- und Istposition zu überprüfen. Wählen Sie die Möglichkeit, die für Ihre Anwendung die günstigste ist.

- **Benutzerdefinierter Bereich**
Diese Möglichkeit bietet sich in Fällen mit kleinem Überwachungsbereich an, in denen die Position nicht ständig geändert wird, eine spätere Änderung jedoch leicht möglich ist. Sie wäre also ideal, um z. B. eine Startposition zu überwachen. Dabei ist die Definition von bis zu 8 benutzerdefinierten Bereichen möglich.
Die Vorteile der Methode sind die kontinuierliche Überwachung, auch wenn kein Programmstart ausgeführt wurde, und die leichte Änderung der Bereiche über Parameter, die programmunabhängig durchgeführt werden kann. Weitere Informationen finden Sie in Abschn. 9.9.
- **Funktionen ZONE und ZONE2**
Verwenden Sie diese Funktionen, um zu prüfen, ob der Roboter sich innerhalb eines definierten Bereiches befindet. Bei der Funktion ZONE ist der Bereich quaderförmig, bei der Funktion ZONE2 zylindrisch oder kugelförmig.
- **Funktion DIST**
Eine einfache Methode zur Überprüfung, ob der Roboter eine Zielposition erreicht hat, bietet die Funktion DIST, in denen der Abstand zwischen der aktuellen (P_CURR) und der Zielposition berechnet wird. Beachten Sie jedoch, dass Winkel unberücksichtigt bleiben.
Eine weitere einfache Möglichkeit zur Überprüfung der Abweichung zwischen der aktuellen Position und der Zielposition bei aktivierter Achsenweichheit bietet die Funktion M_CMPDST.

- Vergleich der einzelnen Komponenten von Positionsvariablen
 Folgendes Programm zeigt ein Beispiel, in denen die Abweichung jeder einzelnen Positionskomponente ermittelt wird. Hier erfolgt ein Vergleich der Positionen PX52A und PX52B. Die Abweichung der X-Komponente darf $\pm 0,01$ mm und die der Y-Komponente $\pm 10,0$ mm betragen. Für die Z-Komponente erfolgt keine Prüfung. Ist die Abweichung der Komponenten A, B, und C kleiner als $\pm 1,0$ Grad wird die Ausgangsvariable MY52RET auf den Wert „1“ gesetzt.
 Beachten Sie, dass die Einheit der Komponenten A, B und C Radiant ist. Wandeln Sie daher 1,0 Grad vor der Prüfung mit Hilfe der Funktion RAD zuerst in Radiant um.

Programm	Erläuterung
1000 'PX52A 1010 'PX52B 1020 'MY52RET	'Zu prüfende Koordinate 1 'Zu prüfende Koordinate 2 'Auf „1“ setzen, wenn 'Abweichung zwischen PX52A 'und PX52B kleiner als Vorgabe, 'sonst auf „0“ setzen
1030 *S52PSCHK	'Unterprogramm zur Prüfung 'einer Positionsänderung 'Rückmeldung initialisieren
1040 MY52RET = 0	
1050 IF ABS(PX52B.X - PX52A.X) > 0.01 THEN GOTO *L52END	
1060 IF ABS(PX52B.Y - PX52A.Y) > 10.0 THEN GOTO *L52END	
1070 ' Keine Prüfung der Z-Komponente	
1080 IF ABS(PX52B.A - PX52A.A) > RAD(1.0) THEN GOTO *L52END	
1090 IF ABS(PX52B.B - PX52A.B) > RAD(1.0) THEN GOTO *L52END	
1100 IF ABS(PX52B.C - PX52A.C) > RAD(1.0) THEN GOTO *L52END	
1110 MY52RET = 1	'Rückmeldung auf „1“ setzen
1120 *L52END	
1130 RETURN	

Tab. 11-9: Beispielprogramm für die Überwachung von Positionsabweichungen

11.1.10 Verkürzung der Zykluszeit

In vielen Fällen kann eine Erhöhung der Produktivität durch eine Verkürzung der Zykluszeiten erreicht werden. Wie lässt sich die Zykluszeit auf einfache Weise reduzieren?

Methode

Folgende grundlegende Methoden können zu einer Verkürzung der Zykluszeit führen.

- **Anpassung der Geschwindigkeit**
Prüfen Sie, ob die Geschwindigkeit nach langsam auszuführenden Verfahrbewegungen wieder über die Anweisungen OVRD und SPD erhöht werden kann. Verwenden Sie als Grundgeschwindigkeit im Programm die für Ihre Anwendung maximal zulässige Geschwindigkeit. Reduzieren Sie diese nur für Verfahrbewegungen, die langsam ausgeführt werden müssen. Setzen Sie die Geschwindigkeit nach einer Verringerung wieder mittels der Anweisungen OVRD 100 und SPD M_NSPD (optimale Geschwindigkeit) auf den für Ihre Anwendung maximal zulässigen Wert.
- **Zwischenpositionen einfügen oder entfernen**
In den meisten Fällen durchfährt der Roboter bei der Bewegung von einer Position (z. B. Aufnahmeposition) zu einer anderen Position (z. B. Ablageposition) Zwischenpositionen. Befinden sich diese Zwischenpositionen an geeigneten Stellen? Die Anzahl und Lage der Zwischenpositionen ist so zu wählen, dass die Zielposition mit möglichst wenig Schritten erreicht wird. Oft ist es jedoch sinnvoll, vor der Zielposition eine Zwischenposition einzufügen, obwohl auch eine direkte Anfahrt der Zielposition möglich wäre. Dadurch ist eine Anfahrt der Zwischenposition mit der maximal zulässigen Geschwindigkeit möglich.
- **Entwurf eines Verfahrweges ohne extreme Übergänge**
Entwerfen Sie einen Verfahrweg mit möglichst sanften Übergängen. Besteht keine Kollisionsgefahr mit umliegenden Einrichtungen, geben Sie die CNT-Einstellung für die Durchfahrt der Zwischenpositionen frei. Dadurch wird eine Verminderung der Geschwindigkeit an den Zwischenpositionen verhindert und der Zyklus schneller durchlaufen.

Programmbeispiel

Im Folgenden werden drei Programme mit identischer Funktion gezeigt. Alle Programme bewegen den Roboter von einer Startposition (PT03) über eine Zwischenposition (PT02) zu einer Zielposition (PT01).

Im zweiten und dritten Programm werden Methoden zur Verkürzung der Zykluszeit angewendet.

Block 1: Der Roboter wird von der Startposition (PT03) über die Zwischenposition (PT02) zur Zielposition (PT01) bewegt.

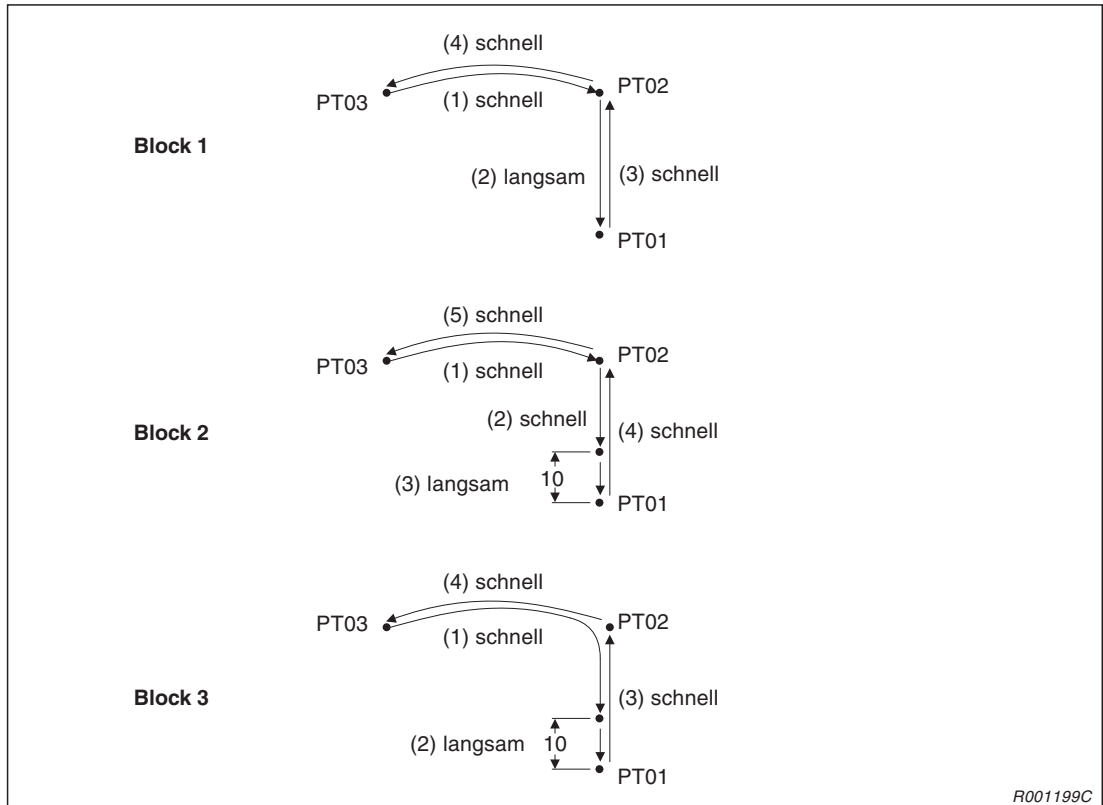
Block 2: Der Roboter fährt mit der maximal zulässigen Geschwindigkeit eine Zwischenposition kurz vor der Zielposition (PT01) an.

Block 3: Die Zwischenposition wird mit freigegebener CNT-Einstellung durchfahren.

Bei allen drei Blöcken wird die Zyklusdauer über die Roboterstatusvariable M_TIMER erfasst und das Ergebnis in die Variablen M01, M02 und M03 übertragen.

Positionen	E/A-Signale
PT01: Zielposition PT02: Zwischenposition PT03: Startposition	—

Tab. 11-10: Übersicht der Positionen und Signale



R001199C

Abb. 11-12: Verkürzung der Zykluszeit

Programm	Erläuterung
<pre> 1000 MOV PT03 1010 '===== Block 1 ===== 1020 M_TIMER(1) = 0 1030 OVRD 100 1040 MOV PT02 1050 OVRD 10 1060 MVS PT01 1070 DLY 0.1 1080 OVRD 100 1090 MVS PT02 1100 MOV PT03 1110 DLY 0.1 1120 M01 = M_TIMER(1) 1130 '===== Block 2 ===== 1140 M_TIMER(1) = 0 1150 OVRD 100 1160 MOV PT02 1170 MVS PT01,-10 1180 OVRD 10 1190 MVS PT01 1200 DLY 0.1 1210 OVRD 100 1220 MVS PT02 1230 MOV PT03 1240 DLY 0.1 1250 M02 = M_TIMER(1) 1260 '===== Block 3 ===== 1270 M_TIMER(1) = 0 1280 OVRD 100 1290 CNT 1 1300 MOV PT02 1310 CNT 0 1320 MVS PT01,-10 1330 OVRD 10 1340 MVS PT01 1350 DLY 0.1 1360 OVRD 100 1370 MVS PT02 1380 MOV PT03 1390 DLY 0.1 1400 M03 = M_TIMER(1) </pre>	<p>Programm ohne Verkürzung der Zykluszeit</p> <p>Verkürzung der Zykluszeit durch Einfügen einer Zwischenposition</p> <p>Verkürzung der Zykluszeit durch Verwendung der CNT-Einstellung</p>

Tab. 11-11: Beispielprogramm zur Verkürzung der Zykluszeit

11.2 Programmiertechniken für fortgeschrittene Einsteiger

11.2.1 Schnelle Anpassung an unterschiedliche Werkstückgeometrien

Im Allgemeinen ist ein System nicht ständig an neue Werkstückanordnungen und -abmessungen anzupassen. Eine häufige Änderung der Bewegungsdaten und eine Steuerung über einen zentralen Rechner sind somit nicht erforderlich. Allerdings ist das erneute Teachen der Positionsdaten bei jeder Änderung sehr aufwändig. Wie können Positionsdaten einfach angepasst und in Abhängigkeit der Werkstückgeometrien verwendet werden?

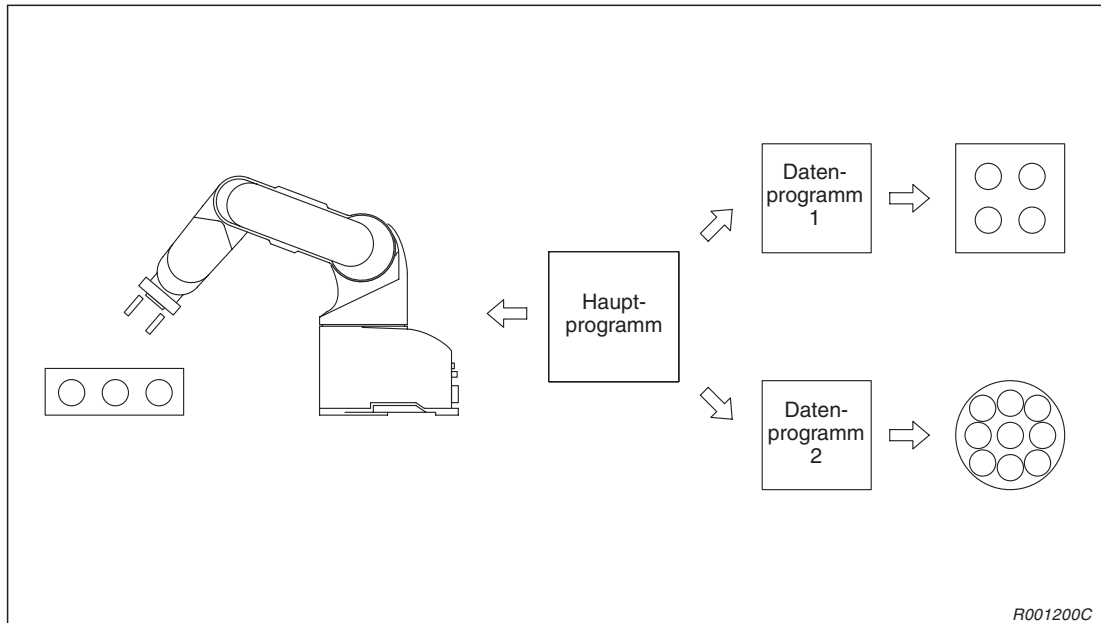


Abb. 11-13: Schnelle Anpassung an unterschiedliche Werkstückgeometrien

Methode

Teilen Sie das Programm in ein Hauptprogramm und Datenprogramme für die verschiedenen Werkstückgeometrien auf. Diese Methode erlaubt die Speicherung anwendungsspezifischer Daten in unterschiedlichen Programmen. Übertragen Sie die Daten zwischen Datenprogramm und Hauptprogramm über externe Variablen. Jedes Datenprogramm besteht aus Positionsdaten und einer Routine, die die Positionsdaten in externe Variablen schreibt. Bei Aufruf eines bestimmten Datenprogramms durch das Hauptprogramm werden die anwendungsspezifischen Daten in externe Variablen übertragen.

Programmbeispiel

Im folgenden Beispiel erfolgt der Aufruf der Datenprogramme (WK1.PRG bis WK63.PRG) in Abhängigkeit externer Eingangssignale. Die Positionsdaten aus den Datenprogrammen werden über die externe Variable P_100(□) in das Hauptprogramm übertragen.

Programm	Erläuterung
<pre> <Hauptprogramm> 1000 DEF IO XL_SETNO = BIT,8 1010 DEF IO XL_PRGNO = BYTE,10,&H3F 1020 ' 1030 *MAIN 1040 IF XL_SETNO = M_OFF THEN *MAIN 1050 C00PRG\$ = "WK" + STR\$(XL_PRGNO) 1060 CALLP C00PRG\$ 1070 ' 1080 FOR M00WK = 1 TO 10 1090 P00TMP = P_100(M00WK) 1100 MOV P00TMP,50 1110 MOV P00TMP 1120 DLY 1 1130 MOV P00TMP,50 1140 NEXT M00WK 1150 GOTO *MAIN 1160 END </pre>	<p>—</p> <p>'Eingabe der Werkstücknummern abgeschlossen 'Eingegebene Werkstücknummer 1 bis 63</p> <p>'Warte, bis alle Werkstücknummern eingelesen sind 'Erzeuge Programmnamen 'Schreibe Daten in eine externe Variable</p> <p>'Zähler für Werkstücke 'Erzeuge Positionsdaten 'Position vor der Zielposition anfahren 'Zielposition anfahren</p> <p>'Position vor der Zielposition anfahren 'Nächstes Werkstück anfahren 'Nächsten Datensatz einlesen</p>
<pre> <Datenprogramme: WK1.PRG bis WK63.PRG> 1000 DIM PTDATA(10) 1010 FOR M01 = 1 TO 10 1020 P_100(M01) = PTDATA(M01) 1030 NEXT M01 1040 END </pre>	<p>—</p> <p>'Speicherbereich der Daten 'Zähler für Werkstücke 'Schreibe Daten in eine externe Variable</p>

Tab. 11-12: Programmbeispiel zur schnellen Anpassung an unterschiedliche Werkstückgeometrien

11.2.2 Vielseitige Anwendung der Palettierungsfunktion

Im Allgemeinen dient die Palettierungsfunktion zum Beladen von Paletten oder Stapelbehältern. Im Folgenden werden weitere Anwendungsmöglichkeiten gezeigt.

Methode

- Anwendung der Palettierungsfunktion in einer beliebig orientierten Ebene
Die Palettierungsfunktion dient nicht nur zum Beladen von Paletten oder Stapelbehältern, die auf dem Boden stehen, sie kann auf beliebig orientierte Ebenen angewendet werden.

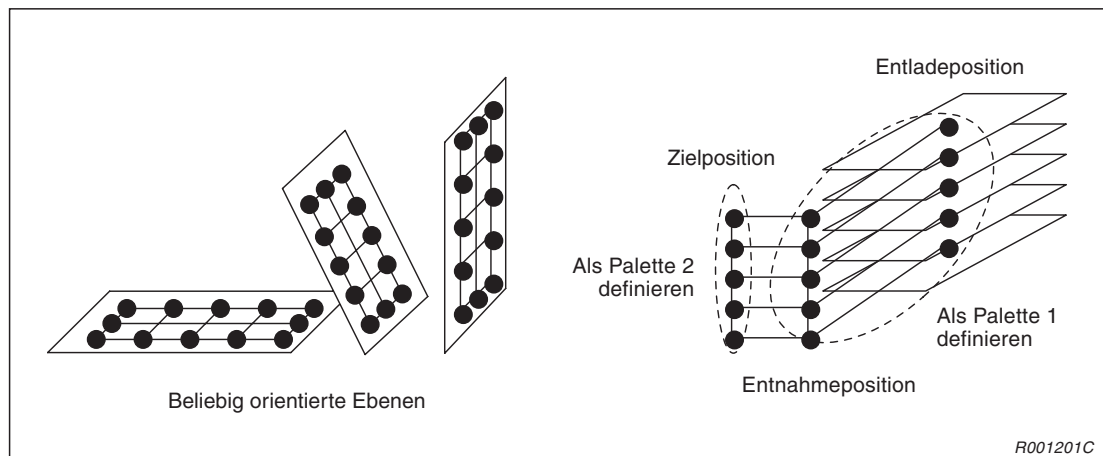


Abb. 11-14: Anwendung der Palettierungsfunktion auf beliebig orientierte Ebenen

Die Palettierungsfunktion ermöglicht die Definition von gitterförmigen Anordnungen (siehe Abbildung oben) mit verschiedenen Orientierungen. Positionen, die z. B. vor der Aufnahme- oder Ablageposition von in Ebenen angeordneten Punkten liegen, können somit leicht festgelegt werden. Eine Festlegung dieser Positionen kann auch ohne die Palettierungsfunktion erfolgen, durch die Verwendung der Funktion wird jedoch zur Berechnung der Positionen nur eine Anweisung benötigt, woraus eine verkürzte Zykluszeit resultiert.

- Anwendung der Palettierungsfunktion auf eine Reihe

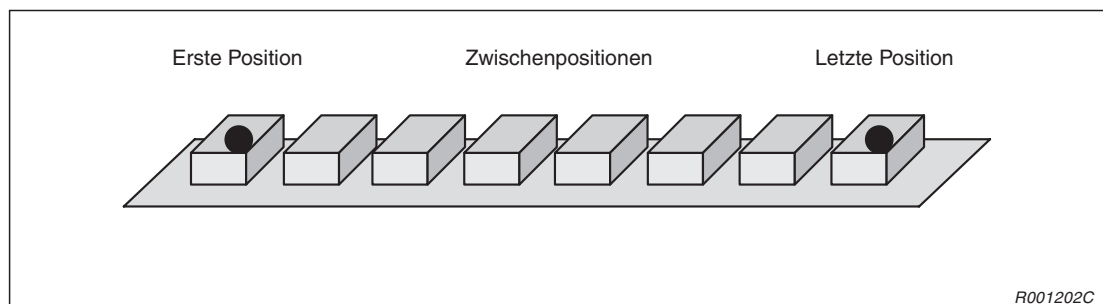


Abb. 11-15: Anwendung der Palettierungsfunktion auf eine Reihe

Wird die Palettierungsfunktion auf Werkstücke angewendet, die in einer Reihe angeordnet sind, entfällt das aufwändige Teachen aller Positionen. Lediglich die erste und letzte Position müssen geteacht werden, die Berechnung aller anderen Positionen erfolgt automatisch.

Eine detaillierte Beschreibung der Palettierungsfunktion finden Sie in Abschn. 4.4 „Palettierung“.

Programmbeispiel

Sind Werkstücke in Ebenen angeordnet, können die Entladepositionen als Palette 1 und die Beladepositionen als Palette 2 definiert werden. Bei Werkstücken, die in einer Reihe angeordnet sind, erfolgt die Definition in derselben Weise, indem die Positionsvariable 2-mal in der Anweisung DEF PLT verwendet wird.

Programm	Erläuterung
<Anordnung der Werkstücke in Ebenen> PT1: unterste Ebene PT2: oberste Ebene PT3: vor der untersten Ebene PT4: vor der obersten Ebene PT5: Ablageposition der untersten Ebene PT6: Ablageposition der obersten Ebene (für 5 Ebenen) ----- 1000 DEF PLT 1,PT1,PT2,PT3,PT4,5,2,2 1010 DEF PLT 2,PT5,PT6,PT5,PT6,5,1,1 1020 PTMP1 = PLT 1,1 'Position der unteren Ebene in PTMP1 übertragen 1030 PTMP2 = PLT 1,5 'Position der oberen Ebene in PTMP2 übertragen 1040 PTMP3 = PLT 1,5 + 1 'Position vor der unteren Ebene in PTMP3 übertragen 1050 PTMP4 = PLT 1,5 + 5 'Position vor der oberen Ebene in PTMP4 übertragen 1060 PTMP5 = PLT 2,1 'Ablageposition der unteren Ebene in PTMP5 übertragen 1070 PTMP6 = PLT 2,5 'Ablageposition der oberen Ebene in PTMP6 übertragen	—
<Anordnung der Werkstücke in Reihen> P1: 1te Position P2: nte Position (bei 10 Werkstücken in einer Reihe) ----- 1000 DEF PLT 1,PT1,PT2,PT1,PT2,10,1,1 1010 PTMP1 = PLT 1,1 '1te Position in PTMP1 übertragen 1020 PTMP2 = PLT 1,10 '10te Position in PTMP2 übertragen	—

Tab. 11-13: Beispielprogramm zur Palettierungsfunktion

11.2.3 Schreiben eines Kommunikationsprogramms

Wie kann eine Kommunikationsverbindung zwischen Roboter und PC oder SPS zur Ausführung von Arbeitsanweisungen programmiert werden?

Methode

Die Programmierung von Kommunikationsverbindungen kann auf verschiedene Arten erfolgen. Von der Verwendung der INPUT-Anweisung über die Kommunikation bei Eintreffen eines bestimmten Befehls bis hin zum Multitasking werden von den verschiedenen Methoden unterschiedliche Merkmale unterstützt. Bei weitläufiger Einteilung kann man drei Methoden unterscheiden:

Programmierniveau	Methode	Beschreibung
Niedrig	INPUT-Anweisung	Ermöglicht die einfache Erstellung von Programmen. Da die INPUT-Anweisung jedoch bis zu einem Dateneingang aktiv ist, kann der Roboter in dieser Zeit keine anderen Aktionen ausführen. Weitere Informationen über diese Methode finden Sie in Abschn. 9.16.
Mittel	Kommunikations-Interrupt	Zur Ausführung des Kommunikationsprozesses ist die Verarbeitung einer weiteren Anweisung nötig. Erfolgt diese Ausführungsanweisung während des Roboterbetriebes, stoppt der Roboter.
Hoch	Multitasking	Bei dieser Methode kann der Kommunikationsprozess parallel zum Roboterbetrieb ausgeführt werden. Dadurch ist es möglich, Verfahrbewegungen auszuführen, während gleichzeitig Anweisungen eingelesen oder komplexe Texte analysiert werden. Nacheinander eingehende Befehle werden somit ausgeführt, ohne den Roboter zu stoppen.

Tab. 11-14: Methoden zur Programmierung von Kommunikationsverbindungen

Zwei der drei Methoden zur Erstellung eines Kommunikationsprogramms werden im Folgenden näher erläutert.

● **Kommunikations-Interrupt**

Folgendes Programmbeispiel zeigt die Verwendung eines Kommunikations-Interrupts. In Abhängigkeit des Befehls („GET“ oder „PUT“), der über die Kommunikationsleitung 1 empfangen wird, erfolgt die Ausführung eines Kommunikationsprozesses. Bei Beendigung des Prozesses wird die Zeichenkette „FIN“ ausgegeben.

Programm	Erläuterung
1000 OPEN "COM1:" AS #1	'Öffnet die Kommunikationsleitung 1 als Datei Nr. 1 (#1)
1010 ON COM(1) GOSUB *S98COMRC	'Einstellung des Kommunikations-Interrupts
1020 '	
1030 ML_REQ = 0	'Merker zur Werkstückanforderung zurücksetzen
1040 COM(1) ON	'Kommunikations-Interrupt freigeben
1050 *MAIN	'Beginn der Hauptschleife
1060 GOSUB *S50CHECK	'Normalbetrieb bis zur Anforderung
1070 SELECT ML_REQ	'Bei einer Werkstückanforderung
1080 CASE 1	
1090 GOSUB *S51GET	'Werkstückroutine aufrufen
1100 GOSUB *S99FIN	'Routine abgeschlossen
1110 BREAK	
1120 GOSUB *S52PUT	'Werkstückroutine aufrufen
1140 GOSUB *S99FIN	'Routine abgeschlossen
1150 BREAK	
1160 END SELECT	
1170 GOTO *MAIN	
1180 '	
1190 *S50CHECK	'Normalbetrieb
1200 COM(1) ON	'Kommunikations-Interrupt freigeben
1210 'Beschreibung des Normalbetriebs	
1220 COM(1) STOP	'Kommunikations-Interrupt stoppen
1230 RETURN	
1240 '	
1250 *S51GET	'Beim Empfang von „GET“
1260 'Beschreibung der Routine beim Empfang von „GET“	
1270 RETURN	
1280 '	
1290 *S52PUT	'Beim Empfang von „PUT“
1300 'Beschreibung der Routine beim Empfang von „PUT“	
1310 RETURN	
1320 '	
1330 *S98COMRC	'Abfrage Kommunikations-Interrupt
1340 INPUT #1,C98CMD\$	'Empfangene Zeichenkette einlesen
1350 ML_REQ = 0	'Merker zum Empfang der Abfrage zurücksetzen
1360 IF C98CMD\$ = "GET" THEN ML_REQ = 1	'Bei „GET“
1370 IF C98CMD\$ = "PUT" THEN ML_REQ = 2	'Bei „PUT“
1380 RETURN 1	
1390 '	
1400 *S99FIN	
1410 PRINT #1,"FIN"	'Abschluss ausgeben
1420 ML_REQ = 0	'Merker zur Werkstückanforderung zurücksetzen
1430 RETURN	

Tab. 11-15: Programmbeispiel zur Verwendung eines Kommunikations-Interrupts

● Multitasking

Folgende Programme zeigen Kommunikationsprozesse unter Verwendung des Multitaskings. In Abhängigkeit eines Befehls (hier „GET“ oder „PUT“), der über die Kommunikationsleitung 1 empfangen wird, erfolgt die Ausführung eines Kommunikationsprozesses. Bei Beendigung des Prozesses wird die Zeichenkette „FIN“ ausgegeben.

Zuerst wird das Hauptprogramm in Programmplatz 1 und das Kommunikationsprogramm in Programmplatz 2 gestartet. Die Startbedingung des Kommunikationsprogrammes ist auf „ALWAYS“ gesetzt.

Die Überprüfung, ob das Kommunikationsprogramm empfangsbereit ist und ob vom Hauptprogramm eine Kommunikationsanforderung an das Kommunikationsprogramm gesendet wurde, erfolgt über externe Variablen. Zur Prüfung der Empfangsbereitschaft des Kommunikationsprogramms dient die externe Variable M_100(10). Die Kommunikationsanforderung vom Hauptprogramm an das Unterprogramm erfolgt über die externe Variable M_05. Für den Fall, dass das Kommunikationsprogramm vor Beendigung der Empfangsbereitschaftsprüfung im Hauptprogramm weitere Befehle empfängt, stehen im Hauptprogramm 10 Empfangspuffer zur Verfügung (externe Variable M_100(10)).

Programm	Erläuterung
<pre> <Hauptprogramm> 1000 *MAIN 1010 IF M_01 = M_02 THEN GOTO *MAIN 1020 M_01 = (M_01 MOD 10) + 1 1030 SELECT M_100(M_01) 1040 CASE 1 1050 GOSUB *S51GET 1060 M_05 = 1 1070 BREAK 1080 CASE 2 1090 GOSUB *S52PUT 1100 M_05 = 1 1110 BREAK 1120 END SELECT 1130 GOTO *MAIN 1140 ' 1150 *S51GET 1160 'Beschreibung der Routine beim Empfang von „GET“ 1170 RETURN 1180 ' 1190 *S52PUT 1200 'Beschreibung der Routine beim Empfang von „PUT“ 1210 RETURN </pre>	<p>Warteschleife, bis Signaleingabe erfolgt Leseindex + 1 Werkstückroutine aufrufen Anfrage zum Senden der Abschlussnachricht Werkstückroutine aufrufen Anfrage zum Senden der Abschlussnachricht Beim Empfang von „GET“ Beim Empfang von „PUT“</p>
<pre> <Kommunikationsprogramm> 1000 CLOSE #1 1010 OPEN "COM1:" AS #1 1020 ON COM(1) GOSUB *S98COMRC 1030 ' 1040 M_01 = 1 1050 M_02 = 1 1060 COM(1) ON 1070 *MAIN 1080 IF M_05 = 1 THEN GOSUB *S99FIN 1090 GOTO MAIN 1100 ' 1110 *S98COMRC 1120 INPUT #1,C98CMD\$ 1130 IF C98CMD\$ = "GET" THEN M_100(M_02) = 1 1140 IF C98CMD\$ = "PUT" THEN M_100(M_02) = 2 1150 M_02 = (M_02 MOD 10) + 1 1160 RETURN 1 1170 ' 1180 *S99FIN 1190 PRINT #1,"FIN" 1200 M_05 = 0 1210 RETURN </pre>	<p>Kommunikationsleitung schließen Öffnet die Kommunikationsleitung 1 als Datei Nr. 1 Einstellung des Kommunikations-Interrupts Leseindex zurücksetzen Schreibindex zurücksetzen Kommunikations-Interrupt freigeben Beginn der Hauptschleife Warte auf Sendeaufforderung Unterprogramm für Kommunikations-Interrupts Empfangene Zeichenkette einlesen Bei „GET“ Bei „PUT“ Addiere nach dem Schreibvorgang eine 1 zum Schreibindex Abschluss ausgeben Merker zur Werkstückanforderung zurücksetzen</p>

Tab. 11-16: Programmbeispiel zur Verwendung der Mutitasking-Methode

11.2.4 Reduzierung von geteachten Positionen

Verläuft ein Fahrweg durch zu viele geteachte Positionen, ist die Fahrbewegung oft nicht gleichmäßig. Wie kann die Anzahl der geteachten Positionen reduziert werden?

Methode

Der Roboter muss bestimmte Positionen durchfahren, um Werkstücke zu handhaben, um auf umliegende Einrichtungen zuzugreifen oder um Punkte in der gewünschten Weise anzufahren. Ein Teachen aller Positionen ist jedoch nicht in jedem Falle notwendig. Die Anzahl der geteachten Positionen lässt sich z. B. durch die Verwendung relativer Positionen reduzieren. Relative Positionen stehen in Bezug zu anderen Positionen, während geteachte Positionen auf die Grundposition des Roboters bezogen und somit absolute Positionen sind.

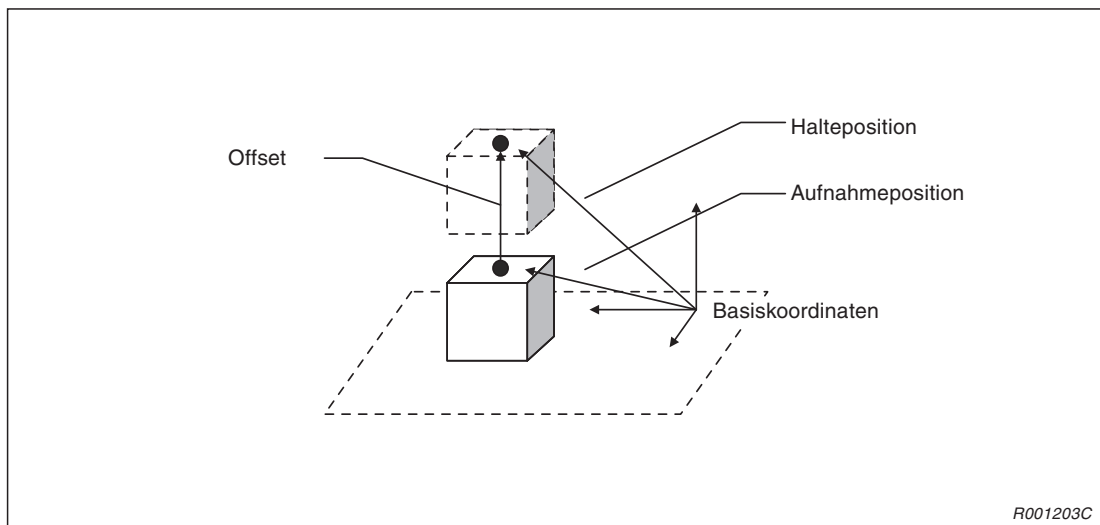


Abb. 11-16: Absolute und relative Positionen

Die Abbildung oben verdeutlicht zwei Methoden des Roboters, um ein Werkstück aufzunehmen und zu heben:

- Beide Positionen, die Aufnahme- und die Halteposition, werden geteacht.
- Nur die Aufnahme-Position wird geteacht. Die Anfahrt der Halteposition erfolgt durch Angabe eines Offsets.

Die zweite Methode erfordert zwar eine Berechnung, sie benötigt jedoch nur eine geteachte Position. Auch bei einer neuen Festlegung der Positionen muss bei der zweiten Methode nur ein Punkt neu geteacht werden, während bei der ersten Methode zwei Punkte neu geteacht werden müssen.

Programmbeispiel

In folgendem Programmbeispiel wird eine Bewegung des Roboters im Roboterkoordinatensystem nach oben durch eine Positions Berechnung mit dem Operator „+“ erreicht. Die Berechnung einer Zielposition im Werkzeugkoordinatensystem eines 6-achsigen Roboters erfolgt über den Operator „*“ oder durch die Angabe des Abstands in den Anweisungen MOV oder MVS.

Programm	Erläuterung
<Teaching-Methode> 1000 MOV GET 'Aufnahmeposition anfahren (geteacht) 1010 MOV PAPER 'Halteposition anfahren (geteacht) 1020 HLT	—
<Offset-Methode (Roboterkoordinatensystem)> 1000 PAPER = PGET + POFB 'Halteposition berechnen 1010 MOV PGET 'Aufnahmeposition anfahren (geteacht) 1020 MOV PAPER 'Halteposition anfahren (berechnet) 1030 HLT	—
<Offset-Methode (Werkzeugkoordinatensystem)> 1000 PAPER = PGET * POFB 'Halteposition berechnen 1010 MOV PGET 'Aufnahmeposition anfahren (geteacht) 1020 MOV PAPER 'Halteposition anfahren (berechnet) 1030 ' 1040 MOV PGET 'Aufnahmeposition anfahren (geteacht) 1050 MOV PGET,-50 'Halteposition anfahren (über Abstand festgelegt) 1060 HLT	siehe ①

Tab. 11-17: Programmbeispiel zum Anfahren von Positionen

- ① Beachten Sie, dass die Richtungen im Werkzeugkoordinatensystem bei SCARA-Robotern und Vertikal-Knickarmrobotern entgegengesetzt sind.

HINWEIS

Sind die Werkstücke in gleichbleibenden Abständen angeordnet, kann zur Berechnung der Zwischenpositionen die Palettierungsfunktion verwendet werden.

11.2.5 Verwendung einer P-Variablen in einem Zähler

Ein Programm soll bei der Installation auf dem Steuergerät initialisiert werden. Wie kann ein Initialisieren eines Zählers, der nicht beim Ausschalten oder beim täglichen Programmstart zurückgesetzt werden soll, erfolgen?

Methode

In diesem Fall kann eine Initialisierung durch Verwendung einer externen Variablen erfolgen. Dazu muss jedoch die Variable nach der Installation zurückgesetzt werden. Hier wird eine Methode unter Verwendung einer Positionsvariablen gezeigt. Positionsvariablen dienen in der Regel zur Speicherung von Koordinatenwerten; allerdings ist auch eine Speicherung anderer Daten in den einzelnen Komponenten möglich. Im Gegensatz zu numerischen Variablen oder Zeichenkettenvariablen können Positionsvariablen ähnlich wie Befehlszeilen als Teil eines Programms gespeichert werden. Diese Eigenschaft ermöglicht den Einsatz einer Positionsvariablen als Zähler, der ausschließlich bei der Installation des Programms initialisiert wird und danach nicht mehr.

Programmbeispiel

Dieses Programm zählt, wie oft die Spannungsversorgung ein- und wieder ausgeschaltet wird.

Dabei wird folgende Positionsvariable verwendet:

PDATA = (0.0, 0.0, 0.0, 0.0, 0.0, 0.0)(0.0, 0.0)

X-Komponente: Initialisierungsmerker, Y-Komponente: Wert des Zählers

Stellen Sie folgende Zeilen an den Beginn eines Programms oder einer Initialisierungsroutine:

Programm	Erläuterung
<pre> <Hauptprogramm> 1000 IF PDATA.X = 1 THEN 1010 PDATA.Y = 0 1020 PDATA.X = 0 1030 ENDIF 1040 PDATA.Y = PDATA.Y + 1 1050 ' 1060 ' 1070 *LOOP 1080 ' Hauptroutine 1090 GOTO *LOOP </pre>	—

Tab. 11-18: Programmbeispiel für die Initialisierung eines Zählers

Setzen Sie PDATA.X auf „1“ und speichern Sie das Programm als Stapeldatei oder erstellen Sie ein Backup. Beim Aufruf oder bei der Wiederherstellung des Programms wird der Zähler PDATA.Y auf „0“ gesetzt. Danach bewirkt jedes Ein- und Ausschalten eine Erhöhung des Zählers um „1“. Bei jeder Ausführung des Programmanfangs erfolgt eine Erhöhung des Zählers. Achten Sie daher darauf, dass keine Sprünge zum Programmanfang ausgeführt werden.

11.2.6 Sensorgesteuerte Übertragung von Positionsdaten

In Abhängigkeit eines Sensorsignals soll eine Übertragung der Roboterkoordinaten, z. B. für Kompensationszwecke, erfolgen, ohne den Roboter zu stoppen. Ist dadurch eine Erhöhung der Genauigkeit möglich?

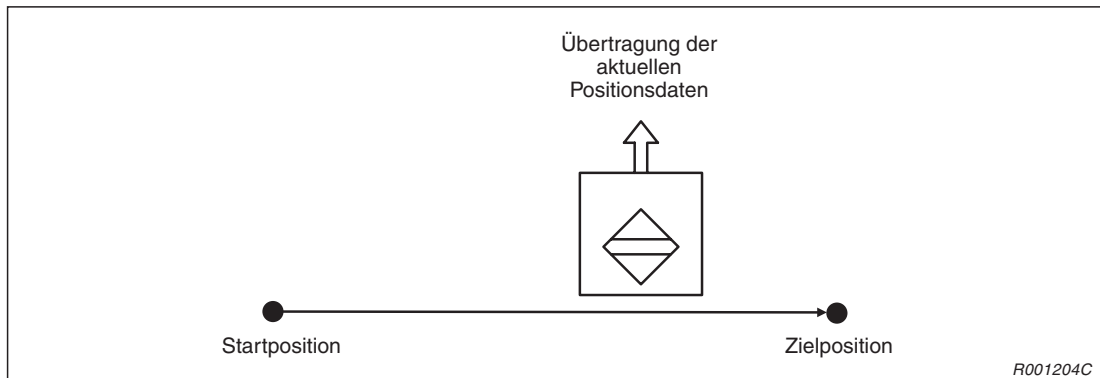


Abb. 11-17: Übertragung von Positionsdaten während der Verfahrbewegung

Methode

Ist ein Stoppen des Roboters während der Übertragung der Positionsdaten zulässig, kann ein Interrupt verwendet werden. Ist eine Unterbrechung der Verfahrbewegung unerwünscht, empfiehlt sich die Anwendung des Multitaskings. Nachfolgend wird die Methode des Multitaskings erläutert.

Dabei sind zwei grundlegende Überlegungen für das Einlesen der Koordinaten beim Ein- und Ausschalten des Sensors anzustellen:

- Das Einlesen der Positionsdaten soll möglichst zeitgleich mit dem Einschalten des Sensors erfolgen. Ein Fortschreiten der Verfahrbewegung ist auch nach der Erfassung des Sensor-Einschaltzeitpunkts durch das Programm ohne Bedeutung bis die Koordinaten eingelesen werden. Daher sollten die Anweisungen zur Erfassung des Sensor-Einschaltzeitpunkts und zum Einlesen der Koordinaten möglichst in einer Zeile programmiert sein. Aus diesem Grunde ist ein Einlesen der Koordinaten innerhalb einer Schleife sinnvoller als die Verwendung eines Interrupts.
- Für eine hohe Genauigkeit der Positionsdaten sollte die Abfrage des Sensorzustandes bei hoher Geschwindigkeit möglichst häufig erfolgen. Dazu ist die Priorität des Sensorprogrammes über die Anweisung PRIORITY so hoch wie möglich zu wählen.

Programmbeispiel

In diesem Beispiel wird das Hauptprogramm im Programmplatz 1 gestartet, das Sensorprogramm mit der Startbedingung „ALWAYS“ im Programmplatz 2. Die Übertragung einer Anweisung vom Haupt- zum Sensorprogramm erfolgt über die externe Variable M_01.

Ein Roboter, dessen Hand über zwei Sensoren verfügt (Eingangssignale 901 und 902), wird während einer linearen Bewegung mit einer Geschwindigkeit von 50 mm/s zwischen den Punkten PT01 und PT02 überwacht. Sobald einer der Sensoren einschaltet, erfolgt die Übertragung der Positionsdaten. Zur Übertragung der Positionsdaten an das Hauptprogramm dienen die beiden externen Variablen P_01 und P_02. Spricht beim Durchfahren der Strecke ein Sensor nicht an, erfolgt die Ausgabe einer Fehlermeldung.

Programm	Erläuterung
<pre> <Hauptprogramm> 1000 M_01 = 0 1010 MOV PT01 'Abtastposition anfahren 1020 SPD 50 'Geschwindigkeit auf Abtastgeschwindigkeit reduzieren 1030 M_01 = 1 'Anforderung zum Start der Überwachung 1040 PRIORITY 1,1 'Hohe Priorität für das Sensorprogramm 1050 PRIORITY 10,2 1060 MVS PT02 'Zielposition anfahren 1070 PRIORITY 1,1 'Anpassung der Priorität 1080 PRIORITY 1,2 1090 IF M_01 = 1 THEN 'Überwachung beenden 1100 M_01 = 0 'falls Überwachung nicht beendet 1110 ERROR 9100 'Fehlerausgabe 1120 ELSE 1130 'Ausführung mittels P_01 und P_02 1140 PTMP = (P_01 + P_02) / 2 1150 MVS PTMP 'Position zwischen den Sensor-Schaltpositionen anfahren 1160 HLT 1170 ENDIF </pre>	
<pre> <Sensorprogramm> 1000 WAIT M_01 = 1 'Warte auf Überwachungsanforderung 1010 MS1 = 0 1020 MS2 = 0 1030 *LOOP 1040 IF MS1 = 0 AND M_IN(901) THEN P_01 = P_CURR(1) 'Aktuelle Position lesen 1050 IF MS1 = 0 AND M_IN(901) THEN MS1 = 1 'Übertragung Sensor 1 abgeschlossen 1060 IF MS2 = 0 AND M_IN(902) THEN P_02 = P_CURR(1) 'Aktuelle Position lesen 1070 IF MS2 = 0 AND M_IN(902) THEN MS2 = 1 'Übertragung Sensor 2 abgeschlossen 1080 IF (MS1 = 0 OR MS2 = 0) AND M_01 = 1 THEN *LOOP 1090 M_01 = 0 1100 END </pre>	<p>Die Anweisung zur Erfassung des Sensor-Schaltzeitpunktes und die Anweisung zur Übertragung der Koordinaten werden in einer Zeile ausgeführt.</p>

Tab. 11-19: Programmbeispiel zur sensorgesteuerten Übertragung von Positionsdaten

11.3 Programmiertechniken für Fortgeschrittene

11.3.1 Einsatz eines Roboters als einfache SPS

Kann die Multitask-Funktion eines Roboters zur Signalverarbeitung ohne Einsatz einer SPS verwendet werden?

Methode

Natürlich kann die Multitask-Funktion eines Roboters keine SPS ersetzen. Für eine einfache Signalverarbeitung, z. B. zur Steuerung eines Transportbandes oder einer Lampe, ist sie jedoch durchaus ausreichend. Die Anzahl der E/A-Signale kann dabei durch den Einsatz einer zusätzlichen, parallelen Schnittstelle vergrößert werden.

Programmbeispiel

Folgendes Programm mit der Startbedingung „ALWAYS“ dient zur Verarbeitung von Signalen. Das Programm startet somit beim Einschalten der Versorgungsspannung des Roboters und wird auch beim Auftreten eines Fehlers nicht unterbrochen. Programmieren Sie keine Unterbrechung der Programmausführung durch die Anweisungen ERROR, HLT oder WAIT.

Das Beispiel beschreibt ein Palettierungssystem, in dem ein Band zur Werkstückzuführung und ein Band zum Abtransport der Paletten vom Roboter gesteuert werden. Nach dem Entwurf eines Kontaktplans zur Signalverarbeitung erfolgt die Programmierung in MELFA-BASIC IV.

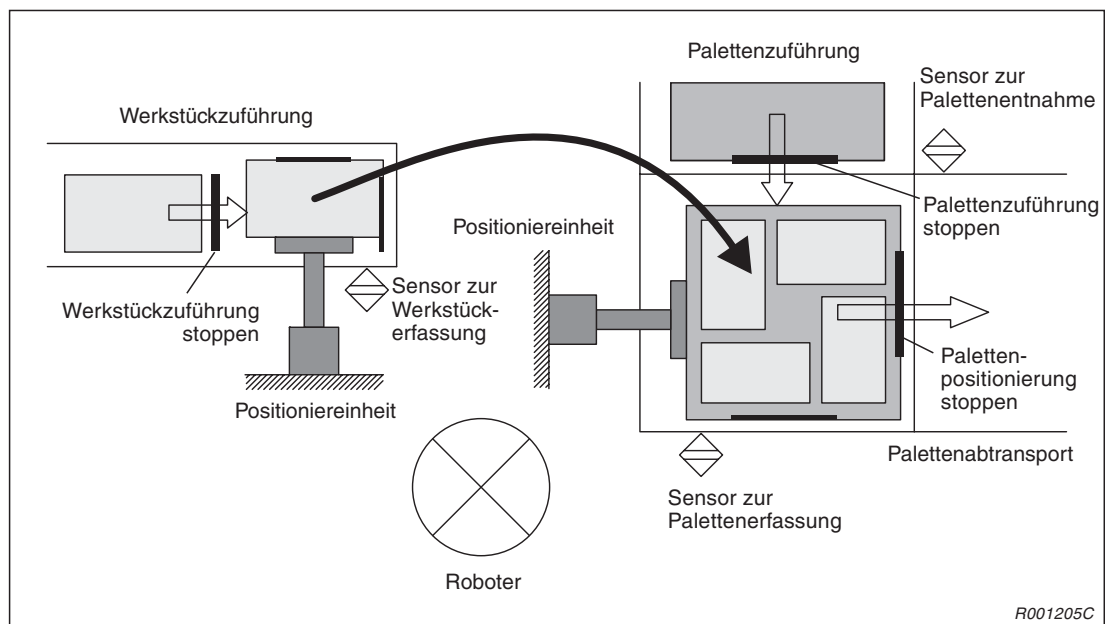


Abb. 11-18: Palettierungssystem

Eingangssignale	Ausgangssignale
M_IN(8): Sensor zur Werkstück erfassung	M_OUT(8): Band zur Werkstückzuführung EIN/AUS
M_IN(9): Sensor zur Palettenerfassung	M_OUT(9): Werkstückzuführung stoppen EIN/AUS
M_IN(10): Sensor zur Palettenentnahme	M_OUT(10): Werkstückpositioniereinheit EIN/AUS
M_IN(11): Anforderung zur Palettenentnahme	M_OUT(11): Band zur Palettenzuführung EIN/AUS
	M_OUT(12): Band zum Palettenabtransport EIN/AUS
	M_OUT(13): Palettenzuführung stoppen EIN/AUS
	M_OUT(14): Palettenpositionierung stoppen EIN/AUS
	M_OUT(15): Palettenpositioniereinheit EIN/AUS

Tab. 11-20: Übersicht der E/A-Signale

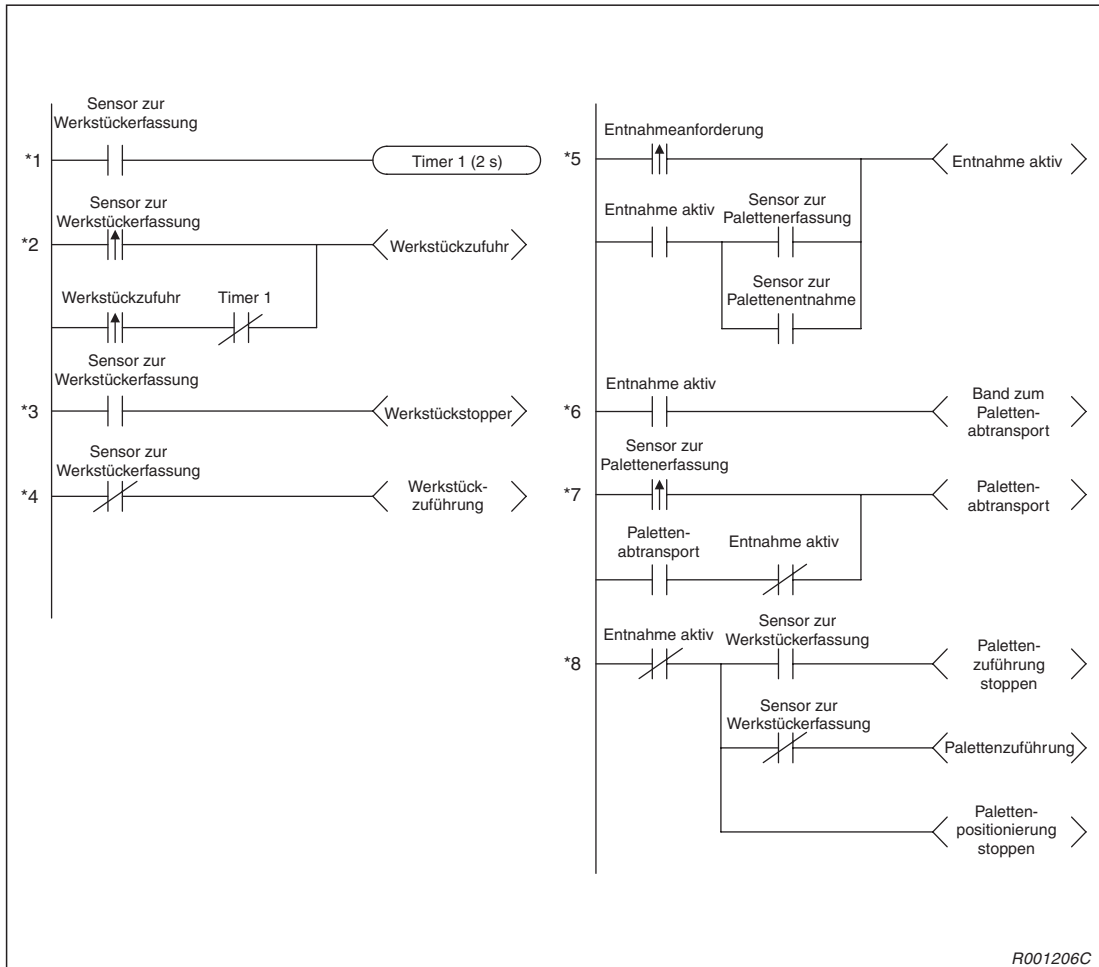


Abb. 11-19: Kontaktplan

Programm	Erläuterung
<pre> 1000 MTMLOOP = 100 * 1000 'Anzahl der Sekunden pro Schleife (100 s) 1010 M_TIMER(0) = 0 'Timer zurücksetzen 1020 MTM01 = 0 'Timer-Variablen zurücksetzen 1030 MIN08 = 0 'Variable für den Status des Sensors zur Werkstückerkennung 1040 MIN09 = 0 'Variable für den Status des Sensors zur Palettenerfassung 1050 MIN11 = 0 'Variable für das Anforderungssignal zur Palettenernahme 1060 MPLTOUT = 0 'Variable zur Anzeige der Palettenernahme 1070 ' 1080 *LOOP 1090 'N-Sekunden-Timer erzeugen 1100 IF M_TIMER(1) > MTMLOOP THEN 'Timer nach Überschreitung der vorgegebenen 1110 M_TIMER(1) = 0 'Sekundenzahl zurücksetzen 1120 IF MTM01 + MTMLOOP > 0 THEN MTM01 = MTMLOOP 'Timer-Variablen 1130 ENDIF 1140 ' 1150 ##### Band zur Werkstückzuführung ##### 1160 MUPPLS08 = (M_IN(8) AND MIN08 = 0) 'Ansteigende Flanke für den Erfassungssensor erzeugen 1170 MIN08 = M_IN(8) 1180 '=== Kontaktplan *1 === 1190 IF MUPPLS08 THEN MTM01 = M_TIMER(1) 'Timer bei ansteigender Flanke zurücksetzen 1200 MTMOUT01 = M_IN(8) AND (M_TIMER(1) - MTM01 > 2000) 'Nach 2 s auf EIN setzen 1210 '=== Kontaktplan *2 === 1220 IF MUPPLS08 OR (M_OUT(10) AND MTMOUT01 = 0) THEN M_OUT(10) = 1 ELSE M_OUT(10) = 0 1230 '=== Kontaktplan *3, *4 === 1240 IF M_IN(8) THEN 'Wenn Sensor zur Werkstückerkennung EIN 1250 M_OUT(9) = 1 'Stopper EIN 1260 M_OUT(8) = 0 'Werkstückzuführung AUS 1270 ELSE 'Wenn Sensor zur Palettenerfassung EIN 1280 M_OUT(9) = 0 'Stopper AUS 1290 M_OUT(8) = 1 'Werkstückzuführung EIN 1300 ENDIF 1310 ' 1320 '=== Palettenband === 1330 MUPPLS09 = (M_IN(9) AND MIN09 = 0) 'Ansteigende Flanke für den Erfassungssensor erzeugen 1340 MIN09 = M_IN(9) 1350 MUPPLS11 = (M_IN(11) AND MIN11 = 0) 'Ansteigende Flanke bei Entnahmeanforderung erzeugen 1360 MIN11 = M_IN(11) 1370 '=== Kontaktplan *5 === 1380 IF MUPPLS11 OR (MPLTOUT AND (M_IN(9) OR M_IN(10))) THEN MPLTOUT = 1 ELSE MPLTOUT = 0 1390 '=== Kontaktplan *6 === 1400 IF MPLTOUT THEN M_OUT(12) = 1 ELSE M_OUT(12) = 0 1410 '=== Kontaktplan *7 === 1420 IF MUPPLS09 OR (M_OUT(15) AND MPLTOUT = 0) THEN M_OUT(15) = 1 ELSE M_OUT(15) = 0 1430 '=== Kontaktplan *8 === 1440 IF MPLTOUT = 0 THEN 'Wenn nicht bei der Entnahme 1450 IF M_IN(9) THEN 'Wenn Sensor zur Palettenerfassung EIN 1460 M_OUT(13) = 1 'Eingangstopper EIN 1470 M_OUT(11) = 0 'Band zur Palettenerfassung AUS 1480 ELSE 1490 M_OUT(13) = 0 'Eingangstopper AUS 1500 M_OUT(11) = 1 'Band zur Palettenerfassung EIN 1510 ENDIF 1520 M_OUT(14) = 1 'Positionierstopper EIN 1530 ELSE 1540 M_OUT(14) = 0 'Positionierstopper AUS 1550 ENDIF 1560 GOTO *LOOP </pre>	<p>Einheit: ms Achten Sie darauf, dass ein Timer durch die Verwendung in anderen Programmen nicht mehrfach genutzt wird.</p> <p>Erzeugen Sie eine 100-s-Schleife zur Überwachung eines Timer-Überlaufs. Setzen Sie den Timer im Falle eines Überlaufs zurück und ziehen Sie die Referenzzeit von der Timer-Variablen ab. Eine ansteigende Flanke wird erfasst. Da eine Zeile zur Programmierung des Timers nicht ausreicht, werden zwei Zeilen verwendet.</p> <p>Stellen Sie sicher, dass die Zustände EIN oder AUS definiert sind.</p> <p>Im Fall eines Staus.</p>

Tab. 11-21: Programmbeispiel zur Signalverarbeitung

11.3.2 Implementierung einer Abbildungsfunktion

Im Allgemeinen fährt der Roboter eine festgelegte Zielposition an und nimmt dort Werkstücke auf. Es kann jedoch vorkommen, dass ein Werkstück fehlt, da es z. B. zur Bearbeitung entnommen wurde oder defekt ist. Der Roboter fährt die leere Position an, erfasst dort, dass kein Werkstück vorhanden ist und bewegt sich zu der nächsten Position. Dies trägt zu einer unnötigen Verlängerung der Zykluszeit bei. Lässt sich das Vorhandensein von Werkstücken im Voraus erfassen?

Methode

Eine Methode, das Vorhandensein von Werkstücken im Voraus zu erfassen, besteht in der Installation eines optischen Sensors an der Handspitze des Roboters. So kann beim Überfahren der Werkstücke eine Abbildung der Werkstückanordnung erstellt werden (Mapping). Aufgrund der begrenzten Verarbeitungsgeschwindigkeit des Steuergerätes kann jedoch bei dieser Methode die Aufnahme der Werkstücke nicht mit der maximalen Geschwindigkeit erfolgen.

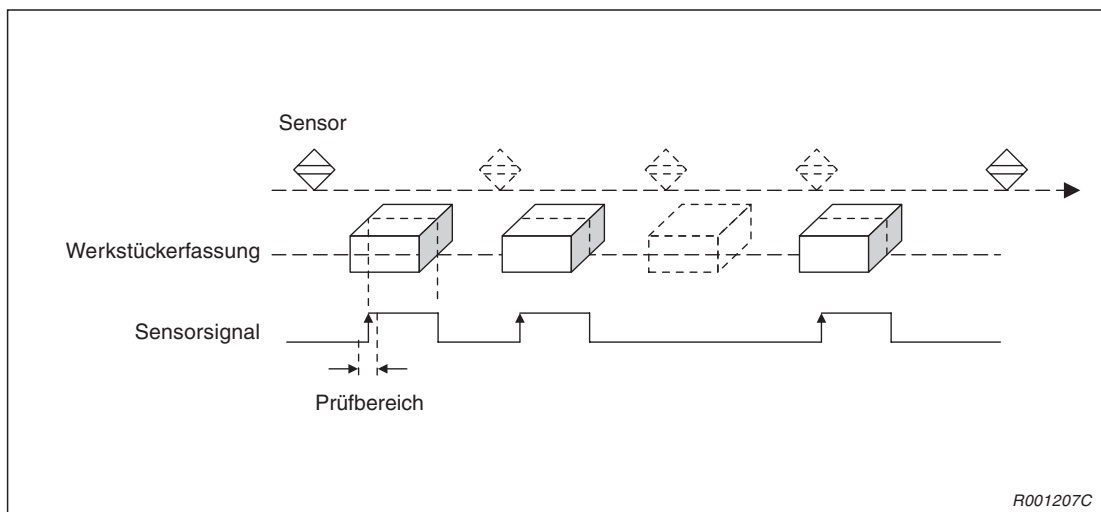


Abb. 11-20: Abbildungsfunktion oder Mapping

Programmbeispiel

Für diese Anwendung empfiehlt sich die Erstellung eines Multitask-Programms. Das Hauptprogramm dient dabei der Steuerung des Roboters, während das Einlesen der Sensordaten in einem Unterprogramm erfolgt. Bei der Erstellung des Programms sollte auf eine hohe Verarbeitungsgeschwindigkeit durch ein möglichst kurzes Programm zum Einlesen der Sensordaten geachtet werden. Dabei sollte die Vergabe der Prioritäten dynamisch erfolgen und die Priorität des Unterprogramms bei normaler Verarbeitung und beim Abtastvorgang umgeschaltet werden.

Der Roboter tastet Werkstücke, die in einer Reihe angeordnet sind, mittels eines kontaktlosen Sensors ab (externes Eingangssignal 901). Dabei wird erfasst, ob an der entsprechenden Position (Referenzposition) ein Werkstück vorhanden ist. Ist ein Werkstück vorhanden, erfolgt die Aufnahme des Werkstücks. Der Block zur Aufnahme des Werkstücks ist aus Gründen der Übersichtlichkeit hier nicht aufgeführt.

Die Erfassung der Werkstücke erfolgt über ein Kalibrierungsprogramm, das beim Systemstart aufgerufen wird, einen Abtastvorgang bei Vorhandensein aller Werkstücken durchführt und daraus eine Referenzposition ableitet. Vor der Aufnahme eines Werkstücks erfolgt ein Vergleich der Referenzposition mit den Abtastdaten. Eine Differenz zwischen Referenzposition und Abtastdaten von kleiner gleich ± 10 mm, wird als „Werkstück vorhanden“ interpretiert.

Gehen Sie bei der Ausführung des Programms wie folgt vor:

- ① Starten Sie das Abtastprogramm mit der Startbedingung „ALWAYS“ in Programmplatz 2.
- ② Starten Sie das Kalibrierungsprogramm zur Festlegung der Referenzposition, wenn alle Werkstücke vorhanden sind.
- ③ Starten Sie das Hauptprogramm.

Variablen	E/A-Signale
M_01: Sensoranforderung	M_IN(8): Signal zur Werkstückanforderung
M_100(): Speicherung der Werkstückerkennung	M_OUT(8): Bearbeitung abgeschlossen
P_100(): Speicherung der Abtastung	M_IN(901): Sensoreingang
P_101(): Speicherung der Referenzposition	

Tab. 11-22: Übersicht der Variablen und Signale

Programm	Erläuterung
<pre> <Hauptprogramm> 1000 WAIT M_IN(8) = M_ON 'Warte, bis Signal zur Werkstückanforderung eingeschaltet 1010 MOV PT1 'Startposition der Abtastung anfahren 1020 M_01 = M_ON 'Abtastung starten 1030 SPD 30 'Auf Abtastgeschwindigkeit reduzieren 1040 MVS PT2 'Endposition der Abtastung anfahren 1050 DLY 0.1 'Warte, bis Stoppvorgang abgeschlossen 1060 M_01 = M_OFF 'Abtastung beenden </pre>	<p>Verwenden Sie die gleichen Bezeichnungen wie im Kalibrierungsprogramm.</p>
<pre> 1070 SPD M_NSPD 'Geschwindigkeit zurücksetzen 1080 MVS PT1 'Zur Startposition der Abtastung zurückkehren 1090 ' 1100 M01RANGE = 10.0 'Werkstückprüfbereich gleich 10 mm 1110 FOR M01 = 1 TO 10 'Vergleich mit 10 Referenzpositionen 1120 M_100(M01) = 0 'Erfassungsmerker zurücksetzen 1130 FOR M00 = 1 TO 10 'Vergleich mit 10 Abtastdaten 1140 M01DIST = DIST(P_101(M01),P_100(M00)) 'Werkstück vorhanden, falls Differenz von 1150 IF M01DIST < M01RANGE THEN 'Abtast- und Referenzwert im def. Bereich 1160 M_100(M01) = 1 'Festlegung, dass Werkstück vorhanden 1170 M00 = 10 'FOR-Schleife verlassen 1180 ENDIF 1190 NEXT M00 1200 NEXT M01 1210 ' 1220 FOR M02 = 1 TO 10 'Es sollten 10 Werkstücke vorhanden sein 1230 IF M_100(M02) = 0 THEN *L01SKIP 'Überspringen, falls kein Werkstück vorhanden 1240 GOSUB *S50WKGET 'Werkstückaufnahme (nicht programmiert) 1250 *L01SKIP 1260 NEXT 1270 M_OUT(8) = M_ON 'Abschlussignal EIN 1280 WAIT M_IN(8) = M_OFF 'Warte, bis das Signal zur Werkstückanforderung ausgeschaltet 1290 M_OUT(8) = M_OOFF 'Abschlussignal ausschalten 1300 END </pre>	
<pre> <Abtastprogramm (in Programmplatz 2 mit Startbedingung „ALWAYS“ starten)> 1000 *S01SENS 1010 PRIORITY 10,1 'Standardstatus 1020 PRIORITY 1,2 1030 M_01 = M_OFF 'Abtastanforderung ausschalten 1040 M01POS = 1 'Zähler löschen 1050 WAIT M_01 = M_ON 'Warte, bis Abtastanforderung eingeschaltet 1060 PRIORITY 1,1 'Abtastprogramm erhält höhere Priorität 1070 PRIORITY 10,2 1080 *L01LOOP 1090 IF M_01 = M_OFF THEN *S01SENS 'Ende, falls Abtastanforderung ausgeschaltet 1100 IF M_IN(901) = M_ON THEN 'Speichert die aktuelle Position, falls 1110 P_100(M01POS) = P_CURR 'Sensor einschaltet 1120 M01POS = M01POS + 1 'Zähler + 1 1130 WAIT M_IN(901) = M_OFF 'Warte, bis Sensor ausschaltet 1140 ENDIF 1150 GOTO *L01LOOP </pre>	<p>Die aktuelle Position wird in einer systemexternen Variablen gespeichert. Ist die Anzahl größer als 10, verwenden Sie benutzerdefinierte externe Variablen.</p>
<pre> <Kalibrierungsprogramm (in Programmplatz 1 starten)> 1000 MOV PT1 'Startposition der Abtastung anfahren 1010 M_01 = M_ON 'Abtastung starten 1020 SPD 30 'Auf Abtastgeschwindigkeit reduzieren 1030 MVS PT2 'Endposition der Abtastung anfahren 1040 DLY 0.1 'Warte, bis Stoppvorgang abgeschlossen 1050 M_01 = M_OFF 'Abtastung beenden 1060 SPD M_NSPD 'Geschwindigkeit zurücksetzen 1070 MVS PT1 'Zur Startposition der Abtastung zurückkehren 1080 ' 1090 FOR M01 = 1 TO 10 'Ergebnis der Abtastung 1100 P_101(M01) = P_100(M01) 'in die Kalibrierungsdaten kopieren 1110 NEXT M01 1120 HLT </pre>	<p>In einigen Fällen empfiehlt sich eine mehrfache Abtastung mit anschließender Mittelwertbildung.</p>

Tab. 11-23: Beispielprogramm zur Anwendung der Abbildungsfunktion

11.3.3 Ausgabe ausgeführter Zeilen

Wie kann die Zeilennummer eines Programms, in der beim Debuggen ein Fehler aufgetreten ist, ausgegeben werden?

Methode

Das Steuergerät arbeitet nach dem sogenannten Round-Robin-Verfahren, bei dem jeweils eine Zeile der Programme ausgeführt wird, solange keine andere Vorgabe über die Priorität erfolgt ist. Bei dieser Methode ist ein Zugriff auf alle aktuell ausgeführten Zeilen über einen Multitask möglich.

Folgende beiden Zeilen des Beispielprogramms definieren die Ausführung von 6 Zeilen dieses Programmes in Programmplatz 2, während eine Zeile des Hauptprogramms in Programmplatz 1 ausgeführt wird. Die 6 Zeilen entsprechen den Zeilennummern 1070 bis 1120.

```
1050 PRIORITY 1,1           'Führe 6 Zeilen dieses Programmes aus, während
1060 PRIORITY 6,2         '1 Zeile des Hauptprogrammes ausgeführt wird
```

Programmbeispiel

Starten Sie dieses Programm mit der Startbedingung „ALWAYS“ in Programmplatz 2 und das Hauptprogramm in Programmplatz 1. Soll das Programm in einem anderen Programmplatz als in Programmplatz 2 gestartet werden, ist auch die Einstellung in der Anweisung PRIORITY in Zeile 1060 anzupassen. Die Speicherung der im Programmplatz 1 ausgeführten Zeile erfolgt in der Variablen MLN(100). Bei Überschreitung des Speicherbereiches werden die Werte, beginnend mit dem ersten Wert, überschrieben.

Programm	Erläuterung
1000 DIM MLN(100)	'Speicherbereich für 100 Zeilen
1010 FOR M01 = 1 TO 100	'Speicherbereich löschen
1020 MLN(M01) = -1	
1030 NEXT	
1040 MIDX = 1	
1050 PRIORITY 1,1	'Führe 6 Zeilen dieses Programmes aus, während
1060 PRIORITY 6,2	'1 Zeile des Hauptprogrammes ausgeführt wird
1070 *LOOP	
1080 MBF = MIDX	'Vorhergehende ID speichern
1090 MIDX = (MIDX MOD 100) + 1	'ID des Ringpuffers
1100 MLN(MIDX) = M_LINE(1)	'Aktuelle Zeilennummer speichern
1110 IF MLN(MBF) = MLN(MIDX) THEN MIDX = MBF	'Rückkehr falls keine Zeile geändert
1120 GOTO *LOOP	

Tab. 11-24: Beispielprogramm zum Auffinden ausgeführter Zeilen

11.3.4 Status bei Auftreten eines Fehlers speichern

Wie kann der Status gespeichert werden, wenn ein Fehler auftritt?

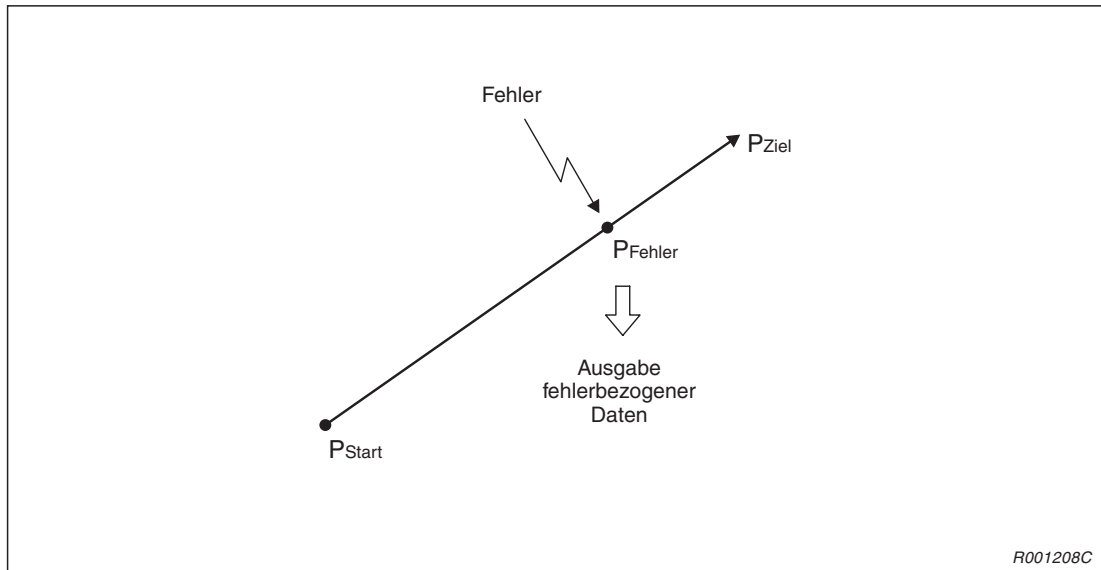


Abb. 11-21: Ausgabe fehlerbezogener Daten

Methode

Die Programmplatzparameter ermöglichen die Definition der Startbedingungen START, ALWAYS und ERROR eines Roboterprogramms. Für die hier geforderte Anwendung ist somit die Erstellung eines Programmes mit der Startbedingung ERROR sinnvoll, dass bei Auftreten eines Fehlers ausgeführt wird und den aktuellen Status speichert.

Programmbeispiel

Stellen Sie die Programmplatzparameter für diesen Programmplatz wie folgt ein:
 Startbedingung = ERROR (Ausführung bei Auftreten eines Fehlers)
 Ausführungsformat = CYC (Beendigung nach Ausführung eines Zyklus)
 Eine detaillierte Beschreibung der Programmplatzparameter finden Sie in Abschn. 9.5.

Das hier gezeigte Programmbeispiel speichert die Informationen der letzten fünf Fehler in der Variablen PERINF(,). Folgende Daten werden gespeichert:

- Fehlernummer
- Nummer der Zeile, in der der Fehler aufgetreten ist
- Verbleibender Verfahrensweg zur Zielposition
- Kartesische Koordinaten der Position beim Auftreten des Fehlers

Die Informationen können in Abhängigkeit des Fehlers geändert werden (siehe Programmbeispiel ab Zeile 1110).

Stellen Sie beim ersten Programmaufruf sicher, dass die Variable M_01 auf „0“ gesetzt ist. Dies bewirkt ein Zurücksetzen des Zählers für den Speicherplatz. Bei jedem Auftreten eines Fehlers erfolgt nun der Aufruf des Programms. Die Informationen werden in die Variable PERINF(,) geschrieben. Die letzte Information im Ringpuffer ist durch M_01 gekennzeichnet.

Programm	Erläuterung
<pre> <Programm zur Erfassung von Fehlerinformationen> 1000 DIM PERINF(5,5) 'Speicherbereich für die letzten 5 Fehler 1010 IF M_01 = 0 THEN 'Variable M_01 zurücksetzen 1020 M_01 = 1 'Anzahl der aufgetretenen Fehler 1030 M_02 = 1 'ID des Ringpuffers 1040 ENDIF 1050 ' Vom Fehlertyp unabhängig 1060 PERINF(M_02,1).X = M_01 'Anzahl der aufgetretenen Fehler 1070 PERINF(M_02,1).Y = M_ERRNO 'Nummer des aufgetretenen Fehlers 1080 PERINF(M_02,1).Z = M_LINE(1) 'Zeilennummer, in der der Fehler aufgetreten ist 1090 PERINF(M_02,2).X = M_RDST 'Restverfahrensweg zur Zielposition 1100 PERINF(M_02,3) = P_CURR 'Aktuelle Position 1110 ' Änderung der zu speichernden Fehlerinformation in Abhängigkeit des Fehlertyps 1120 SELECT M_ERRNO 1130 CASE 2802 1140 ' Informationen, die beim Auftreten des Fehlers 2802 in PERINF gespeichert werden sollen 1150 ' PERINF(M_02,4).X = M_??? 1160 ' PERINF(M_02,5) = P_??? 1170 BREAK 1180 CASE 9100 1190 ' Informationen, die beim Auftreten des Fehlers 9100 in PERINF gespeichert werden sollen 1200 BREAK 1210 DEFAULT 1220 ' Informationen, die beim Auftreten eines unvorhersehbaren Fehlers in PERINF gespeichert ' werden sollen 1230 BREAK 1240 END SELECT 1250 ' 1260 M_01 = M_01 + 1 1270 M_02 = (M_02 MOD 5) + 1 'ID des Ringpuffers generieren 1280 END </pre>	<p>Es ist möglich, die Informationen, die gespeichert werden sollen, in Abhängigkeit des Fehlers zu wählen.</p> <p>Erhöhung des Zählers für den Pufferindex.</p>

Tab. 11-25: Programmbeispiel zur Speicherung des Status im Fehlerfall

A Anhang

A.1 Fehlerdiagnose

Bei Auftreten eines Fehlers wird am Steuergerät eine 5-stellige Fehlernummer auf dem Display „STATUS.NUMBER“ angezeigt (z. B. C0010). Die LED auf dem RESET-Taster leuchtet.

Wird eine Taste der Teaching Box (z. B. MENU-Taste) betätigt, erscheint eine 4-stellige Fehlernummer auf dem Display der Teaching Box. Das erste Zeichen der Fehlernummer wird nicht angezeigt. Es erscheint z. B. „0010“ für „C0010“ und Klartext.

Im Menü „Fehlermeldungen anzeigen“ der Teaching Box kann eine Liste der bisher aufgetretenen Fehler aufgerufen werden. Dazu muss zuerst der Fehler zurückgesetzt werden.

In folgender Tabelle sind die Fehlernummern, die Fehlerursachen und die Gegenmaßnahmen aufgeführt. Lässt sich ein Fehler durch die aufgeführten Gegenmaßnahmen nicht beseitigen, setzen Sie sich mit Ihrem Vertriebspartner in Verbindung.

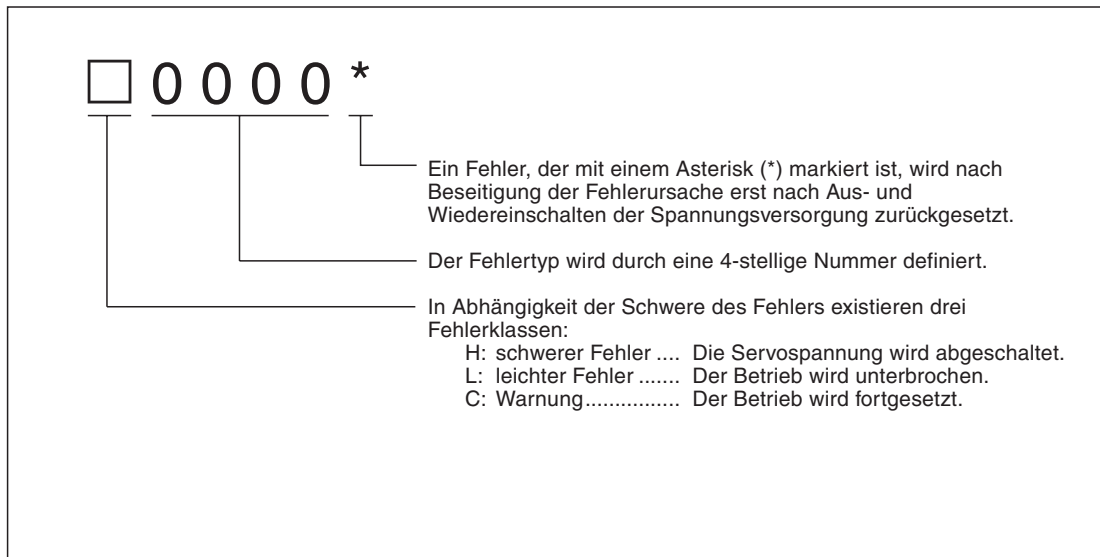


Abb. A-1: Aufbau einer Fehlermeldung

HINWEIS

Die letzte Stelle der Fehlernummer kann eine Achsennummer anzeigen.
Bsp.: Die Fehlernummer H0931 bedeutet Überstrom des Motors der Achse Nr. 1.

A.1.1 Übersicht der Fehlercodes

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
H0001	Fehler beim Speichern einer Datei (SRV OFF)	Systemfehler	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H0002	Fehler beim Speichern einer Datei (STOP)		
H0009*	Update (ALL)	Das System benötigt ein Update	Versorgungsspannung aus- und wieder einschalten
C0010	Ungültige Dateiversion	Versionsunverträglichkeit	Datei wird automatisch initialisiert. Programm wird gelöscht.
C0011	Ungültige Version der Systemdaten	Versionsunverträglichkeit	Datei wird automatisch initialisiert. Versorgungsspannung aus- und wieder einschalten
C0012	Initialisierung des Fehler-Logfiles	Versionsunverträglichkeit oder fehlerhaftes Fehler-Logfile	Fehler zurücksetzen und Betrieb fortsetzen
C0013*	Ungültige Datei	Die Daten bzw. Programme sind beschädigt.	MITSUBISHI-Vertriebspartner kontaktieren, um Initialisierung durchzuführen
H0014*	Systemfehler (ungültiger Mechanismus)	Die Zeichenkette darf nicht aus mehr als 16 Zeichen bestehen.	Namen korrigieren
H0015*	Ungültige Dateiversion	Ungültige Dateiversion	MITSUBISHI-Vertriebspartner kontaktieren
L0016*	Versorgungsspannung aus- und wieder einschalten	Die Zeit zwischen dem Aus- und Wiedereinschalten der Versorgungsspannung ist zu kurz	Die Zeit zwischen dem Aus- und Wiedereinschalten der Versorgungsspannung verlängern
C0018	Ungültige Version (Speicher vergrößern)	Versionsabweichung	Datei wird automatisch initialisiert.
H0020*	Der Backup-Datenname ist bereits vorhanden.	Der Name ist bereits vergeben.	Namen ändern
H0021*	Kein Anlegen weiterer Backup-Daten möglich	Überlauf des Steuerbereiches	Steuerbereich über Software vergrößern
H0022*	Überlauf des Backup-Bereichs	Zu kleiner Bereich	Backup-Bereich über Software vergrößern
C0023	Zusatzspeicher wurde hinzugefügt oder entfernt	Zusatzspeicher wurde hinzugefügt oder entfernt	Speicher prüfen
H0025*	Dateien werden in das ROM geschrieben.	Programme, Parameter und Logfiles werden im ROM gespeichert.	Versorgungsspannung aus- und wieder einschalten
H0026*	Dateien werden aus dem ROM gelesen.	Programme, Parameter und Logfiles aus dem ROM geladen.	Versorgungsspannung aus- und wieder einschalten
H0027*	Backup oder Wiederherstellung abgebrochen	Backup oder Wiederherstellung abgebrochen	Versorgungsspannung aus- und wieder einschalten
L0030	Handfehler Dreistufenschalter betätigen	Einstellfehler der Benutzerdaten	Nach Beseitigung der Fehlerursache Fehler zurücksetzen
L0031	Fehler in der Pneumatikversorgung der Hand	Einstellfehler der Benutzerdaten	Nach Beseitigung der Fehlerursache Fehler zurücksetzen

Tab. A-1: Fehlercodes (1)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
H0039	Die Signalleitung des Tür-Schließkontaktes ist defekt	Die Signalleitung des Tür-Schließkontaktes ist instabil	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H0040	Das Signal vom Tür-Schließkontakt wird geschaltet	Der Tür-Schließkontakt ist geöffnet	Türe schließen
H0041*	Kommunikationsfehler im externen E/A-Kanal 1	Fehler auf der Kommunikationsleitung des externen E/A-Kanals 1	Prüfen des Übertragungskabels und der Spannungsversorgung des externen Geräts
H0042*	Kommunikationsfehler im externen E/A-Kanal 2	Fehler auf der Kommunikationsleitung des externen E/A-Kanals 2	
H0043*	Kommunikationsfehler im externen E/A-Kanal 3	Fehler auf der Kommunikationsleitung des externen E/A-Kanals 3	
H0050	Eingabe eines externen NOT-HALT-Signals	Das externe Stoppsignal wurde eingegeben.	Externen NOT-HALT zurücksetzen
H0051	Fehler im Anschluss des externen NOT-HALT-Kreises		Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H0052	Eingabe des externen NOT-HALT-Signals für die Zusatzachse 1	Das externe Stoppsignal wurde in die Schnittstellenkarte der Zusatzachse eingegeben.	Externen NOT-HALT der Zusatzachse 1 zurücksetzen
H0053	Eingabe des externen NOT-HALT-Signals für die Zusatzachse 2	Das externe Stoppsignal wurde in den Zusatzverstärker eingegeben.	Externen NOT-HALT der Zusatzachse 2 zurücksetzen
H0060	Eingabe des NOT-HALT-Signals über das Steuergerät	Das Stoppsignal wurde über das Steuergerät eingegeben.	NOT-HALT des Steuergeräts zurücksetzen
H0061	Fehler im Anschluss des NOT-HALT-Kreises vom Steuergerät	Die Signalleitung des NOT-HALT-Kreises ist instabil	Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H0070	Eingabe des NOT-HALT-Signals über die Teaching Box	Das Stoppsignal wurde über die Teaching Box eingegeben.	NOT-HALT der Teaching Box zurücksetzen
H0071	Fehler im Anschluss des NOT-HALT-Kreises der Teaching Box	Die Signalleitung des NOT-HALT-Kreises ist instabil	Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H0072	Fehler beim Abschalten der Teaching Box	Fehlerhaftes Abschalten der Teaching Box	Betätigen Sie die Schalter [MODE], [REMOVE T/B] und [T/B ENABLE/DISABLE]
H0073	Fehler in der Verbindungsleitung des Schalters zum Abschalten der Teaching Box	Schalter arbeitet nicht korrekt.	Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H0074	Fehler im System des [ENABLE/DIASABLE]-Schalters der Teaching Box	[ENABLE/DIASABLE]-Schalter arbeitet nicht korrekt.	
H0075	Übertragungsfehler der Teaching Box	Die Kommunikation zwischen Teaching Box und Steuergerät wurde unterbrochen.	

Tab. A-1: Fehlercodes (2)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
H0082*	Sicherung für die pneumatische Greifhand defekt	Sicherung für die pneumatische Greifhand defekt	Sicherung wechseln (siehe auch Technisches Handbuch)
H0083*	Sicherung der Spannungsversorgung der pneumatischen Greifhand defekt	Sicherung der Spannungsversorgung der pneumatischen Greifhand defekt	
H0084*	Sicherung des Steuergerätes defekt	Sicherung des Steuergerätes defekt	
H0085*	Sicherung der Spannungsversorgung des externen NOT-HALT-Kreises defekt	Sicherung der Spannungsversorgung des externen NOT-HALT-Kreises defekt	
H0086	Überstrom der Schnittstellenkarte für die Greifhand	Handmotor oder Platine der motorbetriebenen Greifhand defekt	Handmotor oder Platine der motorbetriebenen Greifhand austauschen
L0091	Signal ist bereits einem speziellen Ausgang zugeordnet.	Signal kann nicht 2-mal zugeordnet werden.	Ausgangsnummer oder zugewiesenen Parameter ändern
H0100*	Temperatur des Steuergerätes zu hoch	Ventilator arbeitet nicht oder Filter ist verunreinigt	Ventilator prüfen oder Filter reinigen bzw. austauschen
L0101	Temperatur des Steuergerätes zu hoch	Ventilator arbeitet nicht oder Filter ist verunreinigt	Ventilator prüfen oder Filter reinigen bzw. austauschen
C0102	Temperatur des Steuergerätes zu hoch	Ventilator arbeitet nicht oder Filter ist verunreinigt	Ventilator prüfen oder Filter reinigen bzw. austauschen
L0110	Batterie entladen	Batterie entladen	Batterie wechseln
C0111	Keine Batteriespannung	Batterie entladen	Batterie wechseln
C0112	Batteriefach geöffnet	Batteriefach geöffnet	Batteriefach schließen
C0460	Überhitzung des Motors der Zusatzachse	Zulässige Temperatur des Zusatzachsenmotors wurde überschritten.	Ventilator des Zusatzachsenmotors austauschen
H0470*	Fehler der Bremseinheit der Zusatzachse	Fehler der Bremseinheit der Zusatzachse	Bremseinheit und Verbindungskabel prüfen
L0480	Fehler des Bremswiderstandes der Zusatzachse	Überhitzung des Bremswiderstandes der Zusatzachse	Ventilator des Bremswiderstandes der Zusatzachse austauschen
C049n* (n: Achse, 1 ≤ n ≤ 8)	Ventilatorfehler im Roboter	Ventilator im Roboter defekt	Ventilator im Roboter austauschen
H050n* (n: Achse, 1 ≤ n ≤ 8)	Einstellfehler Codierschalter	Fehlerhafte Einstellung des Codierschalters zur Festlegung der beiden internen Zusatzachsenansteuerungen	Einstellung korrigieren
H0510*	Einstellung des externen NOT-Halt-Schalters	Widersprüchliche Einstellung des Codierschalters des Spannungswandlers und des Parameters SVPTYP	
H052n* (n: Achse, 1 ≤ n ≤ 8)	Fehlerhafte Achseneinstellung	Die Einstellung der Achse eines Roboters ist auch einem weiteren Roboter zugewiesen.	

Tab. A-1: Fehlercodes (3)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
H053n* (n: Achse, $1 \leq n \leq 8$)	Speicherfehler im Servoverstärker	Prüfsumme im Servoverstärker fehlerhaft	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H054n* (n: Achse, $1 \leq n \leq 8$)	Systemfehler im Servoverstärker	Zeitüberschreitung bei der Verarbeitung der Daten im Servoverstärker	
H055n* (n: Achse, $1 \leq n \leq 8$)	Systemfehler im Servoverstärker	Abweichung der magnetischen Pole des Encoders	
H056n* (n: Achse, $1 \leq n \leq 8$)	Fehler im A/D-Wandler des Servoverstärkers	Fehler im A/D-Wandler des Servoverstärkers bei der Initialisierung	
H057n* (n: Achse, $1 \leq n \leq 8$)	Encoderfehler (E ² PROM)	Encoderdaten im E ² PROM sind fehlerhaft	Versorgungsspannung aus- und wieder einschalten Position auf Abweichung überprüfen und bei Abweichung Einstellung der Grundposition wiederholen (siehe Technisches Handbuch) Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H058n* (n: Achse, $1 \leq n \leq 8$)	Encoderfehler (LED)	Die LED des Encoders ist defekt	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H059n* (n: Achse, $1 \leq n \leq 8$)	Encoderfehler (Positionsdaten)	Fehlerhafte Positionsdaten innerhalb einer Umdrehung des Encoders	Versorgungsspannung aus- und wieder einschalten Position auf Abweichung überprüfen und bei Abweichung Einstellung der Grundposition wiederholen (siehe Technisches Handbuch) Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H060n* (n: Achse, $1 \leq n \leq 8$)	Kein Encodersignal 1	Es liegt kein Eingangssignal vom Encoder des Motors an.	
H061n* (n: Achse, $1 \leq n \leq 8$)	Kein Encodersignal 2	Es liegt kein Eingangssignal vom Encoder der Maschine an.	
H062n* (n: Achse, $1 \leq n \leq 8$)	Fehler auf der Servoplatine	Es ist eine Fehler auf der Platine des Servoverstärkers aufgetreten	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H063n* (n: Achse, $1 \leq n \leq 8$)	Fehler des Servoverstärkers bei nicht verwendeter Achse	Fehler im Leistungsteil einer nicht zum Betrieb verwendeten Achse	
H064n* (n: Achse, $1 \leq n \leq 8$)	Systemfehler (ABS CPU)	Fehler der CPU bei der linearen Skalierung der Absolutwertposition	Versorgungsspannung aus- und wieder einschalten Position auf Abweichung überprüfen und bei Abweichung Einstellung der Grundposition wiederholen (siehe Technisches Handbuch) Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H065n* (n: Achse, $1 \leq n \leq 8$)	Fehler der Absolutwertposition	Fehlermeldung bei der Überwachung der Absolutwertposition innerhalb des linearen Skalierungsbereichs	
H066n* (n: Achse, $1 \leq n \leq 8$)	Fehler der Inkrementalposition	Fehlermeldung bei der Überwachung der relativen Position innerhalb des linearen Skalierungsbereichs	

Tab. A-1: Fehlercodes (4)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
H067n* (n: Achse, 1 ≤ n ≤ 8)	CPU-Fehler des seriellen Encoders	Fehler der CPU durch seriellen Encoder	Versorgungsspannung aus- und wieder einschalten Position auf Abweichung überprüfen und bei Abweichung Einstellung der Grundposition wiederholen (siehe Technisches Handbuch) Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H068n* (n: Achse, 1 ≤ n ≤ 8)	LED-Fehler des seriellen Encoders	Impulse der LED des seriellen Encoders fehlerhaft	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H0690*	Fehler im Bremskreis	Fehler eines Transistors oder Widerstandes im Bremskreis	
H0700*	Schaltfehler des Schützes	Schütz ist trotz READY OFF-Signal eingeschaltet.	
H0710*	Schaltfehler des Überstromrelais der Spannungsversorgung	Überstromrelais der Spannungsversorgung schaltet sich nicht ein.	
H0720*	Fehlerhafte Spannungsversorgung (Watch Dog)	Zeitüberschreitung in der Spannungswandleroutine	
H0730*	Schaltfehler des Überstromrelais der Spannungsversorgung	Überstromrelais der Spannungsversorgung schaltet sich nicht aus.	
H0740*	Fehler im Hauptkreis der Spannungsversorgung	Ladefehler der Kapazität	
H0750*	Speicherfehler des Spannungswandlers	Fehler im Speicherkreis des Spannungswandlers	
H0760*	Spannungswandler- oder A/D-Wandler-Fehler	Fehler im Spannungswandler oder im A/D-Wandler	
H078n* (n: Achse, 1 ≤ n ≤ 8)	Watch Dog des Servoverstärkers	Zeitüberschreitung bei der Datenverarbeitung des Servoverstärkers	
H079n* (n: Achse, 1 ≤ n ≤ 8)	Leiterplattenfehler des Servoverstärkers	Fehler auf der Leiterplatte des Servoverstärkers	
H080n* (n: Achse, 1 ≤ n ≤ 8)	Timerfehler im Servoverstärker	Fehler des Timers im Servoverstärker	
H081n* (n: Achse, 1 ≤ n ≤ 8)	Unterspannung des Servoverstärkers	Die Spannung zwischen den Klemmen P und N ist auf einen Wert kleiner gleich 200 V abgefallen.	Prüfen der Spannungsversorgung
H0820* H082n* (n: Achse, 1 ≤ n ≤ 8)	Erdschluss eines Servomotors	Erdschlussfehler vom Servomotor, defektes Motorkabel	Kabelanschluss prüfen Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren

Tab. A-1: Fehlercodes (5)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
H083n* (n: Achse, 1 ≤ n ≤ 8)	Überspannung des Servoverstärkers	Die Spannung zwischen den Klemmen P und N ist auf einen Wert größer gleich 400 V angestiegen.	Prüfen der Spannungsversorgung Versorgungsspannung aus- und wieder einschalten
H0840*	Kurzzeitiger Spannungsausfall des Servoverstärkers	Es ist ein Spannungsausfall von mindestens 50 ms Dauer aufgetreten.	
H0850*	Offene Phase/fehlerhafter Spannungswert	Eingangsspannung entspricht nicht der eingestellten Spannung.	Eingangsspannung und Wahlschalter zur Einstellung der Eingangsspannung prüfen
H0860	Versorgungsspannung zu hoch	Die Versorgungsspannung zwischen L+ und L- ist größer als 410 V.	
H087n* (n: Achse, 1 ≤ n ≤ 8)	Thermischer Encoderfehler	Der integrierte Thermoschutz des seriellen Encoders wurde aktiviert	Versorgungsspannung aus- und nach einer Wartezeit wieder einschalten Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H0880* H088n* (n: Achse, 1 ≤ n ≤ 8)	Überhitzung des Bremswiderstandes im Leistungsteil	Bremswiderstand wurde überhitzt.	
H089n (n: Achse, 1 ≤ n ≤ 8)	Überhitzung des Servomotors	Thermosicherung des Motors oder des Encoders wurde aktiviert.	Versorgungsspannung aus- und nach einer Wartezeit wieder einschalten Erhöhen der Beschleunigungs-/Bremszeit (siehe auch Befehle ACCEL, OVRD, SPD und Roboterstatusvariablen M_SETADL, M_LDFACT und Parameter JADL)
H090n* (n: Achse, 1 ≤ n ≤ 8)	Geschwindigkeitsüberschreitung beim Anfahren einer Absolutwertposition	Die Geschwindigkeit war beim Anfahren einer Absolutwertposition während der Initialisierung größer gleich 45 mm/s	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H091n* (n: Achse, 1 ≤ n ≤ 8)	Geschwindigkeitsüberschreitung des Servoverstärkers	Überschreitung der maximal zulässigen Motorgeschwindigkeit	
H0920* H092n* (n: Achse, 1 ≤ n ≤ 8)	Überstrom im Leistungsteil des Servoverstärkers	Überstrom im Servoverstärker, im Leistungsteil oder Kurzschluss im Motorkabel	
H093n* (n: Achse, 1 ≤ n ≤ 8)	Motorüberstrom	Der Strom durch den Motor ist zu groß, Fehler am Ausgang des A/D-Wandlers oder Motorkabel defekt	

Tab. A-1: Fehlercodes (6)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
H094n (n: Achse, $1 \leq n \leq 8$)	Überlast 1	Überschreitung der zulässigen Zeit für eine Überlast	Erhöhen der Beschleunigungs-/Bremszeit (siehe auch Befehle ACCEL, OVRD, SPD und Roboterstatusvariablen M_SETADL, M_LDFACT und Parameter JADL)
H095n (n: Achse, $1 \leq n \leq 8$)	Überlast 2	Überschreitung des maximal zulässigen Ausgangsstroms für eine Sekunde oder länger	Last prüfen, Roboter darf nicht mit umliegenden Einrichtungen kollidieren
H096n (n: Achse, $1 \leq n \leq 8$)	Positionsabweichung 1	Bei eingeschaltetem Servo ist die eingestellte Abweichung der aktuellen Position zur Zielposition zu groß.	Last prüfen, Roboter darf nicht mit umliegenden Einrichtungen kollidieren Bei niedrigen Umgebungstemperaturen oder einem Betrieb nach einer langen Betriebspause, Anlaufphase mit einer niedrigen Geschwindigkeit oder Warmlaufbetrieb ausführen.
H097n (n: Achse, $1 \leq n \leq 8$)	Positionsabweichung 2	Bei ausgeschaltetem Servo ist die eingestellte Abweichung der aktuellen Position zur Zielposition zu groß.	Einwirkung externer Kräfte bei eingeschaltetem Servo prüfen
H098n (n: Achse, $1 \leq n \leq 8$)	Positionsabweichung 3	Kein Motorstrom bei Auftreten der Fehlermeldung für Positionsabweichung 1	Motoranschluss prüfen
H101n (n: Achse, $1 \leq n \leq 8$)	Kollisionsüberwachung	Es ist eine Kollision aufgetreten	Kollisionsursache beseitigen
H102n (n: Achse, $1 \leq n \leq 8$)	Regenerative Überlast des Servoverstärkers der Zusatzachse	Überhitzung des regenerativen Bremswiderstandes	Prüfen der regenerativen Lastkapazität und der Parameter für die Zusatzachse
H1030*	Regenerative Überlast des Spannungswandlers	Die regenerative Kapazität des Spannungswandlers ist überschritten worden.	15 min bei eingeschalteter Versorgungsspannung warten, dann Versorgungsspannung aus- und wieder einschalten
H104n* (n: Achse, $1 \leq n \leq 8$)	Keine Kommunikation mit Encoder	Kein Kommunikationsaufbau zwischen Servoverstärker und Encoder; Encoderkabel oder Anschluss fehlerhaft	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H107n* (n: Achse, $1 \leq n \leq 8$)	Kommunikationsfehler mit Encoder	Unterbrechung der Kommunikation mit dem Encoder	
H108n* (n: Achse, $1 \leq n \leq 8$)	Kommunikationsfehler des Servoverstärkers	Unterbrechung der Kommunikation zwischen Steuergerät und Servoverstärker aufgrund eines fehlerhaften Kabels oder Kabelanschlusses	
H1090* H109n* (n: Achse, $1 \leq n \leq 8$)	Initialisierungsfehler des Servoverstärkers	Einstellungen der Achsen fehlerhaft (Parameter, Codierschalter)	
H1100*	Kommunikationsfehler des Servoverstärkers	Unterbrechung der Kommunikation zwischen Steuergerät und Servoverstärker aufgrund eines fehlerhaften Kabels oder Kabelanschlusses	

Tab. A-1: Fehlercodes (7)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
H111n* (n: Achse, 1 ≤ n ≤ 8)	RS232C-Kommunikationsfehler mit Servoverstärker	Kommunikationsfehler zwischen Servoverstärker und PC	Kabelanschluss und Kabel prüfen Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H112n* (n: Achse, 1 ≤ n ≤ 8)	Verlust der Absolutposition	Daten der Absolutposition sind gelöscht.	Batteriespannung und Encoderkabel prüfen
H113n* (n: Achse, 1 ≤ n ≤ 8)	Encoderdaten fehlerhaft	Encoderdaten pro Umdrehung sind fehlerhaft.	Vorgang wiederholen und Umgebungsbedingungen prüfen Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H114n* (n: Achse, 1 ≤ n ≤ 8)	CRC-Fehler der Daten des Servoverstärkers	Prüfsummenfehler in den Daten zum Steuergerät	Kabelanschluss und Kabel prüfen Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H115n* (n: Achse, 1 ≤ n ≤ 8)	Fehlerhafter Befehlswert der Kommunikationsdaten des Servoverstärkers	Die Befehlsdaten für die Zielposition sind zu groß.	
H116n* (n: Achse, 1 ≤ n ≤ 8)	Fehler in der Datenlänge zum Servoverstärker	Die Länge der Daten vom Steuergerät ist fehlerhaft.	
H117n* (n: Achse, 1 ≤ n ≤ 8)	Kommunikationsdaten zum Servoverstärker fehlerhaft	Die Daten vom Steuergerät sind fehlerhaft.	
H118n* (n: Achse, 1 ≤ n ≤ 8)	Fehler 1 der Rückmeldeimpulse des Servoverstärkers	Fehlende Impulse im Rückmeldesignal vom Encoder	Encoderkabel und Anschluss prüfen Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H119n* (n: Achse, 1 ≤ n ≤ 8)	Fehler 2 der Rückmeldeimpulse des Servoverstärkers	Abweichung zwischen den Rückmeldungen der Encoder von Motor und Maschine	
H1200*	CRC-Fehler in den Kommunikationsdaten des Servoverstärkers	In den Kommunikationsdaten vom Servoverstärker ist ein CRC-Fehler aufgetreten.	Kabelanschluss und Kabel prüfen Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H121n* (n: Achse, 1 ≤ n ≤ 8)	ID-Fehler in den Kommunikationsdaten des Servoverstärkers	In den Kommunikationsdaten vom Servoverstärker ist ein ID-Fehler aufgetreten.	
H122n* (n: Achse, 1 ≤ n ≤ 8)	Fehler der Achsennummer in den Kommunikationsdaten des Servoverstärkers	In den Kommunikationsdaten vom Servoverstärker ist ein Fehler der Achsennummer aufgetreten.	
H123n* (n: Achse, 1 ≤ n ≤ 8)	Sub-ID-Fehler in den Kommunikationsdaten des Servoverstärkers	In den Kommunikationsdaten vom Servoverstärker ist ein Sub-ID-Fehler aufgetreten.	
H1240*	Datennummernfehler in den Kommunikationsdaten des Servoverstärkers	In den Kommunikationsdaten vom Servoverstärker ist ein Datennummernfehler aufgetreten.	

Tab. A-1: Fehlercodes (8)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
H125n* (n: Achse, 1 ≤ n ≤ 8)	Parameterfehler des Servoverstärkers	Fehlerhafte Einstellung eines Parameters des Servoverstärkers	Parameter auf einen zulässigen Wert einstellen
C126n (n: Achse, 1 ≤ n ≤ 8)	Kommunikationsfehler des Encoders	Kein Kommunikationsaufbau möglich	Kabelanschluss und Kabel prüfen Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
C127n (n: Achse, 1 ≤ n ≤ 8)	Kommunikationsfehler des Encoders	Fehlerhafte Übertragung der seriellen Daten einer Absolutwertposition	Kabelanschluss und Kabel prüfen Position auf Abweichung überprüfen und bei Abweichung Einstellung der Grundposition wiederholen (siehe Technisches Handbuch) Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
C128n (n: Achse, 1 ≤ n ≤ 8)	Formatfehler des Encodersignals	Fehlerhaftes serielles Datenformat der Absolutwertposition	
C129n (n: Achse, 1 ≤ n ≤ 8)	Absolute Positionsabweichung des Servoverstärkers	Abweichung der Daten der Absolutwertposition beim Einschalten	Prüfen, ob die Achse aufgrund des Eigengewichts oder durch äußere Krafteinwirkung beim Einschalten bewegt wurde
C130n (n: Achse, 1 ≤ n ≤ 8)	Abweichung der Positionsskalierung	Abweichung der Rückmeldepulse von Encoder und Maschinenskalierung	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
C131n (n: Achse, 1 ≤ n ≤ 8)	Abweichung des Positions-Offsets	Abweichung der Rückmeldepulse von Encoder und Maschinenskalierung	
C132n (n: Achse, 1 ≤ n ≤ 8)	Fehler der Multirotationsdaten des Servoverstärkers	Fehlerhafte Multirotationsdaten des Encoders	
C133n (n: Achse, 1 ≤ n ≤ 8)	Niedrige Batteriespannung des Encoders	Batteriespannung für den Encoder ist abgesunken.	Batterie wechseln (siehe auch Technisches Handbuch)
C134n (n: Achse, 1 ≤ n ≤ 8)	Regenerative Überlast des Servoverstärkers	Regenerative Energie des Zusatzachsenverstärkers von 80 % oder mehr	Prüfen der regenerativen Kapazität und des Parameters zur Einstellung der regenerativen Kapazität
C135n (n: Achse, 1 ≤ n ≤ 8)	Überlast des Servoverstärkers	Überlast von 80 % oder mehr	Last prüfen, Roboter darf nicht mit umliegenden Einrichtungen kollidieren
C136n (n: Achse, 1 ≤ n ≤ 8)	Warnung des Zählers für Absolutwertposition	Fehlerhafter Encoderzähler	Batterie und Encoderkabel prüfen
C137n (n: Achse, 1 ≤ n ≤ 8)	Parameterfehler des Servoverstärkers	Einstellung eines Parameters außerhalb des zulässigen Wertebereichs	Parameter auf einen zulässigen Wert einstellen Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren

Tab. A-1: Fehlercodes (9)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
C138n (n: Achse, $1 \leq n \leq 8$)	Achse entfernen	Es wurde vom Steuergerät ein Befehl zum Entfernen der Achse ausgeführt.	Befehl abbrechen
H139n (n: Achse, $1 \leq n \leq 8$)	NOT-AUS des Servoverstärkers	Der NOT-AUS-Schalter des Steuergerätes wurde betätigt	NOT-AUS-Zustand zurücksetzen
C1400	Kurzzeitige Überschreitung der regenerativen Energie des Servoverstärkers	Kurzzeitige Überschreitung der regenerativen Energie des Spannungswandlers	Regenerative Kapazität prüfen Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H1410	Kurzzeitiger Spannungsausfall des Servoverstärkers	Es ist ein Spannungsausfall von mindestens 25 ms Dauer aufgetreten.	Prüfen der Spannungsversorgung
C1420	Regenerative Überlast des Servoverstärkers	Regenerative Energie von 80 % oder mehr	Geschwindigkeit des Roboters verringern
C143n (n: Achse, $1 \leq n \leq 8$)	Hauptkreis des Servoverstärkers der Zusatzachse ausgeschaltet	Der Servo wurden bei ausgeschaltetem Hauptkreis eingeschaltet.	Hauptkreis einschalten
H144n* (n: Achse, $1 \leq n \leq 8$)	Systemfehler 2 im Servoverstärker	Prozessorfehler	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H1450*	Kurzzeitige Unterbrechung der 24-V-DC-Spannungsversorgung	Unterpannung in der 24-V-DC-Spannungsversorgung	Anschluss CN22 prüfen
H1460*	Überstrom in der Spannungsversorgung	In der Spannungsversorgung des Leistungsteils ist ein Überstrom aufgetreten	Anschluss der Versorgungsspannung prüfen Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H1470*	Frequenzschwankung	Die Abweichung der Netzfrequenz ist zu groß	Frequenz der Versorgungsspannung prüfen Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H1480*	Fehler eines Parameters für die Versorgungsspannung	Fehlerhafte Einstellung eines Parameters für die Versorgungsspannung	Parameter korrigieren
H1490*	Überhitzung des Leistungsteils	Leistungsteil oder Bremswiderstand wurde überhitzt	Ventilator auf der Rückseite des Leistungsteils prüfen
H150n (n: Achse, $1 \leq n \leq 8$)	Falscher Servomotor	Die Kombination von Servoverstärker und Servomotor ist nicht korrekt.	Korrekte Kombination verwenden
H156n (n: Achse, $1 \leq n \leq 8$)	Positionsabweichung 4	Die Achse wurde während der Einschaltprozedur des Servoverstärkers bewegt	Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H157n (n: Achse, $1 \leq n \leq 8$)	Unregistrierter Servofehler	Es ist ein unregistrierter Servofehler aufgetreten.	Kann der Fehler nicht zurückgesetzt werden, schalten Sie die Spannungsversorgung aus und wieder ein. Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren

Tab. A-1: Fehlercodes (10)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
C158n (n: Achse, 1 ≤ n ≤ 8)	Unregistrierte Servowarnung	Es ist eine unregistrierte Servowarnung aufgetreten.	Kann die Warnung nicht zurückgesetzt werden, schalten Sie die Spannungsversorgung aus und wieder ein. Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H1600*	Kein Roboter definiert	Keiner der Mechanismen wurde definiert	Mindestens einen Mechanismus definieren Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H1610*	Bezeichnung des Mechanismus ungültig	Die Bezeichnung des Mechanismus ist ungültig oder nicht registriert.	Korrekt einstellen Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
C1620	Roboternummer ungültig	Die gewählte Mechanismennummer ist ungültig.	Korrekte Mechanismennummer einstellen
C1630	Während eines Servofehlers kann die Funktion Servo ON nicht ausgeführt werden.	Während eines Servofehlers kann die Servospannung nicht eingeschaltet werden.	Servofehler zurücksetzen und dann Servospannung einschalten
C1640	Bei ausgeschaltetem Dreistufenschalter kann die Funktion Servo ON nicht ausgeführt werden.	Bei ausgeschaltetem Dreistufenschalter kann die Servospannung nicht eingeschaltet werden.	Dreistufenschalter betätigen und dann Servospannung einschalten
C1650	Bei gelöster Bremse kann die Funktion Servo ON nicht ausgeführt werden.	Bei gelöster Bremse kann die Servospannung nicht eingeschaltet werden.	Bremsen aktivieren und dann Servospannung einschalten
C1660	Während des Servo-ON-Vorgangs kann die Funktion Servo ON nicht ausgeführt werden.	Während des Servo-ON-Vorgangs kann die Servospannung nicht eingeschaltet werden.	Servospannung nicht während des Servo-ON-Vorgangs einschalten
C1670	Während des Servo-OFF-Vorgangs kann die Funktion Servo ON nicht ausgeführt werden.	Der Servo-OFF-Zustand ist aktiv.	Servospannung nicht im Servo-OFF-Zustand einschalten
H1680	Zeitüberschreitung beim Servo-ON-Vorgang	Die Servos sind nicht innerhalb der zulässigen Zeit eingeschaltet worden.	Servoverstärker auf Fehler prüfen Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H1681	Fehlerhaftes Abschalten der Servospannung	Die Servospannung wurde ungewollt abgeschaltet.	Servoverstärker prüfen Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
C1690	Während der Dreistufenschalter ausgeschaltet ist, kann keine Bremse gelöst werden.	Es kann keine Bremse gelöst werden, während der Dreistufenschalter ausgeschaltet ist.	Dreistufenschalter einschalten und dann Bremse lösen

Tab. A-1: Fehlercodes (11)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
C1700	Bei aktiviertem NOT-HALT kann keine Bremse gelöst werden.	Es kann keine Bremse gelöst werden, wenn das NOT-HALT-Signal eingegeben wird.	NOT-HALT-Status aufheben und dann Bremse lösen
C1710	Im Servo-ON-Zustand kann keine Bremse gesteuert werden.	Bei eingeschalteter Servospannung kann keine Bremse gesteuert werden.	Servospannung ausschalten und dann Bremsen steuern
C1720	Während die Bremsen gelöst werden, kann kein weiterer Prozess zum Lösen der Bremsen ausgeführt werden.	Es kann kein weiterer Prozess zum Lösen der Bremsen ausgeführt werden, während die Bremsen gelöst werden.	Bremsen nicht lösen, während der Prozess „Bremsen lösen“ aktiv ist
C1730	Während die Bremsen aktiviert werden, kann kein weiterer Prozess zur Aktivierung der Bremsen ausgeführt werden.	Es kann kein weiterer Prozess zur Aktivierung der Bremsen ausgeführt werden, während die Bremsen aktiviert werden.	Bremsen nicht aktivieren, während der Prozess „Bremsen aktivieren“ aktiv ist
C1740	Servoparameter können nicht geändert werden.	Während der Einstellung von Parametern können keine weiteren Parameter eingestellt werden.	Einstellung wiederholen
C1750	Einstellung der Servoparameter ist fehlgeschlagen.	Es konnte keine Einstellung der Servoparameter vorgenommen werden.	
C1760	Ungültige Daten der Grundposition	Die Daten der Grundposition sind nicht korrekt eingestellt.	Korrekte Daten der Grundposition einstellen
C1770	Einstellung der Daten der Grundposition nicht abgeschlossen	Die Grundposition ist nicht eingestellt.	Einstellung wiederholen
C1780	Ungültige Achseneinstellung für Grundposition	Fehlerhafte Einstellung der Grundposition einer Achse	Setzen der Grundposition dieser Achse
C1781	Keine Einstellung der Grundposition im Servo-ON-Zustand	Es wurde versucht, die Grundposition bei eingeschalteter Servospannung einzustellen.	Vor der Einstellung der Grundposition Servospannung ausschalten
H1790*	Fehlerhafte Einstellung der Verfahrweggrenzen	Einstellung des Parameters MEJAR ist fehlerhaft	Einstellung des Parameters MEJAR korrigieren
H1800*	Fehlerhafte Einstellung der Verfahrweggrenzen für Absolutwertpositionierung	Einstellung des Parameters MEMAR ist fehlerhaft	Einstellung des Parameters MEMAR korrigieren
H1810*	Einstellfehler der benutzerdefinierten Grundposition	Einstellung des Parameters USERORG ist fehlerhaft	Einstellung des Parameters USERORG korrigieren
L182n (n: Achse, 1 ≤ n ≤ 8)	Positionsdaten stimmen nicht überein. Grundposition überprüfen	Positionsdaten bei ausgeschalteter Spannungsversorgung verändert	Überprüfung der Grundposition Bei Verschiebung, diese neu einstellen
L1830	Bereichsüberschreitung bei Ausführung des JRC-Befehls	Bereichsüberschreitung bei Ausführung des JRC-Befehls	Aktuelle Position und Einstellbereich prüfen
L184n (n: Achse, 1 ≤ n ≤ 8)	Fehlerhafte Einstellung des Parameters JRCQTT	Bereichsüberschreitung bei der Einstellung des Parameters JRCQTT	Einstellung des Parameters JRCQTT korrigieren
C1850	Kurzzeitiger Netzausfall	Spannungsausfall von mehr als 20 ms Dauer	Spannungsversorgung prüfen

Tab. A-1: Fehlercodes (12)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
L1860	Ungültiger Parameter (TLC)	Die Einstellung des Parameters TLC für die Anfahrtrichtung ist fehlerhaft	Einstellung des Parameters TLC korrigieren (= X/Y/Z)
C1870*	Stillstand des Ventilators Nr. XX	Der Ventilator Nr. XX im Steuergerät ist evtl. defekt	Ventilator austauschen
L2000	Servospannung ist ausgeschaltet.	Der Roboter kann sich nicht bewegen, da die Servospannung ausgeschaltet ist	Servospannung einschalten und Neustart ausführen
L2010	Keine Impulsausgabe	Fehlerhafte Festlegung des Impulsausgangs	Programm korrigieren
L2020	Lesen externer Positionsdaten	Während des Lesens eines externen Befehls konnte ein Befehl nicht ausgeführt werden.	
L2030	Anforderung JOG-Betrieb nicht akzeptiert	Die Anforderung zum JOG-Betrieb wurde ausgeführt, aber nicht akzeptiert.	JOG-Betrieb deaktivieren
H2031*	Fehlerhafte Einstellung der Dimensionen	Einstellung der Parameter JOGTSJ oder JOGJSP ist fehlerhaft.	Dimension von 5 oder kleiner einstellen
H209n (n: Bereich, 1 ≤ n ≤ 8)	Überschreitung des Überlagerungsbereichs n	Es wurde versucht, den Roboter aus dem benutzerdefinierten Bereich n zu bewegen.	Position korrigieren
H211n (n: Ebene, 1 ≤ n ≤ 8)	Überschreitung der Verfahrwegbegrenzungsebene n	Es wurde versucht, den Roboter außerhalb der Verfahrwegbegrenzungsebene n zu bewegen.	
H2129	Fehlerhafte Daten zur Einstellung der Verfahrwegbegrenzungsebene	Daten zur Festlegung der Verfahrwegbegrenzungsebene sind fehlerhaft	Daten korrigieren
H2130	Geschwindigkeitsüberschreitung	Geschwindigkeitsgrenzwert wurde überschritten.	Geschwindigkeit verringern
H213n (n: Achse, 1 ≤ n ≤ 8)	Geschwindigkeitsüberschreitung Jn-Achse	Geschwindigkeitsgrenzwert der Achse n wurde überschritten.	
H2140	Überschreitung des Absolutwertbereiches	Der Bereich des Absolutwerts wurde überschritten.	Fehler temporär zurücksetzen (siehe Abschn. 3.9) und Roboter im JOG-Betrieb in den zulässigen Bereich zurückfahren
H214n (n: Achse, 1 ≤ n ≤ 8)	Überschreitung des Absolutwertbereiches der Jn-Achse in +-Richtung	Der Bereich des Absolutwerts der Achse n in +-Richtung wurde überschritten.	
H215n (n: Achse, 1 ≤ n ≤ 8)	Überschreitung des Absolutwertbereiches der Jn-Achse in --Richtung	Der Bereich des Absolutwerts der Achse n in --Richtung wurde überschritten.	

Tab. A-1: Fehlercodes (13)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
H2160	Überschreitung der Gelenkwinkel	Der Verfahrwegbereich für Gelenkbewegungen wurde überschritten.	Position korrigieren
H216n (n: Achse, $1 \leq n \leq 8$)	Überschreitung des Gelenkwinkels der Jn-Achse in +-Richtung	Der Verfahrwegbereich der Achse n für Gelenkbewegungen wurde in +-Richtung überschritten.	
H217n (n: Achse, $1 \leq n \leq 8$)	Überschreitung des Gelenkwinkels der Jn-Achse in --Richtung	Der Verfahrwegbereich der Achse n für Gelenkbewegungen wurde in --Richtung überschritten.	
H2180	Überschreitung der orthogonalen Bereichsgrenze	Die orthogonale Bereichsgrenze wurden überschritten.	
H2181	Überschreitung des orthogonalen Bereiches der Achse X in +-Richtung	Die orthogonale Bereichsgrenze der X-Achse wurden in +-Richtung überschritten.	
H2182	Überschreitung des orthogonalen Bereiches der Achse Y in +-Richtung	Die orthogonale Bereichsgrenze der Y-Achse wurden in +-Richtung überschritten.	
H2183	Überschreitung des orthogonalen Bereiches der Achse Z in +-Richtung	Die orthogonale Bereichsgrenze der Z-Achse wurden in +-Richtung überschritten.	
H2191	Überschreitung des orthogonalen Bereiches der Achse X in --Richtung	Die orthogonale Bereichsgrenze der X-Achse wurden in --Richtung überschritten.	
H2192	Überschreitung des orthogonalen Bereiches der Achse Y in --Richtung	Die orthogonale Bereichsgrenze der Y-Achse wurden in --Richtung überschritten.	
H2193	Überschreitung des orthogonalen Bereiches der Achse Z in --Richtung	Die orthogonale Bereichsgrenze der Z-Achse wurden in --Richtung überschritten.	
H2500	Fehlerhafte Encoderdaten	Die Encoderdaten sind fehlerhaft	Wiederholgenauigkeit und Umgebungsbedingungen prüfen
L2600	Positionsdaten außerhalb des Verfahrwegbereiches	Positionsdaten liegen außerhalb des Verfahrwegbereiches.	Position korrigieren
L2601	Startposition außerhalb des Verfahrwegbereiches	Startposition liegt außerhalb des Verfahrwegbereiches.	
L2602	Zielposition außerhalb des Verfahrwegbereiches	Zielposition liegt außerhalb des Verfahrwegbereiches.	
L2603	Zwischenposition außerhalb des Verfahrwegbereiches	Zwischenposition liegt außerhalb des Verfahrwegbereiches.	
L2700	CMP-Fehler (Modus)	Festgelegter Modus weicht vom aktuellen Modus ab.	CMP OFF-Befehl ausführen und dann festlegen
C2710	CMP-Fehler (Abweichung)	Abweichung bei Ausführung des CMP-Befehls zu groß	Programm, Position o. Ä. so ändern, dass die Abweichung verringert wird.

Tab. A-1: Fehlercodes (14)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
C2720	CMP-Fehler (Gelenkwinkel)	Der zulässigen Gelenkwinkel bei Ausführung des CMP-Befehls wurde überschritten.	Positionsdaten korrigieren oder Abweichung verkleinern
C272n (n: Achse, $1 \leq n \leq 8$)	CMP-Fehler (Jn-Gelenkwinkel)	Der zulässigen Gelenkwinkel der Achse n wurde bei Ausführung des CMP-Befehls überschritten.	
C2730	CMP-Fehler (Geschwindigkeit)	Die zulässige Geschwindigkeit wurde bei Ausführung des CMP-Befehls überschritten.	Positionsdaten korrigieren oder Geschwindigkeit verringern
C273n (n: Achse, $1 \leq n \leq 8$)	CMP-Fehler (Geschwindigkeit der Jn-Achse)	Die zulässige Geschwindigkeit der Achse n wurde bei Ausführung des CMP-Befehls überschritten.	
C2740	CMP-Fehler (Konvertierung von Koordinaten)	Fehlerhafte Konvertierung von Koordinaten bei Ausführung des CMP-Befehls	Positionsdaten korrigieren
C2750	Keine Ausführung, wenn hohe Verfahrweggenauigkeit aktiv	Die Ausführung des CMP-Befehls ist bei aktivierter hoher Verfahrweggenauigkeit nicht möglich	Führen Sie den Befehl „TRK OFF“ und anschließend den Befehl „CMP OFF“ aus
L2800	Positionsdaten fehlerhaft	Kann auftreten, wenn die Position außerhalb des Bewegungsbereiches des Roboters liegt	Position korrigieren
L2801	Positionsdaten der Startposition fehlerhaft	Kann auftreten, wenn die Startposition außerhalb des Bewegungsbereiches des Roboters liegt	
L2802	Positionsdaten der Zielposition fehlerhaft	Kann auftreten, wenn die Zielposition außerhalb des Bewegungsbereiches des Roboters liegt	
L2803	Positionsdaten der Hilfsposition fehlerhaft	Kann auftreten, wenn die Hilfsposition außerhalb des Bewegungsbereiches des Roboters liegt	
L2810	Stellungsmerker fehlerhaft	Der Stellungsmerker der Start- und der Endposition muss passen.	Positionsdaten korrigieren
H2820	Werte der Beschleunigung/Abbremsung fehlerhaft	Die Werte zur Festlegung der Beschleunigung/Abbremsung sind zu klein.	Werte vergrößern
H2830	Systemfehler (fehlerhafte Einstellung des Befehlsparameters „TYPE“)	Der Befehlsparameter „TYPE“ des MOV-Befehls wurde auf einen unzulässigen Wert (z. B. „-1“) gesetzt	Wert korrekt einstellen (z. B. „0“, „1“ usw.)
H2840	Systemfehler (Interpolationsparameter)	Parameterdaten sind beschädigt	Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren

Tab. A-1: Fehlercodes (15)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
H2850	Systemfehler (Interpolationsnorm)	Ungültige Norm Fehler bei der internen Verarbeitung	Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
H2860	Systemfehler (Interpolationsmethode)	Es wurde eine ungültige Interpolationsmethode verwendet.	
H2870	Systemfehler (undefinierte Interpolationsdaten)	Die Positionsdaten für die Interpolation wurden nicht definiert. Fehler bei der internen Verarbeitung	
H2880*	Systemfehler (zu wenig Speicherplatz)	Kein Speicherplatz für die Daten des Interpolationsprozesses	
H2890	Systemfehler (undefinierte Fehlernummer)	Bei der internen Verarbeitung wurde eine undefinierte Fehlernummer generiert.	
L2900	Systemfehler MO0	Fehler bei der internen Verarbeitung	
L3100	Speicherüberlauf	Der Speicherplatz nimmt kontinuierlich ab, wenn z. B. ein Sprung über GOSUB ohne Rücksprung über RETURN oder ein Sprung aus einer FOR-NEXT-Schleife über einen GOTO-Befehl erfolgt.	Bei einem Sprung über GOSUB Rücksprung über RETURN und in einer FOR-NEXT-Schleife keinen Sprung über GOTO programmieren
L3110	Bereichsfehler eines Befehlsparameters	Befehlsparameter liegt außerhalb des zulässigen Einstellbereiches.	Bereich prüfen und Eingabe wiederholen
L3120	Anzahl der Befehlsparameter fehlerhaft	Anzahl der Befehlsparameter entspricht nicht dem zulässigen Wert.	Anzahl der Befehlsparameter prüfen und korrigieren
L3130	Datei bereits geöffnet	Es wurde versucht, eine bereits geöffnete Datei zu öffnen.	Dateinummer prüfen und evtl. richtige Datei öffnen
L3140	Öffnen einer Datei nicht möglich	Die Datei kann nicht geöffnet werden.	
L3150	Kein Schreibvorgang möglich, da die Zugriffsmethode auf „INPUT“ gesetzt ist	Kein Schreibvorgang möglich, da die Zugriffsmethode auf „INPUT“ gesetzt ist	Dateinummer prüfen, Zugriffsmethode prüfen und Vorgang wiederholen
L3170	Kein Schreibvorgang möglich, da die Zugriffsmethode auf „OUTPUT“ gesetzt ist	Kein Schreibvorgang möglich, da die Zugriffsmethode auf „OUTPUT“ gesetzt ist	
L3180	Systemfehler (Einlesen von Felddaten nicht möglich)	Systemfehler (Einlesen von Felddaten nicht möglich)	Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
L3200	Datei kann nicht gelesen werden.	Ein Lesen der Datei ist nicht möglich.	Dateiinhalte prüfen
L3210	Variable kann nicht geschrieben werden.	Ein Schreiben der Variablen ist nicht möglich.	Schreibschutzstatus der Variablen prüfen

Tab. A-1: Fehlercodes (16)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
L3220	Zu viele Programmverschachtelungen	Die Anzahl der zulässigen Programmverschachtelungen von IF-Bedingungen oder FOR-Anweisungen ist überschritten.	Programm korrigieren und Ausführung wiederholen
L3230	Anzahl der FOR- und NEXT-Anweisungen ist nicht identisch.	Die Anzahl der FOR- und NEXT-Anweisungen ist nicht identisch.	
L3240	Zu viele Programmverschachtelungen (FOR, WHILE)	Die Anzahl der Programmebenen einer FOR- oder WHILE-Schleife übersteigt 16.	Programm korrigieren
L3250	Die Anzahl der WHILE- und WHEN-Anweisungen ist nicht identisch.	Die Anzahl der WHILE- und WHEN-Anweisungen ist nicht identisch.	Programm korrigieren und Ausführung wiederholen
L3251	Die Anzahl der Sprungziele übersteigt 32.	Die Anzahl der Verzweigungen übersteigt 32.	
L3252	Die Anzahl der IF- und ENDIF-Anweisungen ist nicht identisch.	Die Anzahl der IF- und ENDIF-Anweisungen ist nicht identisch.	
L3253	Zu viele Programmverschachtelungen (IF, ENDIF)	Die Anzahl der Programmebenen übersteigt 8.	
L3254	Die Anzahl der SELECT- und END SELECT-Anweisungen ist nicht identisch.	Die Anzahl der SELECT- und END SELECT-Anweisungen ist nicht identisch.	
L3255	Die Anzahl der IF- und ELSE-Anweisungen ist nicht identisch.	Die Anzahl der IF- und ELSE-Anweisungen ist nicht identisch.	
L3260	Parallele Ausführung der festgelegten Programme nicht möglich	Parallele Ausführung der festgelegten Programme nicht möglich	Einzelne Programme auswählen und ausführen
L3270	Befehlsgröße überschritten	Befehlsgröße überschritten	Maximal 256 Zeichen verwenden
L3280	Ausführung ohne GETM-Befehl nicht möglich	Die Ausführung der Anweisung ist ohne GETM-Befehl nicht möglich, oder es wurde eine ungültige Mechanismnummer festgelegt.	Anweisung nach Ausführung des RELM- und GETM-Befehls in einem anderen Programmplatz ausführen
L3281	Keine Ausführung im Betrieb	Keine Ausführung im Betrieb	Keine Ausführung im Betrieb
L3282	Starten nicht möglich	Fehlerhafte Programmwahl oder Attributeinstellung	Programm in den festgelegten Programmplatz laden oder Attributeinstellung korrigieren
L3283	Keine Ausführung im Betrieb	Vorheriger Befehl wird ausgeführt.	Vorheriger Befehl wird ausgeführt.
L3284	Keine Ausführung während der Editierung möglich	Keine Ausführung während der Editierung möglich	Nicht während einer Editierung ausführen
L3285	Keine Ausführung möglich (RUN oder WAIT)	Keine Ausführung im Betrieb oder Wartezustand möglich	Programm zurücksetzen (Unterbrechung aufheben)
L3286	Leeres Programm	Start eines leeren Programms	Programm schreiben oder korrektes Programm wählen
L3287	Keine Ausführung möglich (ERROR oder ALWAYS)	Programm kann nicht bei Startbedingungen ERROR und ALWAYS gestartet werden.	Programm korrigieren

Tab. A-1: Fehlercodes (17)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
L3288	Keine Ausführung während einer Programmmeditierung möglich	Keine Ausführung des Programms während einer Programmmeditierung möglich	Editierung zuerst beenden und anschließend Programm ausführen
L3289	Das ausgewählte Programm (SLT*) existiert nicht.	Das in der Programmplatzliste ausgewählte Programm existiert nicht.	Programmplatzparameter korrigieren
L3290	Keine Ausführung der Systemanwendung möglich	Keine Ausführung der Systemanwendung möglich	Prüfen, ob eine andere Anwendung (Benutzeranwendung) ausgeführt wird
L3300	Keine Ausführung einer Benutzeranwendung möglich	Keine Ausführung einer Benutzeranwendung möglich	Prüfen, ob die Systemanwendung ausgeführt wird
L3310	Der Befehl XRUN kann nicht ausgeführt werden, wenn das Programm bereits abgearbeitet wird.	Der Befehl XRUN kann nicht ausgeführt werden, wenn das Programm bereits abgearbeitet wird.	Festgelegten Programmplatz stoppen und Programm ausführen
L3320	Der Befehl XRUN kann nicht ausgeführt werden, wenn kein Programm gewählt wurde.	Der Befehl XRUN kann nicht ausgeführt werden, wenn kein Programm gewählt wurde.	Programm korrigieren und ausführen
L3330	Der Befehl XSTP kann nicht ausgeführt werden, wenn kein Programm gewählt wurde.	Der Befehl XSTP kann nicht ausgeführt werden, wenn kein Programm gewählt wurde.	
L3340	Der Befehl XRST kann nicht ausgeführt werden, wenn kein Programm gewählt wurde.	Der Befehl XRST kann nicht ausgeführt werden, wenn kein Programm gewählt wurde.	Ein Rücksetzen kann nur im Wartestatus erfolgen
L3350	Der Befehl XRST kann nicht ausgeführt werden, wenn das Programm abgearbeitet wird.	Der Befehl XRST kann nicht ausgeführt werden, wenn das Programm abgearbeitet wird.	Verarbeitung unterbrechen und Befehl ausführen
L3360	Der Befehl XLOAD kann nicht ausgeführt werden (keine PSA)	Der Befehl XLOAD kann nicht ausgeführt werden, wenn die Programmwahl nicht freigegeben worden ist.	Befehl XRST ausführen und Vorgang wiederholen
L3361	Das Programm kann nicht geladen werden (SLT*)	Das im Programmplatzparameter (SLTn) ausgewählte Programm existiert nicht.	Programmplatzparameter korrigieren
L3370	Der Befehl XCLR kann nicht ausgeführt werden, wenn kein Programm gewählt wurde.	Der Befehl XCLR kann nicht ausgeführt werden, wenn kein Programm gewählt wurde.	Der Befehl XCLR kann nur zur bei einem ausgewählten Programm ausgeführt werden
L3380	Der Befehl XCLR kann nicht ausgeführt werden (keine PSA)	Der Befehl XCLR kann nicht ausgeführt werden, wenn die Programmwahl nicht freigegeben worden ist.	Programm zurücksetzen und Befehl erneut ausführen
L3390	Keine kreisförmige Palettierung möglich	Keine kreisförmige Palettierung möglich	Andere Palettendefinition wählen
L3400	Systemfehler (zu wenig Speicherplatz)	Die Speicherplatzkapazität ist überschritten	Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
L3500	Das mit INPUT angegebene Format ist fehlerhaft.	Der Typ der im INPUT-Befehl angegebenen Variablen passt nicht zu den empfangenen Daten.	Format prüfen
L3600	Sprungziel existiert nicht.	Das im Befehl DEF ACT, ON COM oder ON GOTO angegebene Sprungziel existiert nicht.	Sprungziel prüfen

Tab. A-1: Fehlercodes (18)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
L3700	Verwendung einer nicht definierten Variablen	Verwendung einer nicht definierten Variablen	Variable vor Verwendung definieren und Startwert zuweisen
L3710	Zu viele Programmverschachtelungen durch den CALLP-Befehl.	Es wurden mehr als 7 Programmaufrufe mit dem CALLP-Befehl ausgeführt.	Verschachtelungstiefe reduzieren
L3720	Die Anzahl der RX- und NX-Anweisungen ist nicht identisch.	Die Anzahl der RX- und NX-Anweisungen ist nicht identisch.	Die Anzahl der RX- und NX-Anweisungen anpassen.
L3810	Typ des Arguments fehlerhaft	Typ des Arguments in einer arithmetischen Operation, einer Einzelargument- oder Vergleichsoperation o. Ä. ist fehlerhaft	Korrekten Typ des Arguments festlegen
L3820	Nicht definierter Code	Evtl. ist eine Programm- oder Systemstatusvariable zerstört.	Stellen Sie die Daten mit Hilfe des Backups wieder her. Ist kein Backup verfügbar, muss das Programm neu geschrieben werden.
L3830	Ausführung des GETM-Befehls nicht möglich	Ausführung des GETM-Befehls nicht möglich	Prüfen, ob der festgelegte Roboter mit einem anderen Programmplatz verwendet wird
L3840	RETURN-Befehl ohne GOSUB-Befehl	Ausführung des RETURN-Befehls ohne GOSUB-Befehl	Programm prüfen
L3850	Verwendung einer nicht definierten Palette	Verwendung einer nicht definierten Palette	Palette vor Verwendung über den Befehl DEF PLT definieren
L3860	Ungültige Positionsdaten festgelegt	Fehlerhafte Positionsdaten	Positionsdaten prüfen
L3870	Ungültige Mechanismnummer festgelegt	Der festgelegte Mechanismus kann nicht definiert werden.	Mechanismnummer korrigieren
L3880	Ungültige Programmplatznummer	Die festgelegte Programmplatznummer ist ungültig.	Korrekte Programmplatznummer eingeben
L3890	Systemfehler	Fehlerhafter Betriebsbefehl, evtl. ist das Programm zerstört	Stellen Sie die Daten mit Hilfe des Backups wieder her. Ist kein Backup verfügbar, muss das Programm neu geschrieben werden.
L3900	Ausführung des JRC-Befehls nicht möglich	Der Befehl ist über den Parameter JRCEXE deaktiviert und kann deshalb nicht ausgeführt werden	Parameter JRCEXE ändern und anschließend Befehl JRC ausführen
L3910	Ausführung des Befehls JRC 0 nicht möglich	Der Befehl JRC 0 kann für die festgelegte Achse nicht ausgeführt werden	Einstellung korrigieren
L3930	Ausführung des Befehls nicht möglich	Kollisionsüberwachung ist aktiviert	Kollisionsüberwachung deaktivieren (COLCHK OFF)
L3940	Ausführung des COLCHK-Befehls nicht möglich	Kollisionsüberwachung wird durch eine andere Funktion blockiert	Blockierende Funktion aufheben
L3950	Ausführung des NOERR-Modus nicht möglich	Kein Interrupt-Aufruf über die Statusvariable M_COLSTS freigegeben	Interrupt-Aufruf über die Statusvariable M_COLSTS definieren und freigeben
L3960	Interrupt-Aufruf kann nicht aufgehoben werden	NOERR-Modus wird ausgeführt	Nach Aufheben des NOERR-Modus Interrupt deaktivieren

Tab. A-1: Fehlercodes (19)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
L3970	Ausführung des COLCHK-Befehls nicht möglich	Kollisionsüberwachung über Parameter COL gesperrt	Kollisionsüberwachung über Parameter COL freigegeben
L3980	Einstellung der Variablen M_LDM nicht möglich	Hohe Verfahrweggenauigkeit ist aktiviert (PREC ON)	Hohe Verfahrweggenauigkeit aufheben (PREC OFF)
L3981	Ausführung des PREC-Befehls nicht möglich	Fehlerhafte Einstellung der Variablen M_LDM	Lastmodus 1 einstellen (M_LDM = 0)
L4000	Systemfehler (Zeitüberschreitung)	Programmexterner Fehler	Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
L4100	Zu große Anzahl der registrierten Programme	Zu große Anzahl der registrierten Programme	Nicht verwendete Programme löschen
L4110	Zu wenig Speicherplatz	Die Anzahl der Programme und die Datenmenge sind größer als der verfügbare Speicherplatz.	Nicht verwendete Programme oder Daten löschen oder optionale Speichererweiterung installieren
L4120	Programmname zu lang	Die Länge des Programmnamens darf höchstens 12 Zeichen und 3 Zusatzzeichen betragen.	Länge des Programmnamens auf einen zulässigen Wert kürzen
L4130	Ungültiger Programmname	Programmname enthält ein ungültiges Zeichen.	Es dürfen nur Buchstaben und Ziffern verwendet werden.
L4140	Das festgelegte Programm wurde nicht gefunden.	Das festgelegte Programm wurde nicht gefunden.	Gültiges Programm festlegen oder festgelegtes Programm erstellen
L4150	Fehler im Dateiinhalt	Evtl. sind die Daten durch einen Spannungsausfall während des Schreibvorgangs zerstört	Datei löschen
L4160	Kein gültiges Roboterprogramm	Das ausgewählte Programm ist kein gültiges Roboterprogramm.	Gültiges Programm auswählen
L4170	Programm wird editiert.	Programm wird editiert.	Editiertes Programm schließen
L4180	Programm wird ausgeführt.	Programm wird ausgeführt.	Programm stoppen
L4190	Das Programm ist ausgewählt	Programmausführung wird vorbereitet.	Programm zurücksetzen
L4200	Es können keine Daten in die Datei geschrieben werden.	1: Schreibschutz aktiv oder 2: Speicherplatz zu niedrig	1. Schreibschutz aufheben 2. Nicht verwendete Dateien löschen
L4201	Kein Betrieb im ROM-Modus möglich	Kein Betrieb im ROM-Modus möglich	Umschalten in den RAM-Modus
L4210	Die eingegebene Anweisung ist zu lang.	Die Länge der Anweisung darf höchstens 127 Zeichen betragen.	Länge der Anweisung auf einen zulässigen Wert kürzen
L4220	Syntaxfehler	Die eingegebene Anweisung weist einen Syntaxfehler auf.	Bei der Eingabe Syntax beachten
L4230	Zeilennummer fehlerhaft	Die angegebene Zeilennummer existiert nicht.	Inhalt prüfen und gültige Zeilennummer eingeben
L4240	Schreibgeschütztes Programm	Schreibgeschütztes Programm	Schreibschutz aufheben
L4250	Keine weiteren Zeilen oder Variablen	Die Anzahl der gelesenen Zeilen oder Variablen übersteigt den maximal zulässigen Wert.	Programme prüfen

Tab. A-1: Fehlercodes (20)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
L4300	Der eingegebene Variablenname ist zu lang.	Die maximale Länge eines Variablennamens beträgt 8 Zeichen.	Variablennamen auf 8 Zeichen kürzen
L4310	Ungültiges Zeichen	Es wurde ein anderes Zeichen als A bis Z oder 0 bis 9 verwendet.	Keine ungültigen Zeichen verwenden
L4320	Variable ist schreibgeschützt.	Variable ist schreibgeschützt.	1. Verwenden einer speicherbaren Variablen 2. Schreibschutz aufheben
L4330	Es wurde versucht, eine nicht lesbare Variable zu lesen.	Variable ist lesegeschützt.	1. Verwenden einer speicherbaren Variablen 2. Schreibschutz aufheben
L4340	Nicht definierte Variable	Die Variable ist nicht deklariert.	Variable deklarieren
L4350	Doppelte Deklaration einer Variablen	Bereits definierte Variablen können nicht mit der DIM- oder DEF-Anweisung neu definiert werden.	1. Variablennamen ändern und definieren 2. Definierte Variable löschen
L4360	Dieselbe Variable kann nicht öfter als 65535-mal verwendet werden.	Bei 10 P1 = P1 + P2 wird 2-mal auf P1 und 1-mal auf P2 zugegriffen.	Programm so umschreiben, dass die Zahl der Zugriffe auf dieselbe Variable vermindert wird.
L4370	Fehlerhaftes Feldelement	1. Feldelement liegt nicht im zulässigen Bereich. 2. Variable ist kein Feld.	1. Setzen Sie das Feldelement auf einen Wert zwischen 1 und der max. Anzahl der Feldelemente. 2. Verwenden Sie kein Feldelement.
L4380	In einer Anweisung verwendete Variablen können nicht gelöscht werden.	In einer Anweisung verwendete Variablen können nicht gelöscht werden.	Löschen Sie die Anweisung mit der verwendeten Variablen.
L4390	Fehler in der Kombination der Variablentypen	Die Typen der benutzerdefinierten externen Variablen sind unterschiedlich.	Variablentypen anpassen
L4400	Programmfehler	Unzulässiger Programminhalt	Programm löschen
L4410	Fehler in umnummerierten Daten	Fehler in umnummerierten Daten	Daten zurücksetzen
L4420	Es dürfen keine Zeilennummern größer als 32767 verwendet werden.	Die neue Zeilennummer oder Zeilennummernlücke überschreitet den zulässigen Wert.	Keine Zeilennummern über 32767 verwenden
L4430	Das gesuchte Zeichen konnte nicht gefunden werden.	Das gesuchte Zeichen konnte nicht gefunden werden.	Programm prüfen
L4440	Die Marke existiert bereits.	Eine bereits definierte Marke kann kein zweites Mal definiert werden.	1. Markennamen ändern oder 2. Definierten Markennamen löschen
L4450	Unzulässige Variablennummer	Die Positions-, Zähler oder Zeichenkettennummer liegt außerhalb des zulässigen Bereiches	Variablennummer im zulässigen Bereich festlegen
L4460	Unzulässiger Befehlsparameter	Der Befehlsparameter liegt außerhalb des zulässigen Bereiches	Befehlsparameter im zulässigen Bereich festlegen
L4800	Systemfehler (Basisprogramm)	Das Basisprogramm des Systems kann nicht geöffnet werden oder ist in den Parametern nicht korrekt festgelegt.	Legen Sie das Basisprogramm des Systems korrekt in den Parametern fest.

Tab. A-1: Fehlercodes (21)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
L4810	Kein Zugriff auf eine benutzerdefinierte globale Variable	Der Parameter PRGUSR ist nicht korrekt eingestellt	Zur Verwendung benutzerdefinierter externer Variablen muss das Benutzerbasisprogramm im Parameter PRGUSR registriert sein.
L4811*	Mehrfachdeklaration einer globalen Variablen	Eine globale Systemvariable ist als benutzerdefinierte globale Variable deklariert worden	Programm korrigieren
L4820	Keine Programmeditierung möglich	Das Programm wurde während der Editierung geschlossen. Das kann der Fall sein, wenn z. B. beim Editieren des Programms über den PC der [ENABLE/DISABLE]-Schalter der Teaching Box betätigt wird.	Programmeditierung wiederholen
L4900	Systemfehler	Der Programmname zur internen Verarbeitung ist unzulässig	Im Wiederholungsfall MITSUBISHI-Vertriebspartner kontaktieren
L4910	Fehlerhafte Einstellung der Programmiermethode	Der Parameter RLNG ist falsch eingestellt (1: MELFA-BASIC IV, 0: MOVEMASTER-COMMAND)	Parameter RLNG zurücksetzen oder anderes Programm auswählen
L4920	Keine Backup-Daten im ROM	Es sind keine Backup-Daten im ROM-Speicher	Vor dem Betrieb Backup anlegen
H5000	Fehlerhaftes Einschalten des Teach-Modus	Der [ENABLE/DISABLE]-Schalter der Teaching Box wurde im Automatikbetrieb umgeschaltet.	Stellen Sie den [ENABLE/DISABLE]-Schalter der Teaching Box auf DISABLE oder wählen Sie den Teach-Modus.
L5010	Signal AUTOENA ist AUS	Signal der Freigabe des Automatikbetriebs AUS	Signal zur Freigabe des Automatikbetriebs einschalten oder Teach-Modus wählen.
L5100	Kein Programm ausgewählt	Für den festgelegten Programmplatz wurde kein Programm gewählt.	Programm für den festgelegten Programmplatz wählen.
L5110	Kein kontinuierlicher Betrieb möglich	Es wurde ein anderes Programm gewählt.	Wahl des richtigen Programms
L5120	Programm kann nicht gewählt werden.	Die Programmwahl des festgelegten Programmplatzes ist nicht freigegeben.	Programm zurücksetzen
L5130	Servospannung kann nicht eingeschaltet werden.	Der Vorgang Servo AUS ist aktiv.	Warten Sie, bis die Servospannung ausgeschaltet ist und schalten Sie dann die Servospannung ein.
L5140	Lesen einer Datei nicht möglich	Es wird gerade ein Lese- oder Editierungsvorgang ausgeführt.	Editierte Datei schließen und Daten nach Abschluss des Lesevorgangs lesen
L5150	Keine Ausführung, da Grundposition nicht eingestellt	Grundposition nicht eingestellt	Grundposition einstellen
L5200*	Parameterfehler (TASKMAX)	Anzahl der parallel ausgeführten Programme überschreitet den im Parameter TASKMAX festgelegten Maximalwert (Werkseinstellung: 8, Maximalwert: 32).	Anzahl der parallel ausgeführten Programme reduzieren oder Parameter TASKMAX ändern

Tab. A-1: Fehlercodes (22)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
L5210*	Parameterfehler (TASKMAX)	Anzahl der definierten Mechanismen überschreitet den im Parameter MECHAMAX festgelegten Maximalwert.	Anzahl der definierten Mechanismen reduzieren
L5400	Roboter können nicht definiert werden.	Mechanismen können nicht definiert werden.	Unabhängige Roboternummer definieren
L5410	Ein nicht existierender Modus wurde festgelegt.	Es wurde ein anderer Modus als AUTO/TEACH festgelegt.	MITSUBISHI-Vertriebspartner kontaktieren
L5420	Ungültige Programmplatznummer	Festlegung einer ungültigen Programmplatznummer	Gültigen Programmplatz festlegen
L5430	Ungültige Mechanismusnummer	Festlegung einer ungültigen Mechanismusnummer	Gültigen Mechanismus festlegen
L5600	Keine Ausführung im Fehlerfall möglich	Keine Ausführung im Fehlerfall möglich	Fehler zurücksetzen
C5610	Keine Ausführung bei eingeschaltetem STOP-Signal	Keine Ausführung bei Eingabe des Stopp-Signals möglich	Stopp-Signal ausschalten und Ausführung wiederholen
L5620	Keine Ausführung bei eingeschaltetem CSSTOP-Signal	Keine Ausführung bei Eingabe des Zyklus-Stopp-Signals möglich	Zyklus-Stopp-Signal ausschalten
L5630	Keine Ausführung bei eingeschaltetem SRV OFF-Signal	Keine Ausführung bei Eingabe des Signals Servo OFF möglich	Servo OFF-Signal ausschalten
L5640	Keine Ausführung während des Betriebs	Keine Ausführung während des Betriebs	Betrieb stoppen und Ausführung wiederholen
L5650	Keine Ausführung während des Stopp-Vorgangs	Keine Ausführung während des Stopp-Vorgangs	Stopp-Vorgang beenden und Ausführung wiederholen
L5660	Editierung im Betrieb (inklusive Startbedingung ALWAYS)	Keine Editierung während des Betriebs (inklusive des kontinuierlichen Betriebs)	Programm stoppen und Ausführung wiederholen
L5990	Systemfehler (ungültiger Befehl)	Es wurde ein nicht registrierter Befehl ausgeführt.	Korrekten Befehl ausführen
L6010	Ein ungültiger Befehl wurde gesendet.	Der Fehler kann auftreten, wenn Daten vor Öffnen des Kommunikationskanals gesendet wurden oder ein nicht registrierter Befehl ausgeführt wurde oder die Version der Software für das Steuergerät inkompatibel zu der Version der Programmier-Software ist.	Daten erst nach Öffnen des Kommunikationskanals senden oder Versionen abgleichen
L6020	Betriebsrecht nicht aktiviert	Es wurden keine Betriebsrechte zugewiesen.	Betriebsrecht zuweisen
L6030	Betriebsrecht zum Editieren nicht aktiviert	Es wurden keine Betriebsrechte zum Editieren zugewiesen.	Betriebsrecht zum Editieren zuweisen
L6040	Systemfehler (ungültige Gerätenummer)	Die Gerätenummer ist nicht registriert.	Gültige Gerätenummer angeben
C6050	Datei kann nicht geöffnet werden.	Die Blockdatei kann nicht geöffnet werden.	Datei prüfen und korrekte Datei festlegen

Tab. A-1: Fehlercodes (23)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
C6060	Teach-Modus nicht aktiviert	Parameter im Teach-Modus schreiben	Teach-Modus wählen und dann Parameter schreiben
C6070	Zeit kann nicht eingestellt werden.	Die Zeiteinstellung kann nur bei gestopptem Programm und ausgeschalteter Servoversorgung durchgeführt werden.	Betrieb unterbrechen, Servospannung ausschalten und anschließend Zeit einstellen
C6080	Text zu lang	Die Zeichenkette eines Kommunikationstextes überschreitet die maximal zulässige Länge	Überwachen Sie die Anzahl der Zeichen in einem Kommentar über verschiedene Einstellungen, z. B über Parametern
C6500	Die Kommunikationsleitung wurde nicht über das Anwendungsprogramm geöffnet.	Es wurde im Programm kein OPEN-Befehl ausgeführt.	OPEN-Befehl ausführen und dann „PRN“ senden
H6510	Fehlerhafter Schnittstellenparameter	Parameter für die RS232C-Schnittstelle fehlerhaft	Kommunikationsparameter korrigieren und Spannungsversorgung aus- und wieder einschalten
H6520	Fehlerhafter Schnittstellenparameter	Parameter für die RS422-Schnittstelle fehlerhaft	
H6530*	Fehlerhafte Einstellung des Parameters COMDEV	Fehlerhafte Einstellung des Parameters COMDEV	Einstellung des Parameters COMDEV korrigieren
L6600	Ungültige Signalnummer	Die angegebene Signalnummer wurde nicht definiert.	Korrekte Signalnummer angeben
L6610	Handsensorsignal kann nicht geschrieben werden.	Ungültiges Signal	Gültiges Signal verwenden
L6620	Eingangssignal kann nicht in den für den Roboter festgelegten Bereich geschrieben werden.	Ungültiges Signal	Gültiges Signal verwenden
L6630	Eingangssignal kann nicht geschrieben werden.	Aktueller Eingangssignalmodus	Pseudo-Eingangssignal setzen
H6640	Fehlerhafte Einstellung eines speziellen Parameters	Parametereinstellung fehlerhaft	Einstellung des Parameters korrigieren
H6641	Doppelte Definition eines speziellen Eingangssignals	Parametereinstellung fehlerhaft	
H6642	Das STOP-Signal ist fest dem Signal Nummer 0 zugewiesen	Parametereinstellung fehlerhaft	Speziellen Parameter STOP auf „0“ setzen
H6643	Fehlerhafte Einstellung eines speziellen Parameters	Parametereinstellung fehlerhaft	Die Startnummer muss kleiner als die Endnummer sein
L6650*	Doppelte Definition eines speziellen Ausgangssignals	Ungültige Parametereinstellung	Parameter korrigieren
L6651*	Doppelte Definition eines speziellen Ausgangssignals und des Parameters HANDTYPE	Ungültige Parametereinstellung	
L6660	Schreiben eines speziellen Ausgangssignals nicht möglich	Fehlerhafte Programmeinstellung	Programm korrigieren
L6670	Ungültiges Bitmuster zum Zurücksetzen der Ausgänge	Die Parameter wurden nicht in Gruppen von 8 gesetzt.	Parameter korrigieren

Tab. A-1: Fehlercodes (24)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
L6800*	Pseudo-Eingangszustand löschen	Nach Löschen des Pseudo-Eingangszustands muss die Versorgungsspannung aus- und wieder eingeschaltet werden, um einen fehlerhaften Betrieb des Roboters durch externe Eingangssignale zu verhindern	Nach dem Aus- und Wiedereinschalten der Spannungsversorgung erfolgt die Umschaltung auf ein externes Eingangssignal.
C6900	Start im Pseudo-Eingangssignal-Modus	Parametereinstellungen	Setzen Sie die Parameter bei Verwendung aktueller Signale zurück und schalten Sie die Versorgungsspannung aus und wieder ein.
C7000	Die zu kopierende Quelldatei kann nicht gefunden werden.	Die zu kopierende Quelldatei kann nicht gefunden werden.	Korrekten Dateinamen angeben
C7010	Die zu löschende Zieldatei konnte nicht gefunden werden.	Die zu löschende Zieldatei kann nicht gefunden werden.	
C7020	Die Datei, die umbenannt werden soll, kann nicht gefunden werden.	Die Datei, die umbenannt werden soll, kann nicht gefunden werden.	
H7030*	Parameteränderungsdatei zu groß	Änderungskapazität zu groß	MITSUBISHI-Vertriebspartner kontaktieren
C7040	Änderung des Parameters unzulässig	Änderung des Parameters ist aus Sicherheitsgründen nicht zulässig.	
H7050	Dateifehler	Datei ist beschädigt.	
H7060*	System-RAM-Ressourcen erschöpft	Speicherkapazität wurde überschritten.	
C7070	Speicherkapazität erschöpft	Die Programme und Daten überschreiten die Speicherkapazität.	Nicht verwendete Programme oder Daten löschen
L7071	Zu wenig Speicher für die Ausführung der Funktion CTN „Programm fortsetzen“	Die Funktion benötigt mehr als 100 kB Speicherplatz	Nicht verwendete Programme löschen
C7080	Parameter kann nicht gelesen werden.	Parameter existiert nicht oder Passwort ist nicht korrekt.	1. Korrekten Parameternamen eingeben 2. Korrektes Passwort eingeben
C7081	Parameter kann nicht geschrieben werden.	Parameter existiert nicht oder Passwort ist nicht korrekt.	
C7090	Parameterkommentar fehlerhaft	Die zulässige Zeichenzahl des Parameterkommentars ist überschritten	Der Betrieb wird nicht beeinflusst und kann fortgesetzt werden.
C7100	Kapazität des Erweiterungsspeichers erschöpft	Es sind noch Daten im Erweiterungsspeicher	Programme über die Teaching Box initialisieren, um alle Daten zu löschen
C7200*	Doppelte Vergabe eines Dateinamens für den Erweiterungsspeicher	Doppelte Vergabe eines Dateinamens für den Erweiterungsspeicher	
H7300*	Parameterdatei laden	Damit die Parameter wirksam werden, muss die Spannungsversorgung aus- und wieder eingeschaltet werden.	Spannungsversorgung aus- und wieder einschalten
C7310	Geänderte Variablen wurden nicht gespeichert	Spannungsausfall während der Programmausführung	Spannungsversorgung während der Programmausführung nicht ausschalten

Tab. A-1: Fehlercodes (25)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
L7311	Spannungsversorgung wurde beim Speichern einer Datei ausgeschaltet	Spannungsversorgung wurde beim Speichern einer Datei ausgeschaltet	Spannungsversorgung nicht während des Speichervorgangs ausschalten
L7330	Keine Umschaltung in den ROM-Modus möglich	Keine Änderung des Parameters RLNG oder der CTN-Einstellung im RAM-Modus möglich	RAM-Modus wählen und Vorgang wiederholen
L7331	Keine Änderung der Speicherkapazität im ROM-Modus möglich	Keine Änderung der Speicherkapazität im ROM-Modus möglich	
L7332	Bei aktivierter CTN-Einstellung keine Umschaltung in den ROM-Modus möglich	Bei aktivierter CTN-Einstellung „Programm fortsetzen“ keine Umschaltung in den ROM-Modus möglich	
L7340	Verwendung der CTN-Funktion im DRAM-Modus nicht möglich	Verwendung der CTN-Funktion im DRAM-Modus nicht möglich	
L7341	Bei aktivierter CTN-Einstellung keine Umschaltung in den DRAM-Modus möglich	Bei aktivierter CTN-Einstellung keine Umschaltung in den DRAM-Modus möglich	
L7342	Globale Erweiterung im DRAM-Modus nicht möglich	Globale Erweiterung im DRAM-Modus nicht möglich	
L7343	Im PRGGBL-Modus keine Umschaltung in den DRAM-Modus möglich	Im PRGGBL-Modus keine Umschaltung in den DRAM-Modus möglich	
C7410	Monatliche Inspektion fällig	Monatliches Inspektionsintervall abgelaufen	
C7420	3-monatliche Inspektion fällig	3-monatliches Inspektionsintervall abgelaufen	Führen Sie die 3-monatliche Inspektion durch
C7430	6-monatliche Inspektion fällig	6-monatliches Inspektionsintervall abgelaufen	Führen Sie die 6-monatliche Inspektion durch
C7440	Jahresinspektion fällig	12-monatliches Inspektionsintervall abgelaufen	Führen Sie die Jahresinspektion durch
C7500	Keine Batteriespannung	Batterie im Steuergerät leer	Batterie austauschen und Daten laden
C7510	Batteriespannung zu niedrig	Batterie im Steuergerät leer	Batterie austauschen (siehe Technisches Handbuch)
C7520	Gebrauchszeit der Batterie überschritten	Batterie leer	
C7530	Schmiermittel erneuern	Schmiermittel verbraucht	Schmiermittel erneuern
C7540	Zahnriemen erneuern	Lebensdauer des Zahnriemens abgelaufen	Zahnriemen austauschen
H7600*	Fehlerhafter Parameter (AXMENO)	Die Nummer des Mechanismus der Zusatzachse ist fehlerhaft.	Einstellung der Parameter AXMENO und AXUNUM korrigieren
H7601*	Fehlerhafter Parameter (AXJNO)	Die Nummer der Zusatzachse ist fehlerhaft.	Einstellung des Parameters AXJNO korrigieren
H7602*	Fehlerhafter Parameter (AXJNO)	Die zugewiesene Zusatzachsennummer wird bereits verwendet.	
H7603*	Fehlerhafter Parameter (AXUNT)	Die Einheit für die Zusatzachse ist fehlerhaft.	Einstellung des Parameters AXUNT korrigieren
H7604*	Fehlerhafter Parameter (AXACC)	Die Beschleunigungszeit der Zusatzachse ist nicht korrekt eingestellt.	Einstellung des Parameters AXACC korrigieren

Tab. A-1: Fehlercodes (26)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
H7605*	Fehlerhafter Parameter (AXDEC)	Die Abbremszeit der Zusatzachse ist nicht korrekt eingestellt.	Einstellung des Parameters AXDEC korrigieren
H7606*	Fehlerhafter Parameter (AXGRTN)	Die Einstellung des Nenners zur Festlegung des Übersetzungsverhältnisses der Zusatzachse ist nicht korrekt.	Einstellung des Parameters AXGRTN korrigieren
H7607*	Fehlerhafter Parameter (AXGRTD)	Die Einstellung des Zählers zur Festlegung des Übersetzungsverhältnisses der Zusatzachse ist nicht korrekt.	Einstellung des Parameters AXGRTD korrigieren
H7608*	Fehlerhafter Parameter (AXMOTSET)	Der Freigabemerker des Zusatzachsenmotors ist nicht korrekt.	Einstellung des Parameters AXMOTSET korrigieren
H7609*	Fehlerhafter Parameter (AXMREV)	Die Nenngeschwindigkeit der Zusatzachse ist nicht korrekt.	Einstellung des Parameters AXMREV korrigieren
H7610*	Fehlerhafter Parameter (AXJMX)	Die Maximalgeschwindigkeit der Zusatzachse ist nicht korrekt.	Einstellung des Parameters AXJMX korrigieren
H7611*	Fehlerhafter Parameter (AXENCR)	Die Impulszahl des Encoders ist fehlerhaft eingestellt.	Einstellung des Parameters AXENCR korrigieren
H7612*	Fehlerhafter Parameter (AXJOGTS)	Die Einstellung der Zeitkonstanten der Zusatzachse für den JOG-Betrieb ist nicht korrekt.	Einstellung des Parameters AXJOGTS korrigieren
H7613*	Spannungsversorgung aus- und wieder einschalten	Spannungsversorgung aus- und wieder einschalten, da eine Einstellung des Mechanismus vorgenommen wurde	Spannungsversorgung aus- und wieder einschalten
H7620*	Es sind mehrer Schnittstellenkarten für Zusatzachsen installiert.	Es darf nur eine Schnittstellenkarte für Zusatzachsen installiert werden.	Nur eine Schnittstellenkarte für Zusatzachsen installieren.
C7630	Keine Schnittstellenkarten für Zusatzachsen	Es ist keine Schnittstellenkarten für Zusatzachsen installiert.	Schnittstellenkarte für Zusatzachsen installieren.
H7700*	Steckplatz für Option unzulässig (CC-Link)	In Steckplatz 1 oder 3 darf keine CC-Link-Schnittstellenkarte installiert werden.	CC-Link-Schnittstellenkarte in Steckplatz 2 installieren.
H7710*	Keine Einstellung der Master-Station möglich	CC-Link-Schnittstellenkarte des Roboters kann an der Master-Station nicht eingestellt werden.	Codierschalter auf einen Wert ungleich 0 stellen
H7720*	Es sind mehrere CC-Link-Schnittstellenkarten installiert.	Es darf nur eine CC-Link-Schnittstellenkarte installiert werden.	Nur eine CC-Link-Schnittstellenkarte installieren.
L7730	CC-Link-Datenübertragungsfehler	Fehler in der Kommunikationsleitung oder ungültige Parametereinstellung der Master-Station	Kommunikationsleitung und Parametereinstellung der Master-Station prüfen
L7731	CC-Link-Datenübertragungsfehler		
L7750	Leitung unterbrochen oder fehlerhafter Parameter (CC-Link)	Die Leitung ist unterbrochen oder ein Parameter der Master-Station ist nicht korrekt eingestellt.	Kommunikationsleitung prüfen oder Spannungsversorgung aus- und wieder einschalten und neu starten

Tab. A-1: Fehlercodes (27)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
H7760*	CC-Link-Initialisierungsfehler	Fehlerhafte Eistellung der Parameter der Master-Station	Parameter korrigieren und neu starten
L7780	CC-Link-Registernummer unzulässig	Die eingegebene Registernummer liegt außerhalb des zulässigen Einstellbereiches.	Geben Sie die korrekte Registernummer unter Berücksichtigung der Stationsnummer ein.
L7781	Die Eingangssignalnummer ist dem CC-Link-Betrieb zugeordnet	Es wurde eine Eingangssignalnummer für den CC-Link-Betrieb festgelegt, obwohl keine CC-Link-Schnittstellenkarte installiert ist.	CC-Link-Schnittstellenkarte installieren
H7799*	CC-Link-Systemfehler	CC-Link-Systemfehler	MITSUBISHI-Vertriebspartner kontaktieren
H7800	Fehler bei der Initialisierung der ETHERNET-Schnittstellenkarte	ETHERNET-Schnittstellenkarte defekt	ETHERNET-Schnittstellenkarte prüfen
H7802*	Steckplatz für Option unzulässig (ETHERNET)	In Steckplatz 2 oder 3 darf keine ETHERNET-Schnittstellenkarte installiert werden.	ETHERNET-Schnittstellenkarte in Steckplatz 1 installieren.
H7803*	Es sind mehrere ETHERNET-Schnittstellenkarten installiert.	Es darf nur eine ETHERNET-Schnittstellenkarte installiert werden.	Nur eine ETHERNET-Schnittstellenkarte installieren.
H7810	Parameterfehler ETHERNET (NETIP, NETGW, NETPORT, NETPROC, NETLOGIN, NETPSSWD, NETTOUTR, NETTOUTS, MXTCOM1-3)	Parametereinstellungen fehlerhaft	Parametereinstellungen korrigieren
H7820	Zeitüberschreitung Befehl MXT/MXS	Die im Parameter MXTTOUT eingestellte Zeite wurde überschritten	Wert des Parameters MXTTOUT vergrößern
H7830	Keine ETHERNET-Schnittstellenkarte installiert oder Befehl nicht ausführbar	Keine ETHERNET-Schnittstellenkarte installiert oder Befehl nicht ausführbar	ETHERNET-Schnittstellenkarte installieren
H7840	Empfangene Daten bei Ausführung des Befehls MXT/MXS ungültig	Befehlsparameter und Datentyp stimmen nicht überein	Befehl und gesendete Daten prüfen
H7901	Parameter OPTION#1 unzulässig	Parametereinstellungen fehlerhaft	Parametereinstellungen korrigieren und Spannungsversorgung aus- und wieder einschalten
H7902*	Parameter OPTION#2 unzulässig		
H7903	Parameter OPTION#3 unzulässig		
H7911	Parameter OPTION2#1 unzulässig		
H7912	Parameter OPTION2#2 unzulässig		
H7913	Parameter OPTION2#3 unzulässig		
H7920*	Es sind 3 EX-SIO-Schnittstellenkarten installiert.	Es dürfen maximal zwei EX-SIO-Schnittstellenkarten installiert werden.	Maximal zwei EX-SIO-Schnittstellenkarten installieren.
H7930*	Steckplatz für Option unzulässig (EX-SIO)	In Steckplatz 3 darf keine EX-SIO-Schnittstellenkarte installiert werden.	EX-SIO-Schnittstellenkarte in Steckplatz 1 oder 2 installieren.

Tab. A-1: Fehlercodes (28)

Fehlercode	Bedeutung	Ursache	Gegenmaßnahme
H7990*	Optionskarte fehlerhaft installiert	Nummer des Steckplatzes fehlerhaft	Spannungsversorgung ausschalten und Steckplatz wechseln
H8015*	Es sind mehrere Schnittstellenkarten zum Anschluss optischer Sensoren installiert	Es darf nur eine Schnittstellenkarte zum Anschluss optischer Sensoren installiert werden.	Nur eine Schnittstellenkarte zum Anschluss optischer Sensoren installieren
H8201	Fehlerhafter Parameter (TPTMMG)	Die Einstellung des Parameters TPTMMG ist fehlerhaft.	Einstellung des Parameters TPTMMG korrigieren.
H8202	Fehlerhafter Parameter (TPERRNO)	Die Einstellung des Parameters TPERRNO ist fehlerhaft.	Einstellung des Parameters TPERRNO korrigieren.
H8203	Fehlerhafter Parameter (TPSIGNO)	Die Einstellung des Parameters TPSIGNO ist fehlerhaft.	Einstellung des Parameters TPSIGNO korrigieren.
H8204	Fehlerhafter Parameter (TRAP)	Die Einstellung des Parameters TRAP ist fehlerhaft.	Einstellung des Parameters TRAP korrigieren.
H8206	Fehler bei der Initialisierung (Trap-Funktion)	Fehler bei der Initialisierung (Trap-Funktion)	Versorgungsspannung aus- und wieder einschalten Im Wiederholungsfall, MITSUBISHI-Vertriebspartner kontaktieren
H8207	Fehler beim Schreiben von Daten (Trap-Funktion)	Fehler beim Schreiben von Daten	Schreibvorgang wiederholen
H8208	Schreibkapazität des Flash-ROMs überschritten	Die zulässige Anzahl der Schreibzyklen von 100000 des Flash-ROMs wurden überschritten.	Trap-Funktion deaktivieren
L8300	Maximale Anzahl der GETPOS-Definitionen überschritten	GETPOS darf maximal 8-mal definiert werden	Gleiche Interrupt-Nummer verwenden oder ein nicht benötigtes Programm zurücksetzen
L8310	GETPOS nicht definiert	GETPOS nicht definiert	GETPOS definieren
L8320	Systemfehler (GETPOS)	Interne Daten der GETPOS-Funktion ungültig	Versorgungsspannung aus- und wieder einschalten
L8400	Ungültige CNT-Daten (PREC oder M_LDM)	Die CNT-Daten sind ungültig. Änderung auf PREC OFF und M_LDM = 0	PREC-Modus und M_LDM aktivieren
H9000 : H9099	Ein schwerer benutzerdefinierter Fehler ist aufgetreten.	Das Roboterprogramm hat einen schweren Fehler gemeldet.	Programm prüfen
L9100 : L9199	Ein leichter benutzerdefinierter Fehler ist aufgetreten.	Das Roboterprogramm hat einen leichten Fehler gemeldet.	
C9200 : C9299	Ausgabe einer benutzerdefinierten Warnung	Das Roboterprogramm hat eine Warnmeldung ausgegeben.	

Tab. A-1: Fehlercodes (29)

Index

A

ACL-Befehl	6-8
ACT-Befehl	6-10
Anweisung	5-1
angehängte	4-42
Aufbau	4-2
Ausgangsbitmuster	
festlegen	9-47
Ausgangssignale	4-35
Automatikbetrieb	3-36
Automatische Rückkehr	9-39

B

BASE-Befehl	6-12
Batterie	
anzeigen	3-56
Batteriezüher zurücksetzen	3-55
Befehle	
detaillierte Beschreibung	6-8
Übersicht	6-2
Benutzerdefinierter Bereich	9-36
Beschleunigungszeit	4-14
Betriebsrechte	2-10
Bremszeit	4-14

C

CALLP-Befehl	6-14
CHRSRCH-Befehl	6-16
CLOSE-Befehl	6-17
CLR-Befehl	6-18
CMP JNT-Befehl	6-20
CMP OFF-Befehl	6-28
CMP POS-Befehl	6-22
CMP TOOL-Befehl	6-25
CMPG-Befehl	6-29
CNT-Befehl	6-30
COLCHK-Befehl	6-33
COLLVL-Befehl	6-37
COM OFF-Befehl	6-39
COM ON-Befehl	6-40
COM STOP-Befehl	6-41

D

Datentypen	5-5
Datum	
einstellen	3-57
DEF ACT-Befehl	6-42
DEF ARCH-Befehl	6-45
DEF CHAR-Befehl	6-47
DEF DOUBLE-Befehl	6-50
DEF FLOAT-Befehl	6-50
DEF FN-Befehl	6-48
DEF INTE-Befehl	6-50
DEF IO-Befehl	6-52
DEF JNT-Befehl	6-54
DEF PLT-Befehl	6-55
DEF POS-Befehl	6-58
DIM-Befehl	6-59
DLY-Befehl	6-61

E

Ein-/Ausgänge	
externe	10-1
Einführung	1-1
Eingangssignale	4-34
Einschaltzeit	
anzeigen	3-56
END-Befehl	6-64
ERROR-Befehl	6-63
Externe Signale	
ein- und ausgeben	4-34
Zeitablaufdiagramme	10-22

F

Fehler	
anzeigen	3-51
Codes	A-2
Diagnose	A-1
Übersicht	A-2
zurücksetzen	3-34
Feinpositionierung	4-17
FINE-Befehl	6-65
FOR-NEXT-Befehl	6-67

FPRM-Befehl	6-69
Funktionen	
allgemeine Hinweise	8-1
detaillierte Beschreibung	8-2
fest definierte.	5-24
Steuergerät	2-1
Teaching Box	2-5

G

Gelenkbremsen	
lösen	3-54
Gelenk-Interpolation	4-5
Geschwindigkeit	4-14
GETM-Befehl	6-70
GOSUB-Befehl	6-72
GOTO-Befehl	6-73

H

Handgreifer	
ausrichten	3-15
öffnen und schließen.	3-13
steuern	4-21
Zustand nach Initialisierung	9-45
HCLOSE-Befehl	6-75
Highspeed-RAM-Modus.	9-58
HLT-Befehl	6-74
HOPEN-Befehl	6-75

I

IF ... THEN ... ELSE-Befehl	6-77
INPUT#-Befehl	6-80
Interrupt	4-30

J

JOG-Betrieb	3-4
JOVRD-Befehl.	6-81
JRC-Befehl	6-82

K

Kollisionsüberwachung	6-33
Kommunikation	
Einstellungen.	9-49
Funktionen	4-36
Konstanten	
Einteilung	5-6
Gelenkkonstanten	5-10
Positionskonstanten.	5-7
Zeichenkettenkonstanten	5-7
Kontinuierliche Bewegung.	4-12
Kreis-Interpolation.	4-10

L

LABEL-Befehl	6-85
Linear-Interpolation	4-7
LOADSET-Befehl	6-86
Logische Werte	
Beschreibung	5-23

M

Marken	5-2
MELFA-BASIC IV	
Ausdrücke	4-38
Befehle	6-1
Begriffserklärung	5-1
Funktionsübersicht	4-1
Operationen	4-38
Menüpunkt	
auswählen	3-2
Monitor-Funktion	
Ausgangssignale.	3-49
Eingangssignale	3-48
Fehlermeldungen	3-51
Variable	3-50
MOV-Befehl	6-88
Multirotationsdaten	5-9
ändern	8-40

Multitask-Funktion	
Anwendung	4-50
Anwendungsbeispiel	4-51
Ausführung	4-44
Beschreibung	4-43
Programm erstellen	4-48
Programmplatzparameter	4-46
MVA-Befehl	6-90
MVC-Befehl	6-92
MVR2-Befehl	6-96
MVR3-Befehl	6-98
MVR-Befehl	6-94
MVS-Befehl	6-100

N

NOT-HALT-Schalter	
Anschluss	10-36

O

OADL-Befehl	6-103
ON ... GOTO-Befehl	6-109
ON COM GOSUB-Befehl	6-105
ON-GOSUB-Befehl	6-107
OPEN-Befehl	6-111
Optimale Beschleunigung/Abbremsung	
Hand- und Werkstückbedingung	9-53
OVRD-Befehl	6-113

P

Palettierungsfunktion	4-23
Parallele Ein-/Ausgangsschnittstelle	10-3
Parameter	
anwendungsspezifische Einteilung	9-1
einstellen	3-52
spezielle	10-9
PLT-Befehl	6-115
Positionsdaten	
ändern	3-25
anfahren	3-22
eingeben	3-21
ersetzen	3-24
löschen	3-26
PREC-Befehl	6-118
PRINT-Befehl	6-119

PRIORITY-Befehl	6-121
Programm	
alle löschen	3-53
Aufbau	4-2
auswählen	3-37
editieren	3-19
erstellen	3-16
kopieren	3-45
löschen	3-47
Namen ändern	3-46
schützen	3-43
Start nach Einschalten	9-42
testen	3-28
Verzeichnis anzeigen	3-42
Programmebenen	5-30
Programmiertechniken	
für Einsteiger	11-2
für Fortgeschrittene	11-35
für fortgeschrittene Einsteiger	11-23
Übersicht	11-1
Programmplatzparameter	
Beschreibung	4-46
Programmschleife	4-29
Programmsteuerung	
externe Signale	10-32
Verzweigungen	4-27

R

RAM-Modus	9-58
RELM-Befehl	6-122
REM-Befehl	6-123
Reservierte Wörter	5-30
RESET ERR-Befehl	6-124
RETURN-Befehl	6-125
Roboterbewegung	
steuern	4-5
Roboterstatusvariablen	
allgemeine Hinweise	7-1
detaillierte Beschreibung	7-2
Übersicht	5-18
ROM-Modus	9-58

S

Schnittstelle	
parallele	10-3
serielle	9-50
SELECT CASE-Befehl	6-127
Serielle Ein-/Ausgangsschnittstelle	9-49
SERVO-Befehl	6-129
Servospannung	
ein-/ausschalten	3-31
Sicherheitshinweise	
grundlegende	1-1
Singulärer Punkt	
Beschreibung	6-102
Durchfahren	9-79
Fehlermeldung	9-57
SKIP-Befehl	6-130
SPD-Befehl	6-131
Stellungsmerker	5-8
ändern	8-38
Steuergerät	
Bedien- und Signalelemente	2-1
Stopp	4-33

T

Teaching Box	
Bedienelemente	2-7
Bedienung	3-1
Display-Anzeigen	2-5
Funktionen	2-5
Timer	4-32
TITLE-Befehl	6-133
TOOL-Befehl	6-134
TORQ-Befehl	6-136
TYPE-Festlegung	9-83

U

Uhrzeit	
einstellen	3-57
Unterprogramm	4-31

V

Variablen	
Ein-/Ausgangsvariablen	5-14
Einteilung	5-11
externe	5-15
Feldvariablen	5-14
Gelenkvariablen	5-13
numerische	5-12
Positionsvariablen	5-13
Roboterstatusvariablen	5-18
Zeichenkettenvariablen	5-13
Verfahrweggenauigkeit	4-19
Verfahrwegsgrenzen	
definieren	9-38
Verzweigungen	4-27

W

WAIT-Befehl	6-138
Warmlaufbetrieb	9-70
Wartezeit	4-27
Werkzeugdaten	
umschalten	3-11
WHILE ~ WEND-Befehl	6-139
WTH-Befehl	6-140
WTHIF-Befehl	6-141

X

XCLR-Befehl	6-142
XLOAD-Befehl	6-143
XRUN-Befehl	6-145
XSTP-Befehl	6-147

Z

Zeichen	
mit besonderer Bedeutung	5-4
Typen	5-3
Zeilen	5-2
Zeilennummern	5-2
Zeitablaufdiagramm	10-32

HEADQUARTERS

MITSUBISHI ELECTRIC EUROPE B.V.
 German Branch
 Gothaer Straße 8
D-40880 Ratingen
 Telefon: (02102) 486-0
 Telefax: (02102) 486-1120
 E-Mail: megfamail@meg.mee.com

MITSUBISHI ELECTRIC EUROPE B.V.
 French Branch
 25, Boulevard des Bouvets
F-92741 Nanterre Cedex
 Telefon: +33 1 55 68 55 68
 Telefax: +33 1 55 68 56 85
 E-Mail: factory.automation@fram.mee.com

MITSUBISHI ELECTRIC EUROPE B.V.
 Irish Branch
 Westgate Business Park, Ballymount
IRL-Dublin 24
 Telefon: +353 (0)1 / 419 88 00
 Telefax: +353 (0)1 / 419 88 90
 E-Mail: sales.info@meir.mee.com

MITSUBISHI ELECTRIC EUROPE B.V.
 Italian Branch
 Via Paracelso 12
I-20041 Agrate Brianza (MI)
 Telefon: +39 (0)39 / 60 53 1
 Telefax: +39 (0)39 / 60 53 312
 E-Mail: factory.automation@it.mee.com

MITSUBISHI ELECTRIC EUROPE B.V.
 Spanish Branch
 Carretera de Rubí 76-80
E-08190 Sant Cugat del Vallés (Barcelona)
 Telefon: +34 9 3 / 565 3160
 Telefax: +34 9 3 / 589 1579
 E-Mail: industrial@sp.mee.com

MITSUBISHI ELECTRIC EUROPE B.V.
 UK Branch
 Travellers Lane
GB-Hatfield Herts. AL10 8 XB
 Telefon: +44 (0)1707 276100
 Telefax: +44 (0)1707 278695
 E-Mail: automation@meuk.mee.com

MITSUBISHI ELECTRIC CORPORATION
 Office Tower "Z" 14 F
 8-12,1 chome, Harumi Chuo-Ku
Tokyo 104-6212
 Telefon: +81 3 6221 6060
 Telefax: +81 3 6221 6075

MITSUBISHI ELECTRIC AUTOMATION
 500 Corporate Woods Parkway
Vernon Hills, IL 60061
 Telefon: +1 847 / 478 21 00
 Telefax: +1 847 / 478 22 83

EUROPÄISCHE VERTRETUNGEN

Koning & Hartman b.v. BELGIEN
 Researchpark Zellik
 Pontbeeklaan 43
BE-1731 Brussels
 Telefon: +32 (0)2 / 467 17 51
 Telefax: +32 (0)2 / 467 17 45
 E-Mail: info@koningenhartman.com

Herstad + Piper A/S DÄNEMARK
 Jernholmen 48 C
DK-2650 Hvidovre
 Telefon: +45 (0)36 - 77 40 00
 Telefax: +45 (0)36 - 77 77 40
 E-Mail: mail@herstad-piper.dk

Beijer Electronics OY FINNLAND
 Ansatie 6a
FI-01740 Vantaa
 Telefon: +358 (0)9 / 886 77 500
 Telefax: +358 (0)9 / 886 77 555
 E-Mail: info@beijer.fi

Kouvalias GRIECHENLAND
 Robot + Vision Systems
 25, El. Venizelou Ave
GR-17671 Kallithea
 Telefon: +30 22950 / 42902/3/4
 Telefax: +30 22950 / 42690
 E-Mail: info@kouvalias.com

Koning & Hartman b.v. NIEDERLANDE
 Donauweg 2 B
NL-1000 AK Amsterdam
 Telefon: +31 (0)20 / 587 76 00
 Telefax: +31 (0)20 / 587 76 05
 E-Mail: info@koningenhartman.com

Beijer Electronics AS NORWEGEN
 Teglverksveien 1
NO-3002 Drammen
 Telefon: +47 (0)32 / 24 30 00
 Telefax: +47 (0)32 / 84 85 77
 E-Mail: info@beijer.no

GEVA ÖSTERREICH
 Wiener Straße 89
AT-2500 Baden
 Telefon: +43 (0)2252 / 85 55 20
 Telefax: +43 (0)2252 / 488 60
 E-Mail: office@geva.at

EUROPÄISCHE VERTRETUNGEN

MPL Technology Sp. z o.o. POLEN
 ul. Sliczna 36
PL-31-444 Kraków
 Telefon: +48 (0)12 / 632 28 85
 Telefax: +48 (0)12 / 632 47 82
 E-Mail: krakow@mpl.pl

Beijer Electronics AB SCHWEDEN
 Box 426
S-20124 Malmö
 Telefon: +46 (0)40 / 35 86 00
 Telefax: +46 (0)40 / 35 86 02
 E-Mail: info@beijer.se

ECONOTEC AG SCHWEIZ
 Postfach 282
CH-8309 Nürensdorf
 Telefon: +41 (0)1 / 838 48 11
 Telefax: +41 (0)1 / 838 48 12
 E-Mail: info@econotec.ch

INEA SR d.o.o. SERBIEN & MONTENEGRO
 Karadjordjeva 12/260
SCG-113000 Smederevo
 Telefon: +381 (0)26 / 617 163
 Telefax: +381 (0)26 / 617 163
 E-Mail: vladstoj@yubc.net

AutoCont Control s.r.o. SLOWAKEI
 Radlinského 47
SK-02601 Dolný Kubín
 Telefon: +421 435868 210
 Telefax: +421 435868 210
 E-Mail: info@autocontcontrol.sk

INEA d.o.o. SLOWENIEN
 Stegne 11
SI-1000 Ljubljana
 Telefon: +386 (0)1 513 8100
 Telefax: +386 (0)1 513 8170
 E-Mail: inea@inea.si

AutoCont TSCHECHISCHE REPUBLIK
 Control Systems s.r.o.
 Nemocnici 12
CZ-70200 Ostrava 2
 Telefon: +420 59 / 6152 111
 Telefax: +420 59 / 6152 562
 E-Mail: consys@autocont.cz

GTS TÜRKEI
 Darülaceze Cad. No. 43 Kat. 2
TR-80270 Okmeydani-Istanbul
 Telefon: +90 (0)212 / 320 1640
 Telefax: +90 (0)212 / 320 1649
 E-Mail: gts@turk.net

Axicont Automatika Kft. UNGARN
 Reitter F. U. 132
HU-1131 Budapest
 Telefon: +36 (0)1 / 412-0882
 Telefax: +36 (0)1 / 412-0883
 E-Mail: office@axicont.hu

KUNDEN-TECHNOLOGIE-CENTER DEUTSCHLAND

MITSUBISHI ELECTRIC EUROPE B.V.
 Kunden-Technologie-Center Nord
 Revierstraße 5
D-44379 Dortmund
 Telefon: (0231) 96 70 41-0
 Telefax: (0231) 96 70 41-41

MITSUBISHI ELECTRIC EUROPE B.V.
 Kunden-Technologie-Center
 Süd-West
 Kurze Straße 40
D-70794 Filderstadt
 Telefon: (0711) 77 05 98 0
 Telefax: (0711) 77 05 98 79

MITSUBISHI ELECTRIC EUROPE B.V.
 Kunden-Technologie-Center
 Süd-Ost
 Am Söldnermoos 8
D-85399 Hallbergmoos
 Telefon: (0811) 99 87 40
 Telefax: (0811) 99 87 410

VERTRETUNGEN EURASIEN

ELEKTROSTYLE RUSSLAND
 Poslannikov Per., 9, Str.1
RU-107005 Moscow
 Telefon: +7 095 / 542-4323
 Telefax: +7 095 / 956-7526
 E-Mail: info@estl.ru

ELEKTROSTYLE RUSSLAND
 Krasnij Prospekt 220-1, Office 312
RU-630049 Novosibirsk
 Telefon: +7 3832 / 106618
 Telefax: +7 3832 / 106626
 E-Mail: info@estl.ru

ICOS RUSSLAND
 Industrial Computer Systems Zao
 Ryazanskij Prospekt, 8A, Office 100
RU-109428 Moscow
 Telefon: +7 095 232 0207
 Telefax: +7 095 232 0327
 E-Mail: mail@icos.ru

VERTRETUNG MITTLERER OSTEN

Ilan & Gavish Ltd ISRAEL
 Automation Service
 24 Shenkar St., Kiryat Arie
IL-49001 Petach-Tiqva
 Telefon: +972 (0 3 / 922 18 24
 Telefax: +972 (0 3 / 924 07 61
 E-Mail: iandg@internet-zahav.net

VERTRETUNG AFRIKA

CBI Ltd SÜDAFRIKA
 Private Bag 2016
ZA-1600 Isando
 Telefon: +27 (0 11 / 928 2000
 Telefax: +27 (0 11 / 392 2354
 E-Mail: cbi@cbi.co.za