

# MELFA

## Industrieroboter

### Bedienungs- und Programmieranleitung

# RV-EN

**Bedienungs- und Programmieranleitung  
Industrieroboter MOVEMASTER RV-EN  
Artikel-Nr.: 69134 B**

Version		Änderungen / Ergänzungen / Korrekturen	
A	03/1998 pdp	—	
B	01/2001 pdp	Abs. 2.5.19:	Neue Funktion „Löschen des Batteriezählers“
		Abs. 5.2.35:	Funktionsbeschreibung des Befehls MA
		Tab. 6-2:	Beschreibung Bedienschritt ⑦
		Tab. A-18:	Korrektur der Bemerkung für Parameter XTL
			Korrektur der Standardwerte des Parameters JAR
			Einstellwerte des Parameters GDIR

# **Zu diesem Handbuch**

Die in diesem Handbuch vorliegenden Texte, Abbildungen, Diagramme und Beispiele dienen ausschließlich der Erläuterung zur Installation, Bedienung und Betrieb des Industrieroboters MOVEMASTER RV-EN.

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.

Ohne vorherige ausdrückliche schriftliche Genehmigung der MITSUBISHI ELECTRIC EUROPE B.V. dürfen keine Auszüge dieses Handbuchs vervielfältigt, in einem Informationssystem gespeichert oder weiter übertragen werden.

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

© 01/2001



# 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 Roboter RV-EN sind nur für die Einsatzbereiche vorgesehen, die in diesem Handbuch beschrieben sind. Achten Sie auf die Einhaltung aller im Handbuch angegebenen Kenndaten.

Jede andere darüberhinausgehende 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.***

Darüberhinaus 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 durch elektrische Spannung besteht, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.*



**ACHTUNG:**

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

# Inhaltsverzeichnis

<b>1</b>	<b>Einführung</b>	
1.1	Grundlegende Sicherheitshinweise .....	1-1
1.2	Die ersten Schritte .....	1-3
1.3	Programmiermethoden .....	1-5
<b>2</b>	<b>Bedienung und Programmierung</b>	
2.1	Vorbereitungen zur Bedienung des Roboters .....	2-1
2.1.1	Allgemeine Vorgehensweise bei der Programmierung .....	2-1
2.1.2	Gerätekonfiguration für die Programmierung .....	2-2
2.2	Bedienung des Steuergerätes .....	2-3
2.2.1	Gerätebeschreibung .....	2-3
2.2.2	Gerätefunktionen .....	2-7
2.3	Gerätebeschreibung der Teaching Box .....	2-9
2.4	Roboterprogramm testen .....	2-13
2.5	Bedienung der Teaching Box .....	2-14
2.5.1	Roboter im Jog-Betrieb bewegen .....	2-14
2.5.2	Jog-Geschwindigkeit einstellen .....	2-20
2.5.3	Handgreifer öffnen/schließen .....	2-22
2.5.4	Menübaum .....	2-24
2.5.5	Menüpunkt auswählen .....	2-25
2.5.6	Servospannung ein-/auschalten .....	2-26
2.5.7	Roboterprogramm starten .....	2-27
2.5.8	Programm und Roboterbewegung stoppen .....	2-30
2.5.9	Programmverzeichnis anzeigen .....	2-31
2.5.10	Programm schützen .....	2-32
2.5.11	Roboterprogramm kopieren .....	2-33
2.5.12	Programmname ändern .....	2-35
2.5.13	Roboterprogramm löschen .....	2-36
2.5.14	Monitor-Funktion für Eingangssignale .....	2-37
2.5.15	Monitor-Funktion für Ausgangssignale .....	2-38
2.5.16	Monitor-Funktion für Variablen .....	2-39
2.5.17	Liste der aufgetretenen Fehlermeldungen .....	2-40
2.5.18	Parameter anzeigen/einstellen .....	2-41
2.5.19	Alle gespeicherten Programme löschen .....	2-43

2.5.20	Gelenkbremsen lösen .....	2-44
2.5.21	Encoder zurücksetzen .....	2-46
2.5.22	Batterie und Einschaltzeit anzeigen .....	2-48
2.5.23	Uhrzeit/Datum einstellen .....	2-49
2.5.24	Fehlermeldung quittieren .....	2-51

### **3 Programmiermethoden**

3.1	Einteilung der Programmiermethoden .....	3-1
3.2	Eigenschaften der Programmiermethoden .....	3-2
3.3	Parametereinstellungen .....	3-3
3.3.1	Auswahl der Programmiermethode .....	3-3
3.3.2	Hinweise zur Auswahl der Programmiermethode .....	3-3

### **4 MOVEMASTER COMMAND-Programmierung**

4.1	Programmierung mit der Teaching Box .....	4-1
4.1.1	Roboterprogramm erstellen .....	4-2
4.1.2	Roboterprogramm editieren .....	4-4
4.1.3	Positionsdaten eingeben, anfahren, angleichen, ändern und löschen ..	4-8
4.1.4	Programmzeile direkt aufrufen .....	4-9

### **5 MOVEMASTER-Befehle**

5.1	Allgemeine Hinweise .....	5-1
5.1.1	Gruppeneinteilung der Befehle .....	5-1
5.1.2	Hinweise zu den weiteren Befehlsbeschreibungen .....	5-2
5.1.3	Schutzmaßnahmen bei der Programmierung und Eingabe über die Teaching Box .....	5-6
5.1.4	Vorsichtsmaßnahmen beim Testen eines Programms .....	5-7
5.2	Übersicht der Befehle .....	5-8
5.2.1	ADD (Add) .....	5-11
5.2.2	AN (And) .....	5-12
5.2.3	CF (Change Figure) .....	5-13
5.2.4	CL (Counter Load) .....	5-16
5.2.5	CP (Compare Counter) .....	5-18
5.2.6	CR (Counter Read) .....	5-20
5.2.7	DA (Disable Act) .....	5-22
5.2.8	DC (Decrement Counter) .....	5-23
5.2.9	DIV (Division) .....	5-24
5.2.10	DJ (Draw Joint) .....	5-25



5.2.11	DL ♦ (Delete Line)	5-26
5.2.12	DP (Decrement Position)	5-27
5.2.13	DR (Data Read)	5-28
5.2.14	DS (Draw Straight)	5-30
5.2.15	DW (Draw)	5-32
5.2.16	EA (Enable Act)	5-34
5.2.17	ED (End)	5-36
5.2.18	EQ (Equal)	5-37
5.2.19	ER ♦ (Error Read)	5-39
5.2.20	GC (Grip Close)	5-41
5.2.21	GF (Grip Flag)	5-43
5.2.22	GO (Grip Open)	5-44
5.2.23	GS (Go Sub)	5-45
5.2.24	GT (Go To)	5-47
5.2.25	HE (Here)	5-48
5.2.26	HLT (Halt)	5-49
5.2.27	HO (Home)	5-50
5.2.28	IC (Increment Counter)	5-51
5.2.29	ID (Input Direct)	5-52
5.2.30	INP (Input)	5-53
5.2.31	IP (Increment Position)	5-55
5.2.32	JRC (Joint Roll Change)	5-56
5.2.33	LG (If Larger)	5-57
5.2.34	LR ♦ (Line Read)	5-59
5.2.35	MA (Move Approach)	5-61
5.2.36	MC (Move Continuous)	5-63
5.2.37	MJ (Move Joint)	5-65
5.2.38	ML (Move Linear)	5-67
5.2.39	MO (Move)	5-68
5.2.40	MP (Move Position)	5-69
5.2.41	MPB (Move Playback)	5-71
5.2.42	MPC (Move Playback Continuous)	5-74
5.2.43	MR (Move R)	5-76
5.2.44	MRA (Move R A)	5-78
5.2.45	MS (Move Strait)	5-80
5.2.46	MT (Move Tool)	5-82
5.2.47	MTS (Move Tool Straight)	5-84
5.2.48	MUL (Multiplication)	5-86
5.2.49	N ♦ (Number)	5-87
5.2.50	NE (If Not Equal)	5-88
5.2.51	NT (Nest)	5-90

5.2.52	NW ♦ (New)	5-91
5.2.53	NX (Next)	5-92
5.2.54	OB (Output Bit)	5-93
5.2.55	OC (Output Counter)	5-94
5.2.56	OD (Output Direct)	5-95
5.2.57	OG (Origin)	5-96
5.2.58	OPN (Open)	5-97
5.2.59	OR (Or)	5-98
5.2.60	OVR (Override)	5-99
5.2.61	PA (Pallet Assign)	5-100
5.2.62	PC ♦ (Position Clear)	5-101
5.2.63	PD (Position Define)	5-102
5.2.64	PL (Position Load)	5-104
5.2.65	PMR (Parameter Read)	5-105
5.2.66	PMW (Parameter Writing)	5-106
5.2.67	PR (Position Read)	5-107
5.2.68	PRN ♦ (Print)	5-109
5.2.69	PT (Pallet)	5-111
5.2.70	PW (Pulse Wait)	5-112
5.2.71	PX (Position Exchange)	5-114
5.2.72	QN (Question Number)	5-115
5.2.73	RC (Repeat Cycle)	5-116
5.2.74	RN ♦ (Run)	5-117
5.2.75	RS ♦ (Reset)	5-119
5.2.76	RT (Return)	5-120
5.2.77	SC (Set Counter)	5-121
5.2.78	SD (Speed Define)	5-123
5.2.79	SF (Shift)	5-125
5.2.80	SM (If Smaller)	5-126
5.2.81	SP (Speed)	5-128
5.2.82	STR ♦ (Step Read)	5-131
5.2.83	SUB (Subtraction)	5-133
5.2.84	TB (Test Bit)	5-134
5.2.85	TBD (Test Bit Direct)	5-135
5.2.86	TI (Timer)	5-136
5.2.87	TL (Tool)	5-137
5.2.88	VR (Version Read)	5-138
5.2.89	WH (Where)	5-139
5.2.90	WT (What Tool)	5-141
5.2.91	XO (Exclusive Or)	5-142
5.2.92	' (Comment)	5-143

5.3	Beispiele mit MOVEMASTER-Befehlen . . . . .	5-144
5.3.1	Allgemeine Hinweise . . . . .	5-144
5.3.2	1. Beispiel: Arbeitsgegenstand ergreifen und platzieren . . . . .	5-145
5.3.3	2. Beispiel: Verfahrbewegung über externes Signal unterbrechen . . .	5-147
5.3.4	3. Beispiel: Arbeitsgegenstände palettieren . . . . .	5-149
5.3.5	4. Beispiel: Anschluß externer Ein-/Ausgabegeräte . . . . .	5-155

## **6 MELFA-BASIC III-Programmierung**

6.1	Programmierung mit der Teaching Box . . . . .	6-1
6.1.1	Roboterprogramm erstellen . . . . .	6-2
6.1.2	Roboterprogramm editieren . . . . .	6-5
6.1.3	Positionsdaten eingeben, anfahren, ersetzen, ändern und löschen . . .	6-9

## **7 MELFA-BASIC III**

7.1	Begriffserklärung . . . . .	7-1
7.1.1	Anweisung . . . . .	7-1
7.1.2	Angehängte Anweisung . . . . .	7-1
7.1.3	Zeilen . . . . .	7-2
7.1.4	Zeilennummern und Marken . . . . .	7-2
7.1.5	Zeichentypen . . . . .	7-3
7.1.6	Zeichen mit besonderer Bedeutung . . . . .	7-4
7.1.7	Datentypen . . . . .	7-5
7.1.8	Konstanten . . . . .	7-6
7.1.9	Variablen . . . . .	7-10
7.1.10	Feldvariablen . . . . .	7-16
7.1.11	Programmexterne Variablen . . . . .	7-17
7.1.12	Logische Werte . . . . .	7-23
7.1.13	Komponentendaten . . . . .	7-24
7.2	Ausdrücke und Operationen . . . . .	7-25
7.2.1	Gruppeneinteilung der Ausdrücke und Operationen . . . . .	7-25
7.2.2	Arithmetische Operationen . . . . .	7-25
7.2.3	Vergleichsoperationen . . . . .	7-27
7.2.4	Logische Operationen . . . . .	7-28
7.2.5	Funktionen . . . . .	7-29
7.2.6	Konvertierte Datentypen . . . . .	7-32
7.2.7	Rangfolge von Operationen . . . . .	7-34
7.2.8	Programmebenen . . . . .	7-34
7.2.9	Reservierte Wörter . . . . .	7-35

**8 BASIC-Befehle**

8.1	Allgemeine Hinweise	8-1
8.2	Übersicht der MELFA-BASIC III-Befehle	8-2
8.2.1	ACL (Accelerate)	8-4
8.2.2	ACT (Act)	8-6
8.2.3	ALIGN (Align)	8-8
8.2.4	BASE (Base)	8-9
8.2.5	CALLP (Call P)	8-11
8.2.6	CLOSE (Close)	8-12
8.2.7	CNT (Control)	8-13
8.2.8	COM OFF (Communication OFF)	8-16
8.2.9	COM ON (Communication ON)	8-17
8.2.10	COM STOP (Communication STOP)	8-18
8.2.11	DACL (Deceleration)	8-19
8.2.12	DEF ACT (Define act)	8-21
8.2.13	DEF FN (Define function)	8-22
8.2.14	DEF PLT (Define pallet)	8-24
8.2.15	DIM (Dim)	8-26
8.2.16	DLY (Delay)	8-27
8.2.17	END (End)	8-28
8.2.18	FINE (Fine)	8-29
8.2.19	FOR-NEXT (For-Next)	8-31
8.2.20	FPRM (FPRM)	8-33
8.2.21	GOSUB (Go Subroutine)	8-34
8.2.22	GOTO (Go To)	8-35
8.2.23	HLT (Halt)	8-36
8.2.24	HND ♦♦ (Hand)	8-37
8.2.25	HRE ♦ (Here)	8-39
8.2.26	IF ... THEN ... ELSE (If Then Else)	8-40
8.2.27	IN ♦ (In)	8-41
8.2.28	INPUT# (Input)	8-42
8.2.29	JOVRD (J override)	8-43
8.2.30	LABEL ♦ (Label)	8-44
8.2.31	MOV (Move)	8-45
8.2.32	Movement Position ♦ (Movement Position)	8-46
8.2.33	MVC (Move C)	8-47
8.2.34	MVR (Move R)	8-49
8.2.35	MVR2 (Move R2)	8-51
8.2.36	MVS (Move S)	8-53
8.2.37	OADL (Optimum Acceleration/Deceleration)	8-55

8.2.38	ON COM GOSUB (ON Communication Go Subroutine) . . . . .	8-57
8.2.39	ON-GOSUB (ON GOSUB) . . . . .	8-58
8.2.40	ON ... GOTO (On Go To) . . . . .	8-59
8.2.41	OPEN (Open) . . . . .	8-60
8.2.42	ORG (Origin) . . . . .	8-62
8.2.43	OUT ♦♦ (Out) . . . . .	8-63
8.2.44	OVRD (Override) . . . . .	8-65
8.2.45	PLT ♦ (Pallet) . . . . .	8-67
8.2.46	PRINT# (Print) . . . . .	8-68
8.2.47	REM (Remarks) . . . . .	8-70
8.2.48	RETURN (Return) . . . . .	8-71
8.2.49	SKIP ♦♦ (Skip) . . . . .	8-72
8.2.50	SPD (Speed) . . . . .	8-73
8.2.51	STOP ♦♦ (Stop) . . . . .	8-74
8.2.52	SV (Servo) . . . . .	8-75
8.2.53	TOOL (Tool) . . . . .	8-77
8.2.54	WHILE ~ WEND (While End) . . . . .	8-78
8.2.55	WTH ♦ (With) . . . . .	8-80
8.2.56	WTHIF ♦ (With If) . . . . .	8-81
8.2.57	SUBSTITUTE (Substitute) . . . . .	8-82
8.3	Übersicht der SLIM-Befehle . . . . .	8-83
8.3.1	ACCEL (Accelerate) . . . . .	8-84
8.3.2	CHANGE (Change) . . . . .	8-85
8.3.3	DEFINT (Define Integer) . . . . .	8-86
8.3.4	DEFIO (Define I/O) . . . . .	8-87
8.3.5	DEFJNT (Define Joint) . . . . .	8-88
8.3.6	DEFPOS (Define Position) . . . . .	8-89
8.3.7	DELAY (Delay) . . . . .	8-90
8.3.8	DRIVE (Drive) . . . . .	8-91
8.3.9	GOHOME (Go Home) . . . . .	8-92
8.3.10	GRASP (Grasp) . . . . .	8-93
8.3.11	HALT (Halt) . . . . .	8-94
8.3.12	HAND (Hand) . . . . .	8-95
8.3.13	HOLD (Hold) . . . . .	8-96
8.3.14	IN (Input) . . . . .	8-97
8.3.15	INPUT (Input) . . . . .	8-98
8.3.16	IOBLOCK (I/O Block) . . . . .	8-99
8.3.17	JSPEED (J Speed) . . . . .	8-100
8.3.18	MOVE (Move) . . . . .	8-101
8.3.19	OUT (Output) . . . . .	8-104
8.3.20	PRINT (Print) . . . . .	8-105

8.3.21	RELEASE (Release) . . . . .	8-106
8.3.22	RESET (Reset) . . . . .	8-107
8.3.23	SET (Set) . . . . .	8-108
8.3.24	SPEED (Speed) . . . . .	8-109
8.3.25	WAIT (Wait) . . . . .	8-110

## **A Anhang**

A.1	Übersicht der Befehle . . . . .	A-1
A.1.1	MOVEMASTER . . . . .	A-1
A.1.2	MELFA-BASIC III . . . . .	A-11
A.1.3	SLIM . . . . .	A-16
A.2	Übersicht der Parameter . . . . .	A-19
A.3	Übersicht der Fehlercodes . . . . .	A-23
A.4	Störungssuche . . . . .	A-28
A.5	Stellungsmerker . . . . .	A-30
A.5.1	Definition der einzelnen Stellungsmerker . . . . .	A-30

# 1 Einführung

## 1.1 Grundlegende Sicherheitshinweise

Der MOVEMASTER Roboter ist nach dem neuesten Stand der Technik gebaut und betriebs-sicher ausgeführt. Ungeachtet dessen können von dem Roboter Gefahren ausgehen, wenn er nicht von geschultem oder zumindest eingewiesenem Personal betrieben wird oder unsach-gemäß 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 Ausschaltproze-duren 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 beeinträch-tigt.*

*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.***



## 1.2 Die ersten Schritte

Nachfolgend erhalten Sie eine Darstellung der ersten Schritte mit Ihrem MOVEMASTER ROBOTER vom Typ RV-EN.

- ① **Roboter und Drive Unit auspacken**
- ② **Sicherheitstechnisches Handbuch lesen**  
Vor der ersten Inbetriebnahme des Robotersystems lesen Sie das Sicherheitstechnische Handbuch.
- ③ **Batterien anklemmen**  
Klemmen Sie zunächst die im hinteren Teil des Roboterarms unter der Abdeckklappe befindlichen Batterien an. Nähere Einzelheiten dazu entnehmen Sie dem Technischen Handbuch.
- ④ **Kabel anschließen**  
Verbinden Sie alle Kabel wie im technischen Handbuch beschrieben, und schließen Sie die Teaching Box an.
- ⑤ **Netzspannungsversorgung einschalten**  
Schalten Sie die Netzspannungsversorgung für das Steuergeräte über den POWER-Schalter ein.
- ⑥ **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.
- ⑦ **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 (ENBL/DISABLE-Schalter auf ENBL).
  - System einstellen (DATA-Methode)  
Zur Abgleichung des Systems muß der Origin-Punkt 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.
- ⑧ **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.  
  
Betätigen Sie die Tastenkombination [STEP/MOVE]+[ADD], um in den MOVEMASTER COMMAND-Modus zu gelangen. Hierdurch wird der Teach-Modus für die Erstellung von Positionsdaten aufgerufen.
- ⑨ **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.
- ⑩ **Roboter bewegen**  
Betätigen Sie die [STEP/MOVE]-Taste und halten Sie diese gedrückt. Wenn Sie nun eine der mittleren Tasten für die Achsenbewegung betätigen, wird sich der Roboter in der entsprechenden Achse bewegen.

**⑪ 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 der Taste [+ /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.

**⑫ 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 Taste [+ /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.

## 1.3 Programmiermethoden

Der Bewegungsablauf des Roboters in einem Arbeitsprozeß erfolgt programmgesteuert. Zur Erstellung des Programms stehen zwei Programmiermethoden zur Verfügung:

- Programmierung mit MOVEMASTER COMMAND-Befehlen

Ein Roboterprogramm zur Steuerung des MOVEMASTER-Roboters wird mit Hilfe eines Personalcomputers und der Teaching Box erstellt. Die MOVEMASTER COMMAND-Programmierungsmethode ist besonders für Aufgaben geeignet, die Funktionen wie bedingte Verzweigungen, Programmunterbrechungen oder Palettierung erfordern. Nähere Angaben hierzu entnehmen Sie bitte den Kapiteln 3 und 4.

- Programmierung mit MELFA-BASIC III-Befehlen

Ein Roboterprogramm zur Steuerung des MOVEMASTER-Roboters wird mit Hilfe eines Personalcomputers und der Teaching Box erstellt. Mit Hilfe der MELFA-BASIC III-Programmierungsmethode lassen sich komplexere Programme als mit der MOVEMASTER COMMAND-Methode erstellen. Nähere Angaben hierzu entnehmen Sie bitte den Kapiteln 3 und 6.

Wählen Sie zur Steuerung des Roboters eine der Methoden mittels des Parameters „RLNG“ aus (siehe Abschnitt 3.3).



## 2 Bedienung und Programmierung

### 2.1 Vorbereitungen zur Bedienung des Roboters

In diesem Abschnitt wird die allgemeine Vorgehensweise bei der Programmierung und die hierfür benötigte weitere Geräteausstattung beschrieben.

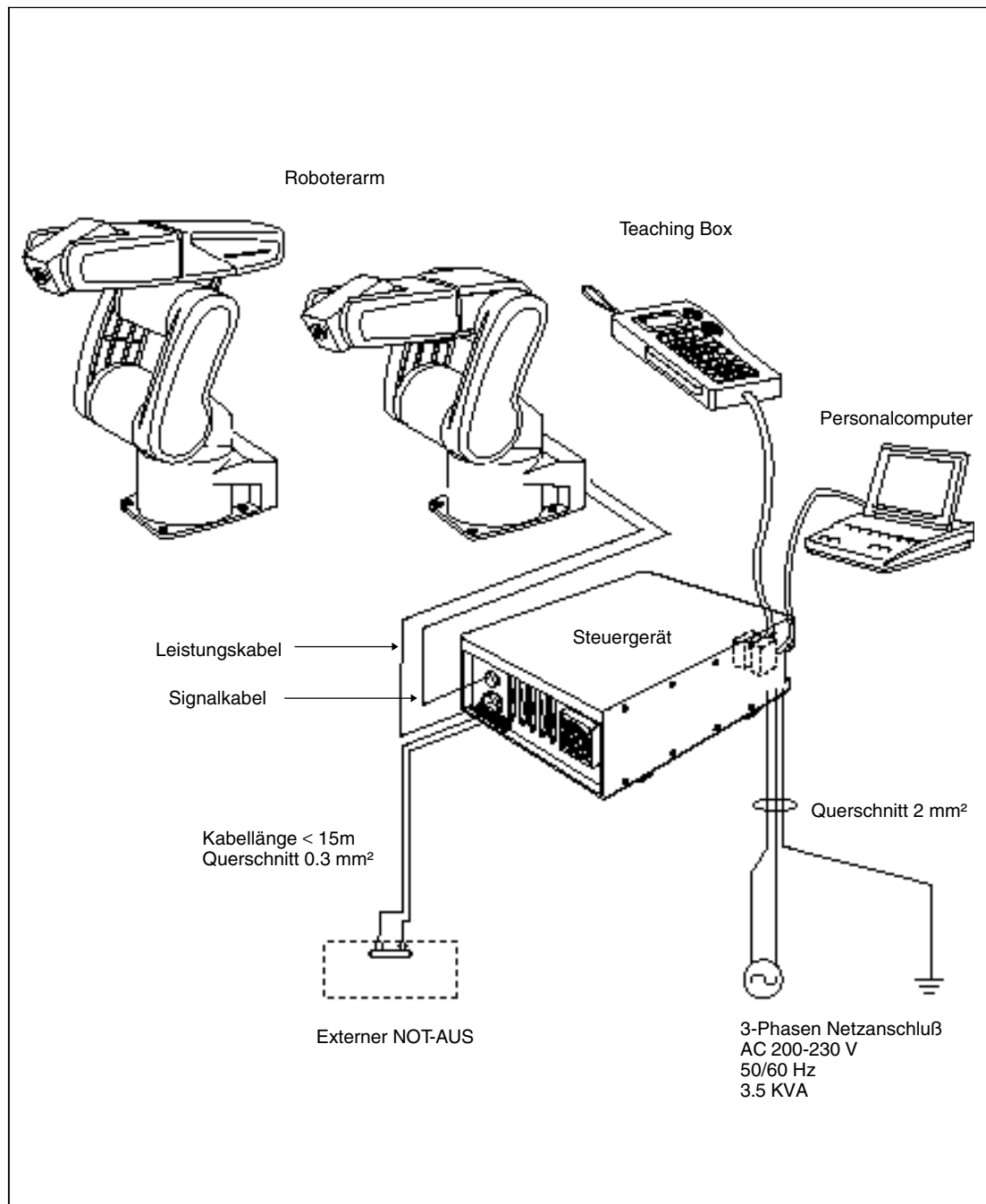
#### 2.1.1 Allgemeine Vorgehensweise bei der Programmierung

Nr.	Tätigkeit	Beschreibung
①	Programm erstellen	Beschreiben Sie die einzelnen Arbeitsschritte, die der Roboter ausführen soll. Erstellen Sie anhand der Arbeitsschritte ein Flußdiagramm für den Programmablauf. Wählen Sie anschließend aus den beiden nachfolgend aufgeführten Methoden eine der Aufgabenstellung entsprechende Programmiermethode aus. Es stehen zwei Programmiermethoden zur Verfügung: 1. Programmierung mit MOVEMASTER COMMAND-Befehlen (unter Verwendung eines Personalcomputers und der Teaching Box) 2. Programmierung mit MELFA-BASIC III-Befehlen (unter Verwendung eines Personalcomputers und der Teaching Box)
②	Programm testen	Vor dem erstmaligen Starten sollten Sie das Programm schrittweise oder zeilenweise ausführen, um die korrekte Arbeitsweise des Bewegungsablaufs, der Ein-/Ausgangssignale und der Zähler zu überprüfen.
③	Programm starten	Starten Sie das Programm im Automatikbetrieb.
④	Programm pflegen	Stellen Sie sicher, daß eine regelmäßige Programmpflege durchgeführt wird.

**Tab. 2-1:** Allgemeine Vorgehensweise bei der Programmierung

## 2.1.2 Gerätekonfiguration für die Programmierung

Abbildung 2.1 zeigt den Anschluß des Steuergerätes. Weitere Einzelheiten über den Anschluß der Peripherie entnehmen Sie den beiliegenden Handbüchern.

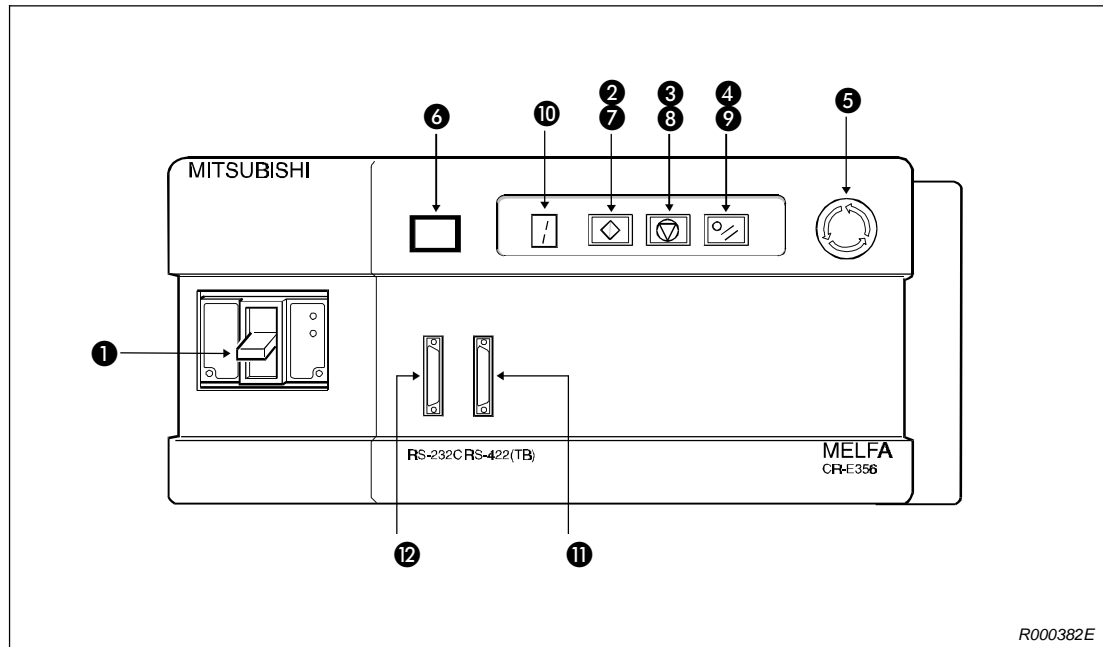


**Abb. 2-1:** Anschluß des Steuergerätes

## 2.2 Bedienung des Steuergerätes

### 2.2.1 Gerätebeschreibung

#### Frontansicht des Steuergeräts



**Abb. 2-2:** Frontansicht des Steuergerätes

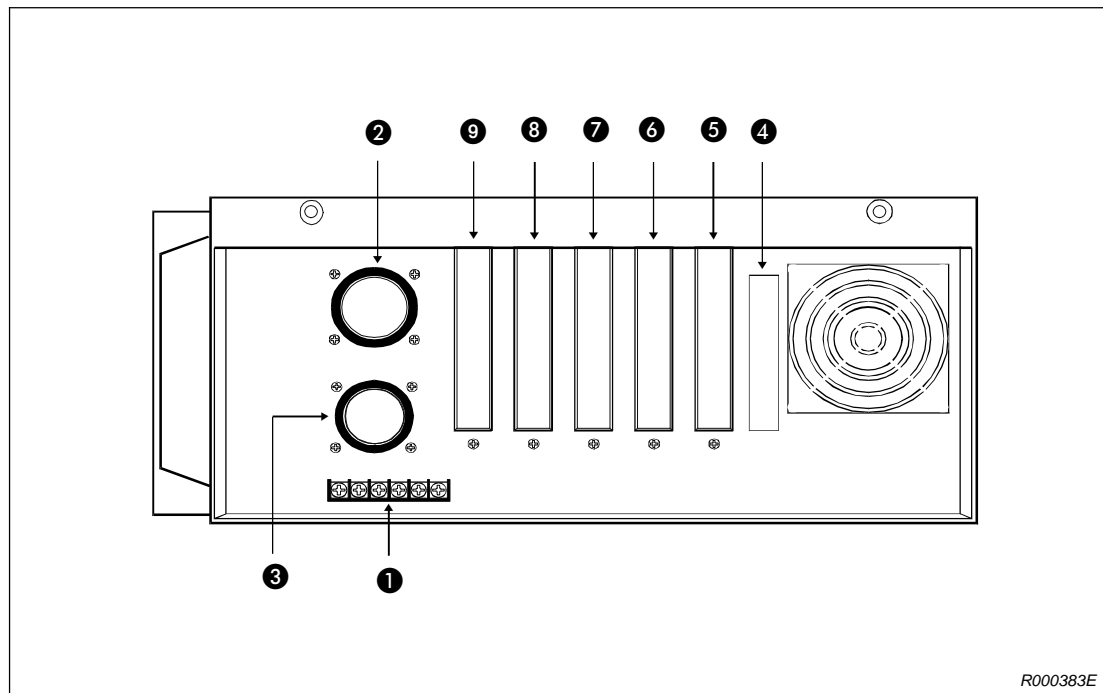
Nr.	Bezeichnung	Funktionsbeschreibung
①	POWER-Schalter	Spannungsversorgung ein-/ausschalten (Steuergerät ein-/ausschalten). Das Einschalten der Spannung für die Servomotoren erfolgt unmittelbar nach dem Einschalten der Spannungsversorgung über den POWER-Schalter.
②	START-Taste	Programm starten oder neustarten. Nach einmaliger Betätigung wird der Roboter im Zyklusmodus betrieben. Nach nochmaliger Betätigung wird der Roboter im Dauermodus betrieben.
③	STOP-Taste	Programm und Roboterbewegung stoppen. Nach einer Betätigung wird die Programmausführung gestoppt und die Roboterbewegung bis zum Stillstand abgebremst (die Tastenfunktion ist in jedem Betriebszustand wirksam).
④	RESET-Taste	Setzt ein zuvor gestopptes Programm zurück. Die Programmverarbeitung startet erneut an der Anfangszeile (die LED der STOP-Taste wird ausgeschaltet). Eine eventuell anstehende Fehlermeldung wird quittiert und der zugehörige Fehlercode gelöscht.
⑤	EMG .STOP-Schalter (NOT-HALT-Schalter)	Roboterbewegung sofort stoppen (NOT-HALT). Nach einer Betätigung wird die Spannung für die Servomotoren ausgeschaltet, die Bremsen werden aktiviert und die Roboterbewegung wird hierdurch gestoppt.
⑥	T/B-EMG-CANCEL-Schalter	Funktion des NOT-HALT-Schalters der Teaching Box unwirksam schalten. Nach einer Betätigung wird die Anzeige-LED des Schalters ausgeschaltet. Das Steuergerät kann jetzt ohne eine angeschlossene Teaching Box bedient werden. Betätigen Sie den Schalter nochmals (Anzeige-LED = EIN), wenn das Steuergerät mit einer angeschlossenen Teaching Box bedient werden soll.

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

Nr.	Bezeichnung	Funktionsbeschreibung
⑦	START-LED (Programmablauf-LED)	LED leuchtet grün: Programm wird abgearbeitet blinkt: Betrieb im Zyklusmodus leuchtet kontinuierlich: Betrieb im Dauermodus erlischt: Programm abgeschlossen oder gestoppt
⑧	STOP-LED	LED leuchtet nach einem Programmstopp so lange, bis das Programm zurückgesetzt wird.
⑨	RESET-LED	LED leuchtet bei einer anstehenden Fehlermeldung und wird nach erfolgter Quittierung mit der RESET-Taste ausgeschaltet.
⑩	PRO/ALM-Anzeige	Bei einem störungsfreien Betriebsablauf wird der Programmname (0 bis 9 oder A bis Z) angezeigt. Ein Programmname, der länger als 2 Zeichen ist, wird mit „-“ gekennzeichnet. Bei einer Störung wird der entsprechende Fehlercode angezeigt. Das Steuergerät kann bedient werden, wenn ein Punkt „.“ angezeigt wird.
⑪	RS422-Schnittstelle	Zum Anschluß der Teaching Box
⑫	RS232C-Schnittstelle	Zum Anschluß eines Personalcomputers

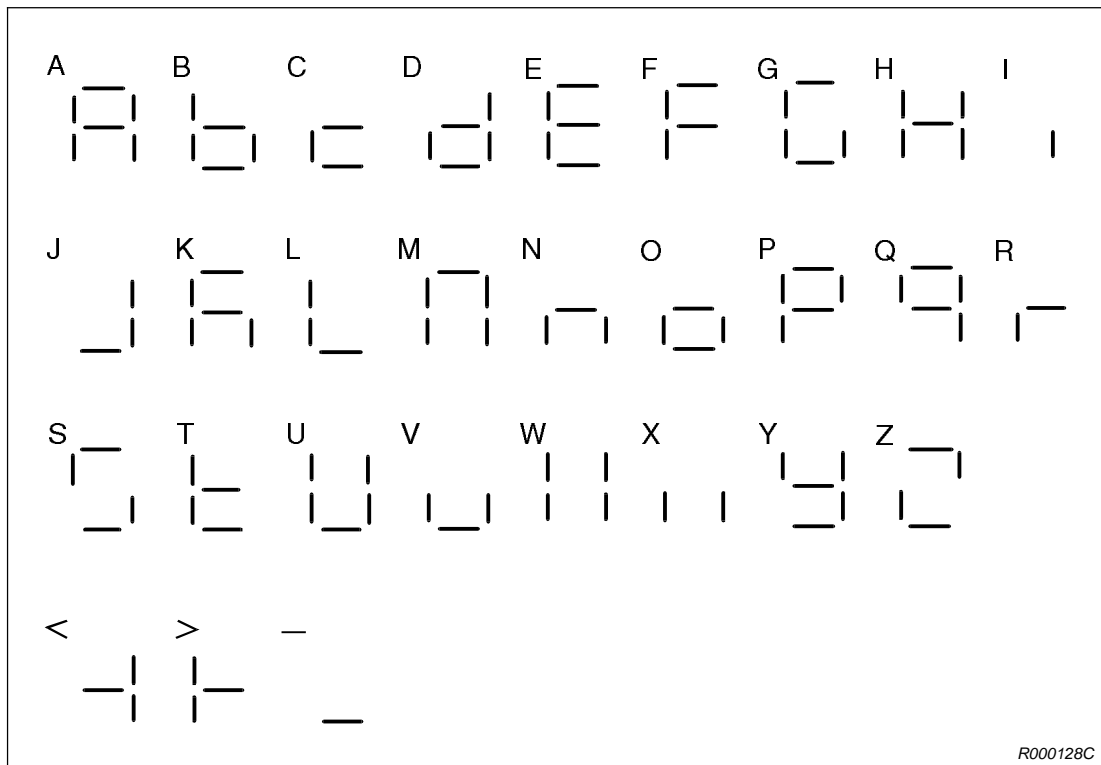
**Tab. 2-2:** Beschreibung der Bedien- und Signalelemente auf der Frontseite des Steuergerätes (2)



**Rückansicht des Steuergeräts****Abb. 2-3:** Rückansicht des Steuergeräts


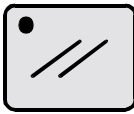
Nr.	Bezeichnung	Beschreibung
①	Klemmenleiste	Klemmen zum Anschluß an externe Sicherheitskreise, Ausgangssignal bei NOT-HALT
②	Anschluß für Leistungskabel	Zum Anschließen des Leistungskabels für die Servospannung zwischen Steuergerät und Roboterarm
③	Anschluß für Signalkabel	Zum Anschließen des Signalkabels für die Signalübertragung zwischen Steuergerät und Roboterarm
④	Einschubplatz für das Steuermodul zur Handsteuerung	Steckkartenplatz für das Hand-Steuermodul
⑤	1. freier Steckkartenplatz (OPT1)	Steckkartenplatz für die parallele E-/A-Schnittstellenkarte
⑥	2. freier Steckkartenplatz (OPT2)	Steckkartenplatz für Zusatzkarte (1. Karte)
⑦	3. freier Steckkartenplatz (OPT3)	Steckkartenplatz für Zusatzkarte (2. Karte)
⑧	nicht belegt	Steckkartenplatz für Erweiterungsfunktionen
⑨	Standardanschluß	Anschluß der standardmäßig eingebauten CPU

**Tab. 2-3:** Anschlüsse und Steckkartenplätze auf der Rückseite des Steuergerätes

**Alphanumerische Anzeige (PRO/ALM-Anzeige)****Abb. 2-4:** Darstellung von alphanumerischen Zeichen auf der PRO/ALM-Anzeige

## 2.2.2 Gerätefunktionen

### Programm auswählen

Funktion	Tastenbetätigungen	Beschreibung
Programme mit einstelligem Programmnamen auswählen	<div style="display: flex; align-items: center; justify-content: center;"> <div style="text-align: center;"> <b>STOP</b>   </div> <div style="margin: 0 10px;">+</div> <div style="text-align: center;"> <b>RESET</b>   </div> </div>	Das ausgewählte Programm kann anschließend gestartet oder editiert werden.

- Auf der PRO/ALM-Anzeige werden nur einstellige Programmnamen angezeigt.
- Betätigen Sie zum Aufrufen eines anderen Programms die RESET-Taste bei gehaltener STOP-Taste.
- Über das Bedienfeld am Steuergerät können Programme mit einstelligem Programmnamen ausgewählt werden. Mit der Teaching Box oder über einen Personal Computer können Programme mit achtstelligem Programmnamen ausgewählt werden.

### Programm starten oder neustarten

Funktion	Tastenbetätigung	Beschreibung
Programm erstmalig starten oder nach einem „Programmstopp“ neustarten	<div style="text-align: center;"> <b>START</b>   </div>	<p>Nach einmaliger Betätigung wird der Zyklusmodus aufgerufen (START-LED blinkt).</p> <p>Nach nochmaliger Betätigung wird der Dauermodus aufgerufen (START-LED leuchtet kontinuierlich).</p>

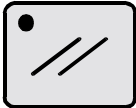
Betätigen Sie die START-Taste zum Neustarten eines Programms, falls ein „Programmstopp“ aufgetreten ist (in diesem Fall leuchtet die STOP-LED).

### Programm und Roboterbewegung stoppen

Funktion	Tastenbetätigung	Beschreibung
Programmabarbeitung und Roboterbewegung stoppen	<div style="text-align: center;"> <b>STOP</b>   </div>	Nach Betätigung der STOP-Taste wird die START-LED aus- und die STOP-LED eingeschaltet.

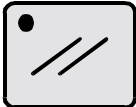
- Während eines Programmstopps kann der Roboter im Jog-Betrieb bewegt werden.
- Betätigen Sie zum Neustarten eines Programms die START-Taste.

**Programm zurücksetzen**

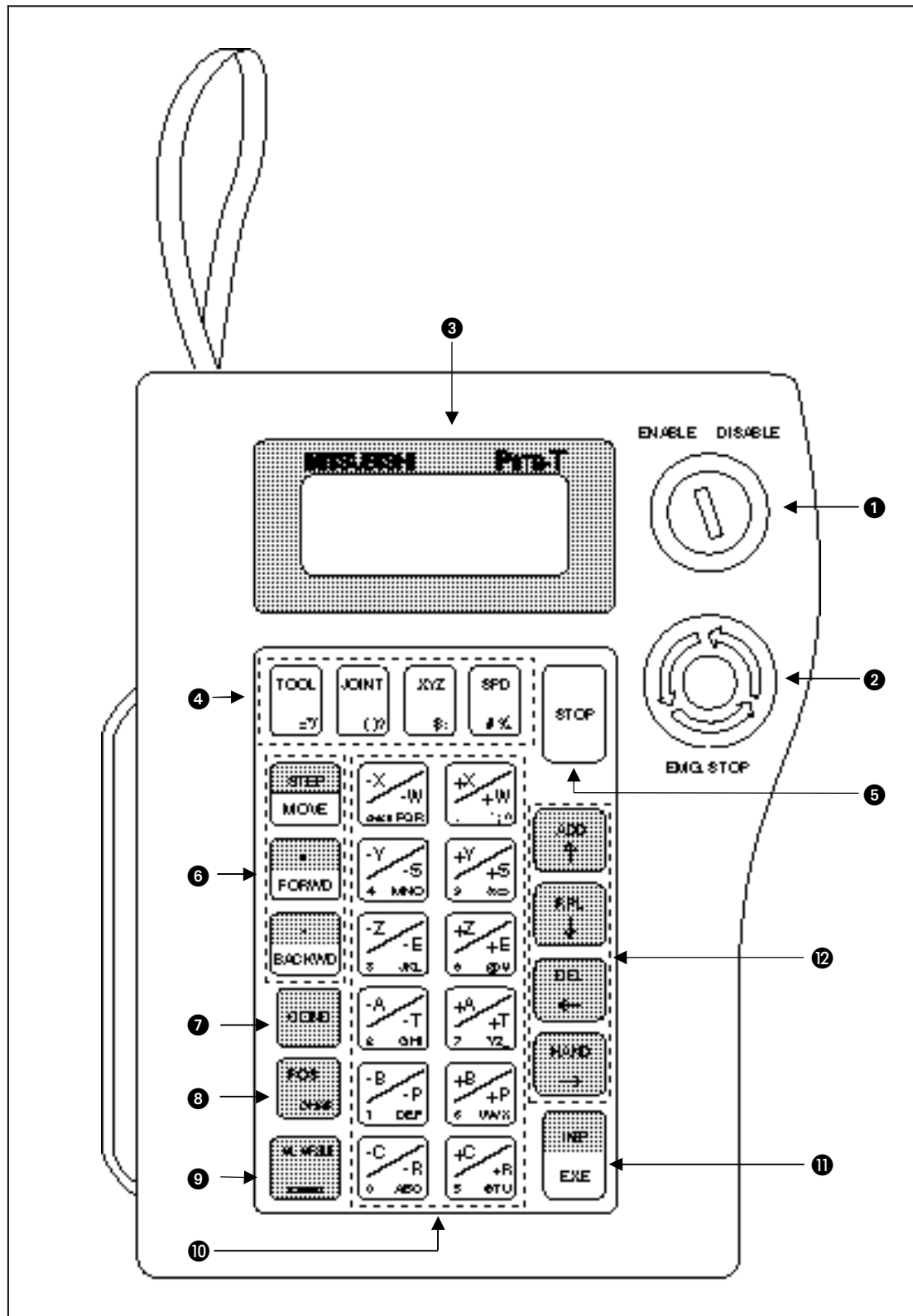
Funktion	Tastenbetätigung	Beschreibung
Programm nach einem Stopp zurücksetzen (erste Programmzeile)	<b>RESET</b> 	Nach Betätigung der RESET-Taste wird die STOP-LED ausgeschaltet.

- Die RESET-Taste kann nur während eines Programmstopps aktiviert werden. Bei laufender Programmabarbeitung ist keine Aktivierung möglich.
- Eine eventuell aufgetretene Fehlermeldung wird nach Betätigung der RESET-Taste quittiert.
- Die Robotersteuerung speichert bei einem Programmstopp die internen Variablenwerte. Rufen Sie zum Löschen der internen Variablenwerte ein anderes Programm auf.

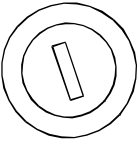
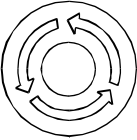










**Fehlermeldung quittieren**

Funktion	Tastenbetätigung	Beschreibung
Fehlermeldung quittieren	<b>RESET</b> 	Die RESET-LED leuchtet bei einer anstehenden Fehlermeldung. Nach Betätigung der RESET-Taste wird die RESET LED ausgeschaltet.




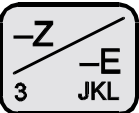

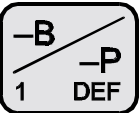
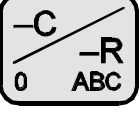
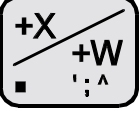
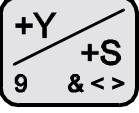
## 2.3 Gerätebeschreibung der Teaching Box



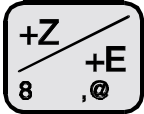
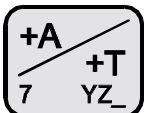
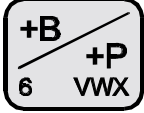
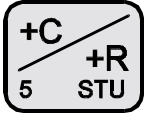
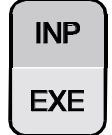




**Abb. 2-5:** Bedienfeld der Teaching Box

Nr.	Schalter / Taste	Beschreibung
①		Teaching Box ein-/auschalten Einschalten: Schalter in Stellung „ENBL“ Ausschalten: Schalter in Stellung „DISABLE“  <b>Bei eingeschalteter Teaching Box kann weder über das Steuergerät noch über externe Geräte in den Steuerungsvorgang eingegriffen werden.</b>
②		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.
③	LCD-Anzeige	Auf der LDC-Anzeige (4 Zeilen x 16 Zeichen) wird das aktuell ausgewählte Programm oder der Betriebszustand des Roboters angezeigt.
④		1.) Betriebsart auswählen: Werkzeug-Jog-Betrieb 2.) Eingabe der Zeichen „ = * / “ zur Programmerstellung
		1.) Betriebsart auswählen: Gelenk-Jog-Betrieb 2.) Eingabe der Zeichen „ ( ) ? “ zur Programmerstellung
		1.) Betriebsart auswählen: XYZ-Jog-Betrieb 2.) Eingabe der Zeichen „ \$ : “ zur Programmerstellung
		1.) Jog-Geschwindigkeit einstellen (2 Stufen) 2.) Eingabe der Zeichen „ # % ! “ zur Programmerstellung
⑤		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.
⑥		1.) Interpolationsmodus für einen Programmschritt ändern 2.) Programmschritt ausführen (bei gleichzeitiger Betätigung der [INP/EXE]-Taste)
		1.) Eingabe des Zeichen „ + “ zur Programmerstellung 2.) Programm schrittweise abarbeiten (vorwärts)
		1.) Eingabe des Zeichen „ - “ zur Programmerstellung 2.) Programm schrittweise abarbeiten (rückwärts)
⑦		Ausführungsbedingungen (Interpolationsmethode, Geschwindigkeit und Timer) einstellen
⑧		1.) Wechsel der Anzeige von „Ausführungsbedingungen“ zu „Befehl/Position“ 2.) Darstellung der Editieranzeige ändern oder Zahlen/Buchstaben auswählen

Tab. 2-4: Bedienelemente der Teaching Box (1)

Nr.	Schalter / Taste	Beschreibung
9		Fehlermeldung und/oder gestopptes Programm zurücksetzen
10		<ol style="list-style-type: none"> <li>1.) Mittelteilgelenk (W-Achse) in Minus-Richtung im Gelenk-Jog-Betrieb bewegen (im Uhrzeigersinn bei Draufsicht)</li> <li>2.) Handspitze in (-X)-Richtung im XYZ-Koordinatensystem bewegen (XYZ-Jog-Betrieb)</li> <li>3.) Handspitze in (-X)-Richtung im Werkzeug-Koordinatensystem bewegen (Werkzeug-Jog-Betrieb)</li> <li>4.) Eingabe von „PQR“ zur Programmerstellung</li> </ol>
		<ol style="list-style-type: none"> <li>1.) Schultergelenk (S-Achse) in Minus-Richtung (aufwärts) im Gelenk-Jog-Betrieb bewegen</li> <li>2.) Handspitze in (-Y)-Richtung im XYZ-Koordinatensystem bewegen (XYZ-Jog-Betrieb)</li> <li>3.) Handspitze der Modelle RV-E4NM/E4NCin (-Y)-Richtung im Werkzeug-Koordinatensystem bewegen (Werkzeug-Jog-Betrieb; Bewegung im Uhrzeigersinn bei Draufsicht)</li> <li>4.) Eingabe von „4“ zur Dateneingabe und „MNO“ zur Programmerstellung</li> </ol>
		<ol style="list-style-type: none"> <li>1.) Ellbogengelenk (E-Achse) in Minus-Richtung (aufwärts) im Gelenk-Jog-Betrieb bewegen</li> <li>2.) Handspitze in (-Z)-Richtung im XYZ-Koordinatensystem bewegen (XYZ-Jog-Betrieb)</li> <li>3.) Handspitze in (-Z)-Richtung im Werkzeug-Koordinatensystem bewegen (Werkzeug-Jog-Betrieb; Bewegung im Uhrzeigersinn bei Draufsicht)</li> <li>4.) Eingabe von „3“ zur Dateneingabe oder „JKL“ zur Programmerstellung</li> </ol>
		<p>5 Achser:</p> <ol style="list-style-type: none"> <li>1.) Eingabe von „2“ zur Dateneingabe oder „GHI“ zur Programmerstellung</li> </ol> <p>6 Achser:</p> <ol style="list-style-type: none"> <li>1.) Unterarmdrehgelenk in Minus-Richtung im Werkzeug-Jog-Betrieb bewegen (im Uhrzeigersinn bei Draufsicht)</li> <li>2.) Eingabe von „2“ zur Dateneingabe oder „GHI“ zur Programmerstellung</li> <li>3.) Unterarmgelenk um die X-Achse im XYZ-Koordinatensystem drehen (XYZ-Jog-Betrieb, Bewegung im Uhrzeigersinn)</li> <li>4.) Unterarmgelenk um die X-Achse im XYZ-Koordinatensystem drehen (Werkzeug-Jog-Betrieb, Bewegung im Uhrzeigersinn)</li> </ol>
		<ol style="list-style-type: none"> <li>1.) Handneigungsgelenk in Minus-Richtung (aufwärts) im Gelenk-Jog-Betrieb bewegen. Die Handspitze des Roboters wird im XYZ- und TOOL-Betrieb unter Beibehaltung des Werkzeugzentrums (TCP) im Uhrzeigersinn um die Y-Achse geschwenkt.</li> <li>2.) Eingabe von „1“ zur Dateneingabe oder „DEF“ zur Programmerstellung</li> <li>3.) Hand 2 schließen (bei gleichzeitiger Betätigung der [HAND/→]-Taste)</li> </ol>
		<ol style="list-style-type: none"> <li>1.) Handdrehgelenk in Minus-Richtung im Gelenk-Jog-Betrieb bewegen (im Uhrzeigersinn bei Draufsicht) Die Handspitze des Roboters wird im XYZ- und TOOL-Betrieb unter Beibehaltung des Werkzeugzentrums (TCP) im Uhrzeigersinn um die Z-Achse geschwenkt.</li> <li>2.) Eingabe von „0“ zur Dateneingabe oder „ABC“ zur Programmerstellung</li> <li>3.) Hand 1 schließen (bei gleichzeitiger Betätigung der [HAND/→]-Taste)</li> </ol>
		<ol style="list-style-type: none"> <li>1.) Mittelteilgelenk (W-Achse) in Plus-Richtung im Gelenk-Jog-Betrieb bewegen (entgegen dem Uhrzeigersinn bei Draufsicht)</li> <li>2.) Handspitze in (+X)-Richtung im XYZ-Koordinatensystem bewegen (XYZ-Jog-Betrieb)</li> <li>3.) Handspitze in (+X)-Richtung im Werkzeug-Koordinatensystem bewegen (Werkzeug-Jog-Betrieb)</li> <li>4.) Eingabe von „ ‘ “ zur Dateneingabe oder „ ‘ ; ^ “ zur Programmerstellung</li> </ol>
		<ol style="list-style-type: none"> <li>1.) Schultergelenk (S-Achse) in Plus-Richtung (abwärts) im Gelenk-Jog-Betrieb bewegen</li> <li>2.) Handspitze in (+Y)-Richtung im XYZ-Koordinatensystem bewegen (XYZ-Jog-Betrieb)</li> <li>3.) Handspitze in (+Y)-Richtung im Werkzeug-Koordinatensystem bewegen (Werkzeug-Jog-Betrieb; entgegen dem Uhrzeigersinn bei Draufsicht)</li> <li>4.) Eingabe von „9“ zur Dateneingabe und „&amp;&lt;&gt;“ zur Programmerstellung</li> </ol>

Tab. 2-4: Bedienelemente der Teaching Box (2)

Nr.	Schalter / Taste	Beschreibung
10		1.) Ellbogengelenk (E-Achse) in Plus-Richtung (abwärts) im Gelenk-Jog-Betrieb bewegen 2.) Handspitze in (+Z)-Richtung im XYZ-Koordinatensystem bewegen (XYZ-Jog-Betrieb) 3.) Handspitze in (+Z)-Richtung im Werkzeug-Koordinatensystem bewegen (Werkzeug-Jog-Betrieb; Bewegung im Uhrzeigersinn bei Draufsicht) 4.) Eingabe von „8“ zur Dateneingabe oder „ ,@“ zur Programmerstellung
		5 Achser: 1.) Eingabe von „7“ zur Dateneingabe oder „YZ_“ zur Programmerstellung  6 Achser: 1.) Unterarmdrehgelenk in Plus-Richtung im Werkzeug-Jog-Betrieb bewegen (entgegen dem Uhrzeigersinn bei Draufsicht) 2.) Eingabe von „7“ zur Dateneingabe oder „YZ_“ zur Programmerstellung 3.) Unterarmdrehgelenk um die X-Achse im XYZ-Koordinatensystem drehen (XYZ-Jog-Betrieb, Bewegung entgegen dem Uhrzeigersinn) 4.) Unterarmdrehgelenk um die X-Achse im XYZ-Koordinatensystem drehen (Werkzeug-Jog-Betrieb, Bewegung entgegen dem Uhrzeigersinn)
		1.) Handneigungsgelenk in Plus-Richtung (abwärts) im Gelenk-Jog-Betrieb bewegen. Die Handspitze des Roboters wird im XYZ- und TOOL-Betrieb unter Beibehaltung des Werkzeugzentrums (TCP) im Uhrzeigersinn um die Y-Achse geschwenkt. 2.) Eingabe von „6“ zur Dateneingabe oder „VWX“ zur Programmerstellung 3.) Hand 2 öffnen (bei gleichzeitiger Betätigung der [HAND/→]-Taste)
		1.) Handdrehgelenk in Plus-Richtung im Gelenk-Jog-Betrieb bewegen (entgegen dem Uhrzeigersinn bei Draufsicht) Die Handspitze des Roboters wird im XYZ- und TOOL-Betrieb unter Beibehaltung des Werkzeugzentrums (TCP) im Uhrzeigersinn um die Z-Achse geschwenkt. 2.) Eingabe von „5“ zur Dateneingabe oder „STU“ zur Programmerstellung 3.) Hand 1 öffnen (bei gleichzeitiger Betätigung der [HAND/→]-Taste)
11		1.) Daten für Ausführungsbedingungen eingeben 2.) Programm schrittweise abarbeiten (inkremental/dekremental)
12		1.) Positionsdaten und/oder Ausführungsbedingungen in Programmschritt einfügen 2.) Cursor im Hauptmenü nach oben bewegen
		1.) Positionsdaten und/oder Ausführungsbedingungen im Programmschritt ändern 2.) Cursor im Hauptmenü nach unten bewegen
		1.) Positionsdaten und/oder Ausführungsbedingungen im Programmschritt löschen 2.) Cursor im Hauptmenü nach links bewegen
		1.) Hand bewegen 2.) Cursor im Hauptmenü nach rechts bewegen
13	Totmannschalter	Bei eingeschalteter Teaching Box wird der Servoantrieb, bei nicht betätigtem Dreistufen-Totmannschalter, ausgeschaltet. Für ein Einschalten des Servoantriebes muß der Totmannschalter betätigt sein. Ist der Servoantrieb während eines NOT-AUS oder einer Befehlsausführung ausgeschaltet, kann er durch den Totmannschalter nicht eingeschaltet werden. Betätigen Sie in diesem Fall die [ALARM/RESET]-Taste, oder schalten Sie den Servoantrieb ein.

Tab. 2-4: Bedienelemente der Teaching Box (3)



## 2.4 Roboterprogramm testen

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

Mit der Teaching Box können Sie jedes Roboterprogramm testen. Es ist gleichgültig, ob das Programm mit der MOVEMASTER COMMAND-Methode oder mit der MELFA-BASIC III-Methode erstellt wurde.



### ACHTUNG

**Testen Sie unbedingt jedes Roboterprogramm vor einem automatischen Betriebseinsatz!**

### Funktionen zum Testen

Nachfolgend sind alle Funktionen aufgelistet, die Sie beim Testen ausführen und/oder überprüfen sollten.

- Programm schrittweise ausführen (vorwärts)
- Programm schrittweise ausführen (rückwärts)
- Schrittweiser Aufruf (vorwärts/rückwärts)
- Schritt direkt aufrufen
- Zeile direkt aufrufen
- Position direkt anfahren
- Servospannung EIN/AUS
- Programm starten
- Monitor-Funktion für Ein-/Ausgangssignale
- Monitorfunktion für Variablen
- Monitorfunktion für Fehlermeldeliste

Abbildung 2.6 zeigt die Display-Darstellung beim Testen eines Roboterprogramms.

Programmierung mit MOVEMASTER COMMAND-Befehlen	Programmierung mit MELFA-BASIC III-Befehlen
<div> PR:5            ST:1                   LN:10  10 MO 1 </div>	<div> PR:5            ST:1                   LN:10  10 MOV P1 </div>

**Abb. 2-6:** Display-Darstellung eines Roboterprogramms

## 2.5 Bedienung der Teaching Box

### 2.5.1 Roboter im Jog-Betrieb bewegen

In diesem Abschnitt wird das Bewegen des Roboters in den verschiedenen Jog-Betriebsarten beschrieben. Folgende Jog-Betriebsarten stehen zur Verfügung:

- Gelenk-Jog-Betrieb
- XYZ-Jog-Betrieb
- Werkzeug-Jog-Betrieb

#### ① Roboter im Gelenk-Jog-Betrieb bewegen

Die einzelnen Gelenke des Roboters können im Gelenk-Koordinatensystem bewegt werden.

#### Bewegungsrichtungen der Robotergelenke

##### 5 Achsen

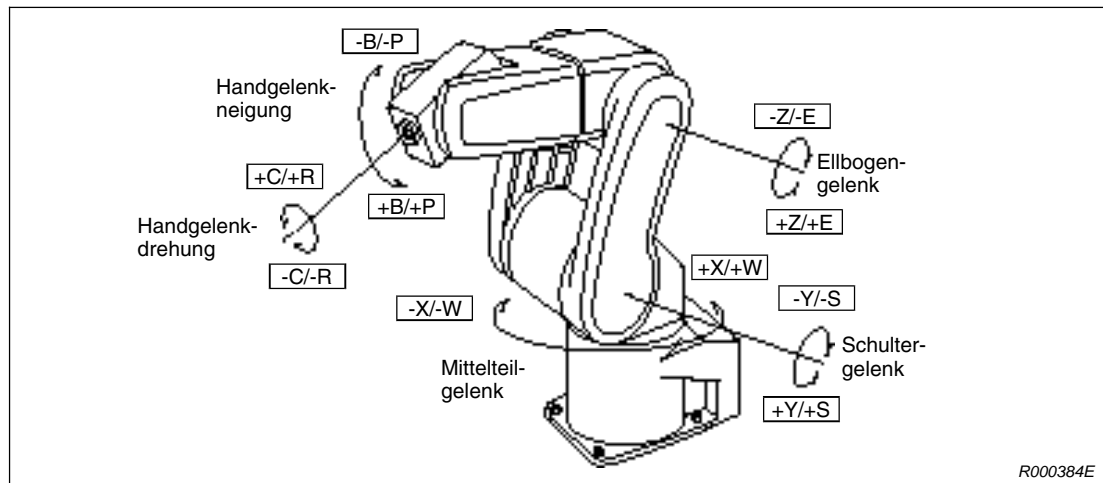


Abb. 2-7: Bewegungsrichtungen der Robotergelenke

##### 6 Achsen

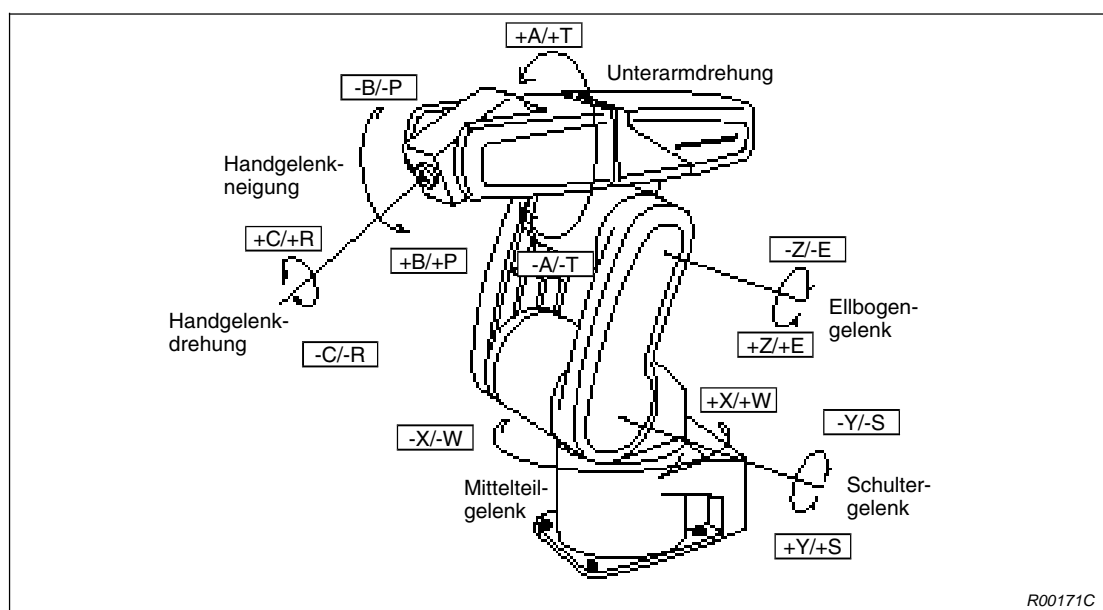


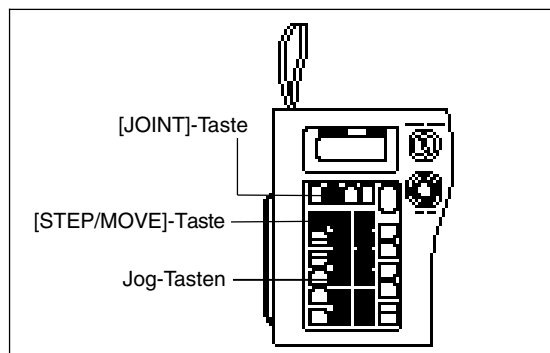
Abb. 2-8: Bewegungsrichtungen der Robotergelenke

## Display-Darstellung

JOINT	low	→ aktuell ausgewählter Jog-Betrieb und Jog-Geschwindigkeit
W	+90.00	
S	+0.00	
E	+90.00	
		→ aktuelle Position (Betrag und Richtung der Gelenkwinkel)

**Abb. 2-9:** Display-Darstellung im Gelenk-Jog-Betrieb

## Anordnung der Tasten



**Abb. 2-10:**

Tasten zum Bewegen des Roboters im Gelenk-Jog-Betrieb

TB-BOX01

## Ausführung

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung								
①	<table><tr><td>JOINT</td><td>low</td></tr><tr><td>W</td><td>+90.00</td></tr><tr><td>S</td><td>+0.00</td></tr><tr><td>E</td><td>+90.00</td></tr></table>	JOINT	low	W	+90.00	S	+0.00	E	+90.00		Gelenk-Jog-Betrieb auswählen.
JOINT	low										
W	+90.00										
S	+0.00										
E	+90.00										
②	<table><tr><td>JOINT</td><td>low</td></tr><tr><td>W</td><td>+90.00</td></tr><tr><td>S</td><td>+0.00</td></tr><tr><td>E</td><td>+90.00</td></tr></table>	JOINT	low	W	+90.00	S	+0.00	E	+90.00		Die einzelnen Gelenke mit den Jog-Tasten bewegen. Die aktuelle Position wird auf dem Display angezeigt.
JOINT	low										
W	+90.00										
S	+0.00										
E	+90.00										

**Tab. 2-5:** Beispiel zum Bewegen des Roboters im Gelenk-Jog-Betrieb

## Beschreibung

- Betätigen Sie die [STEP/MOVE]- in Verbindung mit der [SPD]-Taste zum Ändern der Jog-Geschwindigkeit.
- Der Roboter wird gestoppt, und es ertönt ein Warnton, wenn der zulässige Arbeitsbereich oder die Maximalgeschwindigkeit eines Gelenkes überschritten wird. Auf der Anzeige der Teaching Box wird das Gelenk, das den Arbeitsbereich überschritten hat, mit einem „X“ gekennzeichnet. Bewegen Sie dieses Gelenk in die umgekehrte Richtung.

### 5 Achsen

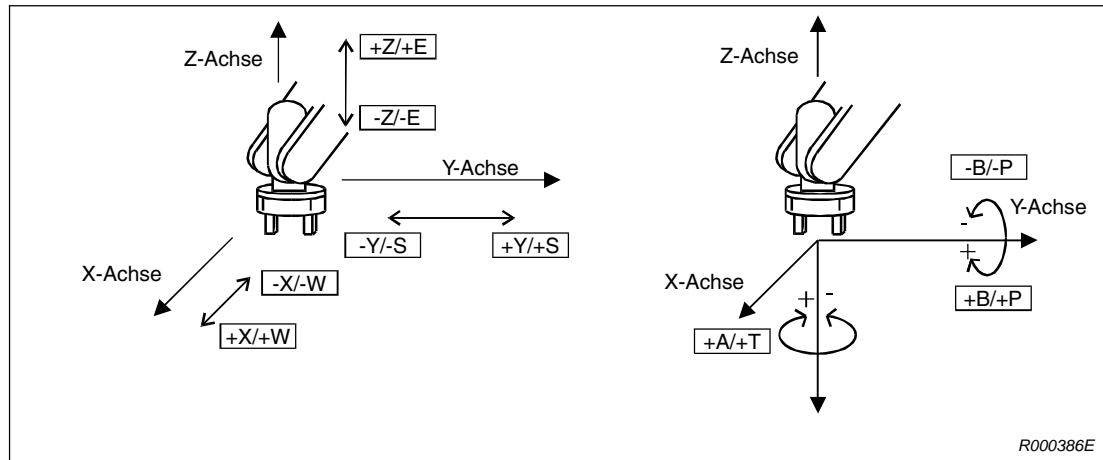
- Die Tasten [+A/+T] und [−A/−T] sind beim 5-achsigen Roboter im Gelenk-Jog-Betrieb ohne Funktion.

## ② Roboter im XYZ-Jog-Betrieb bewegen

Der Roboter kann entlang der Achsen im XYZ-Koordinatensystem bewegt werden.

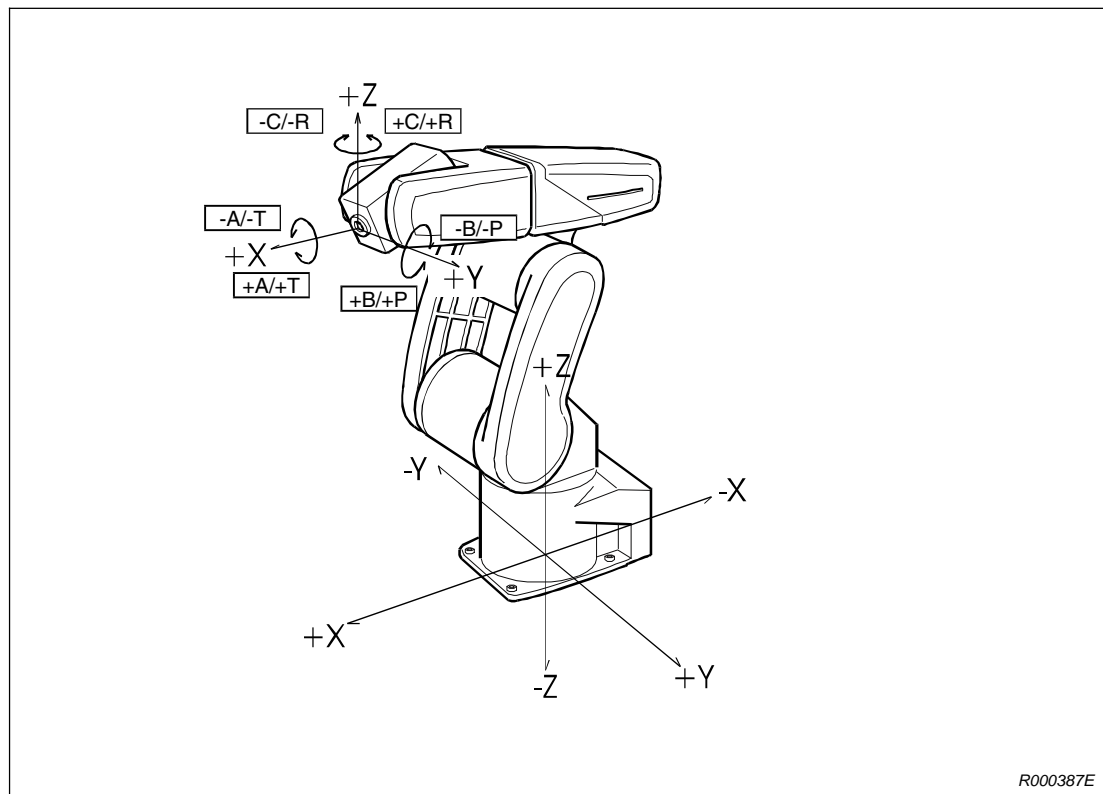
### Bewegungsrichtungen im XYZ-Koordinatensystem

#### 5 Achsen



**Abb. 2-11:** Bewegungsrichtungen im XYZ-Koordinatensystem

#### 6 Achsen



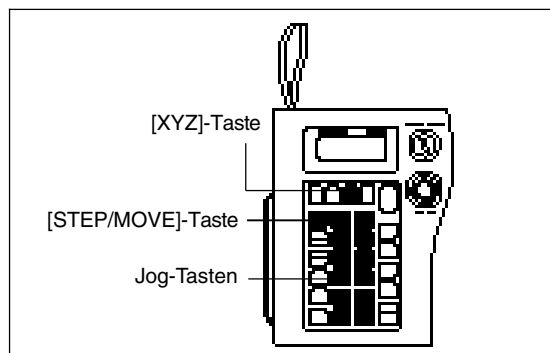
**Abb. 2-12:** Bewegungsrichtungen im XYZ-Koordinatensystem

### Display-Darstellung

X, Y, Z	low	→	aktuell ausgewählter Jog-Betrieb und Jog-Geschwindigkeit
X	+100.00	└─┐	aktuelle Position
Y	+200.00		
Z	+300.00		

**Abb. 2-13:** Display-Darstellung im XYZ-Jog-Betrieb

### Anordnung der Tasten



**Abb. 2-14:**

Tasten zum Bewegen des Roboters im XYZ-Jog-Betrieb

TB-BOX02

### Ausführung

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung								
①	<table><tr><td>X, Y, Z</td><td>low</td></tr><tr><td>X</td><td>+100.00</td></tr><tr><td>Y</td><td>+200.00</td></tr><tr><td>Z</td><td>+300.00</td></tr></table>	X, Y, Z	low	X	+100.00	Y	+200.00	Z	+300.00		XYZ-Jog-Betrieb auswählen.
X, Y, Z	low										
X	+100.00										
Y	+200.00										
Z	+300.00										
②	<table><tr><td>X, Y, Z</td><td>low</td></tr><tr><td>X</td><td>+100.00</td></tr><tr><td>Y</td><td>+200.00</td></tr><tr><td>Z</td><td>+300.00</td></tr></table>	X, Y, Z	low	X	+100.00	Y	+200.00	Z	+300.00		Den Roboter mit den Jog-Tasten entlang der Achsen im XYZ-Koordinatensystem bewegen. Die aktuelle Position wird auf dem Display angezeigt.
X, Y, Z	low										
X	+100.00										
Y	+200.00										
Z	+300.00										

**Tab. 2-6:** Beispiel zum Bewegen des Roboters im XYZ-Jog-Betrieb

### Beschreibung

- Betätigen Sie die [STEP/MOVE]- in Verbindung mit der [SPD]-Taste zum Ändern der Jog-Geschwindigkeit.
- Der Roboter wird gestoppt, und es ertönt ein Warnton, wenn der zulässige Arbeitsbereich oder die Maximalgeschwindigkeit eines Gelenkes überschritten wird. Wechseln Sie in diesem Fall in den Gelenk-Jog-Betrieb. Im Gelenk-Jog-Betrieb wird auf der Anzeige der Teaching Box das Gelenk mit einem „X“ gekennzeichnet, das den Arbeitsbereich überschritten hat. Bewegen Sie dieses Gelenk in die umgekehrte Richtung.

#### 5 Achsen

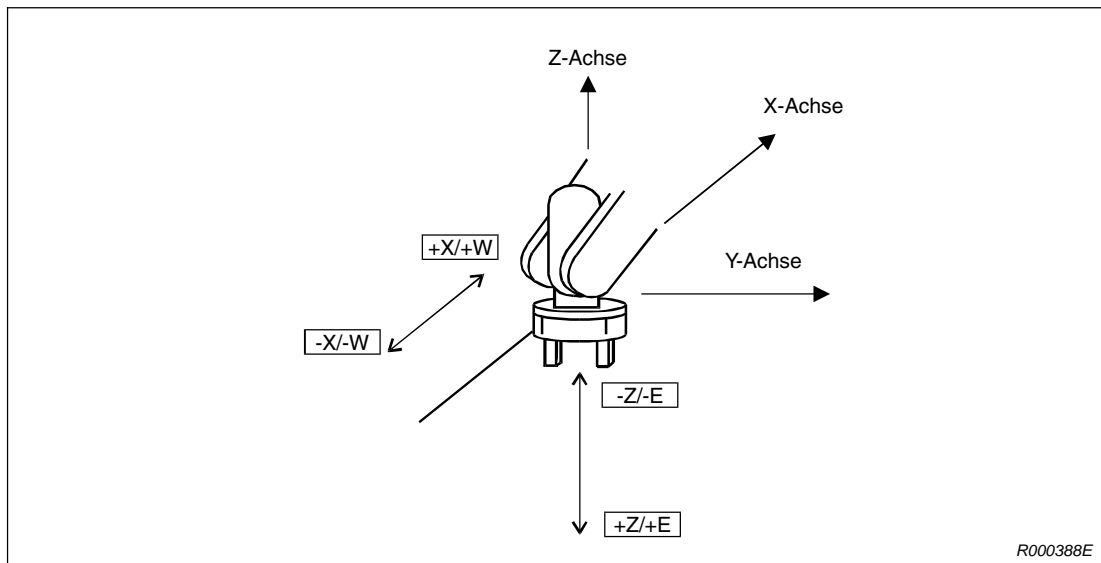
- Die Tasten [+A/+T] und [−A/−T] sind beim 5-achsigen Roboter im XYZ-Jog-Betrieb ohne Funktion.

### ③ Roboter im Werkzeug-Jog-Betrieb (TOOL) bewegen

Der Roboter kann entlang der Achsen im Werkzeug-Koordinatensystem bewegt werden.

#### Bewegungsrichtungen im Werkzeug-Koordinatensystem

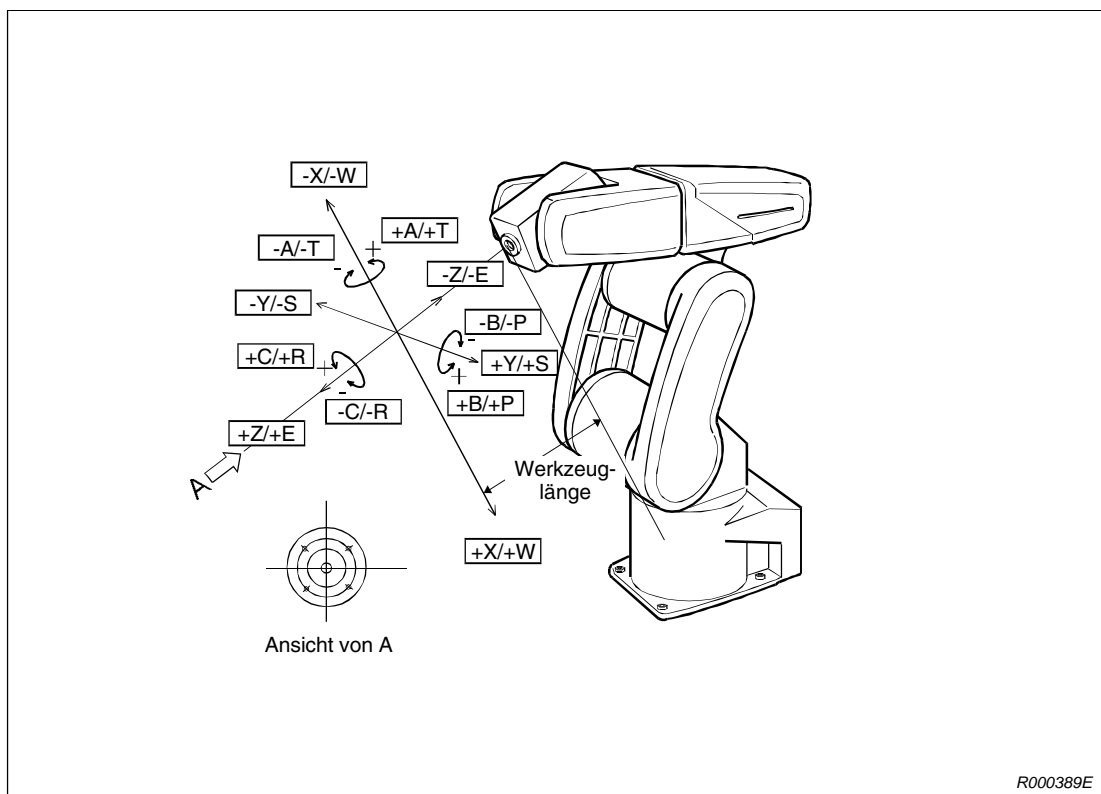
##### 5 Achsen



R000388E

**Abb. 2-15:** Bewegungsrichtungen im Werkzeug-Koordinatensystem

##### 6 Achsen



R000389E

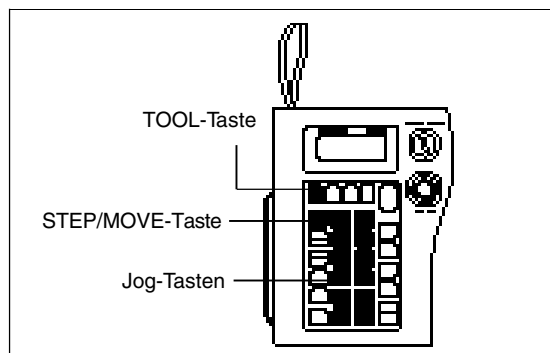
**Abb. 2-16:** Bewegungsrichtungen im Werkzeug-Koordinatensystem

## Display-Darstellung

TOOL	low	→	aktuell ausgewählter Jog-Betrieb und Jog-Geschwindigkeit
X	+100.00	→	aktuelle Position
Y	+200.00		
Z	+300.00		

**Abb. 2-17:** Display-Darstellung im Werkzeug-Jog-Betrieb

## Anordnung der Tasten





**Abb. 2-18:**

Tasten zum Bewegen des Roboters im Werkzeug-Jog-Betrieb

TB-BOX22

## Ausführung

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung								
①	<table><tr><td>TOOL</td><td>low</td></tr><tr><td>X</td><td>+100.00</td></tr><tr><td>Y</td><td>+200.00</td></tr><tr><td>Z</td><td>+300.00</td></tr></table>	TOOL	low	X	+100.00	Y	+200.00	Z	+300.00		Werkzeug-Jog-Betrieb auswählen
TOOL	low										
X	+100.00										
Y	+200.00										
Z	+300.00										
②	<table><tr><td>TOOL</td><td>low</td></tr><tr><td>X</td><td>+100.00</td></tr><tr><td>Y</td><td>+200.00</td></tr><tr><td>Z</td><td>+300.00</td></tr></table>	TOOL	low	X	+100.00	Y	+200.00	Z	+300.00		Den Roboter mit den Jog-Tasten entlang der Achsen im Werkzeug-Koordinatensystem bewegen. Die aktuelle Position wird auf dem Display angezeigt.
TOOL	low										
X	+100.00										
Y	+200.00										
Z	+300.00										

**Tab. 2-7:** Beispiel zum Bewegen des Roboters im Werkzeug-Jog-Betrieb

## Beschreibung

- Betätigen Sie die [STEP/MOVE]- in Verbindung mit der [SPD]-Taste zum Ändern der Jog-Geschwindigkeit.
- Der Roboter wird gestoppt, und es ertönt ein Warnton, wenn der zulässige Arbeitsbereich oder die Maximalgeschwindigkeit eines Gelenkes überschritten wird. Wechseln Sie in diesem Fall in den Gelenk-Jog-Betrieb. Im Gelenk-Jog-Betrieb wird auf der Anzeige der Teaching Box das Gelenk mit einem „X“ gekennzeichnet, das den Arbeitsbereich überschritten hat. Bewegen Sie dieses Gelenk in die umgekehrte Richtung.

### 5 Achsen

Betätigen Sie die Taste [+Y/+S] oder [-Y/-S] zum Bewegen des Mittelteilgelenks.

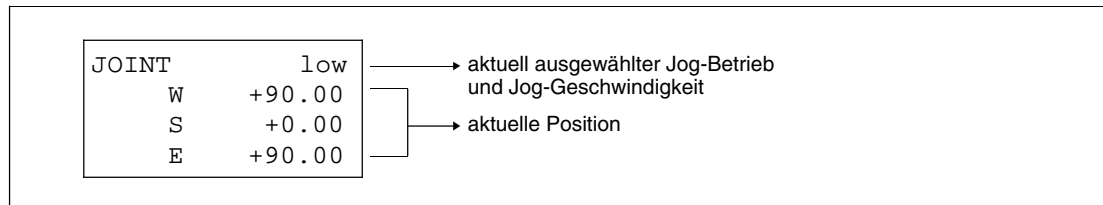
Die Tasten [+A/+T] und [-A/-T] sind beim 5-achsigen Roboter im Werkzeug-Jog-Betrieb ohne Funktion.

## 2.5.2 Jog-Geschwindigkeit einstellen

### Funktion

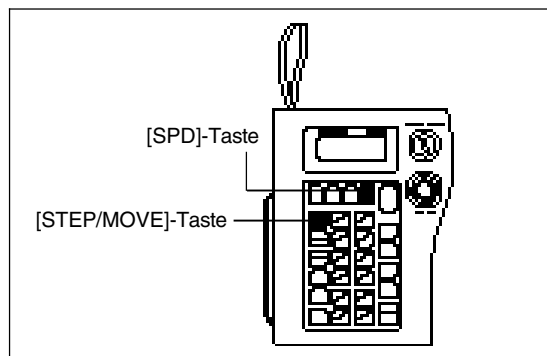
Einstellen der Geschwindigkeit im Jog-Betrieb.

### Display-Darstellung



**Abb. 2-19:** Display-Darstellung im Jog-Betrieb

### Anordnung der Tasten

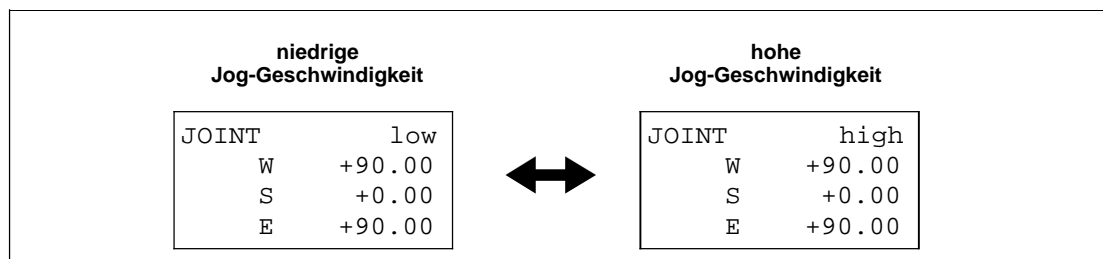


**Abb. 2-20:**  
Tasten zum Einstellen der Jog-Geschwindigkeit

TB-BOX23

### Ausführung

Die Geschwindigkeit kann in 2 Stufen eingestellt werden.



**Abb. 2-21:** Display-Darstellung zum Einstellen der Jog-Geschwindigkeit



Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<div>JOINT            low</div> <div>W        +90.00</div> <div>S        +0.00</div> <div>E        +90.00</div>		Die Einstellung der Jog-Geschwindigkeit wird im Beispiel von „low“ (niedrige) auf „high“ (hohe) Geschwindigkeit geändert.
②	<div>JOINT            high</div> <div>W        +90.00</div> <div>S        +0.00</div> <div>E        +90.00</div>		

**Tab. 2-8:** Beispiel zum Einstellen der Jog-Geschwindigkeit

### Beschreibung

Nach Einschalten der Teaching Box über den [ENBL/DISABLE]-Schalter wird von der Robotersteuerung automatisch die niedrige Jog-Geschwindigkeit eingestellt.

### Geschwindigkeitswerte für die einzelnen Jog-Betriebsarten

Der Roboter wird zuerst mit einer sehr niedrigen konstanten Anfahrgeschwindigkeit (0,1 Grad/s oder 0,1 mm/s) bewegt. Erst nach ca. 0,4 s erfolgt eine Beschleunigung bis zum Erreichen der festgelegten Jog-Geschwindigkeit.

Jog-Betriebsart	Gelenk	XYZ	Werkzeug
Geschwindigkeit			
Anfahrgeschwindigkeit	0,1°	0,1 mm	0,1 mm
Niedrige Jog-Geschwindigkeit	1 %	1,5 mm/s	1,5 mm/s
Hohe Jog-Geschwindigkeit	13 %	100 mm/s	100 mm/s

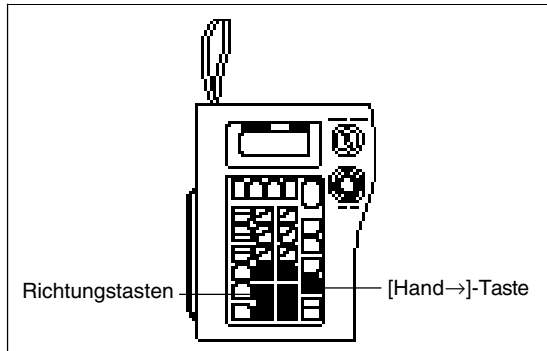
**Tab. 2-9:** Geschwindigkeitswerte für die einzelnen Jog-Betriebsarten

### 2.5.3 Handgreifer öffnen/schließen

#### Funktion

Öffnen oder Schließen des Greifers der pneumatisch betriebenen Roboterhand.

#### Anordnung der Tasten



**Abb. 2-22:**  
Tasten zum Öffnen/Schließen des Handgreifers

TB-BOX24

#### Ausführung

Nr.	Auswahl der Hand	Tastenbetätigungen	Beschreibung
①	Hand 1 pneumatisch betrieben		Greifer der Hand 1 wird geöffnet.
②			Greifer der Hand 1 wird geschlossen.
①	Hand 2 pneumatisch betrieben		Greifer der Hand 2 wird geöffnet.
②			Greifer der Hand 2 wird geschlossen.
①	Hand 3 pneumatisch betrieben		Greifer der Hand 3 wird geöffnet.
②			Greifer der Hand 3 wird geschlossen.

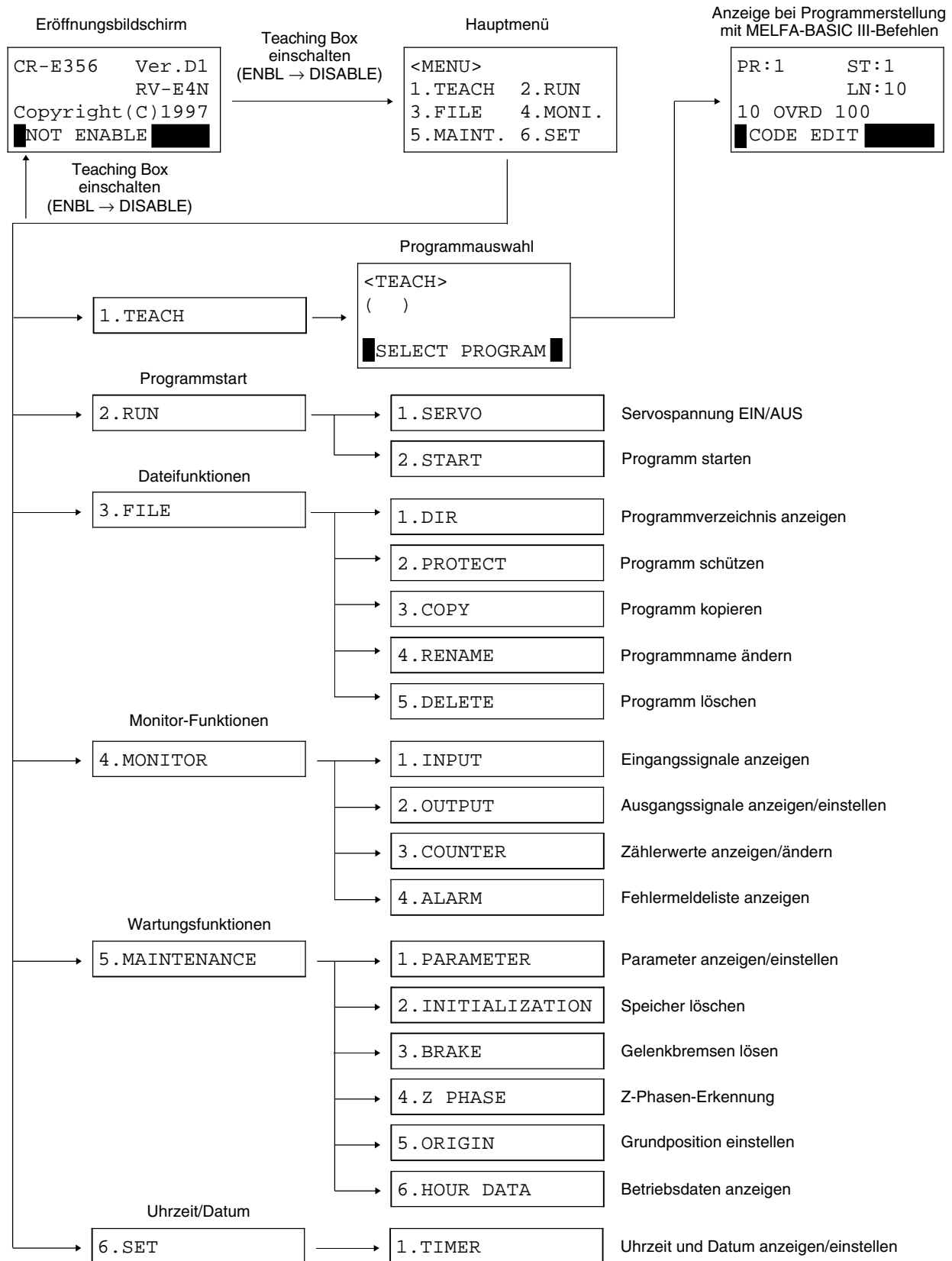
**Tab. 2-10:** Beispiel zum Öffnen/Schließen des Handgreifers

**Beschreibung**

- Beim Einsatz einer pneumatisch betriebenen Hand wird über die Steuerung des Pneumatikventils (24V DC) der Handgreifer geöffnet oder geschlossen.
- Wenn ein Positionsschritt zusätzlich eingefügt wird, speichert die Robotersteuerung den Handgreiferzustand als Teil der Ausführungsbedingungen ab.
- Den Handgreiferzustand können Sie auch über externe Handkontrollsignale festlegen (siehe Seite 5-93).

## 2.5.4 Menübaum

Die untere Abbildung zeigt den Menübaum der Teaching Box und die entsprechenden Funktionen.



## 2.5.5 Menüpunkt auswählen


### Funktion

Zur Auswahl eines Menüpunktes stehen zwei Möglichkeiten zur Verfügung.

### Ausführung



In den nachfolgenden Tabellen werden die beiden Möglichkeiten beispielhaft an der Auswahl des Menüpunktes „3. FILE“ gezeigt.

#### 1.) Menüauswahl über Eingabe der Nummer

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<div style="border: 1px solid black; padding: 5px;">           &lt;MENU&gt;            1. TEACH    2. RUN            3. FILE    4. MONI.            5. MAINT.   6. SET         </div>		Das Menü FILE wird durch Eingabe der Ziffer 3 aufgerufen.
②	<div style="border: 1px solid black; padding: 5px;">           &lt;FILE&gt;            1. DIR    2. PROTECT            3. COPY   4. RENAME            5. DELETE         </div>		Das Menü FILE wird angezeigt.

**Tab. 2-11:** Beispiel zur Menüauswahl über Eingabe der Nummer

#### 2.) Menüauswahl über Cursor

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<div style="border: 1px solid black; padding: 5px;">           &lt;MENU&gt;            1. TEACH    2. RUN            3. FILE    4. MONI.            5. MAINT.   6. SET         </div>		Der Cursor wird zum Menüpunkt „3. FILE“ bewegt.
②	<div style="border: 1px solid black; padding: 5px;">           &lt;MENU&gt;            1. TEACH    2. RUN            3. FILE    4. MONI.            5. MAINT.   6. SET         </div>		Die Auswahl wird bestätigt.
③	<div style="border: 1px solid black; padding: 5px;">           &lt;FILE&gt;            1. DIR    2. PROTECT            3. COPY   4. RENAME            5. DELETE         </div>		Das Menü FILE wird angezeigt.

**Tab. 2-12:** Beispiel zur Menüauswahl über Cursor

### Beschreibung

- Den Cursor können Sie mit den Tasten [ADD ↑], [RPL ↓], [DEL ←] und [HAND →] bewegen.
- Wenn Sie eine falsche Taste betätigt haben, dann gelangen Sie durch das einmalige Betätigen der [SPD]-Taste wieder in das Hauptmenü zurück.

## 2.5.6 Servospannung ein-/auschalten

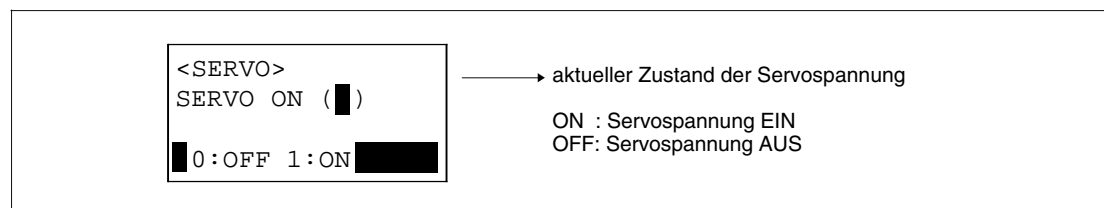
### Funktion

Die Servospannung kann über die Teaching Box ein-/ausgeschaltet werden.

Schalten Sie die Servospannung immer vor dem Aufruf der Funktion „Bremsen lösen“ oder zur Sicherstellung eines Roboterstillstands aus. Bei ausgeschalteter Servospannung kann der Roboter über Steuerungsbefehle nicht mehr bewegt werden.

Die Abfrage „Servospannung EIN/AUS“ erfolgt immer nach Einschalten des Steuergerätes oder nach Rücksetzen eines Programms.

### Display-Darstellung



**Abb. 2-23:** Display-Darstellung zum Ein-/Ausschalten der Servospannung

### Ausführung

Betätigen Sie den Totmannschalter, während Sie die Servospannung einschalten. Bei nicht betätigtem Totmannschalter lässt sich die Servospannung nicht einschalten.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<div> &lt;MENU&gt;  1. TEACH 2. RUN  3. FILE 4. MONI  5. MAINT 6. SET </div>		Das Menü RUN wird durch Eingabe der Ziffer 2 ausgewählt.
②	<div> &lt;RUN&gt;  1. SERVO 2. START </div>		Der Menüpunkt SERVO wird ausgewählt.
③	<div> &lt;SERVO&gt;  SERVO ON (■)  0:OFF 1:ON </div>		Die Servospannung wird ein- oder ausgeschaltet.
④	<div> &lt;SERVO&gt;  SERVO ON (■)  0:OFF 1:ON </div>		Die Servospannung ist jetzt ein- oder ausgeschaltet.

**Tab. 2-13:** Beispiel zum Ein-/Ausschalten der Servospannung

### Beschreibung

- Nach Ausschalten der Servospannung werden die Bremsen automatisch aktiviert.

## 2.5.7 Roboterprogramm starten

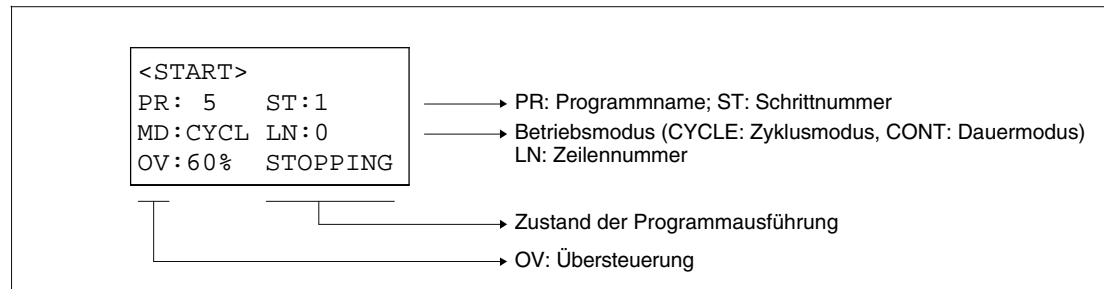
### ① Auswählen und Starten eines Programmes

#### Funktion

Die Robotersteuerung startet das über die Nummer ausgewählte Programm, ab dem festgelegten Schritt oder der festgelegten Programmzeile.

Über die Teaching Box können Sie den Betriebsmodus (Dauer- oder Zyklusmodus) auswählen.

#### Display-Darstellung



**Abb. 2-24:** Display-Darstellung im Menü **START**

#### Ausführung

Im nachfolgenden Beispiel wird das Programm Nr. 5 ab Schritt 3 im Dauermodus ausgeführt.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
	<div>&lt;MENU&gt;</div> <div>1. TEACH 2. RUN</div> <div>3. FILE 4. MONI</div> <div>5. MAINT 6. SET</div>		Das Menü <b>RUN</b> wird durch Eingabe der Ziffer 2 aufgerufen.
②	<div>&lt;RUN&gt;</div> <div>1. SERVO 2. START</div>		Der Menüpunkt <b>START</b> wird ausgewählt.
③	<div>&lt;START&gt;</div> <div>PR: 1 ST:1</div> <div>MD:CYCL LN:0</div> <div>OV:60 % STOPPING</div>		Die Eingabe der Programmnummer wird aufgerufen.
④	<div>&lt;START&gt;</div> <div>PR: (5 ) ST:1</div> <div>MD:CYCL LN:0</div> <div>SELECT PROGRAM</div>		Die Programmnummer (5) wird eingegeben. Danach wird der Cursor zum Eingabefeld für die Schrittnummer bewegt.

**Tab. 2-14:** Beispiel zum Starten eines Programms (1)

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
⑤	<pre> &lt;START&gt; PR:5      ST:(3) MD:CYCL.LN:0 SET STEP NUMBER </pre>		Die Schrittnummer (3) wird eingegeben. Danach wird der Cursor zum Eingabefeld für den Betriebsmodus bewegt.
⑥	<pre> &lt;START&gt; PR:5      ST:3 MD:(1)    LN:0 0:CYCLE 1:CONT </pre>		Der Betriebsmodus (1: Dauermodus) wird ausgewählt.
⑦	<pre> &lt;START&gt; PR:5      ST:3 1:START OV:60%    HALT( ) </pre>		Das Programm wird gestartet und ab Schritt 3 kontinuierlich abgearbeitet.

**Tab. 2-14:** Beispiel zum Starten eines Programms (2)

### Beschreibung

- Geben Sie bei Programmen, die mit MOVEMASTER COMMAND-Befehlen erstellt wurden, die Start-Zeilenummer (LN) an.
- Bewegen Sie den Cursor mit der Taste [ADD ↑] nach oben und der Taste [RPL ↓] nach unten.
- Im Dauermodus wird das Programm nach dem erstmaligen Starten ab dem festgelegten Schritt oder der festgelegten Programmzeile ausgeführt. Die Programmausführung erfolgt ab dem ersten Schritt oder der ersten Programmzeile, wenn das Programm zum zweitenmal im Dauermodus gestartet wird (das gleiche gilt für mehrmaligen Aufruf).
- Betätigen Sie zum Stoppen des Programms entweder die Taste [STOP] auf dem Bedienfeld der Teaching Box oder die Taste [STOP] auf der Frontseite des Steuergerätes.

Nach der Tastenbetätigung wird das Programm gestoppt und die LED der [STOP]-Taste eingeschaltet (LED leuchtet kontinuierlich).

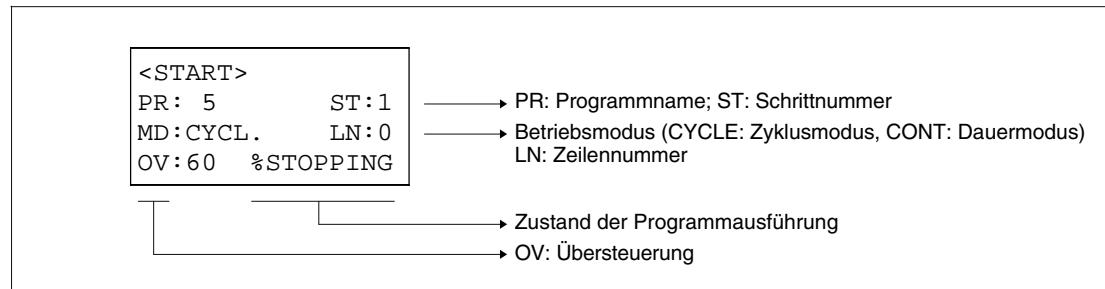


## ② Starten eines bereits ausgewählten Programms

### Funktion

Nach Auswahl eines Programms erfolgt der Start ab dem im Menü START festgelegten Schritt.

### Display-Darstellung



**Abb. 2-25:** Display-Darstellung im Menü START

### Ausführung

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<div> &lt;MENU&gt;  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> &lt;RUN&gt;  1. SERVO    2. START </div>		Der Menüpunkt START wird ausgewählt.
③	<div> &lt;START&gt;  PR: 5                      ST: 1  MD: CYCL.                LN: 0  OV: 60    %STOPPING </div>		Das Menü START wird angezeigt.

**Tab. 2-15:** Beispiel zum Starten eines Programms

### Ausführung

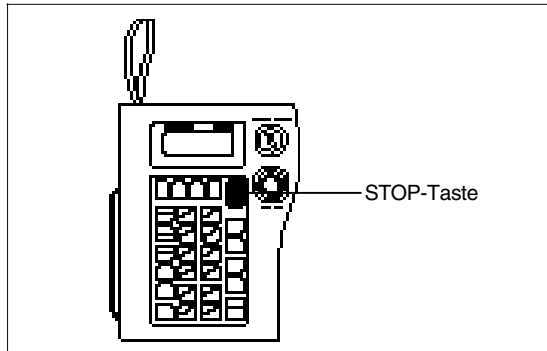
Die Meldung „[START]“ wird blinkend auf dem Display angezeigt, wenn das Programm im Zyklusmodus abgearbeitet wird. Falls die Meldung „[START]“ kontinuierlich auf dem Display angezeigt wird, erfolgt die Programmabarbeitung im Dauermodus.

## 2.5.8 Programm und Roboterbewegung stoppen

### Funktion

Die Programmabarbeitung und die Roboterbewegung wird gestoppt.

### Anordnung der STOP-Taste



**Abb. 2-26:**

*STOP-Taste zum Stoppen des Programms und der Roboterbewegung*

TB-BOX25

### Ausführung

Betätigen Sie die STOP-Taste.

### Beschreibung

Die Roboterbewegung können Sie während des Programmablaufs mit der STOP-Taste anhalten. Anschließend verlischt die Meldung „[START]“ und es wird „[STOP]“ angezeigt.

Wenn Sie den Programmablauf mit der STOP-Taste angehalten haben, können Sie durch Betätigen der START-Taste den Programmablauf ab der momentanen Position wieder starten.

## 2.5.9 Programmverzeichnis anzeigen

### Funktion








Es wird ein Verzeichnis der gespeicherten Programme angezeigt.

### Display-Darstellung

<pre> &lt;DIR&gt;          7 1      94-05-05 2      94-02-10 3      94-03-24 </pre>	<p>→ Gesamtanzahl der gespeicherten Programme</p> <p>→ Programmname und Erstellungsdatum</p>
---	--

**Abb. 2-27:** Display-Darstellung zum Anzeigen des Programmverzeichnisses

### Ausführung

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre> &lt;MENU&gt; 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> &lt;FILE&gt; 1. DIR  2. PROTECT 3. COPY 4. RENAME 5. DELETE </pre>		Der Menüpunkt DIR wird ausgewählt.
③	<pre> &lt;DIR&gt;          7 1      94-05-05 2      94-02-10 3      94-03-24 </pre>	 	Mit den Cursortasten können die weiteren Programmnamen zur Anzeige gebracht werden.
④	<pre> &lt;DIR&gt;          7 1      09:28:48 2      11:30:50 3      19:08:30 </pre>		Es wird die Zeit der Programmerstellung angezeigt (Stunde, Minute, Sekunde).
⑤	<pre> &lt;DIR&gt;          60455 1      199 2      3202 3      1205 </pre>		Es wird die Dateigröße in Bytes angezeigt.
⑥	<pre> &lt;DIR&gt;  PROTECT 1      ON (1) 2      OFF (0) 0=OFF  1=ON </pre>		Es wird angezeigt, ob ein Programm geschützt ist.

**Tab. 2-16:** Beispiel zum Anzeigen des Programmverzeichnisses

### Beschreibung


Die Programmnamen werden in aufsteigender Reihenfolge angezeigt.

## 2.5.10 Programm schützen

### Funktion

Schützt Programme gegen unbeabsichtigtes Löschen und Programmänderungen.






### Display-Darstellung

<div style="border: 1px solid black; padding: 5px; display: inline-block;"> <div style="display: flex; justify-content: space-between;"> <span>&lt;DIR&gt;</span> <span>PROTECT</span> </div> <div> 1            ON ( 1 )  2            OFF ( 0 )  0=OFF    1=ON </div> </div>		Programnummer (Name) und Schutzstatus 0 = ungeschützt, 1 = geschützt
--	---	---

**Abb. 2-28:** Display-Darstellung zum Schützen eines Programms

### Ausführung

Im nachfolgenden Beispiel wird das Programm Nr. 2 geschützt.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<div style="border: 1px solid black; padding: 5px; display: inline-block;"> <div style="display: flex; justify-content: space-between;"> <span>&lt;DIR&gt;</span> <span>7</span> </div> <div> 1            97-05-05  2            97-02-10  5            97-03-24 </div> </div>	3 x 	Der Schutzstatus des Programms wird angezeigt.
②	<div style="border: 1px solid black; padding: 5px; display: inline-block;"> <div style="display: flex; justify-content: space-between;"> <span>&lt;DIR&gt;</span> <span>PROTECT</span> </div> <div> 1            ON ( <b>1</b> )  2            OFF ( 0 )  0=OFF    1=ON </div> </div>	 	Der Programmschutz für Programm Nr. 2 wird eingeschaltet.
③	<div style="border: 1px solid black; padding: 5px; display: inline-block;"> <div style="display: flex; justify-content: space-between;"> <span>&lt;DIR&gt;</span> <span>PROTECT</span> </div> <div> 1            ON ( 1 )  2            OFF ( <b>1</b> )  0=OFF    1=ON </div> </div>	 	Die Eingabe des Programmschutzes wird bestätigt.
④	<div style="border: 1px solid black; padding: 5px; display: inline-block;"> <div style="display: flex; justify-content: space-between;"> <span>&lt;DIR&gt;</span> <span>PROTECT</span> </div> <div> 1            ON ( 1 )  2            ON ( <b>1</b> )  0=OFF    1=ON </div> </div>		Das Programm Nr. 2 ist jetzt geschützt.

**Tab. 2-17:** Beispiel zum Schützen eines Programms

### Beschreibung

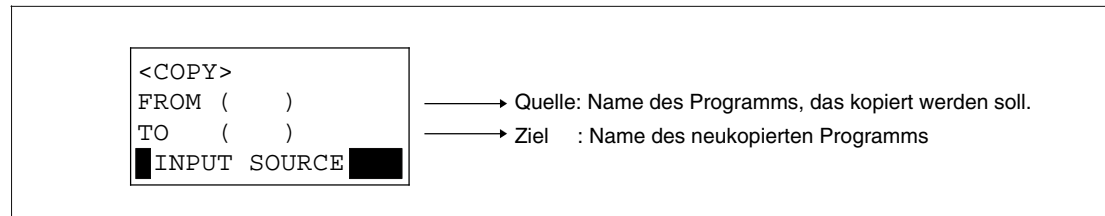
- Programme werden gegen die Operationen DELETE, RENAME und Befehlsänderungen geschützt.
- Beim Kopieren eines Programmes wird der Schutzstatus nicht mitkopiert.
- Beim Löschen des Speichers über „INITIALIZATION“ (siehe S. 2-24, Unterpkt. 5.2) wird der Schutzstatus ignoriert und das Programm gelöscht.

## 2.5.11 Roboterprogramm kopieren

### Funktion

Es kann ein Roboterprogramm kopiert werden.

### Display-Darstellung



**Abb. 2-29:** Display-Darstellung zum Kopieren eines Roboterprogramms

### Ausführung

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

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

**Tab. 2-18:** Beispiel zum Kopieren eines Programms

**Beschreibung**

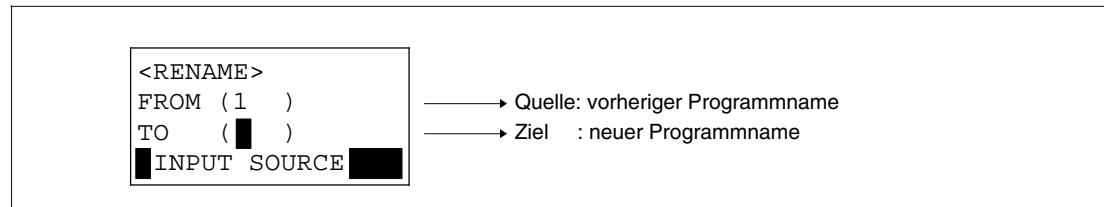
- Es wird eine Fehlermeldung angezeigt, wenn der gleiche Programmname zweimal angegeben wird, d. h. wenn Quelle und Ziel eines Kopiervorgangs identisch sind.
- Der Schutzstatus eines Programmes wird nicht mitkopiert.

## 2.5.12 Programmname ändern

### Funktion

Der Name eines Roboterprogramms kann geändert werden.

### Display-Darstellung



**Abb. 2-30:** Display-Darstellung zum Ändern eines Programmnamens

### Ausführung

Im nachfolgenden Beispiel wird der Programmname von „1“ auf „5“ geändert.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<div> &lt;MENU&gt;  1. TEACH    2. RUN  3. FILE    4. MONI  5. MAINT   6. SET </div>		Das Menü FILE wird ausgewählt.
②	<div> &lt;FILE&gt;  1. DIR    2. PROTECT  3. COPY   4. RENAME  5. DELETE </div>		Der Menüpunkt RENAME wird ausgewählt.
③	<div> &lt;RENAME&gt;  FROM ( )  TO ( )  INPUT SOURCE </div>		Der vorherige Programmname (Nr. 1) wird eingegeben.
④	<div> &lt;RENAME&gt;  FROM ( 1 )  TO ( )  INPUT DEST. </div>		Der neue Programmname (Nr. 5) wird eingegeben.
⑤	<div> &lt;FILE&gt;  1. DIR    2. PROTECT  3. COPY   4. RENAME  5. DELETE </div>		Nach Abschluß des Änderungsvorgangs wird das Menü FILE wieder angezeigt.

**Tab. 2-19:** Beispiel zum Ändern eines Programmnamens

### Beschreibung

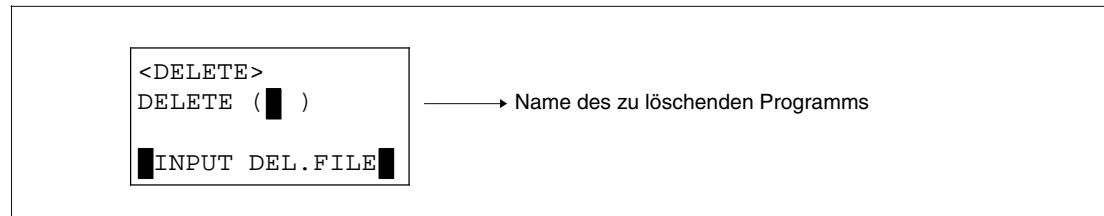
- Wird derselbe Programmname zweimal angegeben, erfolgt eine Fehlermeldung.
- Der Programmname eines geschützten Programmes kann nicht geändert werden.

## 2.5.13 Roboterprogramm löschen

### Funktion

Ein abgespeichertes Roboterprogramm kann gelöscht werden.

### Display-Darstellung



**Abb. 2-31:** Display-Darstellung zum Löschen eines Roboterprogramms

### Ausführung

Im nachfolgenden Beispiel wird das Programm Nr. 1 gelöscht.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre>&lt;MENU&gt; 1. TEACH    2. RUN 3. FILE     4. MONI 5. MAINT    6. SET</pre>		Das Menü FILE wird ausgewählt.
②	<pre>&lt;FILE&gt; 1. DIR      2. PROTECT 3. COPY     4. RENAME 5. DELETE</pre>		Der Menüpunkt DELETE wird ausgewählt.
③	<pre>&lt;DELETE&gt; DELETE ( ) INPUT DEL.FILE</pre>		Der Name des zu löschen- den Programms wird eingegeben. Nach der Eingabe wird eine Bestätigungsabfrage angezeigt.
④	<pre>&lt;DELETE&gt; DELETE 1           OK ? ( ) 1: EXECUTE</pre>		Die Abfrage wird durch Eingabe der Ziffer „1“ bestätigt.
⑤	<pre>&lt;FILE&gt; 1. DIR      2. PROTECT 3. COPY     4. RENAME 5. DELETE</pre>		Nach Abschluß des Löschvorgangs wird das Menü FILE wieder angezeigt.

**Tab. 2-20:** Beispiel zum Löschen eines Roboterprogramms

### Beschreibung

- Ein geschütztes Programm kann nicht gelöscht werden.



## 2.5.14 Monitor-Funktion für Eingangssignale

### Funktion

Es kann der Status der Eingangssignale überprüft werden.




### Display-Darstellung

<pre>&lt;INPUT&gt; NUMBER (    ) BIT :76543210 DATA(00000000)</pre>	<p>————→ Signaltyp (Ein- oder Ausgangssignal)</p> <p>————→ Bitnummer</p> <p>————→ aktueller Zustand der Ein-/ oder Ausgangssignale (0. AUS, 1: EIN)</p>
---	---

**Abb. 2-32:** Display-Darstellung zur Monitor-Funktion für Eingangssignale

### Ausführung

Im nachfolgenden Beispiel wird der Status der ersten acht Eingangsbits angezeigt.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre>&lt;MENU&gt; 1. TEACH    2. RUN 3. FILE     4. MONI 5. MAINT    6. SET</pre>		Der Menüpunkt MONITOR wird ausgewählt.
②	<pre>&lt;MONI&gt; 1. INPUT    2. OUTPUT 3. VAR      4. ALARM</pre>		Der Menüpunkt INPUT wird ausgewählt.
③	<pre>&lt;INPUT&gt; NUMBER (    ) BIT :76543210 DATA(00000000)</pre>		Der Bitstatus wird angezeigt.

**Tab. 2-21:** Beispiel zum Überprüfen der Bitzustände der Eingangssignale

## 2.5.15 Monitor-Funktion für die Ausgangssignale

### Funktion

Es können die Bedingungen für die Ausgangssignale überprüft und eingestellt werden.







### Display-Darstellung

<div style="border: 1px solid black; padding: 5px; width: fit-content;">         &lt;OUTPUT&gt;          NUMBER (        )          BIT :76543210          DATA(00000000)       </div>	<div style="display: flex; align-items: center;"> <div style="width: 20px; border-bottom: 1px solid black; margin-right: 5px;"></div>         Signaltyp (Ein- oder Ausgangssignal)       </div> <div style="display: flex; align-items: center;"> <div style="width: 20px; border-bottom: 1px solid black; margin-right: 5px;"></div>         Bitnummer       </div> <div style="display: flex; align-items: center;"> <div style="width: 20px; border-bottom: 1px solid black; margin-right: 5px;"></div>         aktueller Zustand der Ein-/ oder Ausgangssignale (0. AUS, 1: EIN)       </div>
--	---

**Abb. 2-33:** Display-Darstellung zur Monitor-Funktion für die Ausgangssignale

### Ausführung

Im nachfolgenden Beispiel wird das 3. Ausgangsbit eingeschaltet.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<div style="border: 1px solid black; padding: 5px; width: fit-content;">           &lt;MENU&gt;            1. TEACH    2. RUN            3. FILE     4. MONI            5. MAINT    6. SET         </div>		Der Menüpunkt MONITOR wird ausgewählt.
②	<div style="border: 1px solid black; padding: 5px; width: fit-content;">           &lt;MONI&gt;            1. INPUT   2. OUTPUT            3. VAR     4. ALARM         </div>		Der Menüpunkt OUTPUT wird ausgewählt.
③	<div style="border: 1px solid black; padding: 5px; width: fit-content;">           &lt;OUTPUT&gt;            NUMBER ( ■     )            BIT :76543210            DATA(00000000)         </div>		Der Bitstatus wird angezeigt.
④	<div style="border: 1px solid black; padding: 5px; width: fit-content;">           &lt;OUTPUT&gt;            NUMBER ( 0     )            BIT: 76543210            DATA( ■ 0000000)         </div>	4 x  	Der Wert „1“ wird im Datenfeld an der 3. Stelle eingegeben.
⑤	<div style="border: 1px solid black; padding: 5px; width: fit-content;">           &lt;OUTPUT&gt;            NUMBER ( 0     )            BIT :76543210            DATA(0000 1 000)         </div>		Die Dateneingabe wird bestätigt.
⑥	<div style="border: 1px solid black; padding: 5px; width: fit-content;">           &lt;OUTPUT&gt;            NUMBER ( ■     )            BIT :76543210            DATA(00001000)         </div>		Der Signalzustand von Bit 3 ist auf „1“ gesetzt.

**Tab. 2-22:** Beispiel zum Einstellen der Bitzustände der Ausgangssignale

## 2.5.16 Monitor-Funktion für Variablen

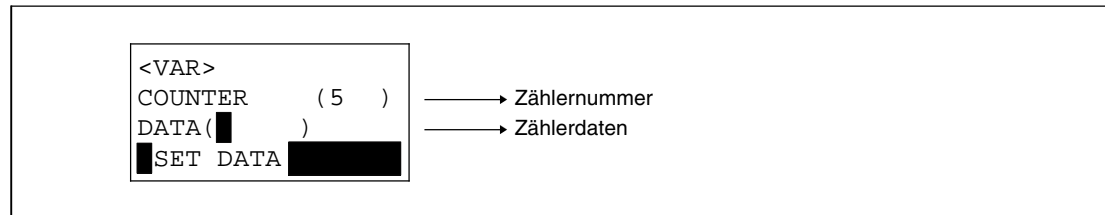
### Funktion

Es können die Variablenwerte überprüft und neu festgelegt werden.

In der MOVEMASTER COMMAND-Programmiermethode bedeutet eine Variable eine Zählernummer, in der MELFA-BASIC III-Programmiermethode eine arithmetische Variable. Während eines Interrupt-Status kann der Inhalt einer Variablen überprüft und geändert werden.

Die Monitor-Funktion für Variablen kann nur im gestoppten Betriebszustand benutzt werden.

### Display-Darstellung



**Abb. 2-34:** Display-Darstellung zur Monitor-Funktion für Variablen

### Ausführung

Im nachfolgenden Beispiel wird der Datenwert für den Zähler Nr. 5 auf „10“ eingestellt.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre>&lt;MENU&gt; 1. TEACH    2. RUN 3. FILE     4. MONI 5. MAINT    6. SET</pre>		Der Menüpunkt MONITOR wird ausgewählt.
②	<pre>&lt;MONI.&gt; 1. INPUT  2. OUTPUT 3. VAR     3. ALARM</pre>		Der Menüpunkt VAR wird ausgewählt.
③	<pre>&lt;VAR.&gt; COUNTER ( ) DATA ( ) SET DATA</pre>	→	Die Zählernummer (5) wird eingegeben. Anschließend wird der Cursor zum Dateneingabefeld bewegt.
④	<pre>&lt;VAR.&gt; COUNTER ( 5 ) DATA ( ) SET DATA</pre>	→  →	Der neue Datenwert (10) wird eingegeben.
⑤	<pre>&lt;VAR.&gt; COUNTER ( 5 ) DATA ( 10 ) SET DATA</pre>		Für den Zähler Nr. 5 wurde der Datenwert auf „10“ eingestellt.

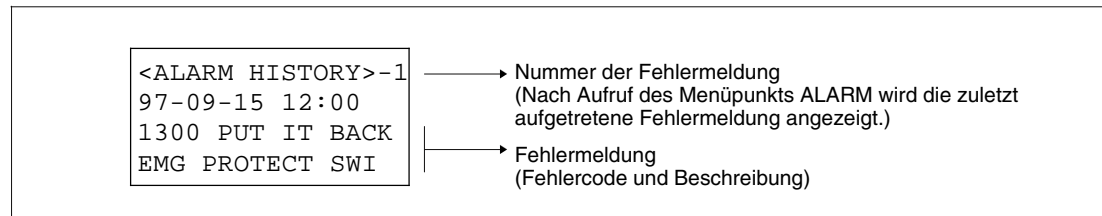
**Tab. 2-23:** Beispiel zum Ändern von Variablenwerten

## 2.5.17 Liste der aufgetretenen Fehlermeldungen

### Funktion





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

### Display-Darstellung



**Abb. 2-35:** Display-Darstellung zum Menüpunkt ALARM

### Ausführung

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<div style="border: 1px solid black; padding: 5px; display: inline-block;"> <code>&lt;MENU&gt; 1.TEACH    2.RUN 3.FILE    4.MONI 5.MAINT   6.SET</code> </div>		Der Menüpunkt MONITOR wird ausgewählt.
②	<div style="border: 1px solid black; padding: 5px; display: inline-block;"> <code>&lt;MONI.&gt; 1.INPUT   2.OUTPUT 3.VAR    4.ALARM</code> </div>		Der Menüpunkt ALARM wird ausgewählt.
③	<div style="border: 1px solid black; padding: 5px; display: inline-block;"> <code>&lt;ALARM HISTORY&gt;-1 97-09-15 12:00 1300 PUT IT BACK EMG PROTECT SWI</code> </div>	 	Mit den Cursortasten können die weiteren Fehlermeldungen zur Anzeige gebracht werden.

**Tab. 2-24:** Beispiel zum Anzeigen der Liste der aufgetretenen Fehlermeldungen

### Beschreibung

- Die Robotersteuerung kann bis zu 128 Fehlermeldungen abspeichern.
- Wenn mehr als 128 Fehlermeldungen auftreten, wird jeweils die zeitlich am weitesten zurückliegende Fehlermeldung überschrieben und durch die neue Fehlermeldung ersetzt (First In First Out).

## 2.5.18 Parameter anzeigen/einstellen

### Funktion

Es können Parameter, wie Geschwindigkeit und Werkzeuglänge, für den Jog- oder den Automatikbetrieb angezeigt oder gesetzt werden. Bei Auslieferung sind die Standardwerte wirksam. Weitere Informationen über Parameter finden Sie im Anhang A.2 „Übersicht der Parameter“.



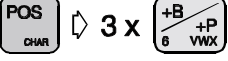
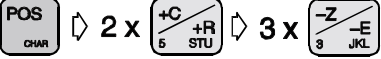

### Display-Darstellung

<pre>&lt;PARAM&gt; (XTL  ) ( 3  ) (123.00      ) SET DATA</pre>	<p>—————&gt; Parameternamen und Nummer des Elements (Achsennummer)</p>
	<p>—————&gt; Eingestellter Parameterwert</p>

**Abb. 2-36:** Display-Darstellung zur Anzeige eines Parameters

### Ausführung

Im nachfolgenden Beispiel wird der Wert des Parameters „XTL“ (Werkzeugdaten) auf der Z-Achse auf 100 mm gesetzt.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre>&lt;MENU&gt; 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>&lt;MAINT&gt; 1. PARAM    2. INIT 3. BRAKE    4. Z 5. ORIGIN   6. POWER</pre>		Das Menü PARAMETER wird durch Eingabe der Ziffer 1 aufgerufen.
③	<pre>&lt;PARAM&gt; (  ) (  ) (  ) SET PARAM.NAME</pre>		Das Zeichen „X“ wird eingegeben.
④	<pre>&lt;PARAM&gt; (X  ) (  ) (  ) SET PARAM.NAME</pre>		Die Zeichen „T“ und „L“ werden eingegeben.
⑤	<pre>&lt;PARAM&gt; (XTL  ) (  ) (  ) SET PARAM.NAME</pre>		Der Cursor wird zur Achsennummer bewegt.

**Tab. 2-25:** Beispiel zum Einstellen eines Parameters (1)

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
⑥	<pre> &lt;PARAM&gt; (XTL  ) (  ) ( SET ELEMENT NO. </pre>		Es wird die Achsennummer „3“ eingegeben.
⑦	<pre> &lt;PARAM&gt; (XTL  ) (3  ) ( SET ELEMENT NO. </pre>		Die Eingabe wird bestätigt.
⑧	<pre> &lt;PARAM&gt; (XTL  ) (3  ) (123.00 ) SET DATA </pre>		Es wird der Wert der Z-Achse des Parameters „XTL“ angezeigt.
⑨	<pre> &lt;PARAM&gt; (XTL  ) (3  ) (123.00 ) SET DATA </pre>		Es wird der neue Wert „100“ eingegeben.
	<pre> &lt;PARAM&gt; (XTL  ) (3  ) (100.00 ) SET DATA </pre>		Der neue Wert „100.00“ wird bestätigt.
⑪	<pre> &lt;PARAM&gt; (XTL  ) (3  ) (100.00 ) SET PARAM.NAME </pre>		Schalten Sie die Spannungsversorgung aus und wieder ein, damit der neue Wert wirksam werden kann.

**Tab. 2-25:** Beispiel zum Einstellen eines Parameters (2)

### Beschreibung

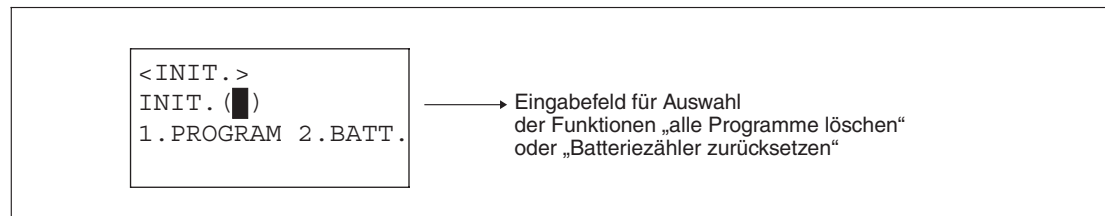
- Damit der neue Wert eines Parameters wirksam wird, muß die Spannungsversorgung einmal aus- und wiedereingeschaltet werden. Bis dahin ist der alte Parameterwert gültig. (Die Parameter UNG und HOE sind direkt nach ihrer Änderung wirksam.)
- Verwenden Sie einen nicht existierenden Parameternamen, so wird er durch einen Namen, der ihm in der alphabetischen Reihenfolge am nächsten kommt, ersetzt.
- Betätigen Sie bei einer fehlerhaften Eingabe die [POS/CHAR]-Taste und gleichzeitig die [DEL ←]-Taste, um das falsche Zeichen zu löschen.

## 2.5.19 Alle gespeicherten Programme löschen

### Funktion

Es können alle abgespeicherten Roboterprogramme mit einem Bedienschritt gelöscht oder der Batteriezähler nach dem Austausch der Pufferbatterien zurückgesetzt werden.

### Display-Darstellung



**Abb. 2-37:** Display-Darstellung zum Löschen aller gespeicherten Roboterprogramme oder des Batteriezählers

### Ausführung

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<div> &lt;MENU&gt;  1. TEACH    2. RUN  3. FILE    4. MONI  5. MAINT    6. SET </div>		Das Menü MAINTENANCE wird durch Eingabe der Ziffer 5 aufgerufen.
②	<div> &lt;MAINT&gt;  1. PARAM    2. INIT.  3. BRAKE    4. Z  5. ORIGIN </div>		Der Menüpunkt INIT wird ausgewählt.
③	<div> &lt;INIT&gt;  INIT. (■)  1. PROGRAM    2. BATT. </div>	oder	Nach der Tastenbetätigung wird eine Bestätigungsabfrage angezeigt.
④	<div> &lt;INIT&gt;  PROGRAM                    OK ? (■)  1: EXECUTE </div>	→	Die Abfrage wird durch Eingabe der Ziffer „1“ bestätigt. Betätigen Sie die SPD-Taste der Teaching Box zum Abbrechen eines Löschvorgangs auf.
⑤	<div> &lt;MAINT&gt;  1. PARAM    . INIT.  3. BRAKE    2. Z  5. ORIGIN </div>		Nach Abschluß des Löschvorgangs wird das Menü FILE wieder angezeigt.

**Tab. 2-26:** Beispiel zum Löschen aller gespeicherten Roboterprogramme oder des Batteriezählers

#### HINWEIS

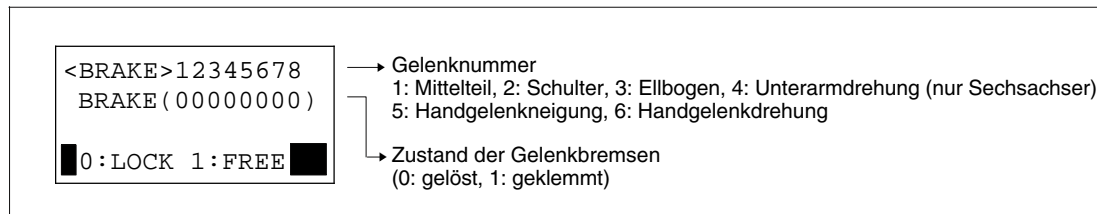
| Es werden auch geschützte Programme gelöscht!

## 2.5.20 Gelenkbremsen lösen

### Funktion

Die Bremsen für die Roboterarme können bei ausgeschalteter Servospannung gelöst werden. Da bei gelösten Gelenkbremsen die Steuerung ausgeschaltet ist, können Sie den Roboterarm aus dem Arbeitsbereich herausschwenken.

### Display-Darstellung



**Abb. 2-38:** Display-Darstellung zum Lösen der Gelenkbremsen

### Ausführung



#### ACHTUNG


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

Im nachfolgenden Beispiel wird die Bremse für das Schultergelenk gelöst. Betätigen Sie während dieser Operation den Totmannschalter auf der Rückseite der Teaching Box. Bei nicht betätigtem Totmannschalter sind die Bremsen wieder aktiv.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre>&lt;MENU&gt; 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>&lt;MAINT&gt; 1. PARAM.  2. INIT. 3. BRAKE   4. Z 5. ORIGIN  6. POWER</pre>		Der Menüpunkt BRAKE wird ausgewählt.
③	<pre>&lt;BRAKE&gt;12345678 BRAKE(00000000) 0:LOCK 1:FREE</pre>		Der Cursor wird zur zweiten Eingabestelle (Schultergelenk) bewegt. Danach wird der Wert „1“ für Bremsen lösen eingegeben.

**Tab. 2-27:** Beispiel zum Lösen der Gelenkbremsen (1)



Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
④	<pre>&lt;BRAKE&gt;12345678 BRAKE(01000000)  0:LOCK 1:FREE</pre>		<p>Betätigen Sie die Tasten zusammen mit dem Totmannschalter. Die Bremsen sind nur gelöst, solange der Totmannschalter betätigt ist.</p>
⑤	<pre>&lt;BRAKE&gt;12345678 BRAKE(01000000)  0:LOCK 1:FREE</pre>		<p>Es sind wieder alle Bremsen aktiv, sobald der Totmannschalter oder die [+X/+W]-Taste losgelassen werden.</p>

**Tab. 2-27:** Beispiel zum Lösen der Gelenkbremsen (2)

### Beschreibung

- Wird der Totmannschalter losgelassen, während die Bremsen gelöst sind, werden die Bremsen sofort wieder aktiv.
- Betätigen Sie zum Lösen aller Bremsen die Taste [STEP/MOVE] zusammen mit der Taste [+X/+W]. Die Bremsen werden automatisch wieder aktiv, sobald Sie die Taste [+X/+W] loslassen.

## 2.5.21 Encoder zurücksetzen

### Funktion

Es können die Encoder zurückgesetzt und die Z-Phasen-Signale der Encoder angezeigt werden. Nach einem Ausfall der Backup-Batterie können von der Robotersteuerung Fehlermeldungen der Encoder nicht mehr angezeigt werden. Deshalb sollten Sie in diesem Fall die Z-Phasen-Erkennung überprüfen. Der Encoder muß nach einer Encoder-Fehlermeldung zurückgesetzt werden.






### Display-Darstellung

<pre> &lt;Z&gt;    12345678 BRAKE (1111110 ) Z RST (00000000) Z PHS : ZZZZZZ- </pre>	<p>→ Gelenknummer 1: Mittelteil, 2: Schulter, 3: Ellbogen, 4: Unterarmdrehung (nur Sechsscher) 5: Handgelenkneigung, 6: Handgelenkdrehung</p> <p>→ Zuordnung der rückgesetzten Encoder-Signale für die Gelenke</p> <p>→ Anzeige der Z-Phasen-Erkennung für die einzelnen Gelenke</p>
--	--




**Abb. 2-39:** Display-Darstellung zur Z-Phasen-Erkennung

### Ausführung

Im nachfolgenden Beispiel werden die Encoder zurückgesetzt.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre> &lt;ALARM&gt;    No311 Encoder error (miss count) PLEASE ALM RST </pre>		Der [ENABLE/DISABLE]-Schalter wird auf ENBL. geschaltet. Das Display zur Menüauswahl wird angezeigt.
②	<pre> &lt;MENU&gt; 1. TEACH    2. RUN 3. FILE     4. MONI 5. MAINT    6. SET </pre>		Das Menü MAINTENANCE wird aufgerufen.
③	<pre> &lt;MAINT&gt; 1. PARAM    2. INIT 3. BRAKE    4. Z 5. ORIGIN   6. POWER </pre>		Der Menüpunkt Z wird ausgewählt.
④	<pre> &lt;Z&gt; SERVO OFF       OK ? ( ) 1: EXECUTE </pre>	 	Der Servoantrieb wird ausgeschaltet.
⑤	<pre> &lt;Z&gt;    12345678 BRAKE (00000000) ZRST (00000000) ZPHS : ZZZZZZ- </pre>		Der Cursor wird auf die Zeile für das Zurücksetzen der Z-Phase positioniert.

**Tab. 2-28:** Beispiel zum Zurücksetzen der Encoder (1)

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
⑥	<pre> &lt;Z&gt;      12345678 BRAKE (00000000) Z RST (11111100) Z PHS :ZZZZZZ- </pre>	6 x  	Es wird für jede Achse, die zurückgesetzt werden soll, die Ziffer „1“ eingetragen. Durch Betätigen der [INP/EXE]-Taste wird die Rückstellung aktiviert.
⑦	<pre> &lt;Z&gt;      12345678 BRAKE (00000000) Z RST (11111100) EXECUTING </pre>		Der Encoder wird zurückgesetzt.
⑧	<pre> &lt;ALARM&gt;   No311 Encoder error (miss count) PLEASE ALM RST </pre>		Die Fehlermeldung wird zurückgesetzt.
⑨	<pre> &lt;Z&gt;      12345678 BRAKE (00000000) Z RST (00000000) Z PHS :ZZZZZZ- </pre>		Der Encoder ist jetzt zurückgesetzt.

**Tab. 2-28:** Beispiel zum Zurücksetzen der Encoder (2)

2.5.22 Batterie und Einschaltzeit anzeigen

Funktion

Zeigt die Betriebszeit des Steuergerätes und die verbleibende Lebensdauer der Batterie in Stunden an.

Display Darstellung

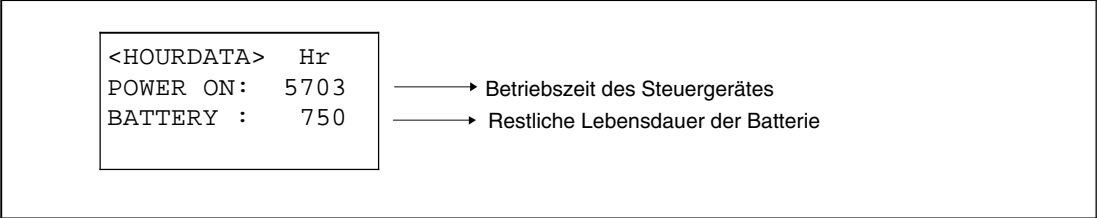


Abb. 2-40: Display-Darstellung Einschaltzeit und Restpufferung

Ausführung

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<div>&lt;MENU&gt; 1. TEACH 2. RUN 3. FILE 4. MONI 5. MAINT 6. SET</div>	<div>+C 5 +R STU</div>	Das Menü MAINTENANCE wird aufgerufen.
②	<div>&lt;MAINT&gt; 1. PARAM 2. INIT 3. BRAKE 4. Z 5. ORIGIN 6. POWER</div>	<div>+B 6 +P VWX</div>	Die Wartungsdaten werden angezeigt.
③	<div>&lt;HOURLDATA&gt; Hr POWER ON: 5703 Battery : 750</div>		Es werden die Betriebszeit und die verbleibende Lebensdauer der Batterie angezeigt.

Tab. 2-29: Beispiel zum Anzeigen der Einschaltzeit und Restpufferung

## 2.5.23 Uhrzeit/Datum einstellen

### Funktion

Die Robotersteuerung ist mit einer internen Uhr für Uhrzeit- und Datumsfunktionen ausgerüstet. Diese Datumsfunktion nutzt die Robotersteuerung 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.

### Display-Darstellung

<div style="border: 1px solid black; padding: 5px; width: fit-content;"> <pre>&lt;CLOCK&gt; DATE ( 97-10-10 ) TIME ( 15:00:00 ) INPUT DATE</pre> </div>	<div style="display: flex; align-items: center; margin-top: 10px;"> <div style="width: 20px; border-bottom: 1px solid black; margin-right: 5px;"></div> <div>aktuelles Datum</div> </div> <div style="display: flex; align-items: center; margin-top: 5px;"> <div style="width: 20px; border-bottom: 1px solid black; margin-right: 5px;"></div> <div>aktuelle Uhrzeit</div> </div>
---	---

**Abb. 2-41:** Display-Darstellung zum Einstellen von Datum/Uhrzeit

### Ausführung

Im nachfolgenden Beispiel wird die Uhrzeit für die Robotersteuerung auf 15:00 eingestellt.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<div style="border: 1px solid black; padding: 5px; width: fit-content;"> <pre>&lt;MENU&gt; 1. TEACH    2. RUN 3. FILE     4. MONI 5. MAINT    6. SET</pre> </div>		Das Menü SET wird ausgewählt.
②	<div style="border: 1px solid black; padding: 5px; width: fit-content;"> <pre>&lt;SET&gt; 1. CLOCK</pre> </div>		Im Menü SET wird der Menüpunkt CLOCK aufgerufen. Nach der Tastenbetätigung wird die aktuelle Uhrzeit und das aktuelle Datum angezeigt.
③	<div style="border: 1px solid black; padding: 5px; width: fit-content;"> <pre>&lt;CLOCK&gt; DATE ( 97-10-10 ) TIME ( 12:34:23 ) INPUT DATE</pre> </div>	→   →	Der Cursor wird zum Eingabefeld TIME bewegt. Dort wird die Uhrzeit auf „15:00:00“ eingestellt.
④	<div style="border: 1px solid black; padding: 5px; width: fit-content;"> <pre>&lt;CLOCK&gt; DATE ( 97-10-10 ) TIME ( 15:00:00 ) INPUT TIME</pre> </div>		Die Eingabe wird bestätigt.
⑤	<div style="border: 1px solid black; padding: 5px; width: fit-content;"> <pre>&lt;CLOCK&gt; DATE ( 97-10-10 ) TIME ( 15:00:00 ) INPUT DATE</pre> </div>		Die Uhrzeiteinstellung ist abgeschlossen.

**Tab. 2-30:** Beispiel zum Einstellen von Datum/Uhrzeit

**Beschreibung**

Lassen Sie die Spannungsversorgung nach Einstellen von Uhrzeit/Datum mindestens noch ca. 3 min eingeschaltet. Andernfalls könnte es zu einer Fehlermeldung kommen.

Die Funktion der Steuerung wird dadurch nicht beeinträchtigt.

2.5.24 Fehlermeldung quittieren

Funktion

Eine angezeigte Fehlermeldung wird gelöscht.

Anordnung der Taste

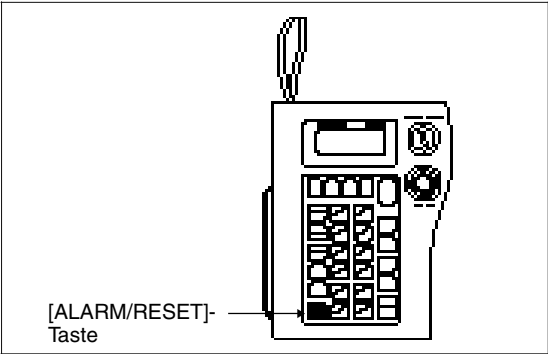


Abb. 2-42:  
Taste zum Rücksetzen einer Fehlermeldung

TB-BOX26

Ausführung

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<div>&lt;ALARM&gt; ALREADY PROGRAM EXIST █PLEASE ALM RST█</div>	<div>ALARM RESET</div>	Die angezeigte Fehlermeldung wird zurückgesetzt (quittiert).
②	<div>&lt;COPY&gt; FROM( 1 ) TO ( 5 ) █INPUT SOURCE█</div>		Nach dem Quittieren der Fehlermeldung wird der vorherige Originalbildschirm wieder angezeigt.

Tab. 2-31: Beispiel zum Rücksetzen einer Fehlermeldung

Beschreibung

- Anhang A.3 in diesem Handbuch enthält eine Übersicht über alle Fehlercodes und deren mögliche Störungsursachen.
- Die Servospannung wird von der Robotersteuerung nach Abschluß der Tasteneingaben nicht eingeschaltet. Schalten Sie die Servospannung über die RESET-Taste am Steuergerät ein.





# 3 Programmiermethoden

## 3.1 Einteilung der Programmiermethoden

Es stehen zwei Programmiermethoden zur Verfügung:

- MOVEMASTER COMMAND-Befehle
- MELFA-BASIC III-Befehle

Für beide Programmiermethoden benötigen Sie die Teaching Box und einen Personalcomputer.

Modus	Merkmal	Bemerkung
Programm-Modus	<p>Sie geben die Positionen über die Teaching Box ein und erstellen das Roboterprogramm mit einem PC (Angabe von Zeilennummer und Befehlen). Die MOVEMASTER-Befehle ermöglichen die Erstellung komplexer Anwendungsprogramme.</p> <p>Diese Methode ist besonders für Aufgaben geeignet, die Funktionen wie bedingte Verzweigungen, Programmunterbrechungen, Palettierung oder Berechnungen erfordern.</p>	<p>Es kann wahlweise die MELFA-BASIC III- oder die MOVEMASTER COMMAND-Programmierungsmethode verwendet werden. Die Auswahl erfolgt mittels dem Parameter RLNG.</p> <p>Diese Methode benötigt für die Programmerstellung einen PC mit einer Roboter-Programmiersoftware, z.B. COSIROP.</p> <p>Das Programm kann mit der Teaching Box angepaßt werden.</p>
Direkt-Modus	<p>Der Roboter führt die vom PC übertragenen MOVEMASTER-Befehle sofort schrittweise aus.</p> <p>In diesem Modus wird kein Speicher in der Robotersteuerung belegt. Die Speicherverwaltung erfolgt ausschließlich über den PC.</p>	<p>Es kann wahlweise die MELFA-BASIC III- oder die MOVEMASTER COMMAND-Programmierungsmethode verwendet werden. Die Auswahl erfolgt mittels dem Parameter RLNG.</p> <p>Der Roboter sollte immer mit dem Personalcomputer verbunden sein.</p> <p>Sie können den Roboter während der Ausführung eines Befehls nur durch Betätigung der STOP-Taste und des NOT-AUS-Schalters anhalten.</p>

**Tab. 3-1:** Programmiermodus

## 3.2 Eigenschaften der Programmiermethoden

Den verschiedenen Programmiermethoden lassen sich nachfolgende Eigenschaften zuordnen:

### **MOVEMASTER COMMAND-Programmiermethode**

Die MOVEMASTER COMMAND-Programmiermethode ist besonders für Aufgaben geeignet, die folgende Funktionen erfordern:

- bedingte Verzweigungen
- Programmunterbrechungen
- Palettierung

Die Programmiermethode ist kompatibel zu den MOVEMASTER-Befehlen der M1-, M2- und E-Serie.

### **MELFA-BASIC III-Programmiermethode**

Es kann die vom JIS (Japan Industrial Standards Committee) entwickelte Programmiersprache SLIM (Standard Language for Industrial Manipulators) verwendet werden.

Mit Hilfe der MELFA-BASIC III-Programmiermethode lassen sich komplexere Programme als mit der MOVEMASTER COMMAND-Programmiermethode erstellen.

Im nachfolgenden Abschnitt wird die Auswahl der Programmiermethode mittels der Parameter beschrieben. Eine detaillierte Beschreibung der Befehle finden Sie in Kapitel 5 (MOVEMASTER-Befehle) und Kapitel 8 (BASIC-Befehle).

## 3.3 Parametereinstellungen

### 3.3.1 Auswahl der Programmiermethode

Die Auswahl der Programmiermethode erfolgt über die Einstellung des Parameters RLNG:

- Setzen Sie den Wert auf „1“, um die MELFA-BASIC III-Programmierungsmethode zu wählen.
- Setzen Sie den Wert auf „0“, um die MOVEMASTER COMMAND-Programmierungsmethode zu wählen.

Wie Sie die Parameter einstellen, finden Sie in Abschnitt 2.4.18 „Parameter anzeigen/einstellen“.

#### HINWEIS

Nachdem Sie die Parametereinstellungen verändert haben, müssen Sie die Versorgungsspannung des Steuergerätes aus- und wieder einschalten, damit die neue Einstellung aktiv wird.

Parameter	Parametername	Zahl der Felder Zahl der Zeichen	Bemerkung	Standardwert
Programmierungsmethode	RLNG	Integer	Freigabe der MELFA-BASIC III-Programmierungsmethode 0: MOVEMASTER COMMAND 1: MELFA-BASIC III	0

**Tab. 3-2:** Parameterinstellung zur Programmierungsmethode

### 3.3.2 Hinweise zur Auswahl der Programmierungsmethode

- Werden beim Editieren eines Programmes oder bei der Eingabe über die Teaching Box Befehle verwendet, die nicht zu der gewählten Programmierungsmethode gehören, erfolgt eine Fehlermeldung.  
(Fehlernummer 2800: Fehler in der Eingabesyntax)  
Geben Sie in diesem Fall den korrekten Befehl ein.
- Wird ein Programm, das Befehle einer Programmierungsmethode verwendet, die nicht der gewählten Programmierungsmethode entsprechen über die Teaching Box editiert, erfolgt die Meldung „NO DATA“ auf dem Display der Teaching Box. Das Programm kann trotzdem gespeichert werden.  
Wird dieses Programm hinzugefügt, überschrieben oder gelöscht, erfolgt eine Fehlermeldung. Ein Editieren ist nicht mehr möglich.  
(Fehlernummer 2830: Programmiersprachen stimmen nicht überein)  
Wählen Sie in diesem Fall die richtige Programmierungsmethode (siehe Abschnitt 3.3.1).
- Bei Ausführung eines Programmes, das von der gewählten Programmierungsmethode abweicht, erfolgt eine Fehlermeldung.  
(Fehlernummer 2830: Programmiersprachen stimmen nicht überein)  
Wählen Sie in diesem Fall die richtige Programmierungsmethode (siehe Abschnitt 3.3.1), und führen Sie das Programm erneut aus.

- Beachten Sie folgende Punkte, wenn Sie ein Roboter-Programm mittels eines PC-Programms aus der Drive Unit lesen oder dorthinein schreiben:
  - Durch ein Auslesen werden nur Programme gesichert, die in der ausgewählten Programmiermethode erstellt worden sind (siehe Abschnitt 3.3.1). Wählen Sie die passende Programmiermethode, bevor Sie das Lesen durchführen.
  - Überprüfen Sie vor dem Schreiben, daß die Programmiermethoden übereinstimmen. Stimmen die Methoden nicht überein, erfolgt eine Fehlermeldung. (Fehlernummer 2800: Fehler in der Eingabesyntax)
  
- Bei einem Wechsel von MOVEMASTER COMMAND- zur MELFA-BASIC III-Programmierungsmethode:
  - Wenn keine MELFA-BASIC III-Datei mit gemeinsamen Variablen existiert, wird beim Öffnen des Programms eine Speicherkapazität von mindestens 5253 Bytes benötigt.
  - Ein Programm, das mit der MOVEMASTER COMMAND-Programmierungsmethode erstellt wurde, kann gespeichert, aber nicht editiert, ausgeführt oder gelesen werden.
  - Gemeinsame Variablen werden in das MOVEMASTER COMMAND-Format für MELFA-BASIC III übertragen.
  
- Bei einem Wechsel von MELFA-BASIC III- zur MOVEMASTER COMMAND-Programmierungsmethode:
  - Ein Programm, daß mit der MELFA-BASIC III-Programmierungsmethode erstellt wurde, kann gespeichert, aber nicht editiert, ausgeführt oder gelesen werden.
  - Gemeinsame Variablen werden in das MOVEMASTER COMMAND-Format für gemeinsame Variablen übertragen.
  - Die Datei für gemeinsame Variablen im MELFA-BASIC III-Format wird gespeichert. Winkelangaben von Positionskomponenten werden automatisch in Grad umgewandelt.

## 4 MOVEMASTER COMMAND-Programmierung

### 4.1 Programmierung mit der Teaching Box

Sie können ein Roboterprogramm unter Einsatz der MOVEMASTER-Befehle mit einem PC und der Teaching Box erstellen. Für eine komfortable Art der Erstellung ist der PC vorzuziehen. In diesem Abschnitt wird die MOVEMASTER COMMAND-Programmierungsmethode unter Verwendung der Teaching Box beschrieben.

Die nachfolgende Tabelle zeigt die Möglichkeiten der Weiterverarbeitung der Daten einer Zeile.

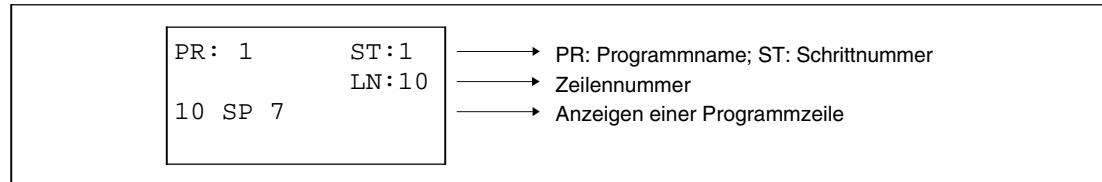
Eingabeformat	Verarbeitung
Zeilennummer und Befehl	Eingabe wird als Zeile des Roboterprogramms verarbeitet
Nur Zeilennummer	Löscht die angegebene Zeile aus dem Programm (durch Überschreiben)
Nur Befehl	Führt den Befehl sofort aus (Direkt-Modus)

**Tab. 4-1:** Weiterverarbeitung der übertragenen Daten einer Zeile

### 4.1.1 Roboterprogramm erstellen

In diesem Abschnitt wird die Erstellung eines neuen Programms beschrieben.

#### Display-Darstellung









**Abb. 4-1:** Display-Darstellung

#### Ausführung



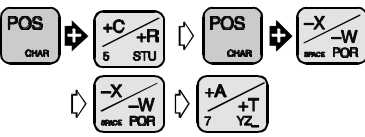

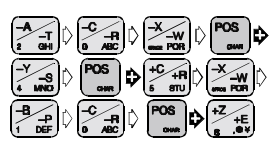
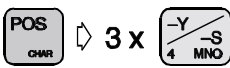

Folgendes Beispielprogramm Nr. 3 soll erstellt werden.

##### Beispielprogramm

10	SP	7
20	MS	10,O
30	TI	3
40	GT	20

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<div style="border: 1px solid black; padding: 5px;">                     &lt;MENU&gt;                      1. TEACH    2. RUN                      3. FILE     4. MONI.                      5. MAINT.   6. SET                 </div>	 	Das Menü TEACH zur Programmauswahl wird durch Eingabe der Ziffer 1 aufgerufen.
②	<div style="border: 1px solid black; padding: 5px;">                     &lt;TEACH&gt;                      ( 3 )                      ■ SELECT PROGRAM ■                 </div>	 	Die Programmnummer (3) wird eingegeben. Anschließend wird die Auswahl bestätigt.
③	<div style="border: 1px solid black; padding: 5px;">                     PR:3            ST:1                      INTP:---                      SPD :--                      TIME:0.0 sec                 </div>	  <p>Weitere mögliche Tastenbetätigungen:                      Taste [COND] + [RPL ↓] oder                      Taste [COND] + [DEL ←]</p>	Nach der Tastenbetätigung wird das Menü zur Programmeditierung aufgerufen. <b>HINWEIS:</b> Nach der Tastenbetätigung erfolgt eine Umschaltung in den MOVEMASTER COMMAND-Befehls-Modus!

**Tab. 4-2:** Erstellung eines neuen Programms (1)

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
④	PR: 3            ST: 1 LN: 0 - NO DATA -	3 x 	Der Cursor wird zum Eingabefeld für die Befehle bewegt.
⑤	PR: 3            ST: 1 LN: 0 CODE EDIT		Die Zeilennummer „10“ wird eingegeben.
⑥	PR: 3            ST: 1 LN: 0 10 CODE EDIT		Der Befehl „SP 7“ wird eingegeben.
⑦	PR: 3            ST: 1 LN: 0 10 SP 7 CODE EDIT		Jetzt ist die Programmzeile „10 SP 7“ wirksam.
⑧	PR: 3            ST: 2 LN: 0 CODE EDIT		Es wird die Programmzeile „20 MS 10“ eingegeben.
⑨	PR: 3            ST: 2 LN: 0 20 MS 10, CODE EDIT		Der Cursor wird hinter die „10“ bewegt, und es wird ein „O“ eingegeben.
⑩	PR: 3            ST: 2 LN: 0 CODE EDIT		Jetzt ist die neue Programmzeile „20 MS 10,O“ wirksam. Das Fenster zur Editierung des 3-ten Schrittes wird angezeigt. Wiederholen Sie die Schritte Nr. 5 bis Nr. 7, um das Programm fertigzustellen.

**Tab. 4-2:** Erstellung eines neuen Programmes (2)

## 4.1.2 Roboterprogramm editieren

Mit der Teaching Box kann ein zuvor erstelltes Roboterprogramm editiert werden. In diesem Abschnitt wird die Editierung eines Roboterprogramms beschrieben.

### ① Menü zur Programmeditierung öffnen

Wählen Sie das gewünschte Programm aus, und öffnen Sie anschließend das Menü zur Programmeditierung (siehe nachfolgendes Beispiel). Es wird automatisch das vorherige Programm im Editiermenü angezeigt, wenn Sie keine Programmauswahl vornehmen.

#### Display-Darstellung

<div style="border: 1px solid black; padding: 5px; display: inline-block;">                 PR: 1            ST:1                                   LN:10                  10 SP 7             </div>	→ PR: Programmname; ST: Schrittnummer → Zeilennummer → Anzeigen einer Programmzeile
---	---







**Abb. 4-2:** Display-Darstellung im Menü zur Programmeditierung

#### HINWEIS

Beachten Sie, daß nach Eingabe der Programmnummer mit anschließender Eingabebestätigung der MOVEMASTER COMMAND-Befehlsmodus aufgerufen wird.

#### Ausführung

In der unteren Tabelle wird das Programm Nr. 3 ausgewählt und anschließend das Menü zur Programmeditierung geöffnet.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<div style="border: 1px solid black; padding: 5px;">                     &lt;MENU&gt;                      1. TEACH    2. RUN                      3. FILE     4. MONI.                      5. MAINT.   6. SET                 </div>	 oder 	Das Menü TEACH zur Programmauswahl wird durch Eingabe der Ziffer 1 aufgerufen.
②	<div style="border: 1px solid black; padding: 5px;">                     &lt;TEACH&gt;                      ( 3 )  <div style="background-color: black; color: white; padding: 2px; display: inline-block;">SELECT PROGRAM</div> </div>	 → 	Die Programmnummer (3) wird eingegeben. Anschließend wird die Auswahl bestätigt.
③	<div style="border: 1px solid black; padding: 5px;">                 PR:3            ST:1                  INTP:---                  SPD :--                  TIME:0.0 sec             </div>	 →  Weitere mögliche Tastenbetätigungen: Taste [COND] + [RPL ↓] oder Taste [COND] + [DEL ←]	Nach der Tastenbetätigung wird das Menü zur Programmeditierung aufgerufen. <b>HINWEIS:</b> Nach der Tastenbetätigung erfolgt eine Umschaltung in den MOVEMASTER COMMAND-Befehls-Modus!
④	<div style="border: 1px solid black; padding: 5px;">                 PR:3            ST:1                                   LN:10                  10 SP 7             </div>		Nach der Tastenbetätigung wird das Menü zur Programmeditierung aufgerufen. Es wird die erste Programmzeile angezeigt.

**Tab. 4-3:** Beispiel zum Öffnen des Menüs zur Programmeditierung

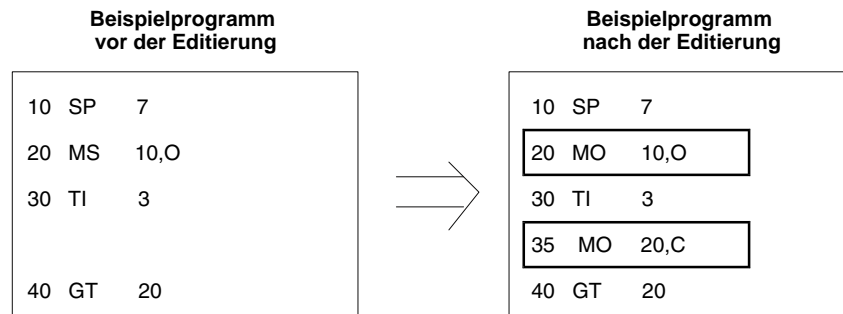















## ② Programm editieren

Nehmen Sie die Programmeditierungen vor.
















### Ausführung

Im nachfolgenden Beispiel wird die Editierung eines Programmes gezeigt.



Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<div style="border: 1px solid black; padding: 5px;"> PR:3 ST:1 LN:10 10 SP 7 </div>	2 x 	Der Cursor wird zum Eingabefeld für die Zeilennummer bewegt.
②	<div style="border: 1px solid black; padding: 5px;"> PR:3 ST:1 LN:20 10 SP 7 LINE NUMBER </div>	    	Die Zeilennummer (20) wird eingegeben. Danach wird Eingabe zur Editierung der Programmzeile aufgerufen (CODE EDIT).
③	<div style="border: 1px solid black; padding: 5px;"> PR:3 ST:2 LN:20 20 MS 10,O CODE EDIT </div>	5 x 	Der Cursor wird nach rechts bewegt und eine Stelle hinter „S“ platziert.
④	<div style="border: 1px solid black; padding: 5px;"> PR:3 ST:2 LN:20 20 MS 10,O CODE EDIT </div>	 	Das Zeichen „S“ wird gelöscht.
⑤	<div style="border: 1px solid black; padding: 5px;"> PR:3 ST:2 LN:20 20 M 10,O CODE EDIT </div>	 3 x 	Das neue Zeichen „O“ wird eingegeben.
⑥	<div style="border: 1px solid black; padding: 5px;"> PR:3 ST:2 LN:20 20 MO 10,O CODE EDIT </div>	 	Die Programmzeile „20 MO 10,O“ ist wirksam.

**Tab. 4-4:** Beispiel zum Editieren eines Programms (1)

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
⑦	PR:3 ST:2 LN:20 20 MO 10,O CODE EDIT	2 x 	Der Cursor wird nach rechts bewegt und eine Stelle hinter „0“ platziert.
⑧	PR:3 ST:2 LN:20 20 MO 10,O CODE EDIT	 2 x 	Die Zeilennummer „20“ wird gelöscht.
⑨	PR:3 ST:2 LN:20 MO 10,O CODE EDIT	 	Die Zeilennummer „35“ wird eingegeben.
⑩	PR:3 ST:2 LN:20 35MO 1,O CODE EDIT	5 x   	Der Cursor wird auf „0“ platziert. Anschließend wird das Zeichen „1“ gelöscht.
⑪	PR:3 ST:2 LN:20 35 MO 20,O CODE EDIT		Das neue Zeichen „2“ wird eingegeben.
⑫	PR:3 ST:2 LN:20 35 MO 20,O CODE EDIT	4 x   	Der Cursor wird nach rechts bewegt und eine Stelle hinter „0“ platziert. Anschließend wird das Zeichen „O“ gelöscht.
⑬	PR:3 ST:2 LN:20 35 MO 20,C CODE EDIT	 3 x 	Das neue Zeichen „C“ wird eingegeben.
⑭	PR:3 ST:2 LN:20 35 MO 20,C CODE EDIT		Jetzt ist die neue Programmzeile „35 MO 20,C“ wirksam.
⑮	PR:3 ST:5 LN:30 40 GT 20 CODE EDIT		Die nächste Programmzeile wird angezeigt.

Tab. 4-4: Beispiel zum Editieren eines Programms (2)

### Beschreibung

- 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.
- 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.
- Geben Sie am Eingabefeld „LN“ die gewünschte Programmzeile an.
- Mit den Tasten [+ /FORWD] (vorwärts) und [- /BACKWD] (rückwärts) können Sie im Programm „blättern“.
- Nach erfolgter Editierung einer Zeile wird dies auf dem Display angezeigt. Bei einer Überschreibung wird die überschriebene Zeile angezeigt. Nach Einfügen einer neuen Zeile wird die nächste Zeile angezeigt.
- Es besteht die Möglichkeit, ein neues Programm ohne Einsatz eines PCs zu erstellen. Fügen Sie hierzu eine neue Zeile in ein noch nicht belegtes Programm ein.
- Das Menü zur Programmeditierung läßt sich, durch Betätigung der Tasten [COND] + [ADD ↑], von jedem anderen Menü aus öffnen.
- Die nachfolgende Tabelle zeigt die Möglichkeiten der Weiterverarbeitung der mit einem PC übertragenen Daten einer Zeile.

Eingabeformat	Verarbeitung
Zeilennummer und Befehl	Eingabe wird als Zeile des Roboterprogramms verarbeitet
Nur Zeilennummer	Löscht die angegeben Zeile aus dem Programm (durch Überschreiben)
Nur Befehl	Führt den Befehl sofort aus (Direkt-Modus)

**Tab. 4-5:** Weiterverarbeitung der übertragenen Daten einer Zeile

### 4.1.3 Positionsdaten eingeben, anfahren, angleichen, ändern und löschen

#### ① Eingabe der aktuellen Positionsdaten

Die aktuelle Position (Koordinatenwerte) und der Status des Handgreifers können in einer Positionsnummer abgespeichert werden.

Die Vorgehensweise ist die gleiche wie bei der MELFA-BASIC III-Programmierung. Siehe Abschnitt 6.1.3 in diesem Handbuch: „MELFA-BASIC III-Programmierung, Positionsdaten eingeben“.

#### ② Positionsdaten anfahren

Die gespeicherte Positionsnummer wird angefahren.

Die Vorgehensweise ist die gleiche wie bei der MELFA-BASIC III-Programmierung. Siehe Abschnitt 6.1.3 in diesem Handbuch: „MELFA-BASIC III-Programmierung, Position anfahren“.

#### ③ Positionsdaten angleichen

Gleicht eine gespeicherte Position und den Handgreiferstatus an die aktuelle Position an.

Die Vorgehensweise ist die gleiche wie bei der MELFA-BASIC III-Programmierung. Siehe Abschnitt 6.1.3 in diesem Handbuch: „MELFA-BASIC III-Programmierung, Position angleichen“.

#### ④ Positionsdaten ändern

Es können die Positionsdaten einer definierten Position geändert werden. Der gespeicherte Handgreiferzustand wird auf „geschlossen“ gesetzt.

Die Vorgehensweise ist die gleiche wie bei der MELFA-BASIC III-Programmierung. Siehe Abschnitt 6.1.3 in diesem Handbuch: „MELFA-BASIC III-Programmierung, Positionsdaten ändern“.

#### ⑤ Positionsdaten löschen

Es können die Positionsdaten einer definierten Position gelöscht werden.

Die Vorgehensweise ist die gleiche wie bei der MELFA-BASIC III-Programmierung. Siehe Abschnitt 6.1.3 in diesem Handbuch: „MELFA-BASIC III-Programmierung, Positionsdaten löschen“.

4.1.4 Programmzeile direkt aufrufen

Funktion

Es kann eine definierte Programmzeile über die Angabe der Zeilennummer aufgerufen werden. Mit Hilfe dieser Funktion können Sie das Programm testen.

Anordnung der Tasten

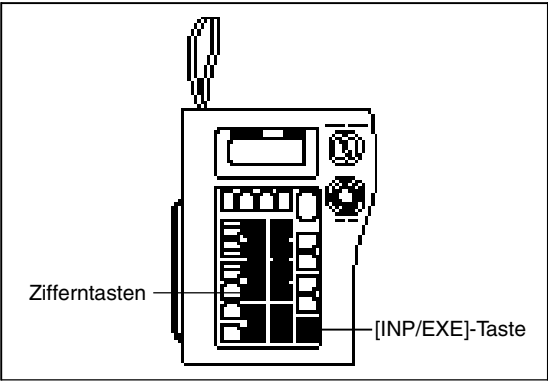


Abb. 4-3:  
Tasten zum direkten Aufruf einer Programmzeile

Ausführung

Im nachfolgenden Beispiel wird die Programmzeile 30 aufgerufen

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<div>PR:1      ST:5             LN:10 10 SP 7</div>	2 x	Der Cursor wird zum Eingabefeld für die Zeilennummer bewegt.
②	<div>PR:5      ST:1             LN: ( 0 ) 10 SP 7 LINE NUMBER</div>	→  →	Die Zeilennummer wird eingegeben.
③	<div>PR:5      ST:1             LN: ( 30 ) 30 TI 3 LINE NUMBER</div>		Die Zeile 30 wird angezeigt.

Tab. 4-6: Beispiel zum direkten Aufruf einer Programmzeile

Beschreibung

- Es wird automatisch die letzte Zeile des Programms aufgerufen, wenn die angegebene Nummer größer als die höchste im Programm existierende Zeilennummer ist.



# 5 MOVEMASTER-Befehle

## 5.1 Allgemeine Hinweise

In diesem Abschnitt finden Sie eine Auflistung aller MOVEMASTER-Befehle und deren Anwendungsmöglichkeiten.

### 5.1.1 Gruppeneinteilung der Befehle

Die Befehle können in die nachfolgend aufgeführten sechs Gruppen eingeteilt werden.

- **Befehle für Positionierungen und Roboterbewegungen**  
Diese Gruppe umfaßt die Befehle zum Definieren von Positionen und Koordinaten. Desweiteren lassen sich mit den Befehlen dieser Gruppe die Interpolationsmethode, die Geschwindigkeit, die Timer, die Werkzeuglängen, die Gitterkoordinaten für Paletten usw. einstellen.
- **Befehle für die Programmsteuerung**  
Mit diesen Befehlen wird der Programmablauf gesteuert. Hierzu gehören Befehle für folgende Aufgaben: bedingte Programmsprünge, Programmschleifen, Programmunterbrechungen über Interrupt-Signale, Programm starten und stoppen, Zähleroperationen usw.
- **Befehle für die Handsteuerung**  
Diese Gruppe umfaßt die Befehle zum Steuern des Handgreifers (offen/geschlossen). Für die motorisch angetriebene Hand (Option) kann die Handgreiferkraft über Parametereinstellungen festgelegt werden.
- **Steuerbefehle für die Ein- und Ausgabe**  
Mit diesen Befehlen kann die externe Ein-/Ausgabe gesteuert werden. Für logische Rechenoperationen können sowohl einzelne Bitzustände als auch parallele Daten übertragen werden.
- **Befehle für die Kommunikationssteuerung über die RS232C-Schnittstelle**  
Mit diesen Befehlen können interne Daten der Robotersteuerung, wie z. B. Zählerwerte, Positionsdaten, Programme, Signalzustände der Ein-/Ausgänge und Parameterwerte, zum PC übertragen werden.
- **Zusatzfunktionen**  
In dieser Gruppe sind die Befehle für allgemeine Zusatzfunktionen zusammengefaßt. Zusatzfunktionen: Parameter einstellen, Programm auswählen, Fehlermeldungen zurücksetzen und Programmbemerkungen schreiben.

## 5.1.2 Hinweise zu den weiteren Befehlsbeschreibungen

In diesem Kapitel werden die Befehle in alphabetischer Reihenfolge beschrieben.

### Beschreibung des verwendeten Formats

Bedeutung des Symbols ♦	Befehle, die mit dem Symbol ♦ gekennzeichnet sind, dürfen innerhalb eines Roboterprogramms <b>nicht</b> programmiert werden. Diese Befehle können nur direkt über einen PC oder die Teaching Box ausgeführt werden.
Funktion	Die Funktion des Befehls wird kurz beschrieben.
Eingabeformat	Die Befehlsparameter und die Reihenfolge bei der Programmierung wird beschrieben (Befehlssyntax). < > kennzeichnet die notwendigen Befehlsparameter. [ ] kennzeichnet die wahlfreien Befehlsparameter. Die vertikale, unterbrochene Linie (!) bedeutet „oder“. Bei Angabe von <Zeilennummer ! Marke>, muß entweder die Zeilennummer oder die Marke angegeben werden.
Term	Die Bedeutung und die zulässigen Wertbereiche für die Befehlsparameter werden beschrieben.
Erläuterung	Die Funktion und Programmierung des Befehls wird ausführlich beschrieben.
Parameter	Es wird die werkseitige Grundeinstellung der Parameter beschrieben.
Programmbeispiel	Die Programmierung des Befehls wird anhand eines typischen Programms gezeigt.

### Programmzeile

Die Programmzeile (Abkürzung: Zeile) besteht aus einer Programmzeilennummer und dem Befehlssatz. Die Länge einer Programmzeile darf maximal 120 Zeichen betragen.

### Positionsnummer

Der Wertebereich für die Positionsnummern beträgt 1 – 999. Bei der Programmierung von mehreren verschiedenen Programmen werden die Daten der Positionen 1 – 900 als individuelle Daten abgespeichert. Die Daten der Positionen 901 – 999 werden jedoch für alle Programme als gemeinsame Positionsdaten abgespeichert. Desweiteren können Positionsnummern auch indirekt über die Zuweisung eines Zählerwertes angegeben werden.

Beispiel:

10	SC	21,10	Wert 10 in den Zähler 21 laden
20	MO	@21	Position 10 anfahren (Wert des Zählers 21)

Die Hinweise zu den Positionsnummern beziehen sich auf folgende Befehle: CF, HE, MA, MC, MO, MR, MRA, MS, MT, MTS, PC, PL, PR, PX und SF.



### Zählernummer

Der Wertebereich für die Zählernummern beträgt 1 – 99. Bei der Programmierung von mehreren verschiedenen Programmen werden die Werte der Zähler 1 – 90 als individuelle Daten abgespeichert. Die Werte der Zähler 91 – 99 werden jedoch für alle Programme als gemeinsame Zählerdaten abgespeichert. Desweiteren können Zählernummern auch indirekt über die Zuweisung eines anderen Zählerwertes angegeben werden.

Beispiel:

10 SC 21,10 Wert 10 in den Zähler 21 laden

20 IC @21 Wert 1 zum Zähler 10 addieren (Wert des Zählers 21)

Die Hinweise zu den Zählernummern beziehen sich auf folgende Befehle: CL, CP, CR, DC, IC, OC und SC.

### Zeichenkettennummer

Mittels der Zeichenkettennummer können Zeichenketten (maximal 127 alphanumerische Zeichen oder Symbole) über den seriellen Kanal zu externen Geräten übertragen werden.

Der Wertebereich für Zeichenkettennummern beträgt \$1 bis \$99. Bei der Programmierung von mehreren verschiedenen Programmen werden die Werte \$1 – \$90 als individuelle Daten abgespeichert. Die Werte von \$91 – \$99 werden jedoch für alle Programme als gemeinsame Daten abgespeichert. In Abhängigkeit des Wertes der Zählernummer, können Zeichenkettennummern nicht indirekt angegeben werden.

Beispiel:

10 SC \$1,"ABC" Zeichenkette "ABC" in Zeichenkette Nummer 1 laden

20 CP \$1 Zeichenkette in das Zeichenkettenregister laden

30 CL \$2 Inhalt des Zeichenkettenregisters in die Zeichenkette Nummer 2 laden

Die Hinweise beziehen sich auf die Befehle: CL, CP, CR, INP, LG, NE, EQ, SM und SC.

### Direkte Befehlsausführung

Wenn Sie einen Befehl ohne Angabe der Zeilennummer über den PC oder die Teaching Box direkt zum Steuergerät übertragen, wird dieser sofort ausgeführt. Gehen Sie deshalb bei der direkten Programmierung von Befehlen für Roboterbewegungen sehr vorsichtig vor.

Befehle für bedingte Programmsprünge (z. B. TB und EQ) können nicht direkt ausgeführt werden. Die Befehle HLT, PRN und die Kommunikationsbefehle für die RS232C-Schnittstelle können auch während der Programmabarbeitung direkt ausgeführt werden.

### Internes Register

Die von der externen Ein-/Ausgabe kommenden Daten werden im sogenannten „internen Register“ abgespeichert. Die Daten können dann für bedingte Programmsprünge, Wertevergleiche, logische Bitoperationen, Zählereinstellungen usw. eingesetzt werden. Das interne Register wird ebenfalls eingesetzt, wenn auf der Basis von Zählerwerten bedingte Sprünge ausgeführt werden sollen.

Die Hinweise zum internen Register beziehen sich auf folgende Befehle: AN, CL, CP, DR, EQ, ID, NE, OR, SM, TB und XO.

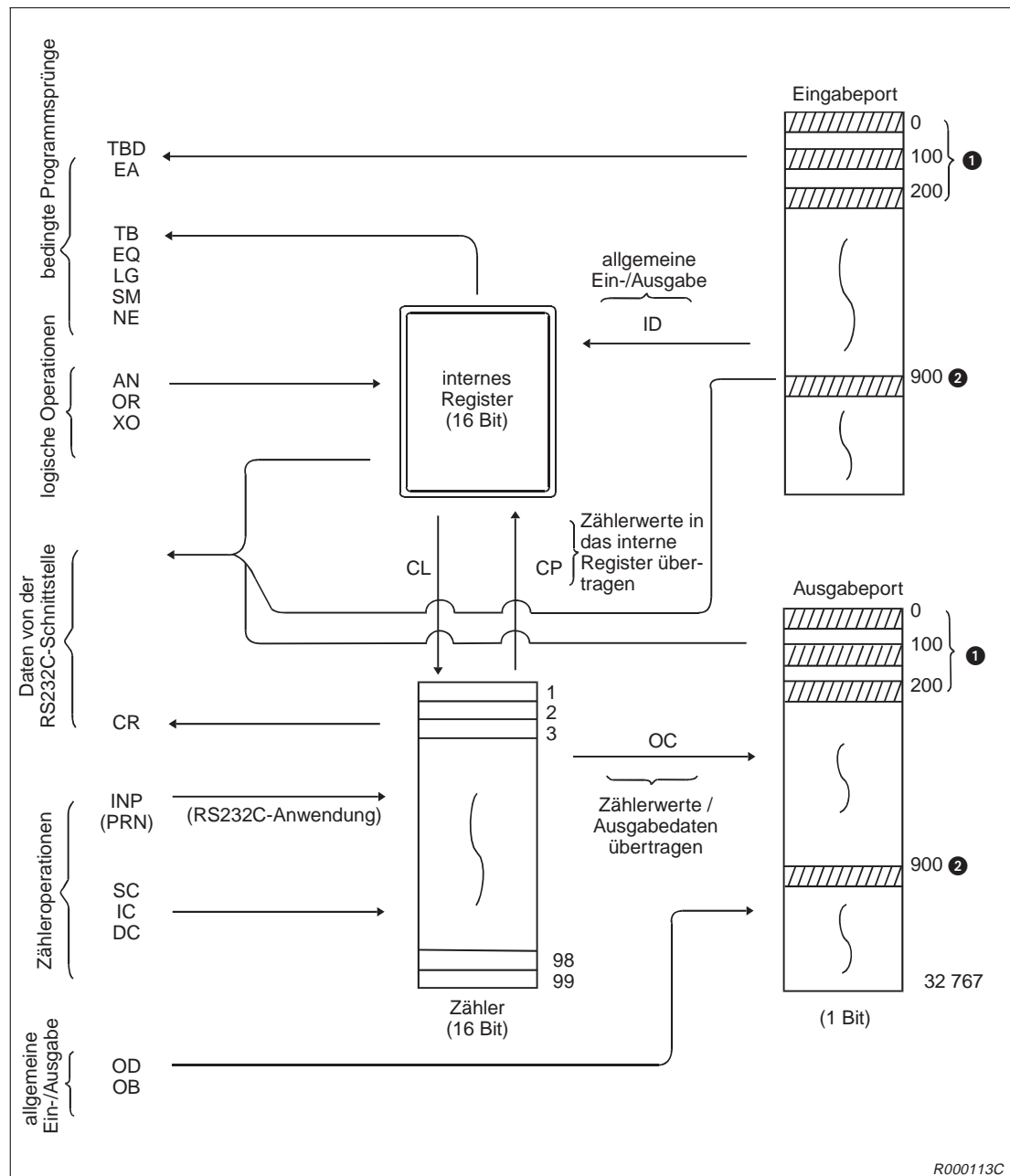
### Zeichenkettenregister

Zeichenketten, die über den seriellen Kanal von externen Eingabegeräten kommen, werden im sogenannten „Zeichenkettenregister“ abgespeichert. Es wird zur Aufnahme von Werten für Zeichenkettenvergleiche oder Zeichenkettennummern zur Ausführung von bedingten Sprüngen verwendet.

Die Hinweise beziehen sich auf die Befehle: CL, CP, INP, LG, NE, EQ und SM.

### Internes Register und Befehlsverarbeitung

In der unteren Schemazeichnung ist die Befehlsverarbeitung und die interne Abspeicherung von Zählerwerten und Ein-/Ausgabedaten schematisch dargestellt.

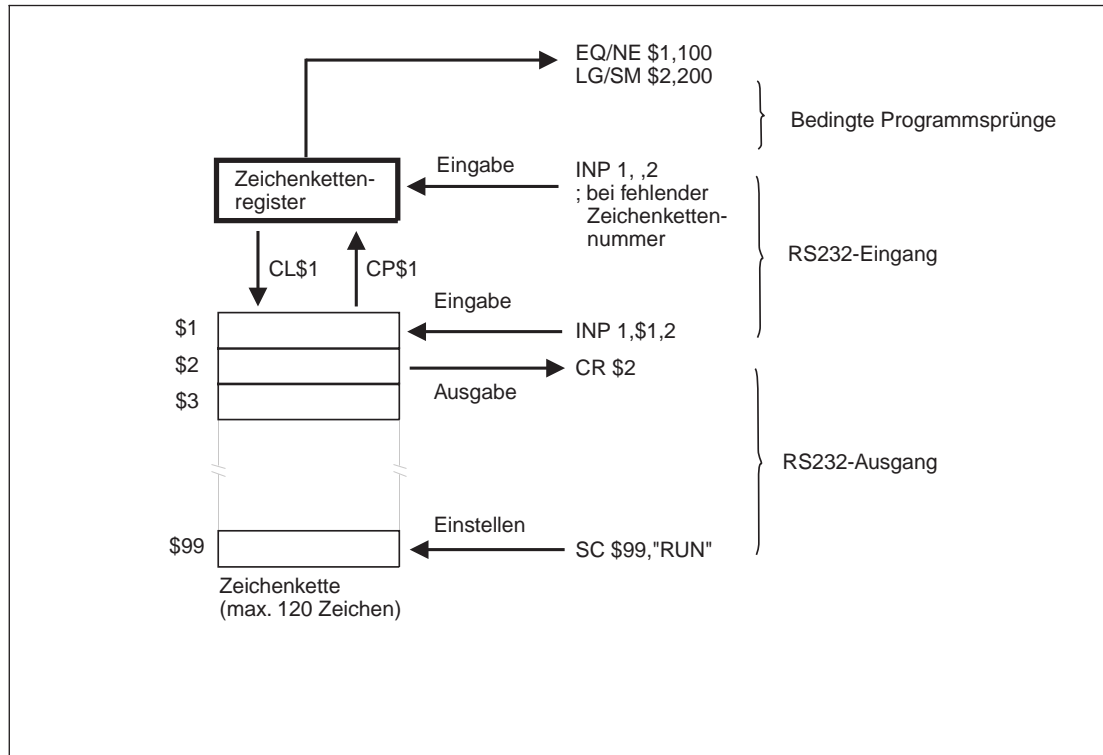


**Abb. 5-1:** Schematische Ablaufdarstellung der internen Befehlsverarbeitung

- ① Null entspricht dem ersten Bereich der EA-Karte. 100 entspricht dem zweiten Bereich der EA-Karte. 200 entspricht dem dritten Bereich der EA-Karte.
- ② 900 entspricht dem Bereich für die Handsteuersignale.

### Zeichenkettenregister und Befehlsverarbeitung

In Abbildung 5-2 ist die Befehlsverarbeitung und die Abspeicherung von Zeichenketten im Zeichenkettenregister schematisch dargestellt.



**Abb. 5-2:** Schematische Darstellung der Befehlsverarbeitung im Zeichenkettenregister

### 5.1.3 Schutzmaßnahmen bei der Programmierung und Eingabe über die Teaching Box

- Wenn Programmierarbeiten durchgeführt werden, dürfen sich generell keine Personen innerhalb des Schutzbereiches aufhalten. Läßt es sich nicht vermeiden, daß sich Personal bei der Programmierung innerhalb des Schutzbereiches aufhält, so sind Vorkehrungen für sichere Arbeitsabläufe zu treffen und diese zu überwachen. Das beinhaltet die folgenden Punkte:
  - Betriebsart und Durchführung; dazu gehören Autorisierung eines Bedieners.
  - Geschwindigkeit bei Langsamfahrt (ein Automatikbetrieb ist nicht erlaubt!)
  - Warnsignale für das Arbeitspersonal
  - Maßnahmen für den Notfall
  - Maßnahmen zur Verhinderung von Fehlfunktionen
- Auch wenn die Möglichkeit der Wahl des Automatikbetriebs über die Teaching-Box gegeben ist, darf der Automatikbetrieb niemals aktiviert werden, solange sich Personal innerhalb des Schutzbereiches aufhält!
- Wenn der Schutzbereich mit der Teaching Box betreten wird, muß vorher sichergestellt sein, daß die vorrangige Steuerung über die Teaching Box freigegeben ist. Ansonsten kann der Roboter unerwartet aktiviert werden, wobei sich äußerst gefährliche Situationen ergeben können.
- Alle Bewegungen von Zusatzeinrichtungen innerhalb des Schutzbereiches müssen entweder verhindert werden oder sich unter der alleinigen Kontrolle des Programmierers befinden. Bei der Kontrolle durch den Programmierer müssen alle Aktionen eindeutig und separat von den Roboteraktionen steuerbar sein.
- Es ist empfehlenswert, eine Hilfsperson unmittelbar an einen NOT-AUS-Schalter zu platzieren, um in einem Notfall die Sicherheit des Teaching-Box-Bedieners zu gewährleisten. Diese Hilfsperson muß mit einer tragbaren NOT-AUS-Steuereinheit ausgerüstet sein, um im Notfall diese zu betätigen.
- Für eine ausreichende Beleuchtung der Arbeitsbereiche ist zu sorgen.
- Es ist dafür zu sorgen, daß das Personal geeignete Kleidung, Sicherheitsschuhe, einen Helm und gegebenenfalls eine Schutzbrille trägt.
- Aus Sicherheitsgründen sollten Sie während des Teaching-Box-Betriebs dem Roboter niemals den Rücken zukehren. Halten Sie sich immer einen Fluchtweg offen.
- Für Wartungszwecke muß immer eine Kopie der Anwendungsprogramme mit allen Änderungen zur Verfügung stehen (z. B. auf einem PC).
- Das Robotersystem darf solange nicht in den Automatikbetrieb geschaltet werden, bis alle Schutzmaßnahmen durch den Programmierer wieder in Funktion gesetzt wurden.

### 5.1.4 **Vorsichtsmaßnahmen beim Testen eines Programms**

Führen Sie nach erfolgter Installation und Programmerstellung immer einen Programmtest durch. Beachten Sie dabei besonders folgende Punkte:

- Ein Programmtest muß zunächst immer erst im Schrittbetrieb bei langsamer Geschwindigkeit erfolgen.
- Ist es unumgänglich, einen Programmtest durchzuführen, während sich eine Person innerhalb des Schutzbereiches aufhält, so sind die gleichen Schutzvorkehrungen wie beim Teaching-Box-Betrieb und beim Programmieren zu treffen. Für sichere Arbeitsabläufe und deren Überwachung ist zu sorgen.
- Wird ein Programm getestet, bei dem Zusatzeinrichtungen über Ein-/Ausgabesignale des Roboters gesteuert werden, ist besondere Vorsicht bei Bewegungen dieser Zusatzeinrichtungen geboten. Wie beim Teaching-Box-Betrieb und Programmieren müssen alle gefährlichen Bewegungen von Teilen der Zusatzeinrichtungen innerhalb des Schutzbereiches entweder verhindert werden oder sich unter der alleinigen Kontrolle des Programmierers befinden.
- Wird ein Programm in der Mitte gestartet, muß sichergestellt sein, daß keine Kollision zwischen dem Roboterarm und den Zusatzeinrichtungen auftreten kann.
- Besonders lange und umfangreiche Programme sollten Sie zuvor ausdrucken und dann schrittweise prüfen. Für das Austesten solcher Programme muß genügend Fachwissen vorhanden sein.

## 5.2 Übersicht der Befehle

Befehl	Funktion	Abschnitt	Seite
<b>ADD</b> (Addition)	Addition	5.2.1	5-11
<b>AN</b> (And)	UND-Verknüpfung	5.2.2	5-12
<b>CF</b> (Change Figure)	Stellungsdaten ändern	5.2.3	5-13
<b>CL</b> (Counter Load)	Zähler laden	5.2.4	5-16
<b>CP</b> (Compare Counter)	Zählerwert vergleichen	5.2.5	5-18
<b>CR</b> (Counter Read)	Zählerwert lesen	5.2.6	5-20
<b>DA</b> (Disable Act)	Interrupt-Möglichkeit sperren	5.2.7	5-22
<b>DC</b> (Decrement Counter)	Zählerwert dekrementieren	5.2.8	5-23
<b>DIV</b> (Division)	Division	5.2.9	5-24
<b>DJ</b> (Draw Joint)	Relative Gelenkbewegung	5.2.10	5-25
<b>DL</b> (Delete Line)	Programmzeile löschen	5.2.11	5-26
<b>DP</b> (Decrement Position)	Positionsnummer dekrementieren	5.2.12	5-27
<b>DR</b> (Data Read)	Daten lesen	5.2.13	5-28
<b>DS</b> (Draw Straight)	Relative geradlinige Bewegung	5.2.14	5-30
<b>DW</b> (Draw)	Relative Bewegung	5.2.15	5-32
<b>EA</b> (Enable Act)	Interrupt-Eingang festlegen	5.2.16	5-34
<b>ED</b> (End)	Programmende	5.2.17	5-36
<b>EQ</b> (Equal)	Datenwertvergleich: =	5.2.18	5-37
<b>ER</b> (Error Read)	Fehler lesen	5.2.19	5-39
<b>GC</b> (Grip Close)	Handgreifer schließen	5.2.20	5-41
<b>GF</b> (Grip Flag)	Handgreiferzustand festlegen	5.2.21	5-43
<b>GO</b> (Grip Open)	Handgreifer öffnen	5.2.22	5-44
<b>GS</b> (Go Sub)	Sprung zu einem Unterprogramm	5.2.23	5-45
<b>GT</b> (Go To)	Sprung zu einer Programmzeile	5.2.24	5-47
<b>HE</b> (Here)	Aktuelle Position speichern	5.2.25	5-48
<b>HLT</b> (Halt)	Programmablauf stoppen	5.2.26	5-49
<b>HO</b> (Home)	Nullpunkt einstellen	5.2.27	5-50
<b>IC</b> (Increment Counter)	Zählerwert inkrementieren	5.2.28	5-51
<b>ID</b> (Input Direct)	Eingänge einlesen	5.2.29	5-52
<b>INP</b> (Input)	Zählerwert oder Positionsdaten lesen	5.2.30	5-53
<b>IP</b> (Increment Position)	Positionsnummer inkrementieren	5.2.31	5-55
<b>JRC</b> (Joint Roll Change)	Addition von $\pm 360^\circ$ zur aktuellen Position der R-Achse	5.2.32	5-56
<b>LG</b> (If Larger)	Datenwertvergleich: >	5.2.33	5-57
<b>LR</b> ♦ (Line Read)	Programmzeile lesen	5.2.34	5-59
<b>MA</b> (Move Approach)	Relative Koordinatenaddition	5.2.35	5-61
<b>MC</b> (Move Continuous)	Kontinuierliche Bewegung	5.2.36	5-63
<b>MJ</b> (Move Joint)	Relative Mehrfach-Gelenkbewegung	5.2.37	5-65
<b>ML</b> (Move Linear)	Lineare Bewegung	5.2.38	5-67
<b>MO</b> (Move)	Position anfahren	5.2.39	5-68
<b>MP</b> (Move Position)	Koordinatenposition anfahren	5.2.40	5-69
<b>MPB</b> (Move Playback)	Schritt-Parameter für Teaching-Playback festlegen	5.2.41	5-71

Tab. 5-1: Übersicht der Befehle (1)

Befehl	Funktion	Abschnitt	Seite
<b>MPC</b> (Move Payback Continuous)	Bewegung für Teaching-Playback festlegen	5.2.42	5-74
<b>MR</b> (Move R)	Zwischenposition bei Kreis-Interpolation	5.2.43	5-76
<b>MRA</b> (Move R A)	Kreis-Interpolation	5.2.44	5-78
<b>MS</b> (Move Straight)	Geradlinige Bewegung	5.2.45	5-80
<b>MT</b> (Move Tool)	Werkzeugbewegung mit Gelenk-Interpolation	5.2.46	5-82
<b>MTS</b> (Move Tool Straight)	Geradlinige Werkzeugbewegung	5.2.47	5-84
<b>MUL</b> (Multiplication)	Multiplikation	5.2.48	5-86
<b>N ♦</b> (Number)	Programm auswählen	5.2.49	5-87
<b>NE</b> (If Not Equal)	Datenwertvergleich:	5.2.50	5-88
<b>NT</b> (Nest)	Nullpunkt anfahren	5.2.51	5-90
<b>NW ♦</b> (New)	Programm- und Positionsspeicher löschen	5.2.52	5-91
<b>NX</b> (Next)	Programmschleife beenden	5.2.53	5-92
<b>OB</b> (Output Bit)	Ausgänge ein-/ausschalten	5.2.54	5-93
<b>OC</b> (Output Counter)	Zählerwert ausgeben	5.2.55	5-94
<b>OD</b> (Output Direct)	Direkte Ausgabe	5.2.56	5-95
<b>OG</b> (Origin)	Nullpunkt anfahren	5.2.57	5-96
<b>OPN</b> (Open)	Kommunikationskanäle öffnen	5.2.58	5-97
<b>OR</b> (Or)	ODER-Verknüpfung	5.2.59	5-98
<b>OVR</b> (Override)	Übersteuerung	5.2.60	5-99
<b>PA</b> (Pallet Assign)	Gitterpunkte für Palette definieren	5.2.61	5-100
<b>PC ♦</b> (Position Clear)	Position löschen	5.2.62	5-101
<b>PD</b> (Position Define)	Position definieren	5.2.63	5-102
<b>PL</b> (Position Load)	Position kopieren	5.2.64	5-104
<b>PMR</b> (Parameter Read)	Parameterwerte lesen	5.2.65	5-105
<b>PMW</b> (Parameter Writing)	Parameterwerte schreiben	5.2.66	5-106
<b>PR</b> (Position Read)	Positionsdaten lesen	5.2.67	5-107
<b>PRN ♦</b> (Print)	Daten übertragen	5.2.68	5-109
<b>PT</b> (Pallet)	Koordinaten für Palette berechnen	5.2.69	5-111
<b>PW</b> (Pulse Wait)	Warteimpulse festlegen	5.2.70	5-113
<b>PX</b> (Position Exchange)	Position austauschen	5.2.71	5-114
<b>QN</b> (Question Number)	Programminformationen lesen	5.2.72	5-115
<b>RC</b> (Repeat Cycle)	Programmschleife	5.2.73	5-116
<b>RN ♦</b> (Run)	Programm starten	5.2.74	5-117
<b>RS ♦</b> (Reset)	Programm/Fehlerbedingung zurücksetzen	5.2.75	5-119
<b>RT</b> (Return)	Rücksprung zum Hauptprogramm	5.2.76	5-120
<b>SC</b> (Set Counter)	Zählerwert einstellen	5.2.77	5-121
<b>SD</b> (Speed Define)	Absolute Geschwindigkeit definieren	5.2.78	5-123
<b>SF</b> (Shift)	Positionskoordinaten addieren	5.2.79	5-125
<b>SM</b> (If Smaller)	Datenwertvergleich: <	5.2.80	5-126
<b>SP</b> (Speed)	Betriebsgeschwindigkeit einstellen	5.2.81	5-128
<b>STR ♦</b> (Step Read)	Programmschritt lesen	5.2.82	5-131
<b>SUB</b> Subtraction	Subtraktion	5.2.83	5-133
<b>TB</b> (Test Bit)	Eingang-Bitstatus überprüfen	5.2.84	5-134

Tab. 5-1: Übersicht der Befehle (2)

Befehl	Funktion	Abschnitt	Seite
<b>TBD</b> (Test Bit Direct)	Eingang-Bitstatus direkt überprüfen	5.2.85	5-135
<b>TI</b> (Timer)	Timer (Zeitglied)	5.2.86	5-136
<b>TL</b> (Tool)	Werkzeuglänge einstellen	5.2.87	5-137
<b>VR</b> (Version Read)	Software-Version lesen	5.2.88	5-138
<b>WH</b> (Where)	Aktuelle Positionskoordinaten lesen	5.2.89	5-139
<b>WT</b> (Whalt Tool)	Werkzeuglänge lesen	5.2.90	5-141
<b>XO</b> (Exclusive Or)	Exklusiv-ODER-Verknüpfung	5.2.91	5-142
<b>'</b> (Comment)	Bemerkung	5.2.92	5-143

**Tab. 5-1:** Übersicht der Befehle (3)



### 5.2.1 ADD (Add)

#### Funktion: Addition

Addiert einen Wert oder den Inhalt eines festgelegten Zählers zum Wert des internen Registers.

#### Eingabeformat

```
ADD <Wert | @Zählernummer>
```

<Wert>	Legt den zu addierenden hexadezimalen oder dezimalen Wert fest. $-32\,768 \leq \text{dezimaler Wert} \leq 32\,767$ $\&8000 \leq \text{hexadezimaler Wert} \leq \&7FFF$
<Zählernummer>	Legt den Zähler fest. $1 \leq \text{Zählernummer (ganzzahlig)} \leq 99$

#### Erläuterung

- Addiert einen dezimalen oder hexadezimalen Wert, oder den Inhalt eines festgelegten Zählers zum Wert des internen Registers.

#### Beispiele ▾

ADD 10	Addiert den dezimalen Wert 10 zum Wert des internen Registers
ADD &FF	Addiert den hexadezimalen Wert &FF10 zum Wert des internen Registers
ADD @5	Addiert den Inhalt des Zählers 5 zum Wert des internen Registers

#### Programmbeispiel (MOVEMASTER-Befehle)

10 ID	Lädt Eingabedaten in das interne Register
20 ADD 10	Addiert den Wert 10 zu den eingelesenen Daten
30 CL 21	Daten aus dem internen Register in den Zähler 21 laden

## 5.2.2 AN (And)

### Funktion: UND-Verknüpfung

UND-Verknüpfung des festgelegten Wertes mit dem Wert des internen Registers und anschließende Ergebnisabspeicherung im internen Register.

### Eingabeformat

AN    <Datenwert>
-------------------

<Datenwert>

Legt den Datenwert fest

$-32\,768 \leq \text{Datenwert (dezimal)} \leq 32\,767$

$\&8000 \leq \text{Datenwert (hexadezimal)} \leq \&7FFF$

### Erläuterung

- Der festzulegende Datenwert kann dezimal oder hexadezimal definiert werden. Bei einem hexadezimalen Wert muß vor der Zahl ein „&“ stehen.
- Das Ergebnis wird im internen Register abgespeichert und kann mit dem entsprechenden Befehl verändert, verglichen oder gelesen werden (siehe Befehle EQ, NE, LG, SM, CL, DR und OR).
- Durch Ausführung des Befehls AN nach einem Eingabebefehl (ID oder IN) besteht die Möglichkeit, vom einem externen Gerät nur die erforderlichen Bits der parallelen Eingabedaten zu empfangen.

### Programmbeispiel (MOVEMASTER-Befehle)

10	ID		Daten am externen Eingabeport holen
20	AN	&000F	Empfängt nur die vier unteren Bits
30	CL	12	Daten in den Zähler 12 laden
40	EQ	8,200	Sprung zur Zeile 200, wenn die Daten gleich 8 sind
50	ED		Programm beenden
200	MO	99	Position 99 anfahren

### 5.2.3 CF (Change Figure)

#### Funktion: Stellungsdaten ändern

Ändert die Stellungsdaten des Roboters an der festgelegten Position.

#### 5 Achsen

#### Eingabeformat

```
CF  <Positionsnummer>
    [, [<R/L>] [, [<A/B>]
```

<Positionsnummer>	Legt die Position fest, an der die Stellungsdaten geändert werden sollen. $1 \leq \text{Positionsnummer (ganzzahlig)} \leq 999$
<R/L>	Bestimmt die Roboterstellung (rechts oder links) R : rechts (Standardwert) L : links
<A/B>	Bestimmt die Roboterstellung (oben oder unten) A : oben (Standardwert) B : unten

#### 6 Achsen

#### Eingabeformat

```
CF  <Positionsnummer>
    [, [<R/L>] [, [<A/B>] [, [N/F]]]
```

<Positionsnummer>	Legt die Position fest, an der die Stellungsdaten geändert werden sollen. $1 \leq \text{Positionsnummer} \leq 999$
<R/L>	Bestimmt die Roboterstellung (rechts oder links) R : rechts (Standardwert) L : links
<A/B>	Bestimmt die Roboterstellung (oben oder unten) A : oben (Standardwert) B : unten
<N/F>	Bestimmt die Roboterstellung (nicht kippen oder kippen) N : nicht kippen (Standardwert) F : kippen

#### Erläuterung

- Die Stellungsdaten des Roboters werden an der festgelegten Position geändert.
- Die Änderungen beziehen sich nicht auf die Koordinatendaten der festgelegten Position und den Handgreiferzustand (offen/geschlossen).
- Die vorgenommenen Änderungen der Stellungsdaten für die festgelegte Position können Sie sich mit dem PR-Befehl anzeigen lassen.

**5 Achsen****Programmbeispiel (MOVEMASTER-Befehle)**

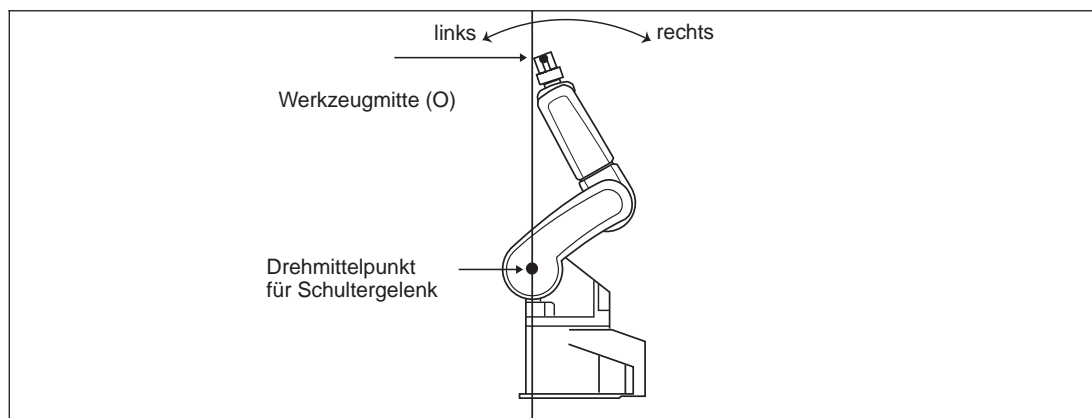
10	PD	1,-280,-50,970,-10,-20,L,A,	Position 1 definieren (links, oben)
20	MO	1	Position 1 anfahren
30	CF	1,R,A	Stellungsdaten der Position 1 ändern (rechts, oben)
40	MO	1	Position 1 anfahren

**6 Achsen****Programmbeispiel (MOVEMASTER-Befehle)**

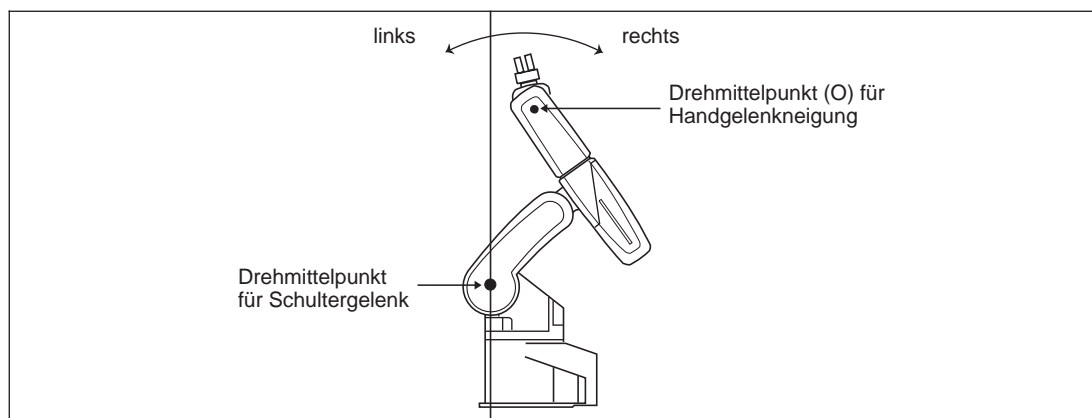
10	PD	1,530,0,470,10,135,-10,R,A,N	Position 1 definieren (rechts, oben, nicht kippen)
20	MO	1	Position 1 anfahren
30	CF	1,R,A	Stellungsdaten der Position 1 ändern (rechts, oben, kippen)
40	MO	1	Position 1 anfahren

**Stellungsmerker**

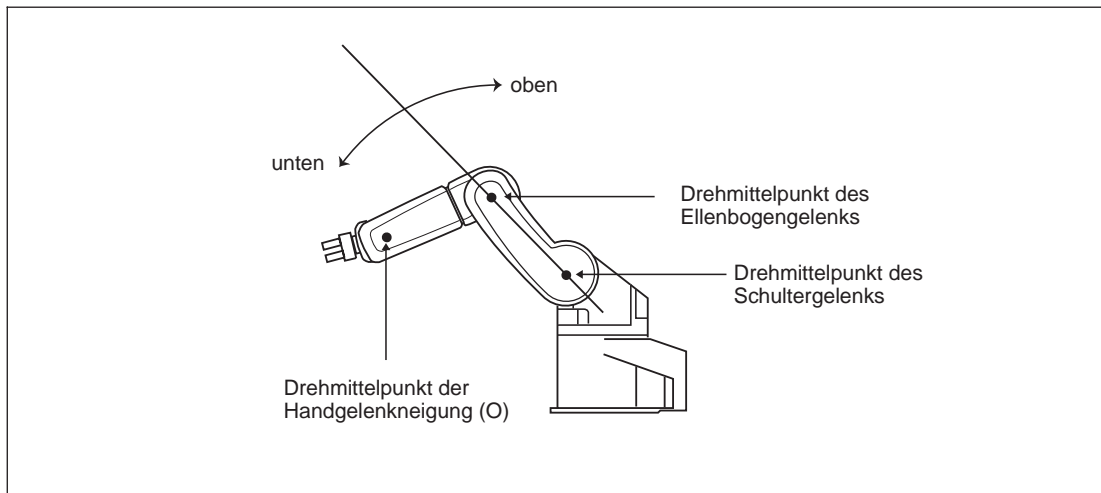
In den nachfolgenden Abbildungen finden Sie eine kurze Darstellung der Stellungsmerker. Für eine detaillierte Beschreibung lesen Sie bitte Anhang A.5.



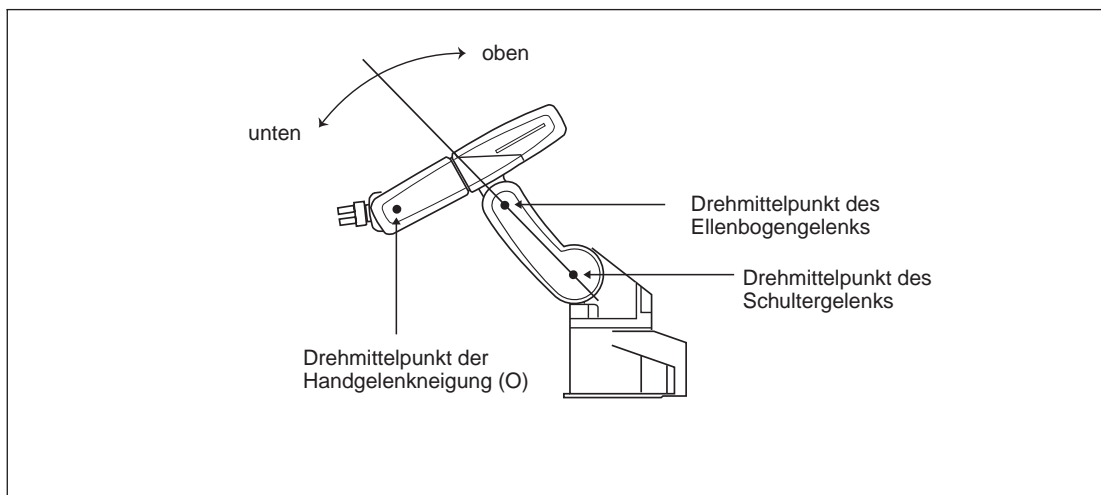
**Abb. 5-3:** Kennungen für die Roboterstellungen am Fünfachser: links/rechts (L/R)



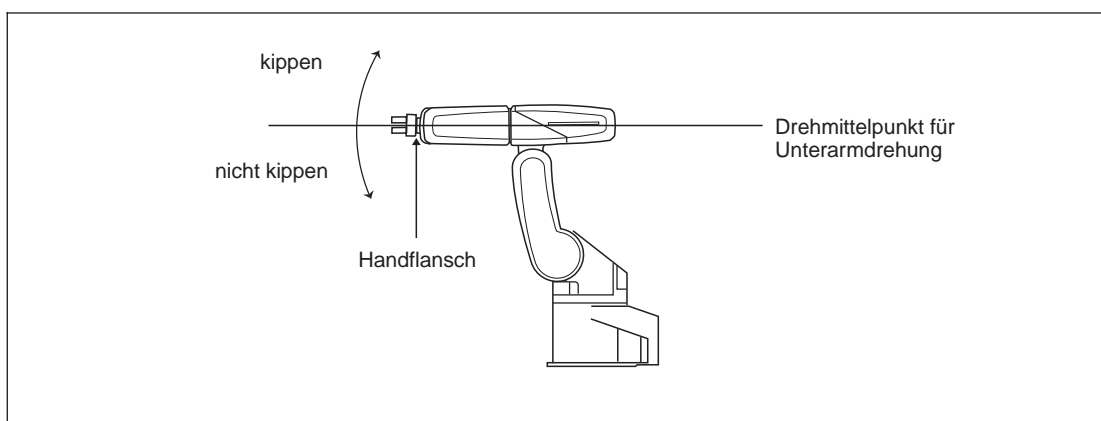
**Abb. 5-4:** Kennungen für die Roboterstellungen am Sechachser: links/rechts (L/R)



**Abb. 5-5:** Kennungen für die Roboterstellung am Fünfachser: oben/unten (A/B)



**Abb. 5-6:** Kennungen für die Roboterstellung am Sechssachser: oben/unten (A/B)



**Abb. 5-7:** Kennungen für die Roboterstellung am Sechssachser: kippen/nicht kippen (F/N)

## 5.2.4 CL (Counter Load)

### Funktion: Zähler laden

Lädt den Wert des internen Registers in den festgelegten Zähler. Der Inhalt des Zeichenkettenregisters wird in die Zeichenkette mit dem angegebenen Wert geladen.

### Eingabeformat

CL    <Zählernummer/Zeichenkettennummer>
--

- <Zählernummer>            Legt den Zähler fest.  
 $1 \leq \text{Zählernummer (ganzzahlig)} \leq 99$
- <Zeichenkettennummer> Legt die Zeichenkette fest. Das führende Zeichen ist „\$“.  
 $\$1 \leq \text{Zeichenkettennummer} \leq \$99$

### Erläuterung

<Bei Angabe einer Zählernummer>

- Der Befehl lädt die vom Eingabeport abgerufenen Daten in den festgelegten Zähler. Deshalb muß dieser Befehl nach dem Eingabebefehl ausgeführt werden (siehe ID-Befehl).
- Weil die Daten vom Eingabeport immer mit dem Vorzeichen verarbeitet werden, wird der Datenwert für den Zähler auch mit dem Vorzeichen verarbeitet (Datenwert zwischen -32 768 und +32 767).
- Der Befehl kann zum Zählen von Arbeitsfolgen und zum Festlegen des Zählerwertes bei Palettenarbeit (über ein externes Gerät, z. B. eine SPS) eingesetzt werden. Gleichzeitig können die eventuell erforderlichen logische Befehle eingesetzt werden (siehe AN, OR oder XO).
- Durch Ausführung des CL-Befehls nach dem CP-Befehl können die Zählerdaten übertragen werden.
- Der Zählerwert kann mit dem entsprechenden Befehlen verändert, verglichen oder gelesen werden (siehe Befehle IC, DC, SC, CP und CR).

<Bei Angabe einer Zeichenkettennummer>

- Der Inhalt des Zeichenkettenregisters wird in die Zeichenkette mit dem angegebenen Wert geladen. Deshalb muß dieser Befehl nach dem Lesebefehl für das Zeichenkettenregister ausgeführt werden (siehe CP- und INP-Befehl).
- Der Befehl wird verwendet, um Arbeitsvorgänge und Arbeitsergebnisse von externen Geräten zu steuern und zu überwachen (z. B. bei Sortiermaschinen). Gleichzeitig können die erforderlichen Vergleichsbefehle (siehe EQ, LG, NE und SM) eingesetzt werden.
- Der Wert des Zeichenkettenregisters wird mit dem CL-Befehl in die Zeichenkette geladen. Wird der Befehl nach dem CP-Befehl ausgeführt, können Daten zwischen Zeichenketten kopiert werden.
- Durch die entsprechenden Befehle können Operationen, Vergleiche und Lesevorgänge mit Zeichenketten durchgeführt werden (siehe CP, CR, EQ, INP, LG, NE, SC und SM).

**Programmbeispiel (MOVEMASTER-Befehle)**

10	ID		Daten vom externen Eingabeport lesen
20	CL	25	Obere Daten in den Zähler 25 laden
30	CP	11	Daten vom Zähler 25 in das interne Register laden
40	CL	21	Daten aus dem internen Register in den Zähler 21 laden
50	SC	\$5,"ABC"	Zeichenkette "ABC" in die Zeichenkette Nummer 5 laden
60	CP	\$5	Zeichenkette Nummer 5 in das Zeichenkettenregister laden
70	CL	\$10	Inhalt des Zeichenkettenregisters in Zeichenkette Nummer 10 laden

## 5.2.5 CP (Compare Counter)

### Funktion: Zählerwert vergleichen

Lädt den Wert des festgelegten Zählers für einen Vergleich in das interne Register. Lädt die festgelegte Zeichenkette in das Zeichenkettenregister.

### Eingabeformat

CP    <Zählernummer/Zeichenkettennummer>
--

- <Zählernummer>            Legt den Zähler fest.  
 $1 \leq \text{Zählernummer (ganzzahlig)} \leq 99$
- <Zeichenkettennummer>    Legt die Zeichenkette fest. Das führende Zeichen ist „\$“.  
 $\$1 \leq \text{Zeichenkettennummer} \leq \$99$

### Erläuterung

<Bei Angabe einer Zählernummer>

- Dieser Befehl muß vor einem bedingten Sprungbefehl (siehe Befehle EQ, NE, LG und SM) ausgeführt werden, wenn der Wert des festgelegten Zählers als Sprungbedingung eingesetzt wird. Der mit dem Befehl CP in das interne Register geladene Datenwert wird als Bedingung für den Sprungbefehl eingesetzt.
- Der Datenwert im internen Register bleibt erhalten, selbst wenn sich der Wert des festgelegten Zählers nach Ausführung des CP-Befehls ändert. Dementsprechend muß dieser Befehl erneut ausgeführt werden, wenn der Zählerwert als Sprungbedingung eingesetzt wird.
- Der Eingabebefehl (ID-Befehl) verwendet dasselbe interne Register wie der CP-Befehl, was bedeutet, daß der alte Inhalt des internen Registers gelöscht wird, wenn ein Eingangsbefehl ausgeführt wird.
- Der Zählerwert kann mit dem entsprechenden Befehlen verändert oder gelesen werden (siehe Befehle SC, IC, DC, CR, CL, AN, OR und XO).

<Bei Angabe einer Zeichenkettennummer>

- Dieser Befehl muß vor einem bedingten Sprungbefehl (siehe Befehle EQ, NE, LG und SM) ausgeführt werden, wenn die Daten der Zeichenkette als Sprungbedingung eingesetzt werden. Der mit dem Befehl CP in das Zeichenkettenregister geladenen Wert wird als Bedingung für den Sprungbefehl eingesetzt.
- Der Datenwert im Zeichenkettenregister bleibt erhalten, selbst wenn sich der Wert der festgelegten Zeichenkette nach Ausführung des CP-Befehls ändert. Dementsprechend muß dieser Befehl erneut ausgeführt werden, wenn sich die Zeichenkette geändert hat.
- Durch die entsprechenden Befehle können Operationen, Vergleiche und Lesevorgänge mit Zeichenketten durchgeführt werden ( siehe CP, CR, EQ, INP, LG, NE, SC und SM).



**Programmbeispiel (MOVEMASTER-Befehle)**

100	IC	21	Wert 1 zum Inhalt des Zählers 12 addieren
110	CP	21	Daten vom Zähler 21 in das interne Register laden
120	EQ	255,500	Programmsprung zur Zeile 500, wenn die Daten des internen Registers gleich 255 sind
130	GT	100	Programmsprung zur Zeile 100
500	SC	21,0	Wert 0 in den Zähler 21 laden
600	SC	\$5,"OK"	Zeichenkette "OK" in die Zeichenkette Nummer 5 laden
610	CP	\$5	Wert der Zeichenkette in das Zeichenkettenregister laden
620	EQ	\$10,800	Sprung zur Zeile 800, wenn der Wert des Zeichenkettenregisters und der Wert der Zeichenkette Nummer 10 gleich sind

## 5.2.6 CR (Counter Read)

### Funktion: Zählerwert lesen

Liest den Inhalt des festgelegten Zählers oder eine Zeichkette (über die RS232C-Schnittstelle).

### Eingabeformat

CR    <Zählernummer/Zeichenkettennummer>
--

<Zählernummer>      Legt den Zähler fest.  
 $1 \leq \text{Zählernummer (ganzzahlig)} \leq 99$

<Zeichenkettennummer> Legt die Zeichenkette fest. Das führende Zeichen ist „\$“.  
 $\$1 \leq \text{Zeichenkettennummer} \leq \$99$

### Erläuterung

- Es werden die Inhalte des festgelegten Zählers über die RS232C-Schnittstelle ausgegeben.
- Das Ausgabeformat ist ein dezimaler ASCII-Code.
- Das Abschlußzeichen der Ausgabedaten ist ein „Carriage Return“ (CR, hex. 0D). Deshalb ist es beim Einsatz eines PCs erforderlich, daß die serielle Datenkette bis zur Stelle „hex. 0D“ empfangen wird. Der Befehl „LINE INPUT #“ ist der entsprechende BASIC-Befehl zum Einlesen der Daten.
- Es wird der Initialisierungswert „0“ zurückgesendet, wenn ein undefinierter Zähler gelesen werden soll. (Der Wert „+0“ wird bei Angabe der Zählernummer 0 zurückgesendet.)
- Der Zählerinhalt bleibt nach Ausschalten der Spannungsversorgung batteriegepuffert abgespeichert.

### Programmbeispiel (BASIC-Befehle)

<Bei Angabe einer Zählernummer>

10	OPEN "COM1 :E83"    AS#1	RS232C-Kommunikationsdatei wird über den PC mittels BASIC-Befehl geöffnet
20	INPUT "Zählernummer = ";N	Eingabe der Zählernummer
30	INPUT "Zählerwert = ";D	Eingabe des Zählerwertes
40	PRINT #1,"SC"+STR\$(N)+","+STR\$(D)	Daten in den Zähler schreiben
50	PRINT #1,"CR"+STR\$(N)	Daten in den PC lesen
60	LINE INPUT #1,A\$	Übertragene Daten in A\$ abspeichern
70	PRINT A\$	Daten auf dem Bildschirm anzeigen
80	END	Programmende

RUN      BASIC-Programm starten

Zählernummer = 1

Zählerwert = 100

100

<Bei Angabe einer Zeichenkettennummer>

10	OPEN "COM1 :E83" AS#1	RS232C-Kommunikationsdatei wird über den PC mittels BASIC-Befehl geöffnet
20	INPUT "Zeichenkettennummer = ";N	Eingabe der Zeichenkettennummer
30	INPUT "Zeichenkettendaten = ";J\$	Eingabe der Zeichenkettendaten
40	PRINT #1,"SC \$"+"STR\$ (N)+",""+CHR\$(&H22)+J\$+CHR\$(&H22)	Daten in den Zähler schreiben
50	PRINT #1,"CR \$"+"STR\$ (N)	Daten in den PC lesen
60	LINE INPUT #1,A\$	Übertragene Daten in A\$ abspeichern
70	PRINT A\$	Daten auf dem Bildschirm anzeigen
80	END	Programmende

RUN	BASIC-Programm starten
-----	------------------------

Zeichenkettennummer = 1

Zeichenkettendaten = ABC

ABC

## 5.2.7 DA (Disable Act)

### Funktion: Interrupt-Möglichkeit sperren

Sperrt die Interrupt-Möglichkeit für die festgelegten Bits am externen Eingabeport.

### Eingabeformat

DA    <Eingangs-Bitnummer>
----------------------------

<Eingangs-Bitnummer>    Legt die zu sperrende Bit-Nummer fest.  
 $0 \leq \text{Eingangs-Bitnummer} \leq 32\,767$

0 – 8999	: Eingangssignal-Interrupt
9003 – 32 767	: Eingangssignal-Interrupt
0 – 299	: allgemeine Eingänge
900 – 903	: Handeingänge
9000	: Alarm-Interrupt
9001	: NOT-HALT-Interrupt
9002	: Kommunikations-Interrupt

### Erläuterung

- Dieser Befehl löscht den Interrupt-Freigabestatus für das durch den Interrupt-Freigabebefehl definierte Bit (siehe Befehl EA).
- Sobald der DA-Befehl ausgeführt wurde, kann durch das festgelegte Bit kein Interrupt während des Programmablaufs ausgeführt werden. Beachten Sie, daß der DA-Befehl die Interrupt-Möglichkeit der anderen Bits nicht beeinflußt.
- Um wiederholte Interrupts durch ein Pegelsignal (0/1) zu verhindern, muß dieser Befehl in der ersten Zeile des Interrupt-Programms stehen, in das der Programmablauf verzweigt.

### Programmbeispiel (MOVEMASTER-Befehle)

Siehe EA-Befehl, Abschnitt 5.2.16.

## 5.2.8 DC (Decrement Counter)

### Funktion: Zählerwert dekrementieren

Subtrahiert 1 vom Wert des festgelegten Zählers.

### Eingabeformat

DC    <Zählernummer>
----------------------

<Zählernummer>

Legt den Zähler fest.

$1 \leq \text{Zählernummer (ganzzahlig)} \leq 99$

### Erläuterung

- Es tritt eine Fehlermeldung auf, wenn der Zählerwert kleiner als –32 768 ist.
- Dieser Befehl kann zum Zählen von Arbeitsgegenständen oder Arbeitsvorgängen und zum Einstellen der Anzahl der Gitterpunkte bei Palettenarbeit eingesetzt werden.
- Der Zählerwert kann mit dem entsprechenden Befehlen verändert, verglichen oder gelesen werden (siehe Befehle SC, IC, CP, CR, CL, AN, OR und XO).

### Programmbeispiel (MOVEMASTER-Befehle)

10	SC	21,15	Wert 15 in den Zähler 21 laden
20	DC	21	Wert des Zählers 21 um 1 erniedrigen

## 5.2.9 DIV (Division)

### Funktion

Dividiert den Wert des internen Registers durch einen Wert oder durch den Inhalt eines festgelegten Zählers.

### Eingabeformat

```
DIV <Wert | @Zählernummer>
```

<Wert>                      Legt den hexadezimalen oder dezimalen Divisor fest.  
                                $-32\,768 \leq \text{dezimaler Wert} \leq 32\,767$   
                                $\&8000 \leq \text{hexadezimaler Wert} \leq \&7FFF$

<Zählernummer>            Legt den Zähler fest.  
                                $1 \leq \text{Zählernummer (ganzzahlig)} \leq 99$

### Erläuterung

- Dividiert den Wert des internen Registers durch einen dezimalen oder hexadezimalen Wert, oder durch den Inhalt eines festgelegten Zählers.

### Beispiele ▾

DIV 10            Dividiert den Wert des internen Registers durch den dezimalen Wert 10  
 DIV &FF          Dividiert den Wert des internen Registers durch den hexadezimalen Wert &FF  
 DIV @5           Dividiert den Wert des internen Registers durch den Inhalt des Zählers 5

### Programmbeispiel (MOVEMASTER-Befehle)

10 ID              Lädt Eingabedaten in das interne Register  
 20 DIV 10          Dividiert die eingelesenen Daten durch den Wert 10  
 30 CL 21           Daten aus dem internen Register in den Zähler 21 laden

## 5.2.10 DJ (Draw Joint)

### Funktion: relative Einzel-Gelenkbewegung

Dreht jedes Gelenk um den festgelegten Drehwinkel (Betrag und Richtung), bezogen auf die aktuelle Position (Gelenk-Interpolation).

#### 5 Achsen

#### Eingabeformat

```
DJ  <Gelenknummer>, <Drehwinkel>
```

<Gelenknummer>

Legt das zu drehende Gelenk fest.

- 1 : Mittelteilgelenk
- 2 : Schultergelenk
- 3 : Ellbogengelenk
- 4 : Handgelenkneigung
- 5 : Handgelenkdrehung

<Drehwinkel>

Legt den Drehwinkel (Betrag und Richtung) fest

#### 6 Achsen

#### Eingabeformat

```
DJ  <Gelenknummer> <Drehwinkel>
```

<Gelenknummer>

Legt das zu drehende Gelenk fest.

- 1 : Mittelteilgelenk
- 2 : Schultergelenk
- 3 : Ellbogengelenk
- 4 : Unterarmdrehung
- 5 : Handgelenkneigung
- 6 : Handgelenkdrehung

<Drehwinkel>

Legt den Drehwinkel (Betrag und Richtung) fest.

### Erläuterung

- Der Drehwinkel kann in Schrittweiten von 0,01° angegeben werden. Geben Sie für z. B. 15,02° den Wert 15.02 an.
- Vor und nach der Roboterbewegung bleibt der Handgreiferzustand (offen/geschlossen) unverändert.
- Es tritt vor einer Gelenkbewegung eine Fehlermeldung auf, wenn der angegebenen Drehwinkel zu einer Überschreitung des zulässigen Roboterarbeitsbereichs führen würde.

### Programmbeispiel (MOVEMASTER-Befehle)

```
10 MO 1          Position 1 anfahren
20 DJ 1,10       Mittelteil um 10° in positiver Richtung drehen
```

## 5.2.11 DL ♦ (Delete Line)

### Funktion: Programmzeile löschen

Löscht die Befehle der festgelegten Programmzeilen und/oder Programmschritte.

### Eingabeformat

```
DL  [<Zeilennummer (a)>] [<Zeilennummer (b)>]
    [, [<(a)>][<(b)>]]]
```

<Zeilennummer (a)>	Legt die erste zu löschende Programmzeile fest (ganzzahlig).
<Zeilennummer (b)>	Legt die letzte zu löschende Programmzeile fest (ganzzahlig).
<(a)>	Legt den ersten zu löschenden Programmschritt fest (ganzzahlig).
<(b)>	Legt den letzten zu löschenden Programmschritt fest (ganzzahlig).
	$1 \leq \text{Zeilennummer (a), (b)} \leq 9999$
	$1 \leq \text{Schritt (a), (b)} \leq 9999$
	$\text{Zeilennummer (a)} \leq \text{Zeilennummer (b)}$
	$\text{Schritt (a)} \leq \text{Schritt (b)}$

### Erläuterung

- Es werden zuerst die Befehle der Schritte (a) bis (b) und anschließend die Befehle der Programmzeile (a) bis (b) gelöscht. Dies schließt den Schritt (b) und die Programmzeile (a) ein.
- Es wird nur die Programmzeile (a) gelöscht, wenn die Programmzeile (b) nicht angegeben wird.
- Es wird nur der Programmschritt (a) gelöscht, wenn der Programmschritt (b) nicht angegeben wird.

### Programmbeispiel (MOVEMASTER-Befehle)

100 MO	10	Position 10 anfahren
110 MO	12	Position 12 anfahren
120 MO	15	Position 15 anfahren
130 MO	17	Position 17 anfahren
140 MO	20	Position 20 anfahren
DL	110	Programmzeile 110 löschen
DL	120,140	Programmzeilen 120 bis 140 löschen



## 5.2.12 DP (Decrement Position)

### Funktion: Positionsnummer dekrementieren

Bewegt die Handspitze von der aktuellen Position an eine vordefinierte Position, deren Positionsnummer kleiner als die aktuelle ist (Gelenk-Interpolation).

### Eingabeformat

DP
----

### Erläuterung

- Dieser Befehl bewegt die Handspitze an die Position der nächst kleineren Positionsnummer (siehe auch IP-Befehl).
- Eine Fehlermeldung tritt auf, wenn keine Position existiert, deren Positionsnummer kleiner als die aktuelle Positionsnummer ist.
- Bei jedem Auftreten einer Fehlermeldung bleibt die aktuelle Position jedoch erhalten.

### Programmbeispiel (MOVEMASTER-Befehle)

100	MO	3	Position 3 anfahren
110	MO	4	Position 4 anfahren
120	MO	5	Position 5 anfahren
130	DP		Position 4 anfahren (nächst kleinere Position)

### 5.2.13 DR (Data Read)

#### Funktion: Daten lesen

Liest den Wert des internen Registers, die Handkontrollsignale und den allgemeinen Status der Ausgänge (über die RS232C-Schnittstelle).

#### Eingabeformat

DR    <Ausgangs-Bitnummer>
----------------------------

<Ausgangs-Bitnummer>       $0 \leq \text{Ausgangs-Bitnummer} \leq 32\,767$   
(0 für Standardwert)

#### Erläuterung

- Es werden die Datenwerte des internen Registers, die Handkontrollsignale und der allgemeine Ausgangsstatus über die RS232C-Schnittstelle ausgegeben. Die Daten vom externen Eingabeport und der Handgreiferzustand (geschlossen/geöffnet) können gelesen werden, wenn der DC-Befehl nach einem Eingabebefehl (ID) ausgeführt wird.
- Es wird jeweils der aktuelle Handgreiferzustand über den DR-Befehl gelesen.
- Durch Angabe der Ausgangs-Bitnummer kann der allgemeine Ausgangsstatus (16-Bit-Daten) beginnend mit der festgelegten Bitnummer gelesen werden.
- Das Ausgabeformat ist eine ASCII-codierte Hexadezimalzahl mit dem Anfangszeichen „&H“ und dem Abschlußzeichen „ , “ (hex. 2C).  
Ausgabeformat: Wert des Handkontrollsignals, Wert des internen Registers, allgemeiner Ausgangsstatus.  
Das erste Byte nach dem Anfangszeichen „&H“ entspricht dem Handkontrolleingang, die nächsten vier Byte entsprechen dem Wert des internen Registers.  
Die vier Byte nach dem nächsten Anfangszeichen „&H“ entsprechen dem allgemeinen Ausgangsstatus.
- Das Abschlußzeichen der Ausgabedaten ist ein „Carriage Return“ (CR, hex. 0D). Deshalb ist es beim Einsatz eines PCs erforderlich, daß die serielle Datenkette bis zur Stelle „hex. 0D“ empfangen wird. Der Befehl „LINE INPUT #“ ist der entsprechende BASIC-Befehl zum Einlesen der Daten.

**Programmbeispiel (BASIC-Befehle)**

10	OPEN "COM1 :E83" AS#1	RS232C-Kommunikationsdatei wird über den PC mittels BASIC-Befehl geöffnet
20	PRINT #1,"ID"	Daten vom Eingabeport in das interne Register schreiben
50	PRINT #1,"DR"	Daten des internen Registers und Handkontroll-eingang lesen
60	LINE INPUT #1,A\$	Daten in A\$ abspeichern
70	PRINT "Daten: ";A\$	Daten von A\$ auf dem Bildschirm anzeigen
80	END	Programmende

RUN

BASIC-Programm starten

Daten: &amp;H10FB2, &amp;H30BA

Folgende Daten werden angezeigt:

Wert des Handkontrollsignals: 1 (hex.)

Wert des internen Registers: 0FB2 (hex.)

Wert des allgemeinen Ausgangsstatus: 30BA (hex.)

## 5.2.14 DS (Draw Straight)

### Funktion: relative geradlinige Bewegung

Bewegt in bezug zur aktuellen Position das Ende der Hand (Handspitze) um den festgelegten Betrag in X-, Y- und Z-Richtung (Linear-Interpolation).

### Eingabeformat

```
DS  [<X-Verfahrbetrag>], [<Y-Verfahrbetrag>],  
    [<Z-Verfahrbetrag>]
```

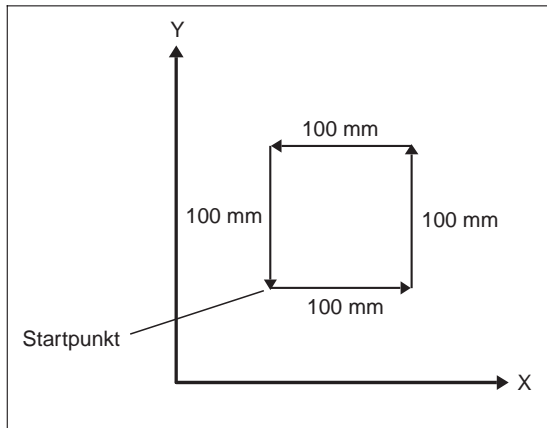
<X-Verfahrbetrag>	Legt den Verfahrwegbetrag in X-Achsenrichtung fest (in bezug zur aktuellen Position).
<Y-Verfahrbetrag>	Legt den Verfahrwegbetrag in Y-Achsenrichtung fest (in bezug zur aktuellen Position).
<Z-Verfahrbetrag>	Legt den Verfahrwegbetrag in Z-Achsenrichtung fest (in bezug zur aktuellen Position). (Verfahrbetrag 0 für Standardwert)

### Erläuterung

- Die Verfahrwegstrecke kann in Einheiten von 0,01 mm angegeben werden. Geben Sie für z. B. 20,01 mm den Wert 20.01 an.
- Die Handstellung, einschließlich des Handgreiferzustandes (offen/geschlossen), bleibt während der Roboterbewegung erhalten.
- Es tritt vor oder während der Roboterbewegung eine Fehlermeldung auf, wenn die Zielposition oder der Verfahrweg außerhalb des zulässigen Roboterarbeitsbereichs liegt. Achten Sie besonders auf die Einhaltung des zulässigen Handdrehwinkels, weil dieser während der Roboterbewegung so gesteuert wird, daß die Handrichtung immer beibehalten bleibt.
- Die Verfahrgeschwindigkeit während der Linear-Interpolation wird über den Befehl SD bestimmt. (Die Handspitze wird mit einer konstanten Geschwindigkeit bewegt.)
- Das Ende der Hand wird durch die Länge des aktuell eingesetzten Werkzeugs bestimmt (siehe TL-Befehl).

**Programmbeispiel (MOVEMASTER-Befehle)**

10	DS	100,0,0	Bewegung in (+X)-Achsenrichtung um 100 mm
20	DS	0,100,0	Bewegung in (+Y)-Achsenrichtung um 100 mm
30	DS	-100,0,0	Bewegung in (-X)-Achsenrichtung um 100 mm
40	DS	0,-100,0	Bewegung in (-Y)-Achsenrichtung um 100 mm



**Abb. 5-8:**  
*Verfahrbewegungen  
in X- und Y-Achsenrichtung*

R000101C

Im obigen Beispiel fährt die Handspitze mittels Linear-Interpolation die vier Ecken eines Quadrates ab. Die Bewegung endet wieder am Startpunkt.

## 5.2.15 DW (Draw)

### Funktion: relative Bewegung

Bewegt in bezug zur aktuellen Position die Handspitze um den festgelegten Betrag in X-, Y- und Z-Richtung (Gelenk-Interpolation).

### Eingabeformat

DW	[<X-Verfahrbetrag>], [<Y-Verfahrbetrag>], [<Z-Verfahrbetrag>]
----	--

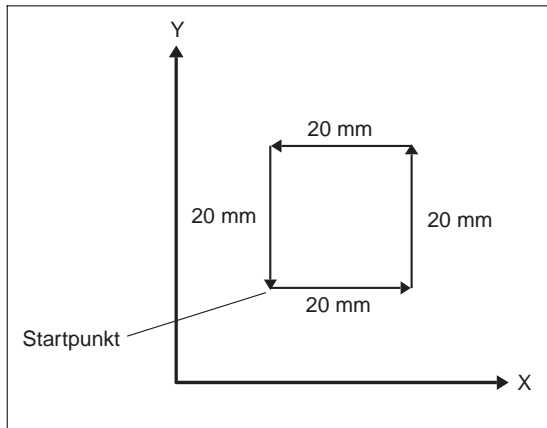
<X-Verfahrbetrag>	Legt den Verfahrwegbetrag in X-Achsenrichtung fest (in bezug zur aktuellen Position).
<Y-Verfahrbetrag>	Legt den Verfahrwegbetrag in Y-Achsenrichtung fest (in bezug zur aktuellen Position).
<Z-Verfahrbetrag>	Legt den Verfahrwegbetrag in Z-Achsenrichtung fest (in bezug zur aktuellen Position). (Verfahrbetrag 0 für Standardwert)

### Erläuterung

- Die Verfahrwegstrecke kann in Einheiten von 0,01 mm angegeben werden. Geben Sie für z. B. 20,01 mm den Wert 20.01 an.
- Die Handstellung, einschließlich des Handgreiferzustands (offen/geschlossen), bleibt während der Roboterbewegung erhalten.
- Es tritt vor oder während der Roboterbewegung eine Fehlermeldung auf, wenn die Zielposition oder der Verfahrweg außerhalb des zulässigen Roboterarbeitsbereichs liegt.
- Weil die Bewegung auf Gelenk-Interpolation basiert, verläuft die Handbewegung bogenförmig, wenn eine längere Verfahrstrecke zurückgelegt werden muß.
- Das Ende der Hand wird durch die Länge des aktuell eingesetzten Werkzeugs bestimmt (siehe TL-Befehl).

**Programmbeispiel (MOVEMASTER-Befehle)**

10	DW	20,0,0	Bewegung in (+X)-Achsenrichtung um 20 mm
20	DW	0,20,0	Bewegung in (+Y)-Achsenrichtung um 20 mm
30	DW	-20,0,0	Bewegung in (-X)-Achsenrichtung um 20 mm
40	DW	0,-20,0	Bewegung in (-Y)-Achsenrichtung um 20 mm



**Abb. 5-9:**  
*Verfahrbewegungen  
in X- und Y-Achsenrichtung*

R000101C

Im obigen Beispiel fährt die Handspitze mittels Gelenk-Interpolation die vier Ecken eines Quadrates ab. Die Bewegung endet wieder am Startpunkt.

## 5.2.16 EA (Enable Act)

### Funktion: Interrupt-Eingang festlegen

Ermöglicht eine Programmunterbrechung (Interrupt) in Abhängigkeit von den Bitzuständen der Eingangssignale.

### Eingabeformat

```
EA    [<+/->] <Eingangs-Bitnummer>, <Zeilennummer>
      [, [<Sprungmethode>]]
```

<+/->

Legt den Bitstatus fest.

- + : Der Programmsprung erfolgt, wenn das festgelegte Bit des externen Eingangssignals eingeschaltet ist.
- : Der Programmsprung erfolgt, wenn das festgelegte Bit des externen Eingangssignals ausgeschaltet ist.

<Eingangs-Bitnummer>

Legt fest, welches Bit am externen Eingabeport als Interrupt-Signal eingesetzt werden soll.

$1 \leq \text{Eingangs-Bitnummer} \leq 32\,767$

0 – 8999 : Eingangssignal-Interrupt

(0 – 299 : allgemein)

(900 – 903 : Handeingänge)

9003 – 32 767 : Eingangssignal-Interrupt

- Ermöglicht ein Interrupt über ein externes Signal.
- Mit dem EA-Befehl können gleichzeitig bis zu acht Eingabebits als Interrupt-Signal festgelegt werden.
- Nach Empfang des Interrupt-Signals wird die Roboterbewegung bis zum Stillstand abgebremst, und ein Programmsprung zur festgelegten Programmzeile ausgeführt.
- Ein einmal ausgelöster Interrupt bleibt so lange bestehen, bis ein DA-Befehl ausgeführt wird oder das Interrupt-Signal nicht mehr ansteht.

9000 : Alarm-Interrupt

Beim Auftreten einer Fehlermeldung, erfolgt die weitere Programmabarbeitung ab der festgelegten Zeilennummer.

9001 : NOT-HALT-Interrupt

Das Programm wird ebenfalls ab der festgelegten Zeile fortgesetzt, wenn ein NOT-HALT über das Bedienfeld, die Teaching Box oder durch ein externes Signal aktiviert wurde.

9002 : Kommunikations-Interrupt

Wenn Daten über die RS232C-Schnittstelle empfangen werden, wird das Programm ab der festgelegten Zeile abgearbeitet.

<Zeilennummer>

Legt die Zeilennummer für den Programmsprung fest.

$1 \leq \text{Zeilennummer (ganzzahlig)} \leq 9999$



<Sprungmethode>	Legt den Programmsprung oder den Unterprogrammaufruf fest.
0	: Sprung (Standardwert) (siehe GT-Befehl)
1	: Unterprogrammaufruf (siehe GS-Befehl)

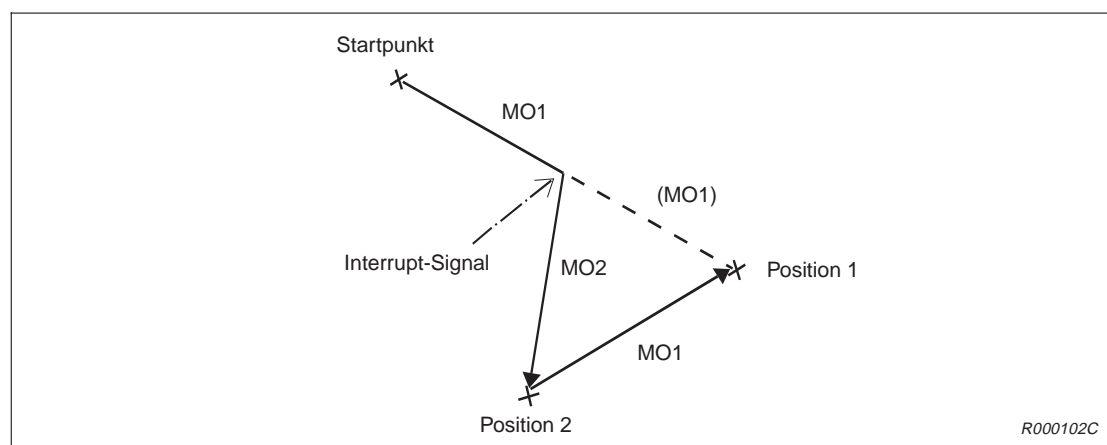
### Erläuterung

- Dieser Befehl bewirkt über ein externes Eingangssignal (Interrupt-Signal) eine Unterbrechung (Interrupt) der Programmabarbeitung.
- Wenn das Interrupt-Signal nach Ausführung des EA-Befehls anliegt, wird die Roboterbewegung bis zum Stillstand abgebremst, und das Programm springt zur festgelegten Programmzeile.
- Es können gleichzeitig bis zu acht Signale als Interrupt-Signale festgelegt werden. Wenn mehr als ein Interrupt-Signal ansteht, hat das Eingangssignal mit der größten Bitnummer Vorrang.
- Sobald dieser Befehl ausgeführt wurde, wird die Interrupt-Freigabebedingung bis zur Ausführung des Befehls DA (Interrupt ausschalten), ED (Programmende) oder RS (Programm zurücksetzen) beibehalten.
- Es tritt während der Programmabarbeitung eine Fehlermeldung auf, wenn die festgelegte Zeile im Programm nicht vorkommt.

### Programmbeispiel (MOVEMASTER-Befehle)

100	EA	+5,500	Bewirkt einen Programmsprung zur Zeile 500, wenn Bit 5 eingeschaltet wird
110	MO	1	Position 1 anfahren
120	ED		Programmende
500	DA	5	Interrupt ausschalten
510	MO	2	Position 2 anfahren
520	GT	110	Sprung zur Zeile 110

Im obigen Beispiel werden in der Zeile 100 die Interrupt-Bedingungen festgelegt. In der Zeile 110 erfolgt eine Roboterbewegung zur Position 1. Wenn das festgelegte Interrupt-Signal während der Roboterbewegung empfangen wird, stoppt der Roboter. Das Programm springt zur Zeile 500, an der der Interrupt wieder ausgeschaltet wird. In der Zeile 510 wird der Roboter zur Position 2 bewegt. Zeile 520 bewirkt einen Programmsprung zur Zeile 110, an der die Roboterbewegung zur Position 1 fortgesetzt wird.



**Abb. 5-10:** Beispiel zur Unterbrechung der Roboterbewegung über ein externes Interrupt-Signal

## 5.2.17 ED (End)

### Funktion: Programmende

Beendet das Programm.

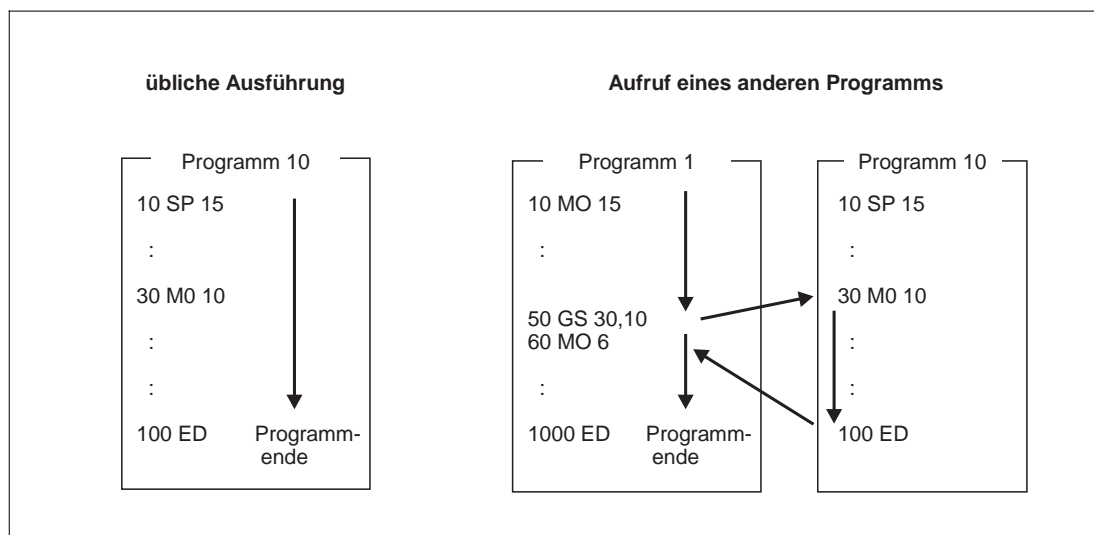
### Eingabeformat

ED

### Erläuterung

- Dieser Befehl markiert das Programmende. Das Programm wird nach Ausführung des ED-Befehls beendet. Es erfolgt ein Rücksprung zum Hauptprogramm, wenn der ED-Befehl in einem Unterprogramm ausgeführt wird.
- Am Programmende ist die Angabe des ED-Befehls immer erforderlich, es sei denn, die Programmbefehle werden direkt über den PC ausgeführt.

### Beispiele zur Ausführung des ED-Befehls



**Abb. 5-11:** Beispiele zur Ausführung des ED-Befehls

Aus einem Programm kann mit dem GS-Befehl ein anderes Programm aufgerufen werden. Das obige rechte Beispiel zeigt diese Aufrufmöglichkeit. Die Pfeile kennzeichnen die Reihenfolge der sequentiellen Programmabarbeitung. Das Programm Nr. 1 wird bis zur Zeile 50 ausgeführt. Danach erfolgt ein Sprung zum Programm Nr. 10. Nach Ausführung der Zeilen 30 bis 100 im Programm Nr. 10 erfolgt der Rücksprung zur Zeile 60 des Programms Nr. 1. Die Programmabarbeitung endet nach Ausführung des ED-Befehls im Programm Nr. 1.

### Programmbeispiel (MOVEMASTER-Befehle)

100	SP	3	Einstellen der Gelenk-Interpolations-Geschwindigkeit auf den Wert 3
110	MO	3	Position 3 anfahren
120	MO	5	Position 5 anfahren
130	ED		Programmende

## 5.2.18 EQ (Equal)

### Funktion: Datenwertvergleich: =

Bewirkt einen Programmsprung, wenn der Wert des internen Registers oder des Zeichenkettenregisters mit dem festgelegten Vergleichswert übereinstimmt.

### Eingabeformat

EQ <Vergleichswert>, <Zeilennummer des Sprungziels>

<Vergleichswert>	Legt den Wert fest, der mit dem Wert des internen Registers verglichen werden soll. $-32\,768 \leq \text{Vergleichswert (dez.)} \leq 32\,767$ $\&8000 \leq \text{Vergleichswert (hex.)} \leq \&7FFF$ $@ \leq \text{Zählernummer} \leq @99$
<Zeichenkettennummer>	Legt die Zeichenkette fest. Das führende Zeichen ist „\$“. $\$1 \leq \text{Zeichenkettennummer} \leq \$99$
<Zeilennummer des Sprungziels>	Legt die Zeilennummer fest, an der das Programm bei Wertegleichheit springen soll. $1 \leq \text{Zeilennummer} \leq 9999$

### Erläuterung

<Bei Angabe einer Zählernummer>

- Dieser Befehl bewirkt einen Programmsprung, wenn die Sprungbedingung über externe Eingabedaten oder durch einen internen Zählerwert gegeben ist.
- Wenn der Wert des internen Registers mit dem Vergleichswert übereinstimmt (d. h. wenn die Bedingung erfüllt ist), springt das Programm zur festgelegten Zeilennummer. Andernfalls (d. h. wenn die Bedingung nicht erfüllt ist) wird der Programmablauf ohne Sprung fortgesetzt.
- Ein Wert kann entweder durch Ausführung des Eingabebefehls für externe Eingabedaten (siehe ID-Befehl) oder durch Ausführung des Zählervergleichsbefehls (siehe CP-Befehl) in das interne Register geladen werden. Um einen bedingten Sprung zu ermöglichen, ist es erforderlich, daß einer der beiden Befehle vor dem EQ-Befehl ausgeführt wird.
- Der Vergleichswert kann entweder dezimal oder hexadezimal definiert werden. Bei einem hexadezimalen Wert muß vor der Zahl ein „&“ stehen.
- Es tritt bei der Sprungausführung eine Fehlermeldung auf, wenn die festgelegte Programmzeilennummer nicht existiert.

<Bei Angabe einer Zeichenkettennummer>

- Dieser Befehl bewirkt einen Programmsprung, wenn die Sprungbedingung über den Inhalt des Zeichenkettenregisters gegeben ist.
- Wenn der Inhalt des Zeichenkettenregisters mit dem der festgelegten Zeichenkette übereinstimmt (d. h. die Bedingung ist erfüllt), springt das Programm zur festgelegten Zeilennummer. Andernfalls (d. h. wenn die Bedingung nicht erfüllt ist) wird der Programmablauf ohne Sprung fortgesetzt.

- Durch Ausführung des INP-Befehls werden externen Daten in das Zeichenkettenregister geladen. Die Werte einer Zeichenkettennummer werden durch Ausführung des CP-Befehls geladen. Um einen bedingten Sprung zu ermöglichen, ist es erforderlich, daß einer der beiden Befehle vor dem EQ-Befehl ausgeführt wird.
- Es tritt bei der Sprungausführung eine Fehlermeldung auf, wenn die festgelegte Programmzeilennummer nicht existiert.

**Programmbeispiel (MOVEMASTER-Befehle)**

100	ID		Daten vom externen Eingabeport holen
110	EQ	100,130	Sprung zur Zeile 130, wenn die Eingabedaten gleich 100 sind
120	ED		Beendet das Programm, wenn die obere Sprungbedingung nicht erfüllt ist
130	SC	\$5,"OK"	Zeichenkette "OK" in die Zeichenkette Nummer 5 laden
140	CP	\$5	Werte der Zeichenkette Nummer 5 in das Zeichenkettenregister laden
150	EQ	\$10,200	Sprung zur Zeile 200, wenn die Daten und die Zeichenkette Nummer 10 gleich sind
200	MO	7	Position 7 anfahren

## 5.2.19 ER ♦ (Error Read)

### Funktion: Fehler lesen

Liest den aktuellen Fehlerstatus und die bis dahin aufgetretenen Fehlercodes (über die RS232C-Schnittstelle).

### Eingabeformat

ER    [<Nummer der Fehlermeldung>]
------------------------------------

<Nummer der Fehlermeldung>

Legt fest, welcher der bisher angezeigten Fehlercodes gelesen werden soll.

$1 \leq \text{Nummer der Fehlermeldung} \leq 16$

(Bei fehlender Nummerangabe wird der aktuelle Fehlerstatus angezeigt.)

### Erläuterung

- Es wird der Fehlerstatus des Roboters über die RS232C-Schnittstelle ausgegeben.
- Die Nummer des Fehlercodes wird auf Basis des aktuellen Fehlercodes (0) festgelegt. Bei Angabe der Nummer 1 wird der zuvor aufgetretene Fehlercode gelesen usw.
- Der ER-Befehl gibt die bisher aufgetretenen Fehlercodes im nachfolgend aufgeführten ASCII-Format aus.  
Ausgabeformat: Nummer der Fehlermeldung, Fehlercode, Jahr, Monat, Tag, Stunde, Minute und Sekunde.
- Bei einer fehlenden Nummernangabe wird über den ER-Befehl der Fehlertyp mit folgenden ASCII-Code ausgegeben.  
0 : kein Fehler  
1 : schwerer Fehler (Fehlercode 0100 – 1900)  
2 : Betriebsfehler (Fehlercode 2300 – 8900)
- Das Abschlußzeichen der Ausgabedaten ist ein „Carriage Return“ (CR, hex. 0D). Deshalb ist es beim Einsatz eines PCs erforderlich, daß die serielle Datenkette bis zur Stelle „hex. 0D“ empfangen wird. Der Befehl „LINE INPUT #“ ist der entsprechende BASIC-Befehl zum Einlesen der Daten.
- Mit diesem Befehl können die während des Roboterbetriebs aufgetretenen Fehlercodes zum PC übertragen werden.

**1. Programmbeispiel (BASIC-Befehle)**

10	OPEN "COM1 :E83" AS#1	RS232C-Kommunikationsdatei wird über den PC mittels BASIC-Befehl geöffnet
20	PRINT #1,"MO1"	Position 1 anfahren
30	GOSUB 100	BASIC-Unterprogramm in Zeile 100 aufrufen
40	PRINT #1,"MO2"	Position 2 anfahren
50	GOSUB 100	BASIC-Unterprogramm in Zeile 100 aufrufen
60	END	Programmende
100	PRIN #1,"ER"	Aktuellen Fehlercode lesen
110	LINE INPUT #1,A\$	Daten in A\$ abspeichern
120	IF A\$ = "0" THEN RETURN	Rücksprung zum Unterprogramm, wenn kein Fehlercode vorhanden
130	PRINT "Fehlertyp = " ;A\$	Fehlertyp wird auf dem Bildschirm des PCs angezeigt
140	END	Programmende
RUN		BASIC-Programm starten
Fehlertyp = 2		Fehlertyp wird auf dem Bildschirm angezeigt (2 : Betriebsfehler)

**2. Programmbeispiel (BASIC-Befehle)**

10	OPEN "COM1 :E83" AS#1	RS232C-Kommunikationsdatei wird über den PC mittels BASIC-Befehl geöffnet
20	INPUT "Fehlernummer = " ;N	Eingabe der Fehlernummer über den PC
30	PRINT #1,"ER+STR (N)"	Fehlerinformationen lesen
40	LINE INPUT#1, A\$	Daten in A\$ abspeichern
50	PRINT "Fehlerinformationen: ",A\$	Daten auf dem Bildschirm des PCs anzeigen
60	END	Programmende
RUN		BASIC-Programm starten
Fehlernummer = 1		Fehlertyp ( 1)
Fehlerinformationen 1,3800,94,10,21,11,34,20		Fehlertyp (1), Fehlercode (3800), Datum (21.10.94), Uhrzeit (11:34:20)

## 5.2.20 GC (Grip Close)

### Funktion: Handgreifer schließen

Schließt den Greifer der Hand.

### Eingabeformat

GC    [ <Handnummer> ]
------------------------

<Handnummer>            Legt fest, welche Hand geschlossen werden soll.  
                                   0 : Hand 1 (Standardwert)  
                                   1 : Hand 2  
                                   3 : Hand 3

### Erläuterung

- **pneumatisch angetriebene Hand**  
 Der Befehl aktiviert das Magnetventil zum Schließen der Hand oder zur Aufnahme des Arbeitsgegenstands. Beachten Sie in diesem Fall die Hinweise zu den Bitmustern der Ausgänge (siehe Beschreibung zum OB-Befehl).
- Zum Positionieren und Greifen eines Arbeitsgegenstands ist eine bestimmte Zeitdauer erforderlich. Deshalb muß gegebenenfalls vor und nach dem GC-Befehl eine Zeitverzögerung mit dem Befehl TI programmiert werden. Die Ausführungszeit des GC-Befehls wird durch die über den Parameter „Haltezeit für Start-Greifkraft“ festgelegte Zeit bestimmt.

### Parameter

Mit den nachfolgenden Parameter kann der Handgreiferzustand (offen/geschlossen) beim Ausführen eines Handbefehls und beim Einschalten der Spannungsversorgung geändert werden.

Parameter GCD :            Hand 1 vorwärts/rückwärts, Standardwert: Hand 1  
                                   Hand 2 vorwärts/rückwärts, Standardwert: Hand 2  
                                   Hand 3 vorwärts/rückwärts, Standardwert: Hand 3

Die Vorwärts-/Rückwärtseinstellung legt fest, welche Handrichtung bei einer Befehlsausführung gewählt werden soll (0: vorwärts, 1: rückwärts).

Die Standardwerteinstellungen legen fest, welcher Handgreiferzustand beim Einschalten der Spannungsversorgung gewählt werden soll. Tabelle 5-2 zeigt die Parametereinstellungen für den Handgreiferzustand der pneumatisch angetriebenen Hand.

Hand 1	0	1	2	3	Hand 2	0	1	2	3	Hand 3	0	1	2	3
Ausg.-bit 900	AUS	EIN	AUS	EIN	Ausg.-bit 902	AUS	EIN	AUS	EIN	Ausg.-bit 904	AUS	EIN	AUS	EIN

**Tab. 5-2:**    *Parametereinstellungen für den Handgreiferzustand der pneumatisch angetriebenen Hand*

Die werkseitige Standardeinstellung ist der Wert 1.

**Programmbeispiel (MOVEMASTER-Befehle)**

10	MO	10,O	Position 10 mit geöffneter Hand anfahren
20	TI	5	Wartezeit von 0,5 s
30	GC		Hand zum Ergreifen des Arbeitsgegenstands schließen
40	TI	5	Wartezeit von 0,5 s
50	MO	15,C	Position 15 mit geschlossener Hand anfahren

**HINWEIS**

Wenn Position 15 mit offener Hand geteacht wurde, wird auf dem Weg zur Position 15 die Hand geschlossen.



## 5.2.21 GF (Grip Flag)

### Funktion: Handgreiferzustand festlegen

Legt den Handgreiferzustand (offen/geschlossen) fest (wird in Verbindung mit dem PD-Befehl eingesetzt).

### Eingabeformat

GF    <Schalter>
------------------

<Schalter>	Legt den Handgreiferzustand fest
0	: Handgreifer offen
1	: Handgreifer geschlossen

### Erläuterung

Dieser Befehl definiert den Handgreiferzustand (offen/geschlossen). Der Befehl wird in Verbindung mit dem PD-Befehl eingesetzt, der die Positionskoordinaten festlegt.

### Programmbeispiel (MOVEMASTER-Befehle)

10	GF	0	Handgreiferzustand festlegen (Hand offen)
20	PD	10,50,320,70,50,40,30,R	Position 10 definieren

Der PD-Befehl wird vorrangig ausgeführt, wenn der Handgreiferzustand mit dem PD-Befehl direkt angegeben wird.

10	PD	10,50,320,70,50,40,30,R, <b>O</b>	Position 10 definieren (Hand offen)
----	----	-----------------------------------	-------------------------------------

## 5.2.22 GO (Grip Open)

### Funktion: Handgreifer öffnen

Öffnet den Greifer der Hand.

### Eingabeformat

GO    [ <Handnummer> ]
------------------------

<Handnummer>            Legt fest, welche Hand geöffnet werden soll  
                                  0 : Hand 1 (Standardwert)  
                                  1 : Hand 2  
                                  2 : Hand 3

### Erläuterung

#### pneumatisch angetriebene Hand

Der Befehl aktiviert das Magnetventil zum Öffnen der Hand oder zur Aufnahme des Arbeitsgegenstands. Beachten Sie in diesem Fall die Hinweise zu den Bitmustern der Ausgänge (siehe Beschreibung zum OB-Befehl).

- Zum Positionieren und Greifen eines Arbeitsgegenstands ist eine bestimmte Zeitdauer erforderlich. Deshalb muß gegebenenfalls vor und nach dem GO-Befehl eine Zeitverzögerung mit dem Befehl TI programmiert werden. Die Ausführungszeit des GO-Befehls wird durch die über den Parameter „Haltezeit für Start-Handkraft“ festgelegte Zeit bestimmt.

### Parameter

Mit den nachfolgenden Parameter kann der Handgreiferzustand (offen/geschlossen) beim Ausführen eines Handbefehls und beim Einschalten der Spannungsversorgung geändert werden.

Parameter GCD :            Hand 1 vorwärts/rückwärts, Standardwert: Hand 1  
                                  Hand 2 vorwärts/rückwärts, Standardwert: Hand 2  
                                  Hand 3 vorwärts/rückwärts, Standardwert: Hand 3

Die Vorwärts-/Rückwärtseinstellung legt fest, welcher Handrichtung bei einer Befehlausführung gewählt werden soll (0: vorwärts, 1: rückwärts)

Die Standardwerteinstellungen legen fest, welcher Handgreiferzustand beim Einschalten der Spannungsversorgung gewählt werden soll. Tabelle 5-3 zeigt die Parametereinstellungen für den Handgreiferzustand der pneumatisch angetriebenen Hand.

Hand 1	0	1	2	3	Hand 2	0	1	2	3	Hand 3	0	1	2	3
Ausg.-bit 900	AUS	EIN	AUS	EIN	Ausg.-bit 902	AUS	EIN	AUS	EIN	Ausg.-bit 904	AUS	EIN	AUS	EIN

**Tab. 5-3:**    *Parametereinstellungen für den Handgreiferzustand der pneumatisch angetriebenen Hand*

Die werkseitige Standardeinstellung ist der Wert 1.

## 5.2.23 GS (Go Sub)

### Funktion: Sprung zu einem Unterprogramm

Bewirkt einen Sprung zu einem Unterprogramm, das mit der festgelegten Programmzeilennummer beginnt.

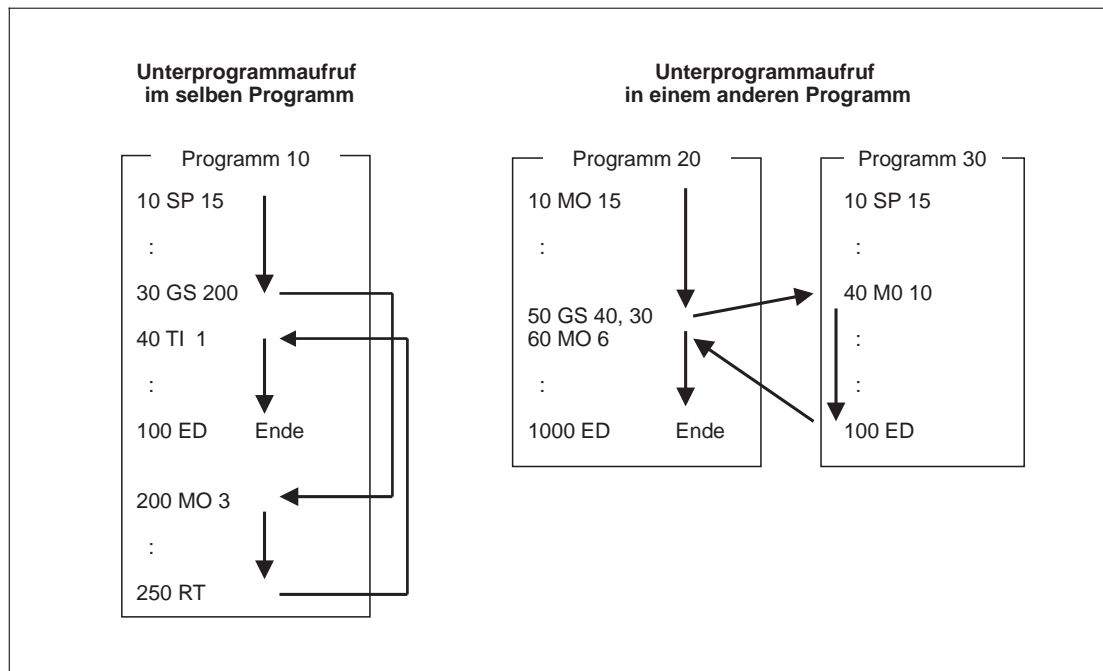
### Eingabeformat

GS    [<Zeilennummer>] [, [<Programmname>]]
---

<Zeilennummer>	Legt die Zeilennummer für den Programmsprung fest $1 \leq \text{Zeilennummer (ganzzahlig)} \leq 9999$
<Programmname>	Legt den Namen des Unterprogramms fest (Zeichenkette mit max. 8 Stellen) $0 \leq \text{Programmname} \leq 8 \text{ (Zeichen)}$
zulässige Zeichen:	Zahlen (0 – 9) Buchstaben (A – Z) Symbole (! @ # \$ % ^ & ( ) _   ^ { } -)
unzulässige Zeichen:	* + , . / : ; = ? [ ¥ ] ‘
spezielle Programmnamen:	Wenn der Programmname ausschließlich aus numerischen Werten besteht, wird dieser wie eine Nummernangabe verarbeitet. Bei der Verwendung von Buchstaben muß der Programmname in Anführungszeichen “ ” gesetzt werden.

### Erläuterung

- Dieser Befehl bewirkt einen Sprung zu einem Unterprogramm. Das Unterprogramm wird über den Unterprogrammnamen und die Startzeile festgelegt.
- Nach Ausführung des Unterprogramms erfolgt wieder ein Rücksprung zum Hauptprogramm. Wenn das Unterprogramm ein Bestandteil desselben Programms ist, müssen Sie zum Abschluß des Unterprogramms den RT-Befehl programmieren. Programmieren Sie am Unterprogrammende den ED-Befehl, wenn das Unterprogramm Bestandteil eines anderen Programms ist.
- Bei der Ausführung des GS-Befehls tritt eine Fehlermeldung auf, wenn die festgelegte Zeilennummer und/oder der Programmname nicht existieren.
- Wurde keine Zeilennummer vorgegeben, wird das festgelegte Programm von der obersten Zeile an abgearbeitet.
- Es erfolgt keine Ausführung, wenn sowohl die Zeilenangabe als auch der Programmname fehlen.
- Innerhalb eines Unterprogramms können weitere Unterprogramme aufgerufen werden. Es können bis zu 9 Unterprogramme ineinander verschachtelt (Nesting) werden.

**Beispiele zur Ausführung des GS-Befehls****Abb. 5-12:** Beispiele zur Ausführung des GS-Befehls

Im obigen linken Beispiel wird das Programm von der Zeile 10 bis zur Zeile 30 abgearbeitet. Dort erfolgt ein Sprung zum Unterprogramm, das mit der Zeile 200 beginnt. Das Unterprogramm wird bis zum RT-Befehl abgearbeitet. Der RT-Befehl bewirkt den Rücksprung zum Hauptprogramm. Die Abarbeitung des Hauptprogramms wird an der Zeile 40 fortgesetzt. Das Hauptprogramm wird durch Ausführung des ED-Befehls beendet.

Durch Einsatz des GS-Befehls kann aus einem Programm ein weiteres Programm aufgerufen werden. Im obigen rechten Beispiel wird das Hauptprogramm Nr. 20 bis zur Zeile 50 abgearbeitet. Dort wird das Programm Nr. 30 aufgerufen. Das Programm Nr. 30 wird von der Zeile 40 bis zur Zeile 100 abgearbeitet. Nach Ausführung des ED-Befehls erfolgt der Rücksprung zum Hauptprogramm (Programm Nr. 20, Zeile 60). Dort wird die Abarbeitung bis zur Zeile 1000 fortgesetzt.

**Programmbeispiel (MOVEMASTER-Befehle)**

10	GS	100	Sprung zum Unterprogramm (Zeile 100)
:			
90	ED		Hauptprogrammende
100	MO	11	Position 11 anfahren
110	MO	12	Position 12 anfahren
120	MO	13	Position 13 anfahren
130	RT		Unterprogrammende

## 5.2.24 GT (Go To)

### Funktion: Sprung zu einer Programmzeile

Bewirkt einen Sprung zu der festgelegten Programmzeile (ohne Sprungbedingung).

### Eingabeformat

GT    [<Zeilennummer>]
------------------------

<Zeilennummer>      Legt die Zeilennummer für den Programmsprung fest  
 $1 \leq \text{Zeilennummer (ganzzahlig)} \leq 9999$

### Erläuterung

- Dieser Befehl bewirkt einen Sprung zu der über die Zeilennummer festgelegten Programmzeile.
- Existiert die festgelegte Programmzeilennummer nicht, tritt eine Fehlermeldung auf.

### Programmbeispiel (MOVEMASTER-Befehle)

10	MO	1	Position 1 anfahren
20	GT	100	Unbedingter Programmsprung zur Zeile 100
	:		
100	MO	12	Position 12 anfahren
110	MO	15	Position 15 anfahren
	:		

## 5.2.25 HE (Here)

### Funktion: aktuelle Position speichern

Speichert die aktuellen Koordinatenwerte als Position ab.

### Eingabeformat

HE    [<Positionsnummer>]
---------------------------

<Positionsnummer>      Legt die Positionsnummer für die Registrierung fest  
 $0 \leq \text{Positionsnummer (ganzzahlig)} \leq 999$   
 (Bei Angabe des Wertes 0 wird die aktuelle Position als benutzerdefinierter Nullpunkt abgespeichert.)

### Erläuterung

- Die Koordinaten der aktuellen Position werden auf der Basis der aktuell eingestellten Werkzeuglänge berechnet (siehe TL-Befehl). Die Werkzeuglänge ergibt sich aus dem Abstand zwischen der Handbefestigungsplatte und der Handspitze. Der Initialisierungswert für die Werkzeuglänge beträgt 107 mm.
- Wenn zwei verschiedene Positionen der gleichen Positionsnummer zugeordnet werden, erhält die zuletzt definierte Position Vorrang. Die andere Position wird gelöscht.
- Der Handgreiferzustand (offen/geschlossen) und die Stellungsdaten des Roboters werden ebenfalls als Positionsdaten abgespeichert.
- Eine Fehlermeldung tritt auf, wenn der HE-Befehl ausgeführt wird, obwohl vorher noch keine Nullpunkteinstellung vorgenommen wurde.
- Bei Angabe der Positionsnummer 0 werden die aktuellen Positionsdaten als benutzerdefinierter Nullpunkt über den Parameter UOG abgespeichert (in Gelenk-Koordinaten). In diesem Fall müssen Sie zuerst den Wert des Parameters HOE ändern. Der Parameter HOE legt die Freigabe für die Nullpunkteinstellung fest. Setzen Sie nach der Nullpunkteinstellung den Parameter wieder auf den vorherigen Wert (Nullpunkteinstellung sperren).

### HINWEIS

Die zuvor beschriebenen Bedienvorgänge sind nur bei einer direkten Befehlsausführung wirksam.

### Programmbeispiel (MOVEMASTER-Befehle)

10	MO	10	Position 10 anfahren
20	DW	10,0,0	Roboter um 10 mm in (+X)-Richtung verfahren
30	HE	11	Aktuelle Koordinatenwerte als Position 11 definieren

### 5.2.26 HLT (Halt)

**Funktion: Programmablauf stoppen**

Stoppt die Roboterbewegung und den Programmablauf.

**Eingabeformat**

HLT
-----

**Erläuterung**

- Der Befehl stoppt die Abarbeitung des Programms. Die Roboterbewegung wird bis zum Stillstand abgebremst. Es gelten die gleichen Ausführungsbedingungen wie bei einem externen Stopp-Signal oder bei Betätigung des STOP-Schalters an der Vorderseite des Steuergerätes.
- Ein Neustart des Programms ist durch Betätigen des START-Schalters, über ein externes Start-Signal oder durch Ausführung des RN-Befehls möglich. Der Programmneustart beginnt eine Zeile nach dem HLT-Befehl.
- Das Programm und die Roboterbewegung wird auch dann gestoppt, wenn der HLT-Befehl bei laufender Programmabarbeitung direkt über einen PC ausgeführt wird.
- Während der Ausführung eines direkten Verfahrbefehls (z. B. ein von einem PC gesendeter Bewegungsbefehl) kann die Roboterbewegung mit dem HLT-Befehl nicht gestoppt werden.

**Programmbeispiel (MOVEMASTER-Befehle)**

10	MO	1	Position 1 anfahren
20	HLT		Programmablauf und Roboterbewegung stoppen
30	MO	2	Position 2 anfahren
40	ED		Programmende

Der Programmneustart beginnt nach Betätigung des START-Schalters bei Zeile 30.

## 5.2.27 HO (Home)

### Funktion: Nullpunkt einstellen

Speichert die aktuellen Koordinatenwerte als Nullpunkt ab.

### Eingabeformat

HO    [<Methode der Nullpunkteinstellung>]
--

<Methode der Nullpunkteinstellung>

Legt die Methode der Nullpunkteinstellung fest (ganzzahliger Wert).

0 : Nullpunkteinstellung am mechanischen Endanschlag

1 : Nullpunkteinstellung mit Kalibriervorrichtung

2 : benutzerdefinierter Nullpunkt

### Erläuterung

- Dieser Befehl legt den Bezugspunkt (Nullpunkt) für das Koordinatensystem fest.
- Eine Nullpunkteinstellung mit dem HO-Befehl ist immer dann erforderlich, wenn der Roboter oder die Kombination aus Roboter und Steuergerät ausgetauscht wurden. Die Nullpunkteinstellung kann auch über die Teaching Box vorgenommen werden (siehe für weitere Details Abschnitt 1.3).
- Ändern Sie vor einer Nullpunkteinstellung zuerst den Wert des Parameters HOE. Der Parameter HOE legt die Freigabe für die Nullpunkteinstellung fest. Setzen Sie nach der Nullpunkteinstellung den Parameter wieder auf den vorherigen Wert (Nullpunkteinstellung sperren), andernfalls kann das Programm nicht gestartet werden.

### Parameter

Parameter HOE: Legt die Freigabe für die Nullpunkteinstellung über den HO-Befehl fest.

Parameterwerte:

0 : Nullpunkteinstellung über den HO-Befehl sperren

1 : Nullpunkteinstellung über den HO-Befehl freigeben

### 1. Programmbeispiel (BASIC-Befehle)

10	OPEN "COM1 :E83"	AS#1	RS232C-Kommunikationsdatei wird über den PC mittels BASIC-Befehl geöffnet
20	PRINT #1,"HO"		HO-Befehl wird über den PC ausgeführt
30	END		Programmende
RUN			BASIC-Programm starten



## 5.2.28 IC (Increment Counter)

### Funktion: Zählerwert inkrementieren

Addiert 1 zum Wert des festgelegten Zählers.

### Eingabeformat

IC    [<Zählernummer>]
------------------------

<Zählernummer>

Legt den Zähler fest.

$1 \leq \text{Zählernummer (ganzzahlig)} \leq 99$

### Erläuterung

- Eine Fehlermeldung tritt auf, wenn der Zählerwert den Maximalwert von 32 767 überschreitet.
- Der Befehl kann zum Zählen von Arbeitsgegenständen und Arbeitsvorgängen oder zum Festlegen der Anzahl der Gitterpunkte bei Palettenarbeit eingesetzt werden.
- Der Zählerwert kann mit dem entsprechenden Befehlen verändert, verglichen oder gelesen werden (siehe Befehle SC, DC, CP, CR, CL, AN, OR und XO).

### Programmbeispiel (MOVEMASTER-Befehle)

10	SC	21,15	Wert 15 in den Zähler 21 laden
20	IC	21	Wert 1 zum Inhalt des Zählers 21 addieren

## 5.2.29 ID (Input Direct)

### Funktion: Eingänge einlesen

Liest Daten vom externen Eingang und Handkontrolleingang (ohne Ausführungsbedingung).

### Eingabeformat

ID    [<Eingangs-Bitnummer>]
------------------------------

<Eingangs-Bitnummer>    Legt die Bitnummer für den Eingabeport fest.  
 Die Datenbreite beträgt 16 Bit (inklusive des festgelegten Bits).  
 $0 \leq \text{Eingangs-Bitnummer (ganzzahlig)} \leq 32\,767$

### Erläuterung

- Dieser Befehl liest ohne Ausführungsbedingung die Signale von externen Geräten (z. B. speicherprogrammierbare Steuerung). Durch Angabe der Nummer 900 für die Eingangs-Bitnummer können die Handkontrolleingänge ausgelesen werden.
- Die eingelesenen Daten werden in das interne Register geladen und können zum Vergleichen oder für Bit-Überprüfungen usw. eingesetzt werden (siehe Befehle EQ, NE, LG, SM und TB).

### Programmbeispiel (MOVEMASTER-Befehle)

100	ID		Lädt zum Vergleichen die Eingabedaten in das interne Register
110	EQ	100,130	Sprung zur Zeile 130, wenn die Eingabedaten gleich 100 sind
120	ED		Andernfalls wird das Programm beendet
130	MO	1	Position 1 anfahren
140	ID	100	Lädt zum Vergleichen die Eingabedaten in das interne Register
150	TB	+0,180	Sprung zur Zeile 180, wenn das Eingangsbit 0 eingeschaltet ist
160	TB	+5,200	Sprung zur Zeile 200, wenn das Eingangsbit 5 eingeschaltet ist
170	ED		Andernfalls wird das Programm beendet
180	MO	2	Position 2 anfahren
190	ED		Programmende
200	MO	3	Position 3 anfahren
210	ED		Programmende

## 5.2.30 INP (Input)

### Funktion: Zählerwert und Positionsdaten lesen

Liest die mit dem PRN-Befehl übertragenen Zählerwerte, Positionsdaten und Zeichenketten-  
daten (vom RS232C-Port).

### Eingabeformat

```
INP <Kanalnummer>, <Zähler/Positionsnummer/Zeichenkettennummer>
    [, [<Datenauswahl>]]
```

<Kanalnummer>	Legt den Kanal fest, der über den OPN-Befehl geöffnet werden soll. $1 \leq \text{Kanalnummer} \leq 4$
<Zähler/Positionsnummer>	Legt den Zähler oder die Position fest. $1 \leq \text{Zählernummer} \leq 99$ $1 \leq \text{Positionsnummer} \leq 999$
<Zeichenkettennummer>	Legt die Zeichenkette fest. Das führende Zeichen ist „\$“. $\$1 \leq \text{Zeichenkettennummer} \leq \$99$
<Datenauswahl>	Legt fest, ob Zählerwerte oder Positionsdaten gelesen werden sollen. 0 : Zählerwerte (Standardwert) 1 : Positionsdaten 2 : Zeichenkettennummer

### Erläuterung

- Dieser Befehl liest die am RS232C-Port anliegenden Daten des festgelegten Zählers, der festgelegten Position oder der festgelegten Zeichenkette.
- Vor der Ausführung eines INP-Befehls, muß zuvor die RS232C-Schnittstelle mit einem OPN-Befehl geöffnet werden.
- Verwenden Sie den PRN-Befehl, um Daten vom PC zum Steuergerät zu übertragen. Während der Datenübertragung wird der Programmablauf gestoppt. Die mit dem PRN-Befehl übertragenen Daten können mit INP-Befehl in den festgelegten Zähler, zur festgelegten Position oder Zeichenkette geladen werden. Es können maximal 256 Zeichen registriert werden. Bei mehrfacher Ausführung des PRN-Befehls und Überschreitung der maximalen Zeichenzahl, wird der Roboter bezogen auf die RS232C-Schnittstellenleitungen ER (DRT) und RS (RTS) in den L-Zustand gesetzt (DR (DSR) und CS (CTS) sind die Signalleitungen auf der PC-Seite). Die Datenübertragung vom Personalcomputer wird in dieser Zeit unterbrochen.
- Bei der Ausführung des INP-Befehls tritt eine Fehlermeldung auf, wenn die mit dem PRN-Befehl übertragenen Daten fehlerhaft sind.

**Programmbeispiel (MOVEMASTER-Befehle)**

10	OPN	2,1	RS232C-Schnittstelle öffnen
20	INP	2,1,0	Wert des Zählers 1 vom RS232C-Port lesen
30	INP	2,5,1	Positionsdaten für Position 5 vom RS232C-Port lesen
40	IC	1	Zahlenwert 1 zum Inhalt des Zählers 1 addieren
50	MO	5	Position 5 anfahren
60	OPN	1,1	RS232C-Schnittstelle öffnen
70	INP	1,\$10,2	Daten der Zeichenkette Nummer 10 vom RS232C-Port lesen

### 5.2.31 IP (Increment Position)

**Funktion: Positionsnummer inkrementieren**

Bewegt den Roboter von der aktuellen Position zu einer vordefinierten Position, deren Positionsnummer größer als die aktuelle ist (Gelenk-Interpolation).

**Eingabeformat**

IP
----

**Erläuterung**

- Dieser Befehl bewegt den Roboter an die Position der nächst größeren Positionsnummer (siehe auch DP-Befehl).
- Eine Fehlermeldung tritt auf, wenn keine Position existiert, deren Positionsnummer größer als die aktuelle Positionsnummer ist.
- Bei jedem Auftreten einer Fehlermeldung bleibt die aktuelle Position jedoch erhalten.

**Programmbeispiel (MOVEMASTER-Befehle)**

10	MO	5	Position 5 anfahren
10	MO	4	Position 4 anfahren
20	MO	3	Position 3 anfahren
40	IP		Position 4 anfahren (nächst größere Position)
50	IP		Position 5 anfahren (nächst größere Position)

## 5.2.32 JRC (Joint Roll Change)

### Funktion: Addition von $\pm 360^\circ$ zur aktuellen Position der R-Achse

Überschreibt die aktuelle Position der R-Achse durch Addition von  $\pm 360^\circ$ . Dies kann zur kurzzeitigen oder zur kontinuierlichen Steuerung der R-Achse erfolgen.

### Eingabeformat

JRC   <[+]   1/-1>
--------------------

<+1>                      Addiert  $360^\circ$  zur aktuellen Position der R-Achse.

<-1>                      Subtrahiert  $360^\circ$  von der aktuellen Position der R-Achse.

### Erläuterung

- Wenn Sie diesen Befehl anwenden wollen, müssen Sie zuvor den Arbeitsbereich der R-Achse auf  $\pm 720^\circ$  einstellen. Setzen Sie dazu im Parameter JAR den Verfahrweggrenzwert +J6 auf 720 und -J6 auf -720.  
Informationen zur Vorgehensweise beim Einstellen der Parameter entnehmen Sie bitte dem technischen Handbuch, Abschnitt 3.2 „Editieren der Parameter“.  
Eine Einstellung der Parameterwerte auf einen Wert kleiner/größer  $\pm 720$  kann zu Positionierfehlern führen.
- Obwohl sich der aktuelle Koordinatenwert ändert, bewegt sich der Roboter nicht.
- Wenn bei Aufruf des JRC-Befehls der Koordinatenwert der R-Achse  $\pm 720^\circ$  überschreitet, bleibt der aktuelle Koordinatenwert erhalten und der Alarm bei Überschreiten des Wertebereichs wird ausgegeben.
- Setzen Sie den RV-E4NM/E4NC ein, und wollen Sie die R-Achse mit diesem Befehl um mehr als  $\pm 180^\circ$  drehen, sollte der Befehl für die Drehung bis  $\pm 180^\circ$  unter Gelenk-Interpolation erfolgen. Verwenden Sie in dem Programm Befehle mit Linear-Interpolation oder Kreis-Interpolation, kommt es zu Alarmmeldungen.  
Beim Einsatz des RV-E5NJM haben Sie diese Einschränkung nicht.
- Verwenden Sie diesen Befehl während einer kontinuierlichen Bewegung ohne Beschleunigung oder Verzögerung, können keine kontinuierlichen Bewegungen mit Beschleunigung oder Verzögerung ausgeführt werden. (Siehe auch Parameter CNT des SP-Befehls.)
- Wenn Sie die R-Achse bei ausgeschalteter Spannungsversorgung der Steuereinheit um mehr als  $\pm 7$  Umdrehungen bewegen, kann es bei Wiedereinschalten der Spannungsversorgung nachfolgend zu Positionsabweichungen der R-Achse kommen. In diesem Fall müssen Sie eine Nullpunktfahrt der R-Achse ausführen. Informationen zur Vorgehensweise entnehmen Sie bitte Abschnitt 1.3.1 „Grundposition einstellen (Nullpunkt)“ in diesem Handbuch und Abschnitt 3.1 „Einstellen der Grundposition (Nullpunkt)“ im technischen Handbuch.

### Programmbeispiel (MOVEMASTER-Befehle)

10	MO	1	Position 1 anfahren (R-Achsenkoordinate ist $-150^\circ$ )
20	MO	2	Position 2 anfahren (R-Achsenkoordinate ist $+170^\circ$ )
30	JRC	-1	Subtraktion von $360^\circ$ von der aktuellen Position der R-Achse (R-Achsenkoordinate ist $-190^\circ$ )
40	MO	1	Position 1 anfahren

### 5.2.33 LG (If Larger)

#### Funktion: Datenwertvergleich: >

Bewirkt einen Programmsprung, wenn der Wert des internen Registers größer als der festgelegte Vergleichswert ist.

Bewirkt einen Programmsprung, wenn der Inhalt des Zeichenkettenregisters größer als die Anzahl der Zeichen einer festgelegten Zeichenkette ist.

#### Eingabeformat

```
LG    <Vergleichswert/Zeichenkettennummer> ,
      <Zeilennummer des Sprungziels>
```

<b>&lt;Vergleichswert&gt;</b>	Legt den Wert fest, der mit dem Wert des internen Registers verglichen werden soll. $-32\,768 \leq \text{Vergleichswert (dez.)} \leq 32\,767$ $\&8000 \leq \text{Vergleichswert (hex.)} \leq \&7FFF$ $@ \leq \text{Zählernummer} \leq @99$
<b>&lt;Zeichenkettennummer&gt;</b>	Legt die Zeichenkette fest. Das führende Zeichen ist „\$“. $\$1 \leq \text{Zeichenkettennummer} \leq \$99$
<b>&lt;Zeilennummer des Sprungziels&gt;</b>	Legt die Zeilennummer fest, an die das Programm springen soll, wenn der Wert des internen Registers größer als der Vergleichswert ist. $1 \leq \text{Zeilennummer} \leq 9999$

#### Erläuterung

**<Bei Angabe eines Vergleichswertes>**

- Dieser Befehl bewirkt einen Programmsprung, wenn die Sprungbedingung über externe Eingabedaten vom Eingangsport oder durch den internen Zählerwert gegeben ist.
- Wenn der Wert des internen Registers größer als der Vergleichswert ist (d. h. wenn die Bedingung erfüllt ist), springt das Programm zur festgelegten Zeilennummer. Andernfalls (d. h. wenn die Bedingung nicht erfüllt ist) wird der Programmablauf ohne Sprung fortgesetzt.
- Ein Wert kann entweder durch Ausführung des Eingabebefehls für externe Eingabedaten (siehe ID-Befehl) oder durch Ausführung des Zählervergleichsbefehls (siehe CP-Befehl) in das interne Register geladen werden. Um einen bedingten Sprung zu ermöglichen, ist es erforderlich, daß einer der beiden Befehle vor dem LG-Befehl ausgeführt wird.
- Der Vergleichswert kann entweder dezimal oder hexadezimal definiert werden. Bei einem hexadezimalen Wert muß vor der Zahl ein „&“ stehen.
- Es tritt bei der Sprungaufführung eine Fehlermeldung auf, wenn die festgelegte Programmzeilennummer nicht existiert.

**<Bei Angabe einer Zeichenkettennummer>**

- Dieser Befehl bewirkt einen Programmsprung, wenn die Sprungbedingung über externe Eingabedaten oder die Anzahl der Zeichen einer festgelegten Zeichenkette gegeben ist.

- Wenn die Anzahl der Zeichen im Zeichenkettenregister größer als die der festgelegten Zeichenkette ist (d. h. wenn die Bedingung erfüllt ist), springt das Programm zur festgelegten Zeilennummer. Ist die Anzahl kleiner (d. h. wenn die Bedingung nicht erfüllt ist) wird der Programmablauf ohne Sprung fortgesetzt. Ist die angegebene Zeilennummer nicht vorhanden, erfolgt eine Fehlermeldung.
- Bei Ausführung des INP-Befehls werden externe Daten in das Zeichenkettenregister geladen. Durch Ausführung des CP-Befehls werden die Daten der Zeichenkettennummer gesetzt. Vor Ausführung des LG-Befehls muß einer dieser Befehle ausgeführt worden sein.

#### Programmbeispiel (MOVEMASTER-Befehle)

100	ID		Daten vom externen Eingabeport holen
110	LG	100,130	Sprung zur Zeile 130, wenn die Eingabedaten größer als 100 sind
120	ED		Beendet das Programm, wenn die obere Sprungbedingung nicht erfüllt wird
130	MO	7	Position 7 anfahren
140	OPN	1,1	RS232C-Schnittstelle öffnen
150	INP	1, ,2	Daten vom RS232C-Port in das Zeichenkettenregister lesen
160	LG	\$5,200	Sprung zur Zeile 200, wenn die Anzahl der Zeichen im Zeichenkettenregister größer als die Anzahl der Zeichen der Zeichenkette Nummer 5 ist
200	ED		Programmende



## 5.2.34 LR ♦ (Line Read)

### Funktion: Programmzeile lesen

Liest die Programminhalte der festgelegten Zeilennummer (über die RS232C-Schnittstelle).

### Eingabeformat

LR    [ <Zeilennummer> ]
--------------------------

<Zeilennummer>      Legt die zu lesende Programmzeile fest.  
 $0 \leq \text{Zeilennummer} \leq 9999$   
 (Bei fehlender Nummerangabe wird die  
 aktuelle Programmstopp-Zeile gelesen)

### Erläuterung

- Dieser Befehl gibt den Programminhalt der festgelegten Zeilen über die RS232C-Schnittstelle aus.
- Die Daten werden im folgenden Format (ASCII-Code) ausgegeben:
  - Angabe einer Zeilennummer  $\Rightarrow$  Programminhalt der festgelegten Zeile wird gelesen
  - keine Zeilennummer (oder Nr. 0)  $\Rightarrow$  aktuelle Programmstopp-Zeile wird gelesen
- Das Abschlußzeichen der Ausgabedaten ist ein „Carriage Return“ (CR, hex. 0D). Deshalb ist es beim Einsatz eines PCs erforderlich, daß die serielle Datenkette bis zur Stelle „hex. 0D“ empfangen wird. Der Befehl „LINE INPUT #“ ist der entsprechende BASIC-Befehl zum Einlesen der Daten.
- Es wird der hexadezimale Wert „0D“ ausgegeben, wenn die festgelegte Zeile im Programm nicht existiert.
- Mit diesem Befehl kann während des Roboterbetriebs beim Auftreten eines Fehlers die Programmzeile ausgelesen werden, in der der Fehler aufgetreten ist. Programmieren Sie hierfür den LR-Befehl ohne Zeilennummerangabe.

### Programmbeispiel (BASIC-Befehle)

10	OPEN "COM1 :E83"    AS#1	RS232C-Kommunikationsdatei wird über den PC mittels BASIC-Befehl geöffnet
20	INPUT "Startzeile = ";S	Eingabe der Startzeile des auszulesenden Programms
30	INPUT "Endzeile = ";E	Eingabe der Endzeile des auszulesenden Programms
40	FOR I=S TO E	Beginn einer Programmschleife
50	PRINT #1; "LR" + STR\$ (I)	Überträgt den LR-Befehl und die Zeilennummer zum Steuergerät
60	LINE INPUT #1,A\$	Übertragene Daten in A\$ abspeichern
70	IF A\$="" THEN 90	Sprung zur Zeile 90, wenn keine Daten vorhanden
80	PRINT I ; :PRINT A\$	Daten auf dem Bildschirm des PCs anzeigen
90	NEXT	Ende der Programmschleife und Sprung zur Zeile 40
100	END	Programmende

## RUN

Startzeile = 1

Endzeile = 5

1 NT

2 MO 6

3 MO 4

4 MO 3

5 ED

BASIC-Programm starten

Eingabe der Startzeile (1)

Eingabe der Endzeile (5)

## 5.2.35 MA (Move Approach)

### Funktion: relative Koordinatenaddition

Addiert die Koordinatenwerte zweier Positionen und bewegt die Handspitze zur berechneten Zielposition (Gelenk-Interpolation).

### Eingabeformat

MA <Positionsnummer (a)>, <Positionsnummer (b)> [, [<O/C>]]

<Positionsnummer (a) >	Legt die Position der Bezugsposition fest $1 \leq \text{Positionsnummer (ganzzahlig)} \leq 999$
<Positionsnummer (b) >	Legt die Position fest, deren Koordinatenwerte addiert werden sollen $1 \leq \text{Positionsnummer (ganzzahlig)} \leq 999$
<O/C>	Legt den Handgreiferzustand fest (offen/geschlossen) O: Handgreifer offen C: Handgreifer geschlossen

### Erläuterung

- Dieser Befehl bewegt die Handspitze an eine Position (Zielposition), die sich aus der Addition der Koordinatenwerte der Positionen (a) und (b) ergibt. Die Koordinatenwerte der Positionen (a) und (b) werden durch die Ausführung des MA-Befehls jedoch nicht verändert (siehe auch SF-Befehl).
- Vor einer Roboterbewegung wird zuerst der festgelegte Handgreiferzustand (offen/geschlossen) eingenommen. Es bleibt der Handgreiferzustand der Position (a) erhalten, wenn der Handgreiferzustand mit dem MA-Befehl nicht festgelegt wurde.
- Es tritt vor der Roboterbewegung eine Fehlermeldung auf, wenn die Zielposition außerhalb des zulässigen Roboterarbeitsbereichs liegt.
- Es kommt ebenfalls zur einer Fehlermeldung, wenn die Positionen (a) und/oder (b) noch nicht definiert worden sind.
- Das Ende der Hand (Handspitze) wird durch die aktuell eingestellte Werkzeuglänge bestimmt (siehe TL-Befehl).

**Programmbeispiele (MOVEMASTER-Befehle)****5 Achsen**

10	HE	1	Aktuelle Koordinatenwerte als Position 1 definieren
20	PD	5,0,0,30,0,0	Z-Koordinatenwert der Position 5 auf 30 mm einstellen
30	MA	1,5,O	Koordinatenwerte der Position 1 und 5 werden addiert. Anschließend wird die Zielposition mit offener Hand angefahren.

**6 Achsen**

10	HE	1	Aktuelle Koordinatenwerte als Position 1 definieren
20	PD	5,0,0,30,0,0,0	Z-Koordinatenwert der Position 5 auf 30 mm einstellen
30	MA	1,5,O	Koordinatenwerte der Position 1 und 5 werden addiert. Anschließend wird die Zielposition mit offener Hand angefahren.

Die Koordinatenwerte der Positionen 1 und 5 bleiben unverändert.

## 5.2.36 MC (Move Continuous)

### Funktion: kontinuierliche Bewegung

Bewegt den Roboter kontinuierlich zwischen zwei festgelegten Positionen über vordefinierte Zwischenpositionen (Linear-Interpolation).

### Eingabeformat

MC   <Positionsnummer (a)> <Positionsnummer (b)> [, [<O/C>]]

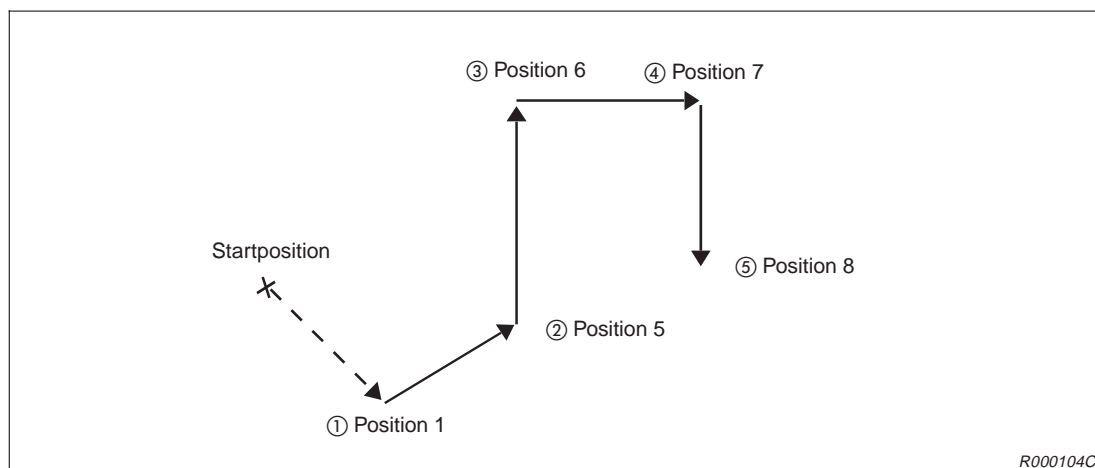
<Positionsnummer (a) >	Legt die Startposition der Roboterbewegung fest
<Positionsnummer (b) >	Legt die Endposition der Roboterbewegung fest $1 \leq \text{Positionsnummer (ganzzahlig)} \leq 999$ $ \text{Positionsnummer (a)} - \text{Positionsnummer (b)}  \leq 99$
<O/C>	Legt den Handgreiferzustand fest (offen/geschlossen) (Bei fehlender Angabe ist der Handstatus der Startposition gültig.) O: Handgreifer offen C: Handgreifer geschlossen

### Erläuterung

- Dieser Befehl bewegt den Roboter entlang einer Reihe von Zwischenpositionen von der Position (a) zur Position (b). Die Roboterbewegung erfolgt mit einer konstanten Geschwindigkeit (Linear-Interpolation).
- Abhängig davon, ob die Positionsnummer von (a) größer oder kleiner als die Positionsnummer von (b) ist, wird die Roboterbewegung entlang der Zwischenpositionen in auf- oder absteigender Reihenfolge erfolgen. Nach der letzten Zwischenposition wird der Roboter zum Erreichen der Endposition abgebremst.
- Vor einer Roboterbewegung wird zuerst der festgelegte Handgreiferzustand (offen/geschlossen) eingenommen.
- Weil der Roboter während der Verfahrbewegung nicht beschleunigt oder abgebremst wird, kommt es zu einer Fehlermeldung, wenn entlang des Verfahrwegs eine größere Richtungsänderung auftritt, die nur durch eine schnelle Bewegung eines Roboter gelenks erreicht werden könnte.
- Die Verfahrgeschwindigkeit während der Linear-Interpolation wird über die Parameter der Befehle SP oder SD bestimmt. Die Handspitze bewegt sich mit einer konstanten Geschwindigkeit.
- Es tritt vor der Ausführung des Befehls eine Fehlermeldung auf,
  - wenn die festgelegten Positionen (a) und (b) vorher nicht definiert worden sind,
  - oder wenn der Betrag der Differenz zwischen den Positionsnummern (a) und (b) größer als 99 ist.
- Es tritt während der Verfahrbewegung eine Fehlermeldung auf, wenn der Verfahrweg ganz oder teilweise außerhalb des zulässigen Roboterarbeitsbereichs liegt.

**Programmbeispiel (MOVEMASTER-Befehle)**

10	SP	10	Geschwindigkeit auf den Wert 10 einstellen
20	MO	1	Position 1 mittels Gelenk-Interpolation anfahren
30	MC	5,8	kontinuierliche Bewegung von der Position 5 zur Position 8 (Linear-Interpolation)



**Abb. 5-13:** Beispiel zur kontinuierlichen Roboterbewegung über Zwischenpositionen (MC-Befehl)

## 5.2.37 MJ (Move Joint)

### Funktion: relative Mehrfach-Gelenkbewegung

Dreht jedes Gelenk um den festgelegten Drehwinkel (Betrag und Richtung), bezogen auf die aktuelle Position (Gelenk-Interpolation).

#### 5 Achsen

#### Eingabeformat

```
MJ  [<Drehwinkel: Mittelteilgelenk>],
    [<Drehwinkel: Schultergelenk>],
    [<Drehwinkel: Ellbogengelenk>],
    [<Neigungswinkel für Handgelenk>],
    [<Drehwinkel für Handgelenk>]
```

<Drehwinkel>            Legt den Betrag und die Richtung des Drehwinkels für jedes Gelenk fest.

#### 6 Achsen

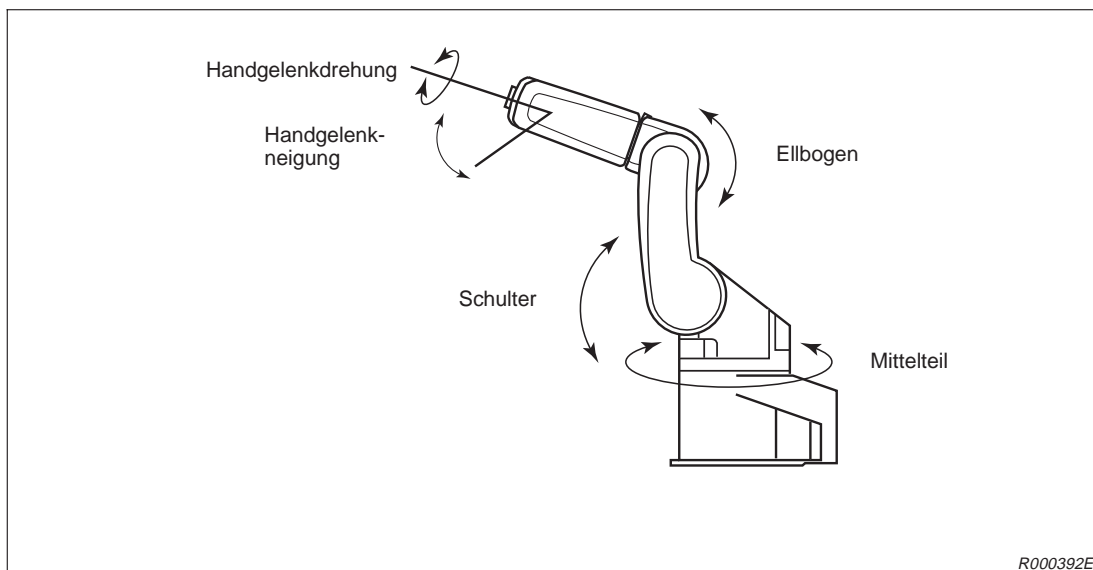
#### Eingabeformat

```
MJ  [<Drehwinkel: Mittelteilgelenk>],
    [<Drehwinkel: Schultergelenk>],
    [<Drehwinkel: Ellbogengelenk>],
    [<Drehwinkel: Unterarmgelenk>],
    [<Neigungswinkel für Handgelenk>],
    [<Drehwinkel für Handgelenk>]
```

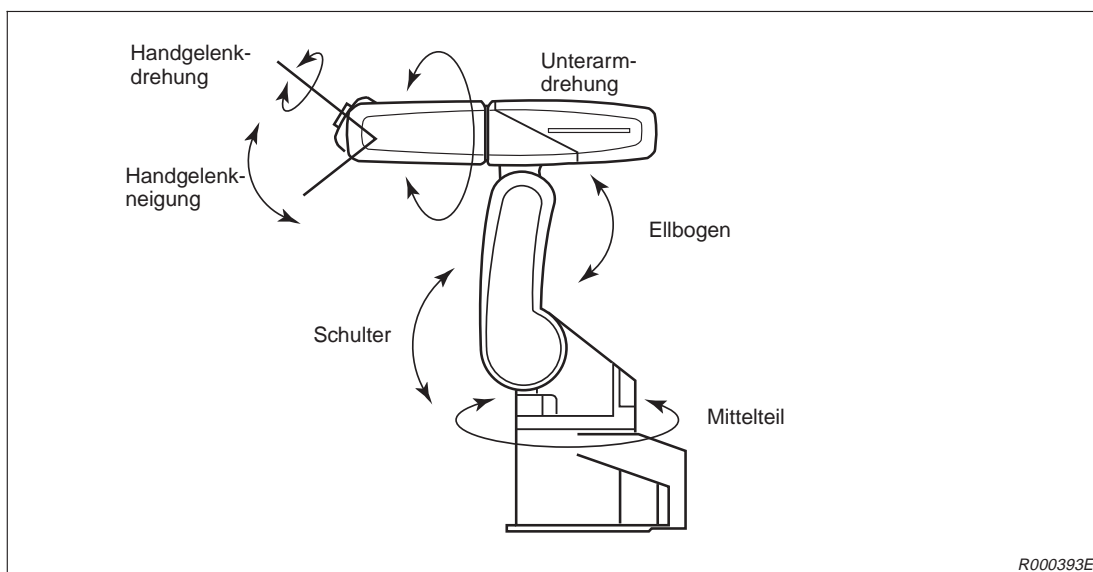
<Drehwinkel>            Legt den Betrag und die Richtung des Drehwinkels für jedes Gelenk fest.

### Erläuterung

- Der Drehwinkel kann in Schrittweiten von  $0,01^\circ$  angegeben werden. Geben Sie für z. B.  $15,02^\circ$  den Wert 15.02 an.
- Vor und nach der Roboterbewegung bleibt der Handgreiferzustand (offen/geschlossen) unverändert.
- Es tritt vor einer Gelenkbewegung eine Fehlermeldung auf, wenn irgendeiner der angegebenen Drehwinkel den zulässigen Roboterarbeitsbereich überschreitet.
- Die Grundeinstellung für den Drehwinkel ist 0.

**5 Achsen****Abb. 5-14:** Gelenke und Drehwinkel des 5-achsigen Roboters**Programmbeispiel (MOVEMASTER-Befehle)**

10	MJ	90,0,0,0,0	Mittelteilgelenk um +90° drehen
20	MJ	0,-30,0,0,0	Schultergelenk um -30° drehen
30	MJ	60,0,0,-20,0	Mittelteilgelenk um +60° drehen und Handgelenk um -20° kippen

**6 Achsen****Abb. 5-15:** Gelenke und Drehwinkel des 6-achsigen Roboters**Programmbeispiel (MOVEMASTER-Befehle)**

10	MJ	90,0,0,0,0,0	Mittelteilgelenk um +90° drehen
20	MJ	0,-30,0,0,0,0	Schultergelenk um -30° drehen
30	MJ	60,0,0,-40,0,0	Mittelteilgelenk um +60° und Unterarm um -40° drehen



## 5.2.38 ML (Move Linear)

### Funktion: lineare Bewegung

#### Funktion

Bewegt eine zusätzliche, unabhängige Achse.

#### Eingabeformat

```
ML [<Wegbetrag der ersten Zusatzachse>] [, <Wegbetrag der
    zweiten Zusatzachse>]
```

- |                                     |  |
|-------------------------------------|--|
| <Wegbetrag der ersten Zusatzachse>  | Legt den Wegbetrag der ersten Zusatzachse in bezug auf die Augenblicksposition fest. Fehlt die Angabe ist der Wegbetrag 0. Die Einheit wird mit dem Parameter AXUN festgelegt.   |
| <Wegbetrag der zweiten Zusatzachse> | Legt den Wegbetrag der zweiten Zusatzachse in bezug auf die Augenblicksposition fest. Fehlt die Angabe, ist der Wegbetrag 0. Die Einheit wird mit dem Parameter AXUN festgelegt. |

#### Erläuterung

- Die Zusatzachse bewegt sich von der Augenblicksposition um den festgelegten Wegbetrag.
- Fehlt die Angabe, ist der Wegbetrag 0.
- Die Einheit des Wegbetrages wird mit dem Parameter AXUN festgelegt.
- Wird der Bewegungsbereich des Roboterarms überschritten, erfolgt eine Fehlermeldung.
- Ungültige Angaben für den Wegbetrag werden ignoriert.
- Enthält der nächste Bewegungsbefehl, der auf den ML-Befehl folgt, keine Angaben für die Zusatzachse, wird der Befehl ausgeführt, auch wenn sich dabei die Zusatzachse mitbewegt.

#### Programmbeispiel (MOVEMASTER-Befehle)

```
10 MO 1      Fährt Position 1 an
20 ML 100    Bewegt die Zusatzachse 100 [mm oder deg.] in die positive Richtung
30 MO 2      Fährt Position 2 an
40 ML        Die Zusatzachse wird nicht bewegt
```

Siehe auch Parameter AXUN und Befehl WRM im Spezifikations-Handbuch.

## 5.2.39 MO (Move)

### Funktion: Position anfahren

Bewegt die Handspitze zur festgelegten Position.

### Eingabeformat

```
MO  <Positionsnummer> [ , [ <O/C> ] ]
```

<Positionsnummer>

Legt die Zielposition fest.

$1 \leq \text{Positionsnummer (ganzzahlig)} \leq 999$

<O/C>

Legt den Handgreiferzustand fest (offen/geschlossen).

(Bei fehlender Angabe ist der Handstatus der Startposition gültig.)

O: Handgreifer offen

C: Handgreifer geschlossen

### Erläuterung

- Dieser Befehl bewegt die Handspitze mittels Gelenk-Interpolation zu den Koordinatenwerten der festgelegten Position. Das Ende der Hand (Handspitze) wird durch die aktuell eingestellte Werkzeuglänge bestimmt (siehe TL-Befehl).
- Vor einer Roboterbewegung wird zuerst der festgelegte Handgreiferzustand (offen/geschlossen) eingenommen. Es wird der Handgreiferzustand der Zielposition eingenommen, wenn mit dem MO-Befehl keine Festlegung angegeben wurde.
- Eine Fehlermeldung tritt auf, wenn die Zielposition vorher nicht definiert wurde oder die Verfahrbewegung den zulässigen Roboterarbeitsbereich überschreitet.

### Programmbeispiel (MOVEMASTER-Befehle)

10	SP	10	Geschwindigkeit auf den Wert 10 einstellen
20	MO	20,C	Position 20 mit geschlossener Hand anfahren
30	MO	30,O	Position 30 mit offener Hand anfahren

## 5.2.40 MP (Move Position)

### Funktion: Koordinatenposition anfahren

Bewegt die Handspitze zu einer über die Koordinaten (XYZ-Koordinaten und Winkel) festgelegten Position.

#### 5 Achsen

#### Eingabeformat

```
MP  [<X-Koordinatenwert>], [<Y-Koordinatenwert>],
    [<Z-Koordinatenwert>], [<Drehwinkel A>], [<Drehwinkel B>],
    [, [<R/L>] [, [<A/B>]]]
```

<XYZ-Koordinaten>	Legen die Position im XYZ-Koordinatensystem des Roboters fest. (Einheit: mm), (0 für Standardwert)
<Drehwinkel A, B>	Legen die Drehwinkel der Roll- und Neigungsgelenke im XYZ-Koordinatensystem des Roboters fest. (Einheit: Grad), (0 für Standardwert)
<R/L>	Bestimmt die Roboterstellung (rechts oder links). R : rechts (Standardwert) L : links
<A/B>	Bestimmt die Roboterstellung (oben oder unten). A : oben (Standardwert) B : unten

#### 6 Achsen

#### Eingabeformat

```
MP  [<X-Koordinatenwert>], [<Y-Koordinatenwert>],
    [<Z-Koordinatenwert>], [<Drehwinkel A>], [<Drehwinkel B>],
    [<Drehwinkel C>], [, [<R/L>] [, [<A/B>] [, [<N/F>]]]]]
```

<XYZ-Koordinaten>	Legen die Position im XYZ-Koordinatensystem des Roboters fest (Einheit: mm), (0 für Standardwert)
<Drehwinkel A, B, C>	Legen die Drehwinkel der Roll- und Neigungsgelenke im XYZ-Koordinatensystem des Roboters fest (Einheit: Grad), (0 für Standardwert)
<R/L>	Bestimmt die Roboterstellung (rechts oder links) R : rechts (Standardwert) L : links
<A/B>	Bestimmt die Roboterstellung (oben oder unten) A : oben (Standardwert) B : unten
<N/F>	Bestimmt die Roboterstellung (nicht kippen oder kippen) N : nicht kippen (Standardwert) F : kippen

### Erläuterung

- Die Koordinatenwerte können in Einheiten von 0,01 mm oder 0,01° angegeben werden.
- Es werden die Standardwerte für die Roboterstellung angenommen, falls hierfür keine Festlegung erfolgte (RV-E5NJM: R, A und RV-E4NM, RV-E4NC-SB/SA: R, A, N).

- Bei der Ausführung des MP-Befehls tritt eine Fehlermeldung auf, wenn durch die vorgenommenen Einstellungen der zulässige Arbeitsbereich des Roboters überschritten wird.
- Der Handgreiferzustand (offen/geschlossen) bleibt vor und nach der Roboterbewegung unverändert erhalten.
- Das Ende der Hand (Handspitze) wird durch die aktuell eingestellte Werkzeuglänge bestimmt (siehe TL-Befehl).

**5 Achsen****Programmbeispiel (MOVEMASTER-Befehle)**

10	MP	400,0,300,0,0	Festgelegte Koordinatenposition anfahren X-Achsenwert: 400, Z-Achsenwert: 300
20	MP	200,200,500,0,0,R	Festgelegte Koordinatenposition anfahren X-Achsenwert: 200, Y-Achsenwert: 200, Z-Achsenwert: 500, Roboterstellung: R (rechts)

**6 Achsen****Programmbeispiel (MOVEMASTER-Befehle)**

10	MP	400,0,300,0,0,0	Festgelegte Koordinatenposition anfahren X-Achsenwert: 400, Z-Achsenwert: 300
20	MP	200,200,500,0,0,0,R,A,N	Festgelegte Koordinatenposition anfahren X-Achsenwert: 200, Y-Achsenwert: 200, Z-Achsenwert: 500 Roboterstellung: R (rechts), A (oben), N (nicht kippen)

## 5.2.41 MPB (Move Playback)

### Funktion: Parameter für Teaching-Playback festlegen

Führt zur festgelegten Position mit folgenden festgelegten Parametern: Interpolation, Geschwindigkeit, Timer, Ein- und Ausgangssignal.

#### 5 Achsen

#### Eingabeformat

```
MPB [<Geschwindigkeit>], [<Timer>], [<Ausgang EIN>],
    [<Ausgang AUS>], [<Eingang EIN>], [<Eingang AUS>],
    [<Interpolation>], [<X-Koordinatenwert>],
    [<Y-Koordinatenwert>], [<Z-Koordinatenwert>],
    [<Drehwinkel A>], [<Drehwinkel B>],
    [, [<R/L>] [, [<A/B>]]] [, [<O/C>]]
```

<Geschwindigkeit>	Legt die Verfahrensgeschwindigkeit fest. $0 \leq \text{Geschwindigkeit} \leq 32\,767$ (Gelenk-Interpolation: %, Linear-Interpolation: mm/s)
<Timer>	Setzt den Timer für die Zielposition (nach der Bewegung). $0 \leq \text{Timer} \leq 255$ 1 : setzen      0 : nicht setzen
<Ausgang EIN>	Setzt das Ausgangssignal, das eingeschaltet wird. $0 \leq \text{Ausgang EIN (hexadezimal)} \leq \&FFFF$ 1 : setzen      0 : nicht setzen
<Ausgang AUS>	Setzt das Ausgangssignal, das ausgeschaltet wird. $0 \leq \text{Ausgang AUS (hexadezimal)} \leq \&FFFF$ 1 : setzen      0 : nicht setzen
<Eingang EIN>	Setzt das Eingangssignal, das eingeschaltet wird. $0 \leq \text{Eingang EIN (hexadezimal)} \leq \&FFFF$ 1 : setzen      0 : nicht setzen
<Eingang AUS>	Setzt das Eingangssignal, das ausgeschaltet wird. $0 \leq \text{Eingang AUS (hexadezimal)} \leq \&FFFF$ 1 : setzen      0 : nicht setzen
<Interpolation>	Legt die Interpolationsmethode für die Verfahrbewegung fest. 0 : Gelenk-Interpolation (Standardwert) 1 : Linear-Interpolation 2 : Kreis-Interpolation
<XYZ-Koordinaten>	Legen die Zielposition im XYZ-Koordinatensystem des Roboters fest. (Einheit: mm), (0 für Standardwert)
<Drehwinkel A, B>	Legen die Drehwinkel der Roll- und Neigungsgelenke im XYZ-Koordinatensystem des Roboters fest. (Einheit: Grad), (0 für Standardwert)
<R/L>	Bestimmt die Roboterstellung (rechts oder links). R : rechts (Standardwert) L : links
<A/B>	Bestimmt die Roboterstellung (oben oder unten). A : oben (Standardwert) B : unten

<O/C>                    Legt den Handgreiferzustand fest (offen/geschlossen).  
 (Bei fehlender Angabe ist der Handstatus der Startposition gültig.)  
 O : Handgreifer offen (Hand 1, Standardwert)  
 C : Handgreifer geschlossen (Hand 1)

**6 Achsen****Eingabeformat**

```
MPB [<Geschwindigkeit>], [<Timer>], [<Ausgang EIN>],
    [<Ausgang AUS>], [<Eingang EIN>], [<Eingang AUS>],
    [<Interpolation>], [<X-Koordinatenwert>],
    [<Y-Koordinatenwert>], [<Z-Koordinatenwert>],
    [<Drehwinkel A>], [<Drehwinkel B>], [<Drehwinkel C>],
    [, [<R/L>] [, [<A/B>] [, [<N/F>]]]] [, [<O/C>]]
```

<Geschwindigkeit>      Legt die Verfahrgeschwindigkeit fest.  
 $0 \leq \text{Geschwindigkeit} \leq 32\,767$   
 (Gelenk-Interpolation: %, Linear-Interpolation: mm/s)

<Timer>                Setzt den Timer für die Zielposition (nach der Bewegung).  
 $0 \leq \text{Timer} \leq 255$   
 1 : setzen              0 : nicht setzen

<Ausgang EIN>        Setzt das Ausgangssignal, das eingeschaltet wird.  
 $0 \leq \text{Ausgang EIN (hexadezimal)} \leq \&FFFF$   
 1 : setzen              0 : nicht setzen

<Ausgang AUS>        Setzt das Ausgangssignal, das ausgeschaltet wird.  
 $0 \leq \text{Ausgang AUS (hexadezimal)} \leq \&FFFF$   
 1 : setzen              0 : nicht setzen

<Eingang EIN>        Setzt das Eingangssignal, das eingeschaltet wird.  
 $0 \leq \text{Eingang EIN (hexadezimal)} \leq \&FFFF$   
 1 : setzen              0 : nicht setzen

<Eingang AUS>        Setzt das Eingangssignal, das ausgeschaltet wird.  
 $0 \leq \text{Eingang AUS (hexadezimal)} \leq \&FFFF$   
 1 : setzen              0 : nicht setzen

<Interpolation>        Legt die Interpolationsmethode für die Verfahrbewegung fest.  
 0 : Gelenk-Interpolation (Standardwert)  
 1 : Linear-Interpolation  
 2 : Kreis-Interpolation

<XYZ-Koordinaten>    Legen die Zielposition im XYZ-Koordinatensystem  
 des Roboters fest. (Einheit: mm), (0 für Standardwert)

<Drehwinkel A, B, C>    Legen die Drehwinkel der Roll- und Neigungsgelenke im  
 XYZ-Koordinatensystem des Roboters fest.  
 (Einheit: Grad), (0 für Standardwert)

<R/L>                  Bestimmt die Roboterstellung (rechts oder links.)  
 R : rechts (Standardwert)  
 L : links

<A/B>                  Bestimmt die Roboterstellung (oben oder unten.)  
 A : oben (Standardwert)  
 B : unten

<N/F>                  Bestimmt die Roboterstellung (nicht kippen oder kippen).  
 N : nicht kippen (Standardwert)  
 F : kippen

&lt;O/C&gt;

Legt den Handgreiferzustand fest (offen/geschlossen).  
(Bei fehlender Angabe ist der Handstatus der Startposition gültig.)  
O : Handgreifer offen (Hand 1, Standardwert)  
C : Handgreifer geschlossen (Hand 1)

**Erläuterung**

- Die Koordinatenwerte können in Einheiten von 0,01 mm oder 0,01° angegeben werden. Geben Sie für z. B. 20,01 mm den Wert 20.01 an.
- Es tritt eine Fehlermeldung auf, wenn die festgelegten Koordinaten den zulässigen Roboterarbeitsbereich überschreiten.
- Der Standardkoordinatenwert ist 0.
- Es werden die Standardwerte für die Roboterstellung angenommen, falls hierfür keine Festlegung erfolgte (RV-E5NJM: R, A und RV-E4NM, RV-E4NC-SB/SA: R, A, N).
- Es wird in jedem Programmschritt automatisch ein MPB-Befehl erzeugt, wenn eine der Bedingungen (Geschwindigkeit, Timer, Ein-/Ausgangssignale) über die Teaching-Playback-Methode festgelegt wurde.
- Die Werte für die Ein-/Ausgangssignale müssen bei Angabe eines hexadezimalen Wertes mit dem Zeichen „&“ beginnen.

## 5.2.42 MPC (Move Playback Continuous)

### Funktion: Bewegung für Teaching-Playback festlegen

Führt mit definierter Interpolationsmethode zur festgelegten Position.

#### 5 Achsen

#### Eingabeformat

```
MPC [<Interpolation>], [<X-Koordinatenwert>],
    [<Y-Koordinatenwert>], [<Z-Koordinatenwert>],
    [<Drehwinkel A>], [<Drehwinkel B>],
    [, [<R/L>] [, [<A/B>]]] [, [<O/C>]]
```

<Interpolation>	Legt die Interpolationsmethode für die Verfahrbewegung fest. 0 : Gelenk-Interpolation (Standardwert) 1 : Linear-Interpolation 2 : Kreis-Interpolation
<XYZ-Koordinaten>	Legen die Zielposition im XYZ-Koordinatensystem des Roboters fest. (Einheit: mm), (0 für Standardwert)
<Drehwinkel A, B>	Legen die Drehwinkel der Roll- und Neigungsgelenke im XYZ-Koordinatensystem des Roboters fest. (Einheit: Grad), (0 für Standardwert)
<R/L>	Bestimmt die Roboterstellung (rechts oder links). R : rechts (Standardwert) L : links
<A/B>	Bestimmt die Roboterstellung (oben oder unten). A : oben (Standardwert) B : unten
<O/C>	Legt den Handgreiferzustand fest (offen/geschlossen). (Bei fehlender Angabe ist der Handstatus der Startposition gültig.) O : Handgreifer offen (Hand 1) C : Handgreifer geschlossen (Hand 1)

#### 6 Achsen

#### Eingabeformat

```
MPC [<Interpolation>], [<X-Koordinatenwert>],
    [<Y-Koordinatenwert>], [<Z-Koordinatenwert>],
    [<Drehwinkel A>], [<Drehwinkel B>], [<Drehwinkel C>],
    [, [<R/L>] [, [<A/B>] [, [<N/F>]]]] [, [<O/C>]]
```

<Interpolation>	Legt die Interpolationsmethode für die Verfahrbewegung fest. 0 : Gelenk-Interpolation (Standardwert) 1 : Linear-Interpolation 2 : Kreis-Interpolation
<XYZ-Koordinaten>	Legen die Zielposition im XYZ-Koordinatensystem des Roboters fest. (Einheit: mm), (0 für Standardwert)
<Drehwinkel A, B, C>	Legen die Drehwinkel der Roll- und Neigungsgelenke im XYZ-Koordinatensystem des Roboters fest. (Einheit: Grad), (0 für Standardwert)



<R/L>	Bestimmt die Roboterstellung (rechts oder links). R : rechts (Standardwert) L : links
<A/B>	Bestimmt die Roboterstellung (oben oder unten). A : oben (Standardwert) B : unten
<N/F>	Bestimmt die Roboterstellung (nicht kippen oder kippen). N : nicht kippen (Standardwert) F : kippen
<O/C>	Legt den Handgreiferzustand fest (offen/geschlossen). (Bei fehlender Angabe ist der Handstatus der Startposition gültig.) O : Handgreifer offen (Hand 1) C : Handgreifer geschlossen (Hand 1)

### Erläuterung

- Die Koordinatenwerte können in Einheiten von 0,01 mm oder 0,01° angegeben werden. Geben Sie für z. B. 20,01 mm den Wert 20.01 an.
- Es tritt eine Fehlermeldung auf, wenn die festgelegten Koordinaten den zulässigen Roboterarbeitsbereich überschreiten.
- Der Standardkoordinatenwert ist 0.
- Verwenden Sie den MPB-Befehl, falls die Einstellung der Geschwindigkeit, des Timers oder des Ein-/Ausgangssignals erforderlich ist.
- Es wird in jedem Programmschritt automatisch ein MPC-Befehl erzeugt, wenn **keine** der Bedingungen (Geschwindigkeit, Timer, Ein-/Ausgangssignale) über die Teaching-Playback-Methode festgelegt wurde.

## 5.2.43 MR (Move R)

### Funktion: Kreis-Interpolation

Bewegt die Handspitze kreisbogenförmig (Kreis-Interpolation).

### Eingabeformat

```
MR   <Positionsnummer (a)>, <Positionsnummer (b)>,
      <Positionsnummer (c)>, [, [<O/C>]]
```

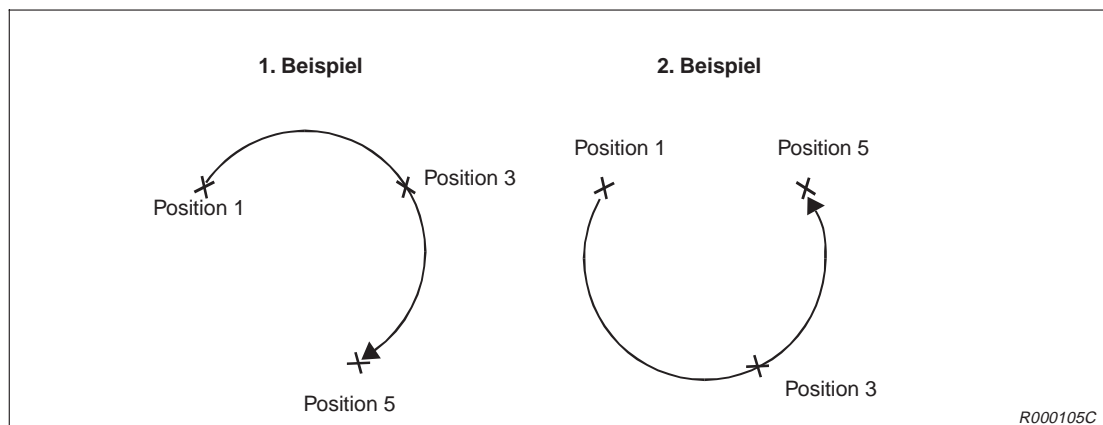
<Positionsnummern>    Legen die Positionen auf dem Kreisbogen fest.  
 $0 \leq \text{Positionsnummer (ganzzahlig)} \leq 999$

<O/C>    Legt den Handgreiferzustand fest (offen/geschlossen).  
 (Bei fehlender Angabe ist der Handstatus der Startposition gültig.)  
 O : Handgreifer offen  
 C : Handgreifer geschlossen

### Erläuterung

- Dieser Befehl verfährt die Handspitze kreisbogenförmig entlang der vordefinierten Zwischenpositionen von Position (a) über (b) nach (c).
- Die Verfahrensgeschwindigkeit bei Kreis-Interpolation wird über die Parameter der Befehle SP oder SD bestimmt. Die Handspitze bewegt sich mit einer konstanten Geschwindigkeit. Weil die Positioniergenauigkeit bei Kreis-Interpolation von der Geschwindigkeit abhängt, sollten Sie eine kleine Verfahrensgeschwindigkeit wählen, wenn eine große Positioniergenauigkeit erforderlich ist.
- Vor und nach der Roboterbewegung bleibt der Handgreiferzustand (offen/geschlossen) unverändert.
- Wenn die Startposition (a) mit der aktuellen Position nicht übereinstimmt, fährt der Roboter zuerst mittels Linear-Interpolation zur Startposition.
- Die Kreisbogenbewegung wird fortgesetzt, wenn die Kreis-Interpolation durch ein Stoppsignal unterbrochen und über ein Startsignal wieder neugestartet wurde. Wenn sich die Handspitze im obigen Fall im JOG-Betrieb nicht an der Stopposition befindet, wird die Stopposition über Gelenk-Interpolation angefahren und danach der verbleibende Kreisbogen mittels Kreis-Interpolation.
- Es tritt eine Fehlermeldung auf, wenn die festgelegten Positionen vorher nicht definiert wurden oder der zulässige Roboterarbeitsbereich überschritten wird. Der Roboter verfährt mit Linear-Interpolation, wenn die drei Positionen (a), (b) und (c) auf einer geraden Linie liegen oder zwei der drei Positionen gleich sind.
- Es tritt eine Fehlermeldung auf, wenn am Beginn der Kreis-Interpolation ein Gelenk um einen sehr großen Winkelbetrag bewegt werden muß. Wählen Sie für diesen Fall eine kleine Anfangsgeschwindigkeit oder setzen Sie einen Timer ein.

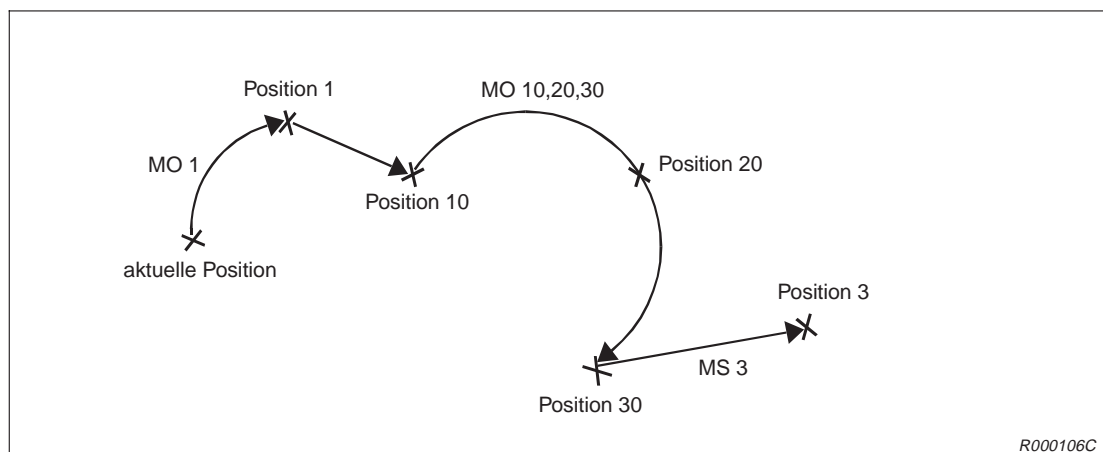
- Die Verfahrrichtung und die Lage des Kreisbogenmittelpunktes ist von der Reihenfolge der festgelegten Positionen abhängig.



**Abb. 5-16:** Beispiele zur Richtungsbestimmung bei kreisbogenförmiger Bewegung (MR-Befehl)

#### Programmbeispiel (MOVEMASTER-Befehle)

10	SP	8	Geschwindigkeit auf den Wert 8 einstellen
20	MO	1	Position 1 anfahren
30	MR	10,20,30	Position 10 mittels Linear-Interpolation anfahren und danach den über die Positionen 10, 20 und 30 definierten Kreisbogen mittels Kreis-Interpolation abfahren
40	MS	3	Position 3 mittels Linear-Interpolation anfahren
50	ED		Programmende



**Abb. 5-17:** Beispiele zur kreisbogenförmigen Bewegung entlang vordefinierter Positionen (MR-Befehl)

## 5.2.44 MRA (Move R A)

### Funktion: Kreis-Interpolation

Bewegt die Handspitze mittels Kreis-Interpolation zur festgelegten Position.

### Eingabeformat

```
MRA <Positionsnummer> [ , [ <O/C> ] ]
```

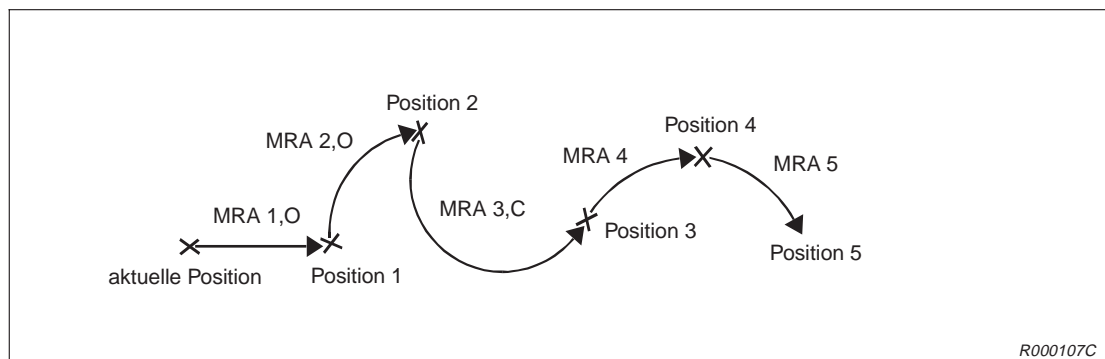
<Positionsnummern>	Legt die Zielposition fest. $1 \leq \text{Positionsnummer (ganzzahlig)} \leq 999$
<O/C>	Legt den Handgreiferzustand fest (offen/geschlossen). (Bei fehlender Angabe ist der Handstatus der Startposition gültig.) O : Handgreifer offen C : Handgreifer geschlossen

### Erläuterung

- Dieser Befehl verfährt die Handspitze entlang eines Kreisbogens, der durch die Positionen definiert wird, die vor und nach dem MRA-Befehl programmiert wurden. Das Ende der Hand (Handspitze) wird durch die aktuell festgelegte Werkzeuglänge bestimmt (siehe TL-Befehl).
- Vor einer Roboterbewegung wird zuerst der festgelegte Handgreiferzustand (offen/geschlossen) eingenommen.
- Es tritt eine Fehlermeldung auf, wenn die festgelegte Position vorher nicht definiert wurde.
- Der MRA-Befehl gleicht in der Funktionsausführung dem MC-Befehl, wenn er nicht dreimal hintereinander programmiert wird. Die folgenden Befehle können jedoch trotzdem zwischen zwei MRA-Befehlen ausgeführt werden: SD, SP, TI, OVR, OB, OC, OD, GC und GO.
- Die Kreisbogenbewegung wird fortgesetzt, wenn die Ausführung des MRA-Befehls durch ein Stoppsignal unterbrochen und über ein Startsignal wieder neugestartet wurde. Wenn sich die Handspitze im obigen Fall im JOG-Betrieb nicht an der Stopposition befindet, wird die Stopposition über Gelenk-Interpolation angefahren und danach der verbleibende Kreisbogen mittels Kreis-Interpolation.

**Programmbeispiel (MOVEMASTER-Befehle)**

			Der Kreisbogen wird durch die Positionen 1, 2 und 3 definiert
10	MRA	1,O	Position 1 mittels Linear-Interpolation anfahren (Hand offen)
20	MRA	2,O	Position 2 mittels Kreis-Interpolation anfahren (Hand offen)
30	MRA	3,C	Position 3 mittels Kreis-Interpolation anfahren (Hand geschlossen)
40	TI	3	Timer mit Wartezeit von 0,3 s
50	MRA	4	Position 4 mittels Kreis-Interpolation anfahren
60	MRA	5	Position 5 mittels Kreis-Interpolation anfahren
70	ED		Programmende



**Abb. 5-18:** Beispiele zur kreisbogenförmigen Bewegung entlang vordefinierter Positionen (MRA-Befehl)

## 5.2.45 MS (Move Strait)

### Funktion: geradlinige Bewegung

Bewegt die Handspitze mittels Linear-Interpolation zur festgelegten Position.

### Eingabeformat

MS   <Positionsnummer> [ , [ <O/C> ] ]
--

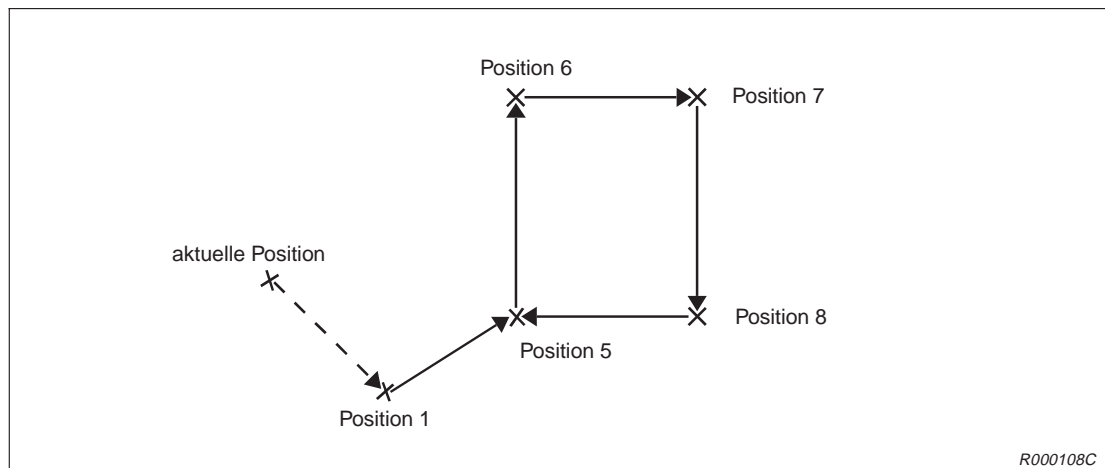
<Positionsnummern>	Legt die Zielposition fest. $0 \leq \text{Positionsnummer (ganzzahlig)} \leq 999$
<O/C>	Legt den Handgreiferzustand fest (offen/geschlossen). (Bei fehlender Angabe ist der Handstatus der Startposition gültig.) O : Handgreifer offen C : Handgreifer geschlossen

### Erläuterung

- Dieser Befehl verfährt die Handspitze entlang einer geraden Linie zur festgelegten Position. Das Ende der Hand (Handspitze) wird durch die aktuell eingestellte Werkzeuglänge bestimmt (siehe TL-Befehl).
- Es tritt vor oder nach der Bewegung eine Fehlermeldung auf, wenn die Zielposition oder der Verfahrweg außerhalb des zulässigen Roboterarbeitsbereichs liegen.
- Vor einer Roboterbewegung wird zuerst der festgelegte Handgreiferzustand (offen/geschlossen) eingenommen.
- Die Verfahrgeschwindigkeit wird über die Parameter der Befehle SP oder SD bestimmt. Die Handspitze bewegt sich mit einer konstanten Geschwindigkeit.
- Verwenden Sie den MC-Befehl, um den Roboter mittels Linear-Interpolation kontinuierlich zwischen verschiedenen Positionen zu verfahren.

**Programmbeispiel (MOVEMASTER-Befehle)**

10	SP	15	Geschwindigkeit auf den Wert 15 einstellen
20	MO	1	Position 1 mittels Gelenk-Interpolation anfahren
30	MS	5	Position 5 mittels Linear-Interpolation anfahren
40	MS	6	Position 6 mittels Linear-Interpolation anfahren
50	MS	7	Position 7 mittels Linear-Interpolation anfahren
60	MS	8	Position 8 mittels Linear-Interpolation anfahren
70	MS	5	Position 5 mittels Linear-Interpolation anfahren
80	ED		Programmende

**Abb. 5-19:** Beispiel zur geradlinigen Roboterbewegung (MS-Befehl)

## 5.2.46 MT (Move Tool)

### Funktion: Werkzeugbewegung mit Gelenk-Interpolation

Bewegt die Handspitze zu einer Position, die um den festgelegten Betrag in Werkzeuglängsrichtung verschoben von der festgelegten Position liegt (Gelenk-Interpolation).

### Eingabeformat

MT <Positionsnummer>, [<Verfahrbetrag>] [, [<O/C>]]

<Positionsnummern>	Legt die Zielposition fest. $1 \leq \text{Positionsnummer (ganzzahlig)} \leq 999$
<Verfahrbetrag>	Legt den Verfahrwegbetrag in Werkzeuglängsrichtung fest. (Abstand zur Zielposition), (Wert 0 für Standardwert). $-3276,80 \leq \text{Verfahrbetrag} \leq 3276,70$
<O/C>	Legt den Handgreiferzustand fest (offen/geschlossen). (Bei fehlender Angabe ist der Handstatus der Startposition gültig.) O : Handgreifer offen C : Handgreifer geschlossen

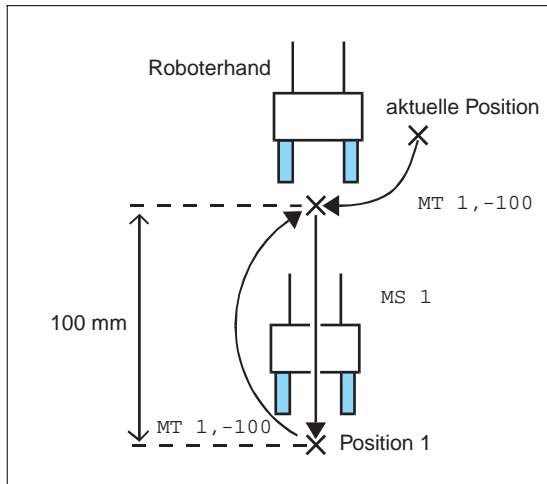
### Erläuterung

- Die Verfahrwegstrecke kann in Einheiten von 0,01 mm angegeben werden.
- Bei Angabe eines positiven Verfahrwegbetrags wird die Handspitze in Werkzeuglängsrichtung nach vorne verschoben. Bei Angabe eines negativen Verfahrwegbetrags wird die Handspitze in Werkzeuglängsrichtung zurückgefahren.
- Vor einer Roboterbewegung wird zuerst der festgelegte Handgreiferzustand (offen/geschlossen) eingenommen.
- Bei der Ausführung des MT-Befehls tritt eine Fehlermeldung auf, wenn die Zielposition oder der Verfahrweg außerhalb des zulässigen Roboterarbeitsbereichs liegt.



**Programmbeispiel (MOVEMASTER-Befehle)**

10	MT	1,-100	Position anfahren, die um 100 mm in Werkzeuglängsrichtung entfernt von Position 1 liegt
20	MS	1	Position 1 anfahren
30	MT	1,-100	Position anfahren, die um 100 mm in Werkzeuglängsrichtung entfernt von Position 1 liegt

**Abb. 5-20:**

*Beispiel zum Verschieben der Handspitze in Werkzeuglängsrichtung (MT-Befehl)*

R000109C

## 5.2.47 MTS (Move Tool Straight)

### Funktion: geradlinige Werkzeugbewegung

Bewegt die Handspitze zur einer Position, die um den festgelegten Betrag in Werkzeuglängsrichtung entfernt von der festgelegten Position liegt (Linear-Interpolation).

### Eingabeformat

MTS <Positionsnummer>, [<Verfahrbetrag>] [, [<O/C>]]

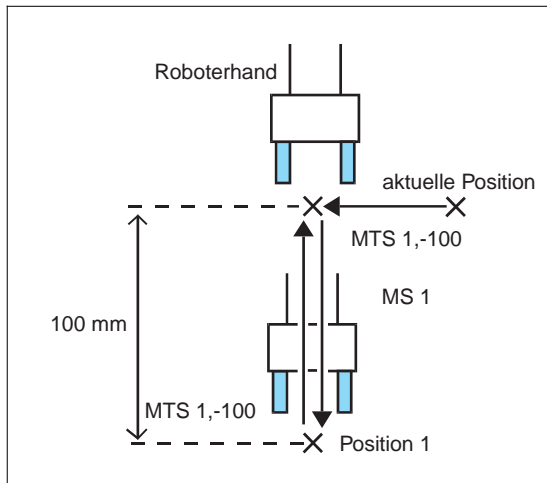
<Positionsnummern>	Legt die Zielposition fest. $1 \leq \text{Positionsnummer (ganzzahlig)} \leq 999$
<Verfahrbetrag>	Legt den Verfahrwegbetrag in Werkzeuglängsrichtung fest (Abstand zur Zielposition), (Wert 0 für Standardwert). $-3276,80 \leq \text{Verfahrbetrag} \leq 3276,70$
<O/C>	Legt den Handgreiferzustand fest (offen/geschlossen).. (Bei fehlender Angabe ist der Handstatus der Startposition gültig.) O : Handgreifer offen C : Handgreifer geschlossen

### Erläuterung

- Die Verfahrwegstrecke kann in Einheiten von 0,01 mm angegeben werden.
- Bei Angabe eines positiven Verfahrwegbetrags wird die Handspitze in Werkzeuglängsrichtung nach vorne verschoben. Bei Angabe eines negativen Verfahrwegbetrags wird die Handspitze in Werkzeuglängsrichtung zurückgefahren.
- Vor einer Roboterbewegung wird zuerst der festgelegte Handgreiferzustand (offen/geschlossen) eingenommen.
- Bei der Ausführung des MTS-Befehls tritt eine Fehlermeldung auf, wenn die Zielposition oder der Verfahrweg außerhalb des zulässigen Roboterarbeitsbereichs liegt.

**Programmbeispiel (MOVEMASTER-Befehle)**

10	MTS	1,-100,O	Position anfahren, die um 100 mm in Werkzeuglängsrichtung entfernt von Position 1 liegt (Hand offen, geradlinige Bewegung)
20	MS	1	Position 1 anfahren
30	MTS	1,-100,C	Position anfahren, die um 100 mm in Werkzeuglängsrichtung entfernt von Position 1 liegt (Hand geschlossen, geradlinige Bewegung)

**Abb. 5-21:**

*Beispiel zum Verschieben der Handspitze in Werkzeuglängsrichtung (MTS-Befehl)*

R000110C

## 5.2.48 MUL (Multiplication)

### Funktion: Multiplikation

Multipliziert einen Wert oder den Inhalt eines festgelegten Zählers mit dem Wert des internen Registers.

### Eingabeformat

```
MUL <Wert | @Zählernummer>
```

<Wert>	Legt den hexadezimalen oder dezimalen Multiplikator fest. $-32\,768 \leq \text{dezimaler Wert} \leq 32\,767$ $\&8000 \leq \text{hexadezimaler Wert} \leq \&7FFF$
<Zählernummer>	Legt den Zähler fest. $1 \leq \text{Zählernummer (ganzzahlig)} \leq 99$

### Erläuterung

- Multipliziert den Wert des internen Registers, oder den Inhalt eines festgelegten Zählers mit einem dezimalen oder hexadezimalen Wert.

### Beispiele ▾

MUL 10	Multipliziert den Wert des internen Registers mit dem dezimalen Wert 10
MUL &FF	Multipliziert den Wert des internen Registers mit dem hexadezimalen Wert &FF
MUL @5	Multipliziert den Wert des internen Registers mit dem Inhalt des Zählers 5

### Programmbeispiel (MOVEMASTER-Befehle)

10	ID	Lädt Eingabedaten in das interne Register
20	MUL 10	Multipliziert die eingelesenen Daten mit dem Wert 10
30	CL 21	Daten aus dem internen Register in den Zähler 21 laden

## 5.2.49 N ♦ (Number)

### Funktion: Programm auswählen

Über diese Funktion kann ein Programm ausgewählt werden.

### Eingabeformat

N     <Programmname>
----------------------

<Programmname>

Legt den Namen des Roboterprogramms fest  
(Zeichenkette mit max. 8 Stellen)  
 $0 \leq \text{Programmname} \leq 8$  (Zeichen)

zulässige Zeichen:

Zahlen (0 – 9)

Buchstaben (A – Z)

Symbole (! @ # \$ % ^ & ( ) \_ | ~ { } - )

unzulässige Zeichen:

\* + , . / : ; = ? [ ¥ ] ‘

spezielle Programmnamen:

Wenn der Programmname ausschließlich aus numerischen Werten besteht, wird dieser wie eine Nummernangabe verarbeitet. Bei der Verwendung von Buchstaben muß der Programmname in Anführungszeichen " " gesetzt werden.

### Erläuterung

- Dieser Befehl wählt das über den Programmnamen festgelegte Programm aus. Es kann ein neues Programm mit dem ausgewählten Programmnamen erstellt werden oder ein bereits vorhandenes Programm ergänzt, geändert oder gestartet werden. Diese Modifizierungsmöglichkeiten beziehen sich so lange auf das aktuelle Programm, bis über den N-Befehl ein Programm mit einem anderen Programmnamen ausgewählt wurde. (Die Programmauswahl bleibt auch nach einem Ausschalten der Spannungsversorgung erhalten.)
- Mit dem QN-Befehl kann der Programmname des aktuell ausgewählten Programms über einen PC gelesen werden (siehe QN-Befehl).
- Werkseitig ist das Programm mit dem Programmnamen „1“ bereits vorausgewählt.
- Die folgenden Namensbezeichnungen werden wie ein und derselbe Name angenommen:  
Beispiel:  
Wird wie ein und derselbe Name angenommen: 1, 01, 001 (nur numerische Werte)  
Wird wie unterschiedliche Namen angenommen: 1, 1 A, A0\_001 (inklusive Buchstaben)
- Die Zeichen 0 – 9 und A – Z kann das Steuergerät über die 7-Segment-Anzeige darstellen.

### Programmbeispiel (BASIC-Befehle)

10	OPEN "COM1 :E83"    AS#1	RS232C-Kommunikationsdatei wird über den PC mittels BASIC-Befehl geöffnet
20	PRINT #1,"N10"	Programm 10 auswählen
30	PRINT #1"10 MO 1"	Neues Programm erstellen (Zeile 10)
40	PRINT #1"20 MS 2"	Neues Programm erstellen (Zeile 20)
50	PRINT #1"30 ED"	Neues Programm erstellen (Zeile 30)
60	END	BASIC-Programmende

## 5.2.50 NE (If Not Equal)

### Funktion: Datenwertvergleich: $\neq$

Bewirkt einen Programmsprung, wenn der Wert des internen Registers nicht mit dem festgelegten Vergleichswert übereinstimmt.

Bewirkt einen Programmsprung, wenn der Wert des Zeichenkettenregisters nicht mit dem einer festgelegten Zeichenkette übereinstimmt.

### Eingabeformat

NE    <Vergleichswert/Zeichenkettennummer>, <Zeilennummer des Sprungziels>

<Vergleichswert>      Legt den Wert fest, der mit dem Wert des internen Registers verglichen werden soll.

$-32\,768 \leq \text{Vergleichswert (dez.)} \leq 32\,767$

$\&8000 \leq \text{Vergleichswert (hex.)} \leq \&7FFF$

<Zeichenkettennummer>      Legt die Zeichenkette fest. Das führende Zeichen ist „\$“.  
 $\$1 \leq \text{Zeichenkettennummer} \leq \$99$

<Zeilennummer des Sprungziels>      Legt die Zeilennummer fest, an die das Programm bei Wertungleichheit springen soll.  
 $1 \leq \text{Zeilennummer} \leq 9999$

### Erläuterung

<Bei Angabe eines Vergleichswertes>

- Dieser Befehl bewirkt einen Programmsprung, wenn die Sprungbedingung über externe Eingabedaten oder durch den internen Zählerwert gegeben ist.
- Wenn der Wert des internen Registers nicht mit dem Vergleichswert übereinstimmt (d. h. wenn die Bedingung erfüllt ist), springt das Programm zur festgelegten Zeilennummer. Andernfalls (d. h. wenn die Bedingung nicht erfüllt ist) wird der Programmablauf ohne Sprung fortgesetzt.
- Ein Wert kann entweder durch Ausführung des Eingabebefehls für externe Eingabedaten (siehe ID-Befehl) oder durch Ausführung des Zählervergleichsbefehls (siehe CP-Befehl) in das interne Register geladen werden. Um einen bedingten Sprung zu ermöglichen, ist es erforderlich, daß einer der beiden Befehle vor dem NE-Befehl ausgeführt wird.
- Der Vergleichswert kann entweder dezimal oder hexadezimal definiert werden. Bei einem hexadezimalen Wert muß vor der Zahl das Zeichen „&“ stehen.
- Bei der Sprungausführung tritt eine Fehlermeldung auf, wenn die festgelegte Programmzeilennummer nicht existiert.

<Bei Angabe einer Zeichenkettennummer>

- Dieser Befehl bewirkt einen Programmsprung, wenn die Sprungbedingung über externe Eingabedaten oder die Anzahl der Zeichen einer festgelegten Zeichenkette gegeben ist.
- Wenn die Anzahl der Zeichen im Zeichenkettenregister nicht mit der Anzahl der Zeichen der festgelegten Zeichenkette übereinstimmt (d. h. wenn die Bedingung erfüllt ist), springt das Programm zur festgelegten Zeilennummer. Andernfalls (d. h. wenn die Bedingung nicht erfüllt ist) wird der Programmablauf ohne Sprung fortgesetzt.

- Durch Ausführung des INP-Befehls werden externen Daten in das Zeichenkettenregister geladen. Die Werte einer Zeichenkettennummer werden durch Ausführung des CP-Befehls geladen. Um einen bedingten Sprung zu ermöglichen, ist es erforderlich, daß einer der beiden Befehle vor dem NE-Befehl ausgeführt wird.
- Es tritt bei der Sprungausführung eine Fehlermeldung auf, wenn die festgelegte Programmzeilennummer nicht existiert.

#### Programmbeispiel (MOVEMASTER-Befehle)

10	ID		Daten vom externen Eingabeport holen
20	NE	80,100	Sprung zur Zeile 100, wenn die Eingabedaten ungleich 80 sind
30	ED		Beendet das Programm, wenn die obere Sprungbedingung nicht erfüllt wird
100	MO	7	Position 7 anfahren
110	OPN	1,1	RS232C-Schnittstelle öffnen
120	INP	1, ,2	Daten von der RS232C-Schnittstelle in das Zeichenkettenregister einlesen
130	NE	\$2,200	Sprung zur Zeile 200, wenn die Daten nicht mit der Zeichenkette Nummer 2 übereinstimmen
200	ED		Programmende

## 5.2.51 NT (Nest)

### Funktion: Nullpunkt anfahren

Nullpunktfahrt (Der Roboter fährt zum benutzerdefinierten Nullpunkt.)

### Eingabeformat

NT
----

### Erläuterung

- Die Bewegungsreihenfolge der einzelnen Gelenke wird bereits im voraus festgelegt. Die Nullpunkteinstellung für das Schulter- und Ellbogengelenk wird als erstes ausgeführt. Danach folgt die Einstellung für die Unterarmdrehung (nur RV-E2), die Handneigung und die Handdrehung.
- Fahren Sie den Roboterarm mit der Teaching Box vorher in eine Sicherheitsstellung, falls der Roboterarm mit umliegenden Gegenständen kollidieren könnte.
- Über den Parameter UNG können Sie die Bewegungsreihenfolge für die Nullpunkteinstellung ändern. Die Koordinatenwerte des benutzerdefinierten Nullpunktes können über den Parameter UOG geändert werden. Der Parameter UNG kann über die Teaching Box eingestellt werden.
- Normalerweise braucht der NT-Befehl für den allgemeinen Betrieb nicht programmiert werden.

### Parameter

Die Bewegungsreihenfolge für die Nullpunkteinstellung kann über den nachfolgenden Parameter geändert werden:

Parameter UNG: Bewegungsreihenfolge  
2,1,1,1,2,2 (Standardwerte)

Die Roboterstellungen für den benutzerdefinierten Nullpunkt können über den nachfolgenden Parameter geändert werden:

Parameter UOG: Roboterstellung am benutzerdefinierten Nullpunkt (Grad)  
RV-E5NJM: -160.00, -45.00, 0.00, 0.00, -120.0, -200.0  
RV-E4NM/E4NC: -160.00, -45.00, 50.00, -160.00, -120.00, -200.00  
(Standardwerte)

### Programmbeispiel (MOVEMASTER-Befehle)

10	NT		Nullpunkt anfahren
20	MO	1	Position 1 anfahren
30	ED		Programmende



## 5.2.52 NW ♦ (New)

### Funktion: Programm- und Positionsspeicher löschen

Löscht das aktuelle Programm und die Positionsdaten.

### Eingabeformat

NW
----

### Erläuterung

- Dieser Befehl löscht alle Positionen und Zähler des aktuellen Programms. Gemeinsame Positionen (901 – 999) und gemeinsame Zähler (91 – 99) werden nicht gelöscht.
- Durch Ausführung des NW-Befehls werden die Einstellungen für die nachfolgend aufgeführten Punkte nicht gelöscht:
  - Nullpunkteinstellung
  - Wert des internen Registers
  - Werkzeuglänge
  - Geschwindigkeitseinstellung
  - Paletteneinstellung
  - Handeinstellung
- Der NW-Befehl kann nur direkt ausgeführt werden. Innerhalb eines Roboterprogramms kann der NW-Befehl nicht ausgeführt werden.

### Programmbeispiel (BASIC-Befehle)

10	OPEN "COM1 :E83" AS#1	RS232C-Kommunikationsdatei wird über den PC mittels BASIC-Befehl geöffnet
30	PRINT #1,"NW"	Befehl „NW“ übertragen
60	END	Programmende

### 5.2.53 NX (Next)

**Funktion: Programmschleife beenden**

Legt das Ende einer Programmschleife fest, die über den RC-Befehl aufgerufen wurde.

**Eingabeformat**

NX
----

**Erläuterung**

- Dieser Befehl legt in Kombination mit dem RC-Befehl den Bereich einer Programmschleife fest.
- Es tritt eine Fehlermeldung auf, wenn der zugehörige RC-Befehl vorher nicht definiert wurde.

**Programmbeispiel**

Siehe RC-Befehl.

## 5.2.54 OB (Output Bit)

### Funktion: Ausgänge ein-/ausschalten

Legt den Ausgangsstatus des festgelegten Bit am externen Ausgabeport fest.

### Eingabeformat

OB [ <+/-> ] <Ausgangs-Bitnummer>

<+/->                      Legt den Bitstatus fest (EIN oder AUS).

+ : Bit EIN

- : Bit AUS

<Ausgangs-Bitnummer>    Legt das Bit am externen Ausgang fest.  
 $0 \leq \text{Bitnummer (dez.)} \leq 32\,767$

### Erläuterung

- Geben Sie „+“ an, um das festgelegte Bit einzuschalten und „-“, um das festgelegte Bit auszuschalten.
- Die Einstellungen des OB-Befehls beziehen sich nur auf das jeweils festgelegte Bit. Alle anderen Bitzustände bleiben unbeeinflusst.
- Der Ausgangsstatus des festgelegten Bits bleibt so lange erhalten, bis eine Neueinstellung über die Befehle OB oder OD erfolgt.
- Es tritt eine Fehlermeldung auf, wenn eines der speziellen Bits des Parameters OT1 bis OT3 für die externe Ausgabe festgelegt wurde.
- Für die pneumatisch angetriebene Hand kann der Handgreiferzustand (offen/geschlossen) der Hand 1, 2 und 3 über den OB-Befehl festgelegt werden (normalerweise wird hierfür der GC- oder GO-Befehl eingesetzt). Für die motorisch angetriebene Hand können diese Einstellungen nicht vorgenommen werden.

Hand	Handgreiferstatus	GR1	GR2	GR3	GR4	GR5	GR6
		Ausg.-bit 900	Ausg.-bit 901	Ausg.-bit 902	Ausg.-bit 903	Ausg.-bit 904	Ausg.-bit 905
Hand 1	offen (GO 0)	EIN	AUS				
	geschlossen (GC 0)	AUS	EIN				
Hand 2	offen (GO 1)			EIN	AUS		
	geschlossen (GC 1)			AUS	EIN		
Hand 3	offen (GO 2)					EIN	AUS
	geschlossen (GC 2)					AUS	EIN

**Tab. 5-4:** Einstellmöglichkeiten für den Handgreiferzustand der pneumatisch angetriebenen Hand

GR1 – GR6 kennzeichnen die Steckernummer des Handkabels im Roboterarm.

### Programmbeispiel (MOVEMASTER-Befehle)

```

10  OD    &FFFF    Bit 0 – 15 des externen Ausgabeport einschalten
20  OB    -10       Nur Bit 10 ausschalten
30  ED
```

Programmende

## 5.2.55 OC (Output Counter)

### Funktion: Zählerwert ausgeben

Gibt den Wert des festgelegten Zählers ohne Ausführungsbedingung über den Ausgabeport aus.

### Eingabeformat

```
OC  <Zählernummer> [, [<Ausgangs-Bitnummer>]
    [, [<Bitdatenbreite>]]]
```

<Zählernummer>	Legt den Zähler fest, dessen Zählerwert ausgegeben werden soll. $1 \leq \text{Zählernummer (ganzzahlig)} \leq 99$
<Bitnummer der Ausgabedaten>	Legt die Startbitnummer der Ausgabedaten fest. $0 \leq \text{Bitnummer (ganzzahlig)} \leq 32\,767$
<Bitdatenbreite>	Legt die Bitdatenbreite der Ausgabedaten fest. $1 \leq \text{Bitdatenbreite (ganzzahlig)} \leq 16$ (16 für Standardwert)

### Erläuterung

- Dieser Befehl gibt den Wert des festgelegten Zählers ohne Ausführungsbedingung über den externen Ausgabeport aus. Die weitere Datenausgabe wird durch die Ausführung des OC-Befehls nicht beeinflusst.
- Bei jeder Ausführung des OC-Befehls bleiben sowohl der Zählerwert als auch der interne Registerwert unverändert.
- Die Datenbreite der Ausgangssignale kann mit dem OC-Befehl über die Bitdatenbreite festgelegt werden.

### Programmbeispiel (BASIC-Befehle)

10	OPEN "COM1 :E83" AS#1	RS232C-Kommunikationsdatei wird über den PC mittels BASIC-Befehl geöffnet
20	PRINT #1,"SC 5,&0008"	Wert 8 in den Zähler 5 laden
30	PRINT #1,"OC5"	Wert des Zählers 5 unbedingt über den externen Ausgabeport ausgeben
40	END	Programmende

## 5.2.56 OD (Output Direct)

### Funktion: direkte Ausgabe

Gibt die festgelegten Daten ohne Ausführungsbedingung über den Ausgabeport aus.

### Eingabeformat

```
OD  <Ausgabedaten> [, [<Bitnummer der Ausgabedaten>]
    [, [<Bitdatenbreite>]]]
```

<b>&lt;Ausgabedaten&gt;</b>	Legt die Ausgabedaten fest. $-32\,768 \leq \text{Ausgabedaten (dezimal)} \leq 32\,767$ $\&8000 \leq \text{Ausgabedaten (hexadezimal)} \leq \&7FFF$
<b>&lt;Bitnummer der Ausgabedaten&gt;</b>	Legt die Startbitnummer der Ausgabedaten fest. $0 \leq \text{Bitnummer (ganzzahlig)} \leq 32\,767$ (0 für Standardwert)
<b>&lt;Bitdatenbreite&gt;</b>	Legt die Bitdatenbreite der Ausgabedaten fest. $1 \leq \text{Bitdatenbreite (ganzzahlig)} \leq 16$ (16 für Standardwert)

### Erläuterung

- Dieser Befehl gibt ohne Ausführungsbedingung über den Ausgabeport ein Signal (Paralleldaten) aus, welches für ein externes Gerät, wie z. B. eine SPS, eingesetzt werden kann. Die weitere Datenausgabe wird durch die Ausführung des OD-Befehls nicht beeinflusst.
- Die Ausgabedaten können entweder dezimal oder hexadezimal angegeben werden. Für eine hexadezimale Angabe muß vor dem Datenwert das Zeichen „&“ stehen.
- Detaillierte Informationen zum Anschluß von externen Geräten enthält das Handbuch zur Installation des Roboters.
- Die Datenbreite der Ausgangssignale kann mit dem OD-Befehl über die Bitdatenbreite festgelegt werden.
- Es tritt eine Fehlermeldung auf, wenn eines der speziellen Bits des Parameters OT1 bis OT3 für die externe Ausgabe festgelegt wurde.
- Die weitere Datenausgabe wird durch die Ausführung des OD-Befehls nicht beeinflusst.

### Programmbeispiel (MOVEMASTER-Befehle)

10	OD	&FFFF	Am Ausgabeport 16 Bits ab Bit 0 einschalten
20	OD	&FFFF,10	Am Ausgabeport 16 Bits ab Bit 10 einschalten
30	OD	&FFFF,10,15	Am Ausgabeport 15 Bits ab Bit 10 einschalten
40	ED		Programmende

## 5.2.57 OG (Origin)

### Funktion: Nullpunkt anfahren

Führt den Roboter zum benutzerdefinierten Nullpunkt (Gelenk-Interpolation).

### Eingabeformat

OG
----

### Erläuterung

- Dieser Befehl fährt den Roboter mittels Gelenk-Interpolation zum benutzerdefinierten Nullpunkt, dessen Koordinatenwerte über den Parameter UOG festgelegt werden.
- Nach Ausführung des OG-Befehls nimmt der Roboter die gleiche Stellung wie nach Ausführung des NT-Befehls ein. Die über den UNG-Parameter definierte Bewegungsreihenfolge hat bei der Ausführung des OG-Befehls keine Gültigkeit.
- Der Parameter UOG kann über die Teaching Box eingestellt werden (siehe Technisches Handbuch).

### Parameter

Die Roboterstellung für den benutzerdefinierten Nullpunkt kann über den nachfolgenden Parameter geändert werden:

Parameter UOG:            Roboterstellung am benutzerdefinierten Nullpunkt (Grad)  
                                  5 Achsen: –160.00, –45.00, 00.0, 00.0, –120.0, –200.00  
                                  6 Achsen: –160.00, –45.00, 50.00, –160.00, –120.00, –200.00  
                                  (Standardwerte)

### Programmbeispiel (MOVEMASTER-Befehle)

10	NT		Nullpunkt anfahren
20	MO	2	Position 2 anfahren
30	OG		Nullpunkt anfahren
40	ED		Programmende

## 5.2.58 OPN (Open)

### Funktion: Kommunikationskanäle öffnen

Öffnet die Kommunikationskanäle und legt die Schnittstellen fest.

### Eingabeformat

```
OPN <Kanalnummer>, <Schnittstellenummer>
```

<Kanalnummer>            Legt den Kommunikationskanal für die Ein-/Ausgabe fest.  
 $1 \leq \text{Kanalnummer} \leq 4$

<Schnittstellenummer>    Legt die Schnittstellenummer für die Ein-/Ausgabe fest.  
 0 : RS422  
 1 : RS232C  
 2 : RS232C-1 (Option)  
 3 : RS232C-2 (Option)

### Erläuterung

- Dieser Befehl öffnet die Kommunikationskanäle und definiert die entsprechende Schnittstelle für die Ein-/Ausgabe.
- Es können bis zu 4 Kommunikationskanäle gleichzeitig geöffnet werden.
- Durch Einsatz des OPN-Befehls in Verbindung mit dem INP-Befehl ist es möglich, Zählerwerte und Positionsdaten vom PC zur Robotersteuerung zu übertragen.

### Programmbeispiel (MOVEMASTER-Befehle)

10	OPN	1,1	RS232C-Kommunikationskanäle öffnen
20	INP	1,1,1	Positionsdaten von der RS232C-Schnittstelle lesen
30	INP	1,1,0	Zählerwerte von der RS232C-Schnittstelle lesen
40	INP	1,\$1,2	Daten der Zeichenkette von der RS232C-Schnittstelle lesen

## 5.2.59 OR (Or)

### Funktion: ODER-Verknüpfung

ODER-Verknüpfung des festgelegten Datenwertes mit dem Wert des internen Registers.

### Eingabeformat

OR    <Datenwert>
-------------------

<Datenwert>

Legt den Datenwert fest.

$-32\,768 \leq \text{Datenwert (dezimal)} \leq 32\,767$

$\&8000 \leq \text{Datenwert (hexadezimal)} \leq \&7FFF$

@ ≤ Zählernummer ≤ @99

### Erläuterung

- Der festgelegte Datenwert kann dezimal oder hexadezimal definiert werden. Bei einem hexadezimalen Wert muß vor der Zahl das Zeichen „&“ stehen.
- Das Ergebnis wird im internen Register abgespeichert und kann mit den entsprechenden Befehlen verändert, verglichen oder gelesen werden (siehe Befehle EQ, NE, LG, SM, CL, DR, ADD, SUB, MUL, DIV, AN und XO).
- Durch Ausführung des OR-Befehls nach einem Eingabebefehl (ID oder IN) besteht die Möglichkeit, vom einem externen Gerät nur die erforderlichen Bits der parallelen Eingabedaten zu empfangen.

### Programmbeispiel (MOVEMASTER-Befehle)

10	ID		Daten am externen Eingabeport holen
20	OR	&FFF0	Alle Bits auf 1 setzen (außer die vier niedrigsten)
30	EQ	&FFFF,100	Sprung zur Zeile 100, wenn bei den oberen Daten alle Bits auf 1 gesetzt sind
40	ED		Programm beenden
100	MO	10	Position 10 anfahren



## 5.2.60 OVR (Override)

### Funktion: Übersteuerung

Legt den Programmwert für die Geschwindigkeits-Übersteuerung fest.

### Eingabeformat

OVR <Übersteuerungswert>
--------------------------

<Übersteuerungswert>      Legt den prozentualen Übersteuerungswert fest.  
 $1 \leq \text{Übersteuerungswert} \leq 200$

### Erläuterung

- Der OVR-Befehl gibt eine globale Geschwindigkeitserhöhung oder Absenkung über den gesamten Programmverlauf vor. Dieser Befehl wird häufig beim erstmaligen Test eines Roboterprogramms eingesetzt, damit der Bewegungsablauf leichter nachvollzogen werden kann und legt den prozentualen Übersteuerungswert für die Arbeitsgeschwindigkeit des Roboters fest.
- Der OVR-Befehl ist bei jeder Interpolationsmethode wirksam, d. h. bei der Gelenk-, Linear- und Kreis-Interpolation.
- Die aktuelle Arbeitsgeschwindigkeit ergibt sich folgendermaßen:

$$\begin{aligned} \text{Gelenk-Interpolation} &= \text{Playback-Übersteuerungswert} \times \text{Einstellwert des OVR-Befehls} \times \text{Einstellwert des SP-Befehls} \\ \text{Linear-Interpolation} &= \text{Playback-Übersteuerungswert} \times \text{Einstellwert des OVR-Befehls} \times \text{Einstellwert des SP- oder SD-Befehls} \end{aligned}$$

Der Playback-Übersteuerungswert kann über die Teaching Box oder ein externes Eingangssignal festgelegt werden. Die mit dem OVR-Befehl festgelegte Übersteuerung wird Programmübersteuerung genannt.

- Der Initialisierungswert beträgt 100 %.
- Der in einem Programm festgelegte Übersteuerungswert bleibt so lange wirksam, bis er durch einen neuen Wert ersetzt oder das Programm beendet wird.
- Es tritt eine Fehlermeldung auf, wenn für den OVR-Befehl der Wert 0 festgelegt wird.

### Programmbeispiel (MOVEMASTER-Befehle)

10	SP	30	Arbeitsgeschwindigkeit auf den Wert 30 (100 %) einstellen
20	OVR	80	Übersteuerung auf den Wert 80 % einstellen
30	MO	2	Position 2 anfahren
40	ED		Programmende

Es ergibt sich die nachfolgende aktuelle Arbeitsgeschwindigkeit, wenn für das obige Beispiel der Wert für die Playback-Übersteuerung auf 50 % festgelegt wurde.

Arbeitsgeschwindigkeit bei Gelenk-Interpolation =  $50 \times 80 \times 100 [\%] = 40 \%$

Der Roboter fährt zur Position 2 mit einer Geschwindigkeit von 40 % des Maximalwertes.

## 5.2.61 PA (Pallet Assign)

### Funktion: Gitterpunkte für Palette definieren

Definiert die Anzahl der Gitterpunkte in Spalten- und Zeilenrichtung für die festgelegte Palette. Es können 9 Paletten pro Programm definiert werden.

### Eingabeformat

```
PA  <Palettennummer>, <Anzahl der Spaltengitterpunkte>,
    <Anzahl der Zeilengitterpunkte>
```

<Palettennummer>	Legt die Nummer der eingesetzten Palette fest. $1 \leq \text{Palettennummer} \leq 9$
<Anzahl der Spalten- gitterpunkte>	Legt die Gitterpunkte der Palette in Spaltenrichtung fest. $1 \leq \text{Anzahl der Spaltengitterpunkte} \leq 32\ 767$
<Anzahl der Zeilen- gitterpunkte>	Legt die Gitterpunkte der Palette in Zeilenrichtung fest. $1 \leq \text{Anzahl der Zeilengitterpunkte} \leq 32\ 767$

### Erläuterung

- Der PA-Befehl muß vor dem Palettenberechnungsbefehl (siehe PT-Befehl) ausgeführt werden.
- Die Anzahl der Gitterpunkte entspricht der Anzahl der tatsächlich auf der Palette angeordneten Arbeitsgegenstände. Bei einer Palette mit z. B. 15 Arbeitsgegenständen (3 x 5) ist die Anzahl der Spaltengitterpunkte 3 und die Anzahl der Zeilengitterpunkte 5.
- Die Spalten- und Zeilenrichtung wird durch die Festlegungen für die Endpositionen (Eckpunkte) bestimmt (siehe PT-Befehl).

### Programmbeispiel (MOVEMASTER-Befehle)

10	PA	5,20,30	Palette 5 mit einem Gitterpunktraster von 20 x 30 definieren
20	SC	51,15	Wert 15 in den Zähler 51 laden (Spaltengitterpunkte)
30	SC	52,25	Wert 25 in den Zähler 52 laden (Zeilengitterpunkte)
40	PT	5	Koordinatenwerte des Gitterpunktes (15, 25) als Position 5 festlegen
50	MO	5	Position 5 anfahren
60	ED		Programmende

## 5.2.62 PC ♦ (Position Clear)

### Funktion: Position löschen

Löscht die Positionsdaten der festgelegten Position[en].

### Eingabeformat

```
PC <Positionsnummer (a)>, [, [<Positionsnummer (b)>]]
```

<Positionsnummer>      Legt die zu löschende Positionsnummer fest.  
 $1 \leq \text{Positionsnummer (a)} \leq 999$   
 $1 \leq \text{Positionsnummer (b)} \leq 999$   
 $\text{Positionsnummer (a)} \leq \text{Positionsnummer (b)}$

### Erläuterung

- Dieser Befehl löscht alle Positionsdaten zwischen der Position (a) und (b). Die Position (a) und (b) wird auch gelöscht.
- Es werden nur die Positionsdaten von Position (a) gelöscht, wenn die Position (b) nicht angegeben wurde.

### Programmbeispiel (BASIC-Befehle)

10 OPEN "COM1 :E83" AS#1	RS232C-Kommunikationsdatei wird über den PC mittels BASIC-Befehl geöffnet
20 PRINT #1,"MO 10"	Position 10 anfahren
30 PRINT #1,"MO 11"	Position 11 anfahren
40 PRINT #1,"MO 12"	Position 12 anfahren
50 PRINT #1,"PC 11"	Position 11 löschen
60 PRINT 31,"DP"	Position 10 anfahren
70 END	BASIC-Programmende

## 5.2.63 PD (Position Define)

### Funktion: Position definieren

Definiert die Koordinaten (XYZ-Koordinaten und Winkel) der festgelegten Position.

#### 5 Achsen

#### Eingabeformat

```
PD  <Positionsnummer>, [<X-Koordinatenwert>],
    [<Y-Koordinatenwert>], [<Z-Koordinatenwert>],
    [<Drehwinkel A>], [<Drehwinkel B>],
    [, [<R/L>] [, [<A/B>]]] [, [<O/C>]]
```

<Positionsnummer>	Legt die zu definierende Positionsnummer fest. $1 \leq \text{Positionsnummer} \leq 999$
<XYZ-Koordinaten>	Legen die Position im XYZ-Koordinatensystem des Roboters fest. (Einheit: mm), (0 für Standardwert)
<Drehwinkel A, B>	Legen die Drehwinkel der Roll- und Neigungsgelenke im XYZ-Koordinatensystem des Roboters fest. (Einheit: Grad), (0 für Standardwert)
<R/L>	Bestimmt die Roboterstellung (rechts oder links). R : rechts (Standardwert) L : links
<A/B>	Bestimmt die Roboterstellung (oben oder unten). A : oben (Standardwert) B : unten
<O/C>	Legt den Greiferzustand für Hand 1 fest (offen/geschlossen). O : Handgreifer offen C : Handgreifer geschlossen

#### 6 Achsen

#### Eingabeformat

```
PD  <Positionsnummer>, [<X-Koordinatenwert>],
    [<Y-Koordinatenwert>], [<Z-Koordinatenwert>],
    [<Drehwinkel A>], [<Drehwinkel B>], [<Drehwinkel C>],
    [, [<R/L>] [, [<A/B>] [, [<N/F>]]]] [, [<O/C>]]
```

<Positionsnummer>	Legt die zu definierende Positionsnummer fest $1 \leq \text{Positionsnummer} \leq 999$
<XYZ-Koordinaten>	Legen die Position im XYZ-Koordinatensystem des Roboters fest (Einheit: mm), (0 für Standardwert)
<Drehwinkel A, B, C>	Legen die Drehwinkel der Roll- und Neigungsgelenke im XYZ-Koordinatensystem des Roboters fest (Einheit: Grad), (0 für Standardwert)
<R/L>	Bestimmt die Roboterstellung (rechts oder links) R : rechts (Standardwert) L : links
<A/B>	Bestimmt die Roboterstellung (oben oder unten) A : oben (Standardwert) B : unten
<N/F>	Bestimmt die Roboterstellung (nicht kippen oder kippen)

N : nicht kippen (Standardwert)

F : kippen

<O/C>

Legt den Greiferzustand für Hand 1 fest (offen/geschlossen)

O : Handgreifer offen

C : Handgreifer geschlossen

### Erläuterung

- Die Koordinatenwerte können in Einheiten von 0,01 mm oder 0,01° angegeben werden. Geben Sie für z. B. 20,01 mm bzw. 20,01° den Wert 20.01 an.
- Beachten Sie, daß es zu keiner Fehlermeldung kommt, wenn die definierten Koordinaten außerhalb des zulässigen Roboterarbeitsbereichs liegen.
- Mit dem PD-Befehl kann in Kombination mit dem SF- oder MA-Befehl ein Verfahrenwegbe-  
trag festgelegt werden.

### 5 Achsen

#### Programmbeispiel (MOVEMASTER-Befehle)

10	PD	10,50,320,70,40,30,R,A,O	XYZ-Koordinaten und Winkel der Position 10 definieren
40	MO	10	Position 10 anfahren
50	ED		Programmende

### 6 Achsen

#### Programmbeispiel (MOVEMASTER-Befehle)

10	PD	10,50,320,70,50,40,30,R,A,N,O	XYZ-Koordinaten und Winkel der Position 10 definieren
40	MO	10	Position 10 anfahren
50	ED		Programmende

## 5.2.64 PL (Position Load)

### Funktion: Position kopieren

Kopiert die Position (b) durch Überschreiben der Position (a).

### Eingabeformat

PL   <Positionsnummer (a)>, <Positionsnummer (b)>
---

<Positionsnummer (a)>    Legt die Positionsnummer fest (Zielposition).  
 $1 \leq \text{Positionsnummer (a)} \leq 999$

<Positionsnummer (b)>    Legt die Positionsnummer fest (Quellposition).  
 $1 \leq \text{Positionsnummer (a)} \leq 999$

### Erläuterung

- Nach Ausführung des PL-Befehls werden die Koordinaten der Position (a) durch die Koordinaten der Position (b) ersetzt. Die alten Koordinaten der Position (a) werden gelöscht. Mit dem PL-Befehl können die Koordinaten der Position (b) kopiert werden.
- Der Kopiervorgang bezieht sich auch auf den Handgreiferzustand. Der Handgreiferzustand von Position (a) wird durch den Handgreiferzustand von Position (b) ersetzt.
- Es tritt eine Fehlermeldung auf, wenn die Position (b) noch nicht definiert wurde.
- Es wird eine neue Position angelegt, wenn die Position (a) nicht definiert ist.

### Programmbeispiel (MOVEMASTER-Befehle)

10	HE	2	Aktuelle Koordinaten und Handgreiferzustand als Position 2 abspeichern
20	PL	3,2	Koordinaten der Position 3 durch Koordinaten der Position 2 ersetzen
30	ED		Programmende

## 5.2.65 PMR (Parameter Read)

### Funktion: Parameterwerte lesen

Liest die Inhalte des festgelegten Parameters.

### Eingabeformat

```
PMR [ "<Parametername>" ]
```

<Parametername>      Legt den Parameter über den zugehörigen Namen fest.  
Es dürfen nur die definierten Parameternamen benutzt werden.  
(vorgegebene Reihenfolge der definierten  
Parameternamen für Grundeinstellung)

### Erläuterung

- Die Inhalte des festgelegten Parameters werden über die RS232C-Schnittstelle ausgegeben.  
Ausgabeformat: Parametername, Inhalte
- Es wird der nächste Parameter der vorgegebenen alphabetischen Reihenfolge ausgegeben, wenn der Parametername nicht festgelegt wurde. Nach dem letzten Parameter wird ein „Carriage Return“ (hex. 0D) ausgegeben.
- Es wird der nächste Parameter der vorgegebenen alphabetischen Reihenfolge ausgegeben, wenn der festgelegte Parameter nicht existiert.

### Programmbeispiel (BASIC-Befehle)

10 OPEN "COM1 :E83" AS#1	RS232C-Kommunikationsdatei wird über den PC mittels BASIC-Befehl geöffnet
20 INPUT "Parametername = " ;J\$	Parametername eingeben
30 PRINT #1,"PMR"+CHR\$(&H22)+J\$+CHR\$(&H22)	PMR-Befehl und Parametername übertragen
40 LINE INPUT #1,A\$	Übertragene Daten in A\$ speichern
50 PRINT A\$	Inhalte von A\$ auf dem Bildschirm anzeigen
60 END	Ende des BASIC-Programms
RUN	BASIC-Programm starten
Parametername = ? ADL	Parametername (ADL) eingeben
ADL,0.20,0.20	Ausgabe der Inhalte von Parameter ADL

## 5.2.66 PMW (Parameter Writing)

### Funktion: Parameterwerte schreiben

Überschreibt die Inhalte des festgelegten Parameters.

### Eingabeformat

```
PMW "<Parametername>", "<Parameterinhalte>"
```

<Parametername>      Legt den Parameter fest, dessen Inhalt geändert werden soll.

<Parameterinhalte>      Legt die neuen Inhalte des Parameters fest.

### Erläuterung

- Die Inhalte des festgelegten Parameters werden über die RS232C-Schnittstelle vom PC an das Steuergerät ausgegeben.  
Ausgabeformat: Parametername, Inhalte
- Wenn der festgelegte Parameter nicht existiert, erfolgt auch keine Funktionsausführung.
- Eine Auflistung aller Parameter enthält der Abs. A.1.7 im Anhang.
- Beachten Sie, daß die alten Inhalte des festgelegten Parameters bis zum Ausschalten der Spannungsversorgung wirksam bleiben. Die neuen Parameterinhalte werden erst nach wiederholtem Einschalten der Spannungsversorgung wirksam (Versorgungsspannung: AUS → EIN).

### Programmbeispiel (MOVEMASTER-Befehle)

PMW "ADL", "0.40,0.40"      Alte Inhalte des Parameters ADL mit den neuen Inhalten (0.40,0.40) überschreiben



## 5.2.67 PR (Position Read)

### Funktion: Positionsdaten lesen

Liest die Koordinaten und den Handgreiferzustand der festgelegten Position (über die RS232C-Schnittstelle).

### Eingabeformat

PR    <Positionsnummer>
-------------------------

<Positionsnummer>	Legt die Position fest, deren Daten gelesen werden sollen. $1 \leq \text{Positionsnummer} \leq 999$ (Bei fehlender Positionsnummer werden die Daten der aktuellen Position ausgegeben)
-------------------	--

### Erläuterung

- Dieser Befehl gibt die Koordinaten und den Handgreiferzustand (offen/geschlossen) der festgelegten Position über die RS232C-Schnittstelle aus. Es werden die Daten der aktuellen Position ausgegeben, wenn die Positionsnummer fehlt oder der Wert 0 festgelegt wurde.
- Die Daten werden im ASCII-Code und im nachfolgenden Format ausgegeben.  
 Ausgabeformat:  
 5 Achsen : XYZ-Koordinaten, Drehwinkel (A und B), R/L, A/B, O/C  
 6 Achsen : XYZ-Koordinaten, Drehwinkel (A, B und C), R/L, A/B, N/F, O/C
- Die kleinste Schrittweite beträgt 0,01 mm oder 0,01°.
- Das Abschlußzeichen der Ausgabedaten ist ein „Carriage Return“ (CR, hex. 0D). Deshalb ist es beim Einsatz eines PCs erforderlich, daß die serielle Datenkette bis zur Stelle „hex. 0D“ empfangen wird. Der Befehl „LINE INPUT #“ ist der entsprechende BASIC-Befehl zum Einlesen der Daten.
- Es wird für jede Koordinate der Wert 0 ausgegeben, wenn eine Position gelesen wird, die im Programm nicht eingesetzt wurde und auch nicht definiert ist.  
 Es wird für jede Koordinate der Wert 0.00 ausgegeben, wenn eine Position gelesen wird, die zwar im Programm eingesetzt wurde, aber nicht definiert ist.
- Wenn nach Ausführung eines Verfahrenbefehls eine Fehlermeldung aufgetreten ist, können Sie sich mit der Hilfe des PR-Befehls die aktuelle Positionsnummer anzeigen lassen. Lassen Sie hierfür die Angabe der Positionsnummer aus oder geben Sie den Wert 0 an.

**5 Achsen****Programmbeispiel (BASIC-Befehle)**

```

10 OPEN "COM1 :E83" AS#1

20 INPUT "Positionsnummer = " ;P
30 PRINT #1,"PR"+STR$(P)
40 LINE INPUT #1,A$
50 PRINT A$
60 END

```

RUN

Positionsnummer = ? 15  
 +10.00,+380.00,300.00,50.00,40.00,  
 R,A,C

RS232C-Kommunikationsdatei wird über  
 den PC mittels BASIC-Befehl geöffnet  
 Positionsnummer eingeben  
 PR-Befehl und Positionsnummer übertragen  
 Übertragene Daten in A\$ speichern  
 Inhalte von A\$ auf dem Bildschirm anzeigen  
 Ende des BASIC-Programms

BASIC-Programm starten

Positionsnummer (15) eingeben  
 Ausgabe der Positionsdaten

**6 Achsen****Programmbeispiel (BASIC-Befehle)**

```

10 OPEN "COM1 :E83" AS#1

20 INPUT "Positionsnummer = " ;P
30 PRINT #1,"PR"+STR$(P)
40 LINE INPUT #1,A$
50 PRINT A$
60 END

```

RUN

Positionsnummer = ? 15  
 +10.00,+380.00,300.00,-70.00,  
 50.00,40.00,R,A,N,C

RS232C-Kommunikationsdatei wird über  
 den PC mittels BASIC-Befehl geöffnet  
 Positionsnummer eingeben  
 PR-Befehl und Positionsnummer übertragen  
 Übertragene Daten in A\$ speichern  
 Inhalte von A\$ auf dem Bildschirm anzeigen  
 Ende des BASIC-Programms

BASIC-Programm starten

Positionsnummer (15) eingeben  
 Ausgabe der Positionsdaten

## 5.2.68 PRN ♦ (Print)

### Funktion: Daten übertragen

Überträgt den Einstellwert für einen Zähler oder Koordinatenwerte für eine Position.

### Eingabeformat

PRN <Zählerwert> | <Positionskoordinaten> | "<Zeichenkette>"

<Zählerwert>	Legt den Einstellwert für den Zähler fest. $-32\,768 \leq \text{Zählerwert (dezimal)} \leq 32\,767$ $\&8000 \leq \text{Zählerwert (hexadezimal)} \leq \&7FFF$
<Positionskoordinaten>	Legen die Koordinateneinstellwerte für die Position fest. Die Einstellung erfolgt in der gleichen Art und Weise wie beim PD-Befehl (siehe Beschreibung zum PD-Befehl).
<Zeichenkette>	Legt die Zeichenkette fest. $1 \leq \text{Zeichenkettendaten} \leq 120$ (Anzahl der Zeichen) Erlaubte Zeichen:    Nummern (0 bis 9) Buchstaben (A bis Z) Symbole (!@#, u.s.w.) Sonderzeichen: *+,./,:□¥'

### Erläuterung

- Dieser Befehl überträgt einen Zählereinstellwert, Koordinatenwerte für eine Position oder eine Zeichenkette vom PC an das Steuergerät über die RS232C-Schnittstelle, die dann mit einem INP-Befehl im Roboterprogramm ausgelesen werden kann.
- Der Roboter bleibt bei der Ausführung des INP-Befehls so lange in Wartestellung, bis die entsprechenden Daten durch Ausführung des PRN-Befehls vom PC zur Robotersteuerung übertragen werden.
- Der PRN-Befehl sollte vor dem zugehörigen INP-Befehl ausgeführt werden.
- Bei der Übertragung von Zeichenketten, muß die Zeichenkette in Anführungszeichen (""") angegeben werden.

### 5 Achsen

### Programmbeispiel

#### 1.) PC-Programm (BASIC-Befehle)

10 OPEN "COM1 :E83" AS#1	RS232C-Kommunikationsdatei wird über den PC mittels BASIC-Befehl geöffnet
20 INPUT "Zählerwert = ";J	Einstellwert für den Zähler über den PC eingeben
30 PRINT #1,"PRN"+STR\$(J)	Zählerwert übertragen und in den Zähler laden
40 PRINT #1,"PRN 100,0,0,0,0"	Positionskoordinaten übertragen
50 END	Ende des BASIC-Programms

**2.) Roboterprogramm (MOVEMASTER-Befehle)**

10	OPN	2,1	RS232C-Schnittstelle öffnen
20	INP	2,1,0	Daten von der RS232C-Schnittstelle lesen und in den Zähler 1 laden
30	INP	2,5,1	Daten von der RS232C-Schnittstelle lesen und in der Position 5 speichern
40	IC	1	Zählerwert von Zähler 1 um 1 erhöhen
50	MO	5	Position 5 anfahren

**6 Achsen****Programmbeispiel****1.) PC-Programm (BASIC-Befehle)**

10	OPEN "COM1 :E83"	AS#1	RS232C-Kommunikationsdatei wird über den PC mittels BASIC-Befehl geöffnet
20	INPUT "Zählerwert = ";J		Einstellwert für den Zähler über den PC eingeben
30	PRINT #1,"PRN"+STR\$(J)		Zählerwert übertragen
40	PRINT #1,"PRN 100,0,0,0,0,0"		Positionskoordinaten übertragen
50	END		Ende des BASIC-Programms

**2.) Roboterprogramm (MOVEMASTER-Befehle)**

10	OPN	2,1	RS232C-Schnittstelle öffnen
20	INP	2,1,0	Daten von der RS232C-Schnittstelle lesen und in den Zähler 1 laden
30	INP	2,5,1	Daten von der RS232C-Schnittstelle lesen und in der Position 5 speichern
40	IC	1	Zählerwert von Zähler 1 um 1 erhöhen
50	MO	5	Position 5 anfahren

## 5.2.69 PT (Pallet)

### Funktion: Koordinaten für Palette berechnen

Berechnet die Koordinaten eines Gitterpunktes der festgelegten Palette und weist die berechneten Koordinaten der festgelegten Position zu.

### Eingabeformat

PT <Palettennummer>

<Palettennummer>      Legt die Nummer der eingesetzten Palette fest.  
 $1 \leq \text{Palettennummer} \leq 9$

### Erläuterung

- Dieser Befehl berechnet die Koordinaten eines Gitterpunktes der festgelegten Palette und weist die berechneten Koordinaten der Position zu, deren Nummer mit der festgelegten Palettennummer übereinstimmt (siehe untere Tabelle). Vor Ausführung des PT-Befehls muß zuerst die Anzahl der Gitterpunkte für die eingesetzte Palette mit dem PA-Befehl festgelegt werden. Nach Ausführung des PT-Befehls werden die vorher definierten Positionsdaten der Gitterzielposition gelöscht.
- Vor Ausführung des PT-Befehls müssen die Palettenpunkte (Gitterpunkte an den vier Eckpunkten), welche eine bestimmte Palette und die Palettenzähler (Spalten und Zeilen) indentifizieren, genau definiert werden. Nur so können von der Robotersteuerung die Koordinaten für einen bestimmten Gitterpunkt richtig berechnet werden. Nachdem die Palettenpositionen und Palettenzähler definiert worden sind, können anschließend mit dem PT-Befehl die Koordinaten der Gitterpunkte als Positionsnummer entsprechend der Palettennummer definiert werden. In der nachfolgenden Tabelle sind für jede Palettennummer die zugehörigen Palettenpositionen und Palettenzähler aufgeführt.

Palettennummer	1	2	3	4	5	6	7	8	9
Bezugsposition der Palette	10	20	30	40	50	60	70	80	90
Spaltenendpunkt der Palette	11	21	31	41	51	61	71	81	91
Zeilenendpunkt	12	22	32	42	52	62	72	82	92
Paletteneckpunkt, der gegenüber der Bezugsposition liegt	13	23	33	43	53	63	73	83	93
Spaltenzähler der Palette	11	21	31	41	51	61	71	81	91
Zeilenzähler der Palette	12	22	32	42	52	62	72	82	92
Palettengitterposition	1	2	3	4	5	6	7	8	9

**Tab. 5-5:** Palettennummern und deren Positionen und Zähler

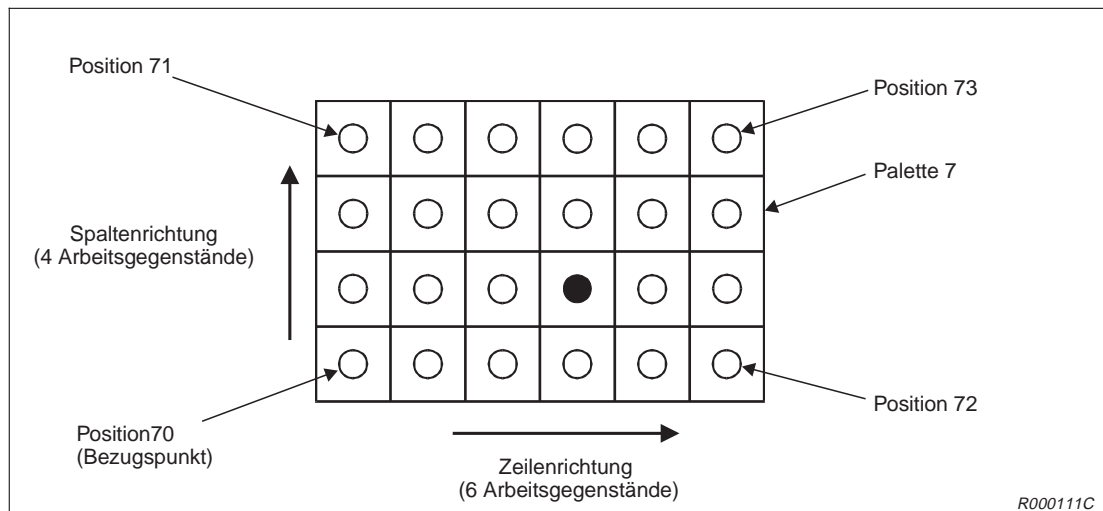
- Es tritt eine Fehlermeldung auf, wenn
  - die Palettenposition nicht definiert wurde, oder
  - der Palettenzähler nicht eingestellt oder
  - mit einem Wert eingestellt wurde, der außerhalb des für den PA-Befehls zulässigen Bereichs liegt.
- Der Handgreiferzustand (offen/geschlossen) der Gitterzielposition ist der gleiche wie der an der Palettenbezugsposition.
- Vor Ausführung des PT-Befehls muß die Werkzeuglänge für die eingesetzte Hand mit dem

TL-Befehl genaustens festgelegt werden. Die Palettenpositionen (vier Paletteneckpunkte) können von der Robotersteuerung erst nach Festlegung der richtigen Werkzeuglänge ermittelt werden.

- Beachten Sie beim Einsatz der Palettennummer 9, daß die zugehörigen Palettenzähler (91, 92) oftmals auch als gemeinsame Zähler im Roboterprogramm verwendet werden.

### Programmbeispiel (MOVEMASTER-Befehle)

Als Beispiel dient eine Palette, auf der 24 Arbeitsgegenstände angeordnet sind (4 in Spalten- und 6 in Zeilenrichtung). Die Robotersteuerung soll die Koordinaten des an der Gitterposition (2, 4) befindlichen Arbeitsgegenstands berechnen, d. h. der zweite Gitterpunkt in Spaltenrichtung und vierte Gitterpunkt in Zeilenrichtung. Die Handspitze des Roboters soll diese Position anfahren. Für die Palette wird die Palettennummer 7 vergeben.



**Abb. 5-22:** Beispiel zur Palettendefinition

Die Positionen der Eckpunkte müssen im voraus mittels der Teaching Box angefahren und abgespeichert werden.

Roboterprogramm:

10	TL	200	Werkzeuglänge in Übereinstimmung mit der eingesetzten Hand festlegen
20	PA	7,4,6	Palettennummer und Anzahl der Gitterpunkte der Spalten und Zeilen festlegen
30	SC	71,2	Gitterpunkte in Spaltenrichtung definieren
40	SC	72,4	Gitterpunkte in Zeilenrichtung definieren
50	PT	7	Legt die Koordinaten der Gitterzielposition als Position 7 fest
60	MO	7	Position 7 anfahren
70	ED		Programmende

## 5.2.70 PW (Pulse Wait)

### Funktion: Warteimpulse

Positioniert über die Servomotoren alle Gelenke so lange, bis die festgelegte Positioniertoleranz erreicht ist.

### Eingabeformat

PW    <Positioniertoleranz>
-----------------------------

<Positioniertoleranz>    Legt die Anzahl der Justierimpulse für die Positioniertoleranz fest.  
 $1 \leq \text{Positioniertoleranz} \leq 10\,000$

### Erläuterung

- Mit diesem Befehl kann festgelegt werden, wie genau eine Position angefahren werden soll. Die Gelenke werden über die Servomotoren so lange positioniert, bis die festgelegte Anzahl der Einstellimpulse erreicht ist.
- Geben Sie einen kleinen Wert für die Positioniertoleranz an, wenn eine große Positioniergenauigkeit erforderlich ist (z. B. beim Greifen eines Arbeitsgegenstands). Geben Sie einen großen Wert an, wenn nur eine geringe Positioniergenauigkeit erforderlich ist. Bei Angabe eines kleinen Wertes für die Positioniertoleranz hat die Wartezeit für die Roboterpositionierung die gleiche Wirkung wie die Ausführung eines TI-Befehls.
- Die Positionierung dauert sehr lange, wenn bei der Handhabung von relativ schweren Lasten oder bei schnellen Roboterbewegungen eine große Positioniergenauigkeit festgelegt wurde, d. h. ein kleiner Wert für die Positioniertoleranz angegeben wurde.
- Der Initialisierungswert für die Positioniertoleranz beträgt 10 000 Justierimpulse.
- Es tritt eine Fehlermeldung auf, wenn der festgelegte Wert außerhalb des oben angegebenen Einstellbereichs liegt.

### Programmbeispiel (MOVEMASTER-Befehle)

10	PW	10	Roboter so lange positionieren, bis die Positioniertoleranz von 10 Justierimpulsen erreicht ist
20	MO	1	Position 1 anfahren
30	GC		Hand schließen
40	ED		Programmende

## 5.2.71 PX (Position Exchange)

### Funktion: Position austauschen

Tauscht die Koordinaten einer festgelegten Position durch die einer anderen festgelegten Position aus.

### Eingabeformat

PX   <Positionsnummer (a)>, <Positionsnummer (b)>
---

<Positionsnummer (a)>    Legt die Positionsnummer fest (alte Position).  
 $1 \leq \text{Positionsnummer (a)} \leq 999$

<Positionsnummer (b)>    Legt die Positionsnummer fest (neue Position).  
 $1 \leq \text{Positionsnummer (b)} \leq 999$

### Erläuterung

- Nach Ausführung des PX-Befehls werden die Koordinaten der Position (a) durch die Koordinaten der Position (b) ausgetauscht.
- Der Austausch bezieht sich auch auf den Handgreiferzustand. Der Handgreiferzustand von Position (a) wird durch den Handgreiferzustand von Position (b) ersetzt.
- Es tritt eine Fehlermeldung auf, wenn die Positionen (a) und (b) vorher nicht definiert worden sind.

### Programmbeispiel (MOVEMASTER-Befehle)

10	HE	2	Aktuelle Koordinaten und Handgreiferzustand als Position 2 abspeichern
20	MJ	20,30,10,0,0,0	Dreht jedes Gelenk um den festgelegten Winkelbetrag
30	GO		Hand öffnen
40	HE	3	Aktuelle Position als Position 3 abspeichern
50	PX	2,3	Koordinaten der Position 2 durch die Koordinaten der Position 3 ersetzen
60	ED		Programmende



## 5.2.72 QN (Question Number)

### Funktion: Programminformationen lesen

Liest den Programmnamen oder die Programminformationen.

### Eingabeformat

QN    [ <Programmname> ]
--------------------------

<Programmname>

Legt den Namen des Roboterprogramms fest  
(Zeichenkette mit max. 8 Stellen).

$0 \leq \text{Programmname} \leq 8$  (Zeichen)

zulässige Zeichen:

Zahlen (0 – 9)

Buchstaben ( A – Z)

Symbole (! @ # \$ % ^ & ( ) \_ | - { } - )

unzulässige Zeichen:

\* + , . / : ; = ? [ ¥ ] ‘

spezielle Programmnamen:

Wenn der Programmname ausschließlich aus numerischen Werten besteht, wird dieser wie eine Nummernangabe verarbeitet.

Bei der Verwendung von Buchstaben muß der Programmname in Anführungszeichen " " gesetzt werden.

### Erläuterung

- Dieser Befehl gibt den aktuellen Programmnamen oder die aktuellen Programminformationen über die RS232C-Schnittstelle aus. Zu dem festgelegten Programmnamen werden die entsprechenden Programminformationen ausgegeben. Es wird der aktuelle Programmname ausgegeben, wenn der QN-Befehl ohne eine weitere Angabe programmiert wird.
- Das Ausgabeformat erfolgt im ASCII-Code.
  - Format des Programmnamens: "N" gefolgt von "Programmname"
  - Format der Programminformationen: Anzahl der Programmschritte, Anzahl der Positionen, Anzahl der Zähler
- Das Abschlußzeichen der Ausgabedaten ist ein „Carriage Return“ (CR, hex. 0D). Deshalb ist es beim Einsatz eines PCs erforderlich, daß die serielle Datenkette bis zur Stelle „hex. 0D“ empfangen wird. Der Befehl "LINE INPUT #" ist der entsprechende BASIC-Befehl zum Einlesen der Daten.

### Programmbeispiel (BASIC-Befehle)

10 OPEN "COM1 :E83" AS#1	RS232C-Kommunikationsdatei wird über den PC mittels BASIC-Befehl geöffnet
20 PRINT #1,"QN"	QN-Befehl übertragen
30 LINE INPUT #1,A\$	Übertragene Daten in A\$ speichern
40 PRINT "Aktuelles Programm = "A\$	Inhalte von A\$ auf dem Bildschirm anzeigen
50 END	Ende des BASIC-Programms
RUN	BASIC-Programm starten
Aktuelles Programm = N10	Programmname (10) anzeigen

## 5.2.73 RC (Repeat Cycle)

### Funktion: Programmschleife

Wiederholt sooft den mit dem NX-Befehl festgelegten Programmabschnitt, bis die festgelegte Anzahl der Wiederholzyklen erreicht ist.

### Eingabeformat

RC    <Anzahl der Wiederholzyklen>
------------------------------------

<Anzahl der  
Wiederholzyklen>

Legt die Anzahl der Wiederholungen für die Programmschleife fest.  
 $1 \leq \text{Anzahl der Wiederholungen (dezimal)} \leq 32\,767$   
 $\&0001 \leq \text{Anzahl der Wiederholungen (hexadezimal)} \leq \&7FFF$

### Erläuterung

- Dieser Befehl wird in Verbindung mit dem NX-Befehl eingesetzt. Der NX-Befehl muß nach dem RC-Befehl programmiert werden. Der Programmabschnitt zwischen dem RC-Befehl und dem NX-Befehl wird sooft wiederholt abgearbeitet, bis der festgelegte Wert für die Anzahl der Wiederholzyklen erreicht ist. Anschließend wird die Programmabarbeitung an der dem NX-Befehl folgenden Programmzeile fortgesetzt.
- Das Einfügen einer weiteren Programmschleife in die existierende Programmschleife (zwischen RC- und NX-Befehl) wird „Verschachtelung“ (Nesting) genannt. Es sind bis zu 9 Verschachtelungen möglich.

### Programmbeispiel (MOVEMASTER-Befehle)

10	MO	1	Position 1 anfahren	
20	RC	3	Abarbeitung der Programmschleife (Programmabschnitt bis NX-Befehl) dreimal wiederholen	
30	MO	2	Position 2 anfahren	} Programmschleife
40	MO	3	Position 3 anfahren	
50	MO	4	Position 4 anfahren	
60	NX		Ende der Programmschleife	
70	MO	5	Position 5 anfahren	
80	ED		Programmende	

## 5.2.74 RN ♦ (Run)

### Funktion: Programm starten

Abarbeitung des festgelegten Programmabschnitts.

### Eingabeformat

```
RN  [<Startzeilennummer> [, <Endzeilennummer>
    [, [<Programmname>]]]
```

<Startzeilennummer>    Legt die Zeilennummer des Programmbeginns fest.  
 $1 \leq \text{Startzeilennummer} \leq 9999$  (oberste Zeile für Standardwert)

<Endzeilennummer>    Legt die Zeilennummer des Programmendes fest.  
 $1 \leq \text{Startzeilennummer} \leq 9999$   
 (letzte Zeile oder ED-Befehl für Standardwert)

<Programmname>    Legt den Namen des Roboterprogramms fest.  
 (Zeichenkette mit max. 8 Stellen)  
 $0 \leq \text{Programmname} \leq 8$  (Zeichen)

zulässige Zeichen:    Zahlen (0 – 9)  
                           Buchstaben ( A – Z)  
                           Symbole (! @ # \$ % ^ & ( ) \_ | ~ { } - )

unzulässige Zeichen:    \* + , . / : ; = ? [ \ ] ' `

spezielle Programmnamen:

Wenn der Programmname ausschließlich aus numerische Werten besteht, wird dieser wie eine Nummernangabe verarbeitet.

Bei der Verwendung von Buchstaben muß der Programmname in Anführungszeichen " " gesetzt werden.

### Erläuterung

- Dieser Befehl startet die Abarbeitung des festgelegten Programmabschnitts. Der Programmabschnitt beginnt mit der festgelegten Startzeile und endet eine Zeile vor der festgelegten Endzeile.
- Wenn das Programm nicht fortlaufend programmiert wurde, kann es durch Angabe der Endzeile neugestartet werden.
- Durch Angabe des Programmnamens können Sie ein bestimmtes Programm auswählen. In diesem Fall wird das ausgewählte Programm zum Zielprogramm und die Abarbeitung beginnt an der festgelegten Startzeile.
- Es kommt zu keiner Funktionsausführung, wenn der RN-Befehl mit einer vorangesetzten Zeilennummer programmiert wurde.
- Bei Ausführung des RN-Befehls bleiben die Zählerwerte erhalten (keine Initialisierung).
- Die folgenden Namensbezeichnungen werden wie ein und derselbe Name angenommen:  
 Beispiel:  
 Wird wie ein und derselbe Name angenommen : 1, 01, 001, (nur numerische Werte)  
 Wird wie unterschiedliche Namen angenommen : 1, 1 A, A0\_001 (inklusive Buchstaben)
- Die Zeichen 0 – 9 und A – Z kann das Steuergerät über die 7-Segment-Anzeige darstellen.

**Programmbeispiel (BASIC-Befehle)**

```
10 OPEN "COM1 :E83" AS#1
20 PRINT #1,"RN 100,,2"
30 END
```

RS232C-Kommunikationsdatei wird über  
den PC mittels BASIC-Befehl geöffnet  
Programm 2 ab der Zeile 100 starten  
BASIC-Programmende

## 5.2.75 RS ♦ (Reset)

### Funktion: Programm/Fehlerbedingung zurücksetzen

Setzt das Programm und die Fehlerbedingung zurück.

### Eingabeformat

RS    [ <Rücksetznummer> ]
----------------------------

<Rücksetznummer>	Legt die Inhalte für den Rücksetzvorgang fest.
0	: Fehlermeldung abbrechen und Programm zurücksetzen
1	: Zähler zurücksetzen
2	: Batterie-Timer zurücksetzen
3	: Alle Programme und Positionen löschen (gleiche Wirkung wie NW-Befehl)
4	: Nullpunkteinstellung zurücksetzen

### Erläuterung

- Dieser Befehl stellt die Fehlerbedingung im Fehlermodus zurück, schaltet die Servomotoren von AUS auf EIN und bewirkt, daß das Programm an den Programmbeginn zurückspringt.
- Die Fehlerbedingung kann nicht zurückgesetzt werden, wenn eine der Achsen die zugehörige Software-Beschränkung überschreitet.
- Die Datenausgabe bleibt während der Rückstellung einer Fehlermeldung unverändert aufrecht erhalten.

### Programmbeispiel (BASIC-Befehle)

10	OPEN "COM1 :E83"    AS#1	RS232C-Kommunikationsdatei wird über den PC mittels BASIC-Befehl geöffnet
20	PRINT #1,"MO 1000"	Es ist aufgrund einer falschen Werteangabe (Positionnummer > 999) eine Fehlermeldung aufgetreten
30	PRINT #1,"RS"	Fehlermeldung abbrechen
40	END	BASIC-Programmende

## 5.2.76 RT (Return)

### Funktion: Rücksprung zum Hauptprogramm

Schließt ein Unterprogramm ab und springt zum Hauptprogramm zurück.

### Eingabeformat

RT    [ <Zeilennummer> ]
--------------------------

<b>&lt;Zeilennummer&gt;</b>	Legt die Zeilennummer für den Programmsprung fest. $1 \leq \text{Zeilennummer} \leq 9999$ (Bei fehlender Angabe der Zeilennummer erfolgt ein Rücksprung zur dem GS-Befehl folgenden Programmzeile.)
-----------------------------	---

### Erläuterung

- Dieser Befehl schließt das über den GS-Befehl aufgerufene Unterprogramm ab und bewirkt anschließend einen Rücksprung zum Hauptprogramm.
- Wurde der entsprechende GS-Befehl vorher nicht programmiert, tritt eine entsprechende Fehlermeldung auf.
- Bei Angabe einer Zeilennummer springt das Programm nach Ausführung des RT-Befehls zur festgelegten Zeilennummer im Hauptprogramm.

### Programmbeispiel

Siehe GS-Befehl.

## 5.2.77 SC (Set Counter)

### Funktion: Zählerwert einstellen

Lädt einen Wert in den festgelegten Zähler.

### Eingabeformat

SC <Zählernummer/Zeichenkettennummer>, [<Einstellwert/Zeichenkettenwert>]

<Zählernummer>	Legt den Zähler fest. $1 \leq \text{Zählernummer} \leq 99$
<Zeichenkettennummer>	Legt die Zeichenkette fest. Das führende Zeichen ist „\$“. $\$1 \leq \text{Zeichenkettennummer} \leq \$99$
<Einstellwert>	Legt den Einstellwert für den Zähler fest. $-32\,768 \leq \text{Einstellwert (dezimal)} \leq 32\,767$ (0 für Standardwert) $\&8000 \leq \text{Einstellwert (hexadezimal)} \leq \&7FFF$
<Zeichenkettenwert>	Legt den Zeichenkettenwert fest. Erlaubte Zeichen:           Zahlen (0 bis 9) Buchstaben (A bis Z) Symbole (!@#, u.s.w.)  Nicht erlaubte Zeichen:    “ Anzahl der Zeichen:        120 inklusive der Zeilennummer und dem SC-Befehl

### Erläuterung

<Bei Angabe einer Zählernummer>

- Werkseitig sind alle Zähler auf 0 voreingestellt.
- Dieser Befehl kann zum Zählen von Arbeitsgegenständen oder Arbeitsvorgängen und zum Einstellen der Anzahl der Gitterpunkte bei Palettenarbeit eingesetzt werden.
- Der Zählerwert kann mit den entsprechenden Befehlen verändert, verglichen oder gelesen werden (siehe Befehle IC, INP, DC, CP, CR, CL, AN, OR und XO).
- Der Zählerwert bleibt bei der Ausführung des RS-, NW- oder ED-Befehls unverändert erhalten.
- Die Inhalte der Zähler werden nach Ausschalten der Gerätespannung batteriegepuffert abgespeichert.

<Bei Angabe einer Zeichenkettennummer>

- Die Zeichenkette muß mit Anführungszeichen angegeben werden. Bei Angabe der Zeichenkette ABC muß “ABC” eingegeben werden.
- Zeichenketten können mit entsprechenden Befehlen verarbeitet, verglichen und gelesen werden (siehe Befehle CP, CR, CL, EQ, NE, LG, SM, und INP).
- Der Wert der Zeichenkette bleibt bei der Ausführung des RS-, NW oder ED-Befehls unverändert.
- Der Wert bleibt durch die Batteriepufferung auch nach Ausschalten der Gerätespannung erhalten.

**Programmbeispiel (MOVEMASTER-Befehle)**

10	SC	21,10	Wert 10 in den Zähler 21 laden
20	IC	21	Wert des Zählers 21 um 1 erhöhen
30	CP	21	Einstellwert des Zählers 21 in das interne Register laden
40	DR		Wert des internen Registers über die RS232C-Schnittstelle ausgeben
50	SC	\$5,"OK"	Zeichenkette "OK" in in Zeichenkette Nummer 5 laden
60	CP	\$5	Zeichenkette Nummer 5 in Zeichenkettenregister laden
70	EQ	\$10,200	Sprung zur Zeile 200, wenn die Daten des Zeichenkettenregisters und der Zeichenkette Nummer 10 gleich sind



## 5.2.78 SD (Speed Define)

### Funktion: absolute Geschwindigkeit definieren

Definiert die absolute Verfahrgeschwindigkeit, die Zeitkonstante für den Bewegungsablauf, die absolute Beschleunigungs-/Abbremszeit und den Einstellmodus für die kontinuierliche gleichmäßige Verfahrbewegung.

### Eingabeformat

```
SD  <Verfahrgeschwindigkeit> [ , <Zeitkonstante>
    [ , <Beschleunigungszeit> [ , <Abbremszeit>
    [ , <CNT-Einstellung>]]]
```

<Verfahrgeschwindigkeit>	Legt die Verfahrgeschwindigkeit für Linear- oder Kreis-Interpolation fest. $0,01 \leq \text{Verfahrgeschwindigkeit} \leq 650,00$ [mm/s]
<Zeitkonstante>	Legt die Zeitkonstante für den Bewegungsablauf fest. $0 \leq \text{Zeitkonstante} \leq 300$ [ms]
<Beschleunigungszeit>	Legt die Zeit für das Beschleunigen bis zur Maximalgeschwindigkeit fest. $0 \leq \text{Zeitkonstante} \leq 2000$ [ms]
<Abbremszeit>	Legt die Zeit für das Abbremsen von der Maximalgeschwindigkeit fest. $0 \leq \text{Zeitkonstante} \leq 2000$ [ms]
<CNT-Einstellung>	Legt den Einstellmodus für die kontinuierliche gleichmäßige Verfahrbewegung fest. 0 : nicht kontinuierlich    1 : kontinuierlich

### Erläuterung

- Die Geschwindigkeit kann in Schrittweiten von 0,01 mm/s angegeben werden. Geben Sie für z. B. 20,05 mm/s den Wert 20.05 an.
- Die Zeitkonstante kann in Schrittweiten von 1 ms angegeben werden.
- Mit dem SD-Befehl kann die Verfahrgeschwindigkeit (oder Winkelgeschwindigkeit) der Handspitze für Linear- oder Kreis-Interpolation absolut und in kleineren Schrittweiten als mit dem SP-Befehl definiert werden.
- Die Einstellung eines großen Wertes für die Zeitkonstante bewirkt eine gleichmäßigere und ruhigere Roboterbewegung.
- Die mit dem SD-Befehl eingestellte Geschwindigkeit bleibt so lange gültig, bis über den SD-Befehl ein neuer Wert eingestellt wird. Es werden die voreingestellten Werte wirksam, wenn Sie die Zeitkonstante und die Beschleunigungs-/Abbremszeit nicht angeben.
- Während der Linear- oder Kreis-Interpolation kann bei bestimmten, über den SD-Befehl eingestellten Geschwindigkeiten eine Fehlermeldung auftreten, weil die Maximalgeschwindigkeit eines Gelenkes überschritten wurde. Stellen Sie in diesem Fall die Geschwindigkeit auf einen kleineren Wert ein.
- Beim Einschalten der Spannungsversorgung wird die Verfahrgeschwindigkeit auf den Initialisierungswert von 63,3 mm/s eingestellt.

- Nach Freigeben der CNT-Einstellung wird die Roboterbewegung so lange kontinuierlich und gleichmäßig verlaufen (ohne Beschleunigung oder Abbremsung), bis die CNT-Einstellung über den SD- oder SP-Befehl wieder gesperrt wird (Bewegungsverlauf). An den Startpunkten wird jedoch der Roboter beschleunigen und an den Endpunkten abbremsen. Das gleiche gilt auch für durch Timer ausgelöste Stopp-Vorgänge oder Eingabebefehle während des Bewegungsverlaufs.
- Die Beschleunigungszeit ist die Zeit, die dem Roboter höchstens zum Erreichen der maximalen Geschwindigkeit zur Verfügung steht. Dementsprechend gilt, daß die tatsächliche Beschleunigungszeit kleiner als der festgelegte Wert ist, wenn die Verfahrgeschwindigkeit nicht die maximale Geschwindigkeit erreicht. (Das gleiche gilt umgekehrt für die Abbremszeit.)

#### Programmbeispiel (MOVEMASTER-Befehle)

10	SP	15	Verfahrgeschwindigkeit auf den Wert 15 einstellen
20	MS	1	Position 1 mittels Linear-Interpolation anfahren
30	SD	100	Geschwindigkeit auf den Wert 100 mm/s einstellen
40	MS	2	Position 2 mittels Linear-Interpolation anfahren (100 mm/s)
50	MO	3	Position 3 mittels Gelenk-Interpolation anfahren
60	MS	4	Position 4 mittels Linear-Interpolation anfahren (100 mm/s)
70	ED		Programmende

## 5.2.79 SF (Shift)

### Funktion: Positionskoordinaten addieren

Addiert die einzelnen Koordinatenwerte der Position (b) zu den entsprechenden Koordinatenwerten der Position (a) und definiert die Position (a) erneut.

### Eingabeformat

SF    <Positionsnummer (a)>, <Positionsnummer (b)>
--

<Positionsnummern>    Legen die Positionen fest.  
 $1 \leq \text{Positionsnummer (a), (b)} \leq 999$

### Erläuterung

- Der Handgreiferzustand (offen/geschlossen) und die Roboterstellungsdaten der Position (a) werden durch die Ausführung des SF-Befehls nicht beeinflusst.
- Es tritt eine Fehlermeldung auf, wenn die Positionen (a) und/oder (b) vorher nicht definiert wurden.
- Dieser Befehl löst keine Roboterbewegung aus.

#### 5 Achsen

#### Programmbeispiel (MOVEMASTER-Befehle)

10	PD	20,0,0,20,0	XYZ-Koordinaten und Winkel der Position 10 definieren
20	HE	10	Aktuelle Position als Position 10 definieren
30	SF	10,20	Zur Position 10 wird der Z-Koordinatenwert (20 mm) von Position 20 addiert
40	MO	10	Position 10 anfahren
50	ED		Programmende

#### 6 Achsen

#### Programmbeispiel (MOVEMASTER-Befehle)

10	PD	20,0,0,20,0,0	XYZ-Koordinaten und Winkel der Position 10 definieren
20	HE	10	Aktuelle Position als Position 10 definieren
30	SF	10,20	Zur Position 10 wird der Z-Koordinatenwert (20 mm) von Position 20 addiert
40	MO	10	Position 10 anfahren
50	ED		Programmende

## 5.2.80 SM (If Smaller)

### Funktion: Datenwertvergleich: <

Bewirkt einen Programmsprung, wenn der Wert des internen Registers kleiner als der festgelegte Wert ist.

### Eingabeformat

SM    <Vergleichswert/Zeichenkettennummer>, <Zeilennummer des Sprungziels>

<Vergleichswert>	Legt den Wert fest, der mit dem Wert des internen Registers verglichen werden soll. $-32\,768 \leq \text{Vergleichswert (dez.)} \leq 32\,767$ $\&8000 \leq \text{Vergleichswert (hex.)} \leq \&7FFF$
<Zeichenkettennummer>	Legt die Zeichenkette fest. Das führende Zeichen ist „\$“. $\$1 \leq \text{Zeichenkettennummer} \leq \$99$
<Zeilennummer des Sprungziels>	Legt die Zeilennummer fest, an die das Programm bei Erfüllung der Bedingung springen soll. $1 \leq \text{Zeilennummer} \leq 9999$

### Erläuterung

<Bei Angabe eines Vergleichswertes>

- Dieser Befehl bewirkt einen Programmsprung, wenn die Sprungbedingung über externe Eingabedaten oder durch den internen Zählerwert gegeben ist.
- Wenn der Wert des internen Registers kleiner als der Vergleichswert ist (d. h. wenn die Bedingung erfüllt ist), springt das Programm zur festgelegten Zeilennummer. Andernfalls (d. h. wenn die Bedingung nicht erfüllt ist) wird der Programmablauf ohne Sprung fortgesetzt.
- Ein Wert kann in das interne Register entweder durch Ausführung des Eingabebefehls für externe Eingabedaten (siehe ID-Befehl) oder durch Ausführung des Zählervergleichsbefehls (siehe CP-Befehl) geladen werden. Um einen bedingten Sprung zu ermöglichen, ist es erforderlich, daß einer der beiden Befehle vor dem SM-Befehl ausgeführt wird.
- Der Vergleichswert kann entweder dezimal oder hexadezimal definiert werden. Bei einem hexadezimalen Wert muß vor der Zahl das Zeichen „&“ stehen.
- Es tritt bei der Sprungausführung eine Fehlermeldung auf, wenn die festgelegte Programmzeilennummer nicht existiert.

<Bei Angabe einer Zeichenkettennummer>

- Dieser Befehl bewirkt einen Programmsprung, wenn die Sprungbedingung über externe Eingabedaten oder die Anzahl der Zeichen einer festgelegten Zeichenkette gegeben ist.
- Wenn die Anzahl der Zeichen im Zeichenkettenregister kleiner als die Anzahl der Zeichen der festgelegten Zeichenkette ist (d. h. wenn die Bedingung erfüllt ist), springt das Programm zur festgelegten Zeilennummer. Andernfalls (d. h. wenn die Bedingung nicht erfüllt ist) wird der Programmablauf ohne Sprung fortgesetzt.
- Durch Ausführung des INP-Befehls werden externe Daten in das Zeichenkettenregister gelesen. Der Inhalt einer Zeichenkettennummer wird durch Ausführung des CP-Befehls gelesen. Um einen bedingten Sprung zu ermöglichen, ist es erforderlich, daß einer der beiden Befehle vor dem SM-Befehl ausgeführt wird.
- Es tritt bei der Sprungausführung eine Fehlermeldung auf, wenn die festgelegte Programmzeilennummer nicht existiert.

#### Programmbeispiel (MOVEMASTER-Befehle)

10	ID		Daten vom externen Eingabeport holen
20	SM	10,100	Sprung zur Zeile 100, wenn die Eingabedaten kleiner als 10 sind
30	MS	1	Position 1 mittels Linear-Interpolation anfahren
40	ED		Programmende
100	MO	10	Position 10 anfahren
140	OPN	1,1	RS232C-Schnittstelle öffnen
150	INP	1, ,2	Daten vom RS232C-Port in das Zeichenkettenregister einlesen
160	SM	\$5,200	Sprung zur Zeile 200, wenn die Daten kleiner als die Zeichenkette Nummer 10 sind
200	MO	2	Position 2 anfahren

## 5.2.81 SP (Speed)

### Funktion: Betriebsgeschwindigkeit einstellen

Legt die relative Betriebsgeschwindigkeit, die relative Beschleunigungs-/Abbremszeit und den Einstellmodus für die kontinuierliche gleichmäßige Verfahrbewegung fest.

### Eingabeformat

SP <Geschwindigkeitswert> [, <H/L> [, <CNT-Einstellung>]]

<Geschwindigkeitswert>	Legt die relative Betriebsgeschwindigkeit fest. $0 \leq \text{Geschwindigkeitswert} \leq 30$
<H/L>	Legt die Beschleunigungs-/Abbremszeit fest. H: (high) große Beschleunigung/Abbremsung (max. 0,2 s) L: (low) kleine Beschleunigung/Abbremsung (max. 0,2 s)
<CNT-Einstellung>	Legt den Einstellmodus für die kontinuierliche gleichmäßige Verfahrbewegung fest. 0: nicht kontinuierlich      1: kontinuierlich

### Erläuterung

- Die Betriebsgeschwindigkeit kann in Stufen zwischen 0 und 30 eingestellt werden (31 Abstufungen). Die Beschleunigungs-/Abbremszeit zum Starten und Stoppen kann über 2 Abstufungen eingestellt werden.
- Die Geschwindigkeitswerte ergeben sich bei Gelenk-Interpolation aus dem Verhältnis zum Maximalwert (U/min) für jedes Gelenk und bei Linear-Interpolation aus dem Verhältnis zur Maximalgeschwindigkeit der Handspitze (650 mm/s).
- Für die Beschleunigungs-/Abbremszeit kann ein großer (H) oder kleiner (L) Betrag eingestellt werden.
- Die Beschleunigungszeit ist die Zeit, die dem Roboter höchstens zum Erreichen der maximalen Geschwindigkeit zur Verfügung steht. Dementsprechend gilt, daß die tatsächliche Beschleunigungszeit kleiner als der festgelegte Wert ist, wenn die Verfahrsgeschwindigkeit nicht die maximale Geschwindigkeit erreicht. (Das gleiche gilt umgekehrt für die Abbremszeit.)
- Bei Linear-Interpolation wird die Handspitze mit der über den Werkzeugbefehl festgelegten konstanten Geschwindigkeit bewegt. In diesem Fall kann es zu einer Fehlermeldung kommen, wenn ein Gelenk die entsprechende Maximalgeschwindigkeit überschreitet. Wenn die Bewegung eines Positionswinkels (Drehwinkel A, B oder C) größer als die Bewegung entlang einer Strecke (X, Y oder Z) ist, bewegt sich der Roboter entsprechend der Winkelgeschwindigkeit. Mit dem SD-Befehl kann die Geschwindigkeit in kleineren Abstufungen eingestellt werden.
- Die einmal festgelegte Geschwindigkeit und Beschleunigungs-/Abbremszeit bleibt so lange gültig, bis ein neuer Wert eingestellt wird. Beim Einschalten der Spannungsversorgung wird die Verfahrsgeschwindigkeit auf den Initialisierungswert von SP 12,H (63,3 mm/s) eingestellt. Die Beschleunigungs-/Abbremszeit wird auf den zuletzt gewählten Wert eingestellt, falls keine andere Festlegung erfolgte.

- Nach Freigeben der CNT-Einstellung wird die Roboterbewegung so lange kontinuierlich und gleichmäßig verlaufen (ohne Beschleunigung oder Abbremsung), bis die CNT-Einstellung über den SP- oder SD-Befehl wieder gesperrt wird (Bewegungsverlauf). An den Startpunkten wird jedoch der Roboter beschleunigen und an den Endpunkten abbremsen. Das gleiche gilt auch für durch Timer ausgelöste Stopp-Vorgänge oder Eingabebefehle während des Bewegungsverlaufs.

#### Programmbeispiel (MOVEMASTER-Befehle)

10	SP	8	Verfahrgeschwindigkeit auf den Wert 8 einstellen
20	MO	5	Position 5 mittels Gelenk-Interpolation anfahren
30	SP	10	Verfahrgeschwindigkeit auf den Wert 10 einstellen
40	MS	7	Position 7 mittels Linear-Interpolation anfahren
50	ED		Programmende

#### Beziehung zwischen Geschwindigkeitswert und Verfahrgeschwindigkeit

Einstellwert	Gelenk-Interpolation [%]	Linear-Interpolation [mm/s]	Einstellwert	Gelenk-Interpolation [%]	Linear-Interpolation [mm/s]
0	0,1	0,2	16	19,0	123,7
1	0,4	2,7	17	22,2	144,5
2	0,6	3,8	18	25,9	168,1
3	0,8	5,3	19	29,8	193,8
4	1,1	7,3	20	34,2	222,0
5	1,5	9,8	21	40,7	264,8
6	2,0	13,3	22	47,3	307,6
7	2,7	17,8	23	53,9	350,4
8	3,7	23,8	24	60,5	393,2
9	4,9	31,7	25	67,1	436,0
10	6,5	42,4	26	73,7	478,8
11	8,2	53,1	27	80,2	521,6
12	9,7	63,3	28	86,8	564,4
13	11,6	75,3	29	93,4	607,2
14	13,7	89,2	30	100,0	650,0
15	16,2	105,2			


**Tab. 5-6:** Beziehung zwischen Geschwindigkeitswert und Verfahrgeschwindigkeit

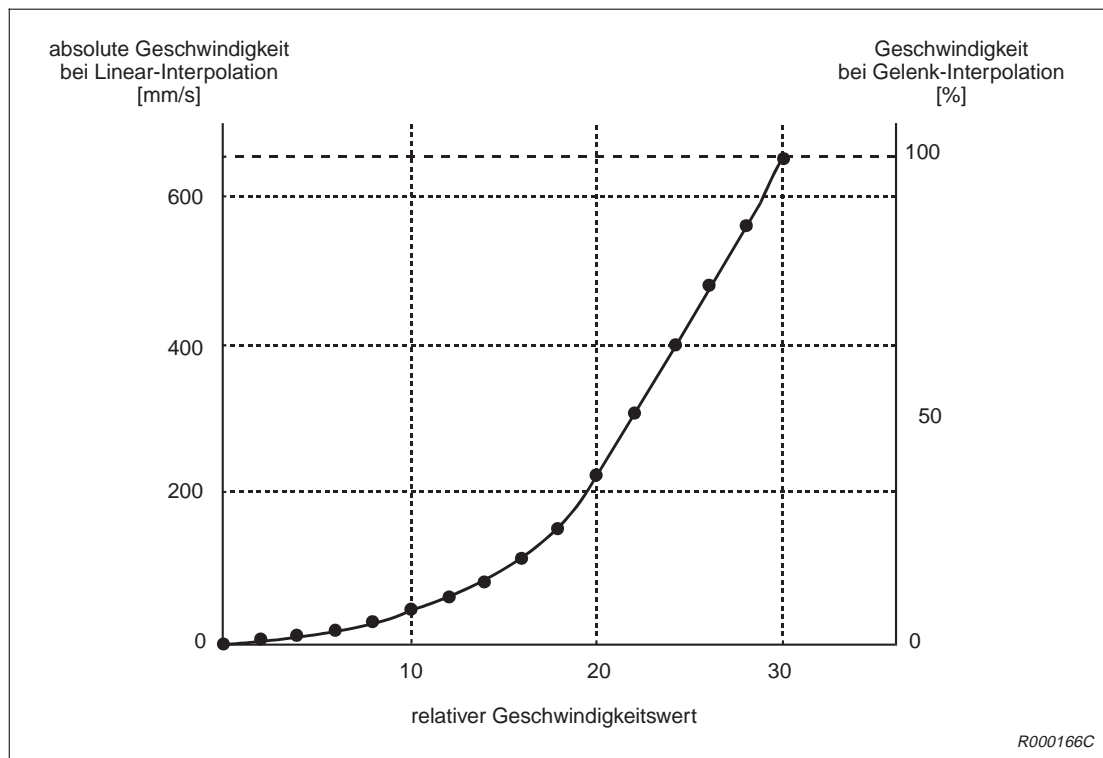
#### Anmerkungen

- Der Roboter bewegt sich entsprechend der Winkelgeschwindigkeit der Position, wenn folgende Bedingung erfüllt ist:

$$\text{Betrag (Drehwinkel A, B, C)} \geq \text{Betrag (Strecke in X-, Y-, Z-Richtung) bei Linear- und Gelenk-Interpolation}$$

(Die Winkelgeschwindigkeit in Grad/Sekunde ist gleich der Streckengeschwindigkeit in Millimeter/Sekunde dividiert durch den Wert 2,12.)

- Die schraffierte Fläche  in der obigen Tabelle kennzeichnet den Initialisierungswert, welcher beim Einschalten der Spannungsversorgung automatisch eingestellt wird.



**Abb. 5-23:** Diagrammdarstellung der Beziehung zwischen Geschwindigkeitswert und Verfahrensgeschwindigkeit



## 5.2.82 STR ♦ (Step Read)

### Funktion: Programmschritt lesen

Liest die Inhalte des festgelegten oder des aktuellen Programmschritts (über die RS232C-Schnittstelle).

### Eingabeformat

```
STR <Schrittnummer>
```

<Schrittnummer>      Legt die zu lesende Schrittnummer fest.  
 $0 \leq \text{Schrittnummer} \leq 9999$   
 (Bei fehlender Schrittnummer werden die Daten  
 des aktuellen Programmschritts ausgegeben.)

### Erläuterung

- Dieser Befehl gibt die Inhalte des festgelegten oder des aktuellen Programmschritts über die RS232C-Schnittstelle aus. Es werden die Inhalte des aktuellen Programmschritts ausgegeben, wenn die Angabe der Schrittnummer fehlt oder der Wert 0 festgelegt wurde.
- Die Daten werden im ASCII-Code mit nachfolgendem Format ausgegeben.
  - 1.) Aufzeichnungsmethode über Teaching Box  
     : (Doppelpunkt) , Programmbefehl
  - 2.) Programmierung über MOVEMASTER-Befehle  
     Zeilennummer, Programmbefehl
- Das Abschlußzeichen der Ausgabedaten ist ein „Carriage Return“ (CR, hex. 0D). Deshalb ist es beim Einsatz eines PCs erforderlich, daß die serielle Datenkette bis zur Stelle „hex. 0D“ empfangen wird. Der Befehl „LINE INPUT #“ ist der entsprechende BASIC-Befehl zum Einlesen der Daten.
- Es wird der hexadezimale Wert 0D zurückgesendet, wenn eine nicht definierte Schrittnummer festgelegt wurde.
- Es werden die Inhalte des aktuellen Programmschritts ausgegeben, wenn die Angabe der Schrittnummer fehlt oder der Wert 0 festgelegt wurde.
- Es wird für jede Koordinate der Wert 0 ausgegeben, wenn eine Position gelesen wird, die im Programm nicht eingesetzt wurde und auch nicht definiert ist.  
 Es wird für jede Koordinate der Wert 0.00 ausgegeben, wenn eine Position gelesen wird, die zwar im Programm eingesetzt wurde, aber nicht definiert ist.
- Wenn nach Ausführung eines Verfahrenbefehls eine Fehlermeldung aufgetreten ist, können Sie sich mit der Hilfe des STR-Befehls die Inhalte des aktuellen Programmschritts über einen PC anzeigen lassen. Lassen Sie hierfür die Angabe der Schrittnummer weg oder geben Sie den Wert 0 an.

**5 Achsen****Programmbeispiel (BASIC-Befehle)**

```
10 OPEN "COM1:E83" AS#1
```

RS232C-Kommunikationsdatei wird über den PC mittels BASIC-Befehl geöffnet

```
20 INPUT "Schrittnummer = " : J$
```

Schrittnummer eingeben

```
30 PRINT #1,"STR"+J$
```

STR-Befehl und Schrittnummer zum Steuergerät übertragen

```
40 LINE INPUT #1,A$
```

Übertragene Daten in A\$ speichern

```
50 PRINT A$
```

Inhalte von A\$ auf dem Bildschirm anzeigen

```
60 END
```

Ende des BASIC-Programms

```
RUN
```

BASIC-Programm starten

```
Schrittnummer = ? 2
```

Schrittnummer (2) eingeben

```
MPC 0,227.85,371.92,581.68,102.83,30.85,R,A,C
```

Ausgabe der Inhalte des Programmschritts Nr. 2

**6 Achsen****Programmbeispiel (BASIC-Befehle)**

```
10 OPEN "COM1:E83" AS#1
```

RS232C-Kommunikationsdatei wird über den PC mittels BASIC-Befehl geöffnet

```
20 INPUT "Schrittnummer = " : J$
```

Schrittnummer eingeben

```
30 PRINT #1,"STR"+J$
```

STR-Befehl und Schrittnummer zum Steuergerät übertragen

```
40 LINE INPUT #1,A$
```

Übertragene Daten in A\$ speichern

```
50 PRINT A$
```

Inhalte von A\$ auf dem Bildschirm anzeigen

```
60 END
```

Ende des BASIC-Programms

```
RUN
```

BASIC-Programm starten

```
Schrittnummer = ? 2
```

Schrittnummer (2) eingeben

```
MPC 0,227.85,371.92,581.68,-60.71,102.83,30.85,R,A,N,C
```

Ausgabe der Inhalte des Programmschritts Nr. 2

## 5.2.83 SUB (Subtraction)

### Funktion: Subtraktion

Subtrahiert einen Wert oder den Inhalt eines festgelegten Zählers vom Wert des internen Registers.

### Eingabeformat

```
SUB <Wert | @Zählernummer>
```

<Wert>                      Legt den zu subtrahierenden hexadezimalen oder dezimalen Wert fest.  
 $-32\,768 \leq \text{dezimaler Wert} \leq 32\,767$   
 $\&8000 \leq \text{hexadezimaler Wert} \leq \&7FFF$

<Zählernummer>            Legt den Zähler fest.  
 $1 \leq \text{Zählernummer (ganzzahlig)} \leq 99$

### Erläuterung

- Subtrahiert einen dezimalen oder hexadezimalen Wert, oder den Inhalt eines festgelegten Zählers vom Wert des internen Registers.

### Beispiele ▾

SUB 10      Subtrahiert den dezimalen Wert 10 vom Wert des internen Registers  
 SUB &FF    Subtrahiert den hexadezimalen Wert &FF10 vom Wert des internen Registers  
 SUB @5     Subtrahiert den Inhalt des Zählers 5 vom Wert des internen Registers

### Programmbeispiel (MOVEMASTER-Befehle)

```
10  ID          Lädt Eingabedaten in das interne Register
20  SUB 10      Subtrahiert den Wert 10 von den eingelesenen Daten
30  CL 21       Daten aus dem internen Register in den Zähler 21 laden
```

## 5.2.84 TB (Test Bit)

### Funktion: Bitstatus überprüfen

Bewirkt einen Programmsprung in Abhängigkeit vom Bitstatus des festgelegten Bits im internen Register.

### Eingabeformat

TB    [<+/->] <Bitnummer>, <Zeilennummer des Sprungziels>

<+/->	Legt die Sprungbedingung für das Vergleichsbit fest. + : Bit ist eingeschaltet (1) - : Bit ist ausgeschaltet (0)
<Bitnummer>	Legt die Bitnummer des internen Registers fest. $0 \leq \text{Bitnummer} \leq 15$
<Zeilennummer des Sprungziels>	Legt die Zeilennummer fest, zu der das Programm bei Erfüllung der Bedingung springen soll. $1 \leq \text{Zeilennummer} \leq 9999$

### Erläuterung

- Dieser Befehl bewirkt einen Programmsprung, wenn die Sprungbedingung über externe Eingabedaten oder durch den internen Zählerwert gegeben ist.
- Das Programm springt zur festgelegten Zeilennummer, wenn die Sprungbedingung erfüllt ist. Hierfür muß das festgelegte Bit des internen Registers entweder eingeschaltet (Sprungbedingung: +) oder ausgeschaltet (Sprungbedingung: -) sein.
- Ein Wert kann entweder durch Ausführung des Eingabebefehls für externe Eingabedaten (siehe ID-Befehl) oder durch Ausführung des Zählervergleichsbefehls (siehe CP-Befehl) in das interne Register geladen werden. Um einen bedingten Sprung zu ermöglichen, ist es erforderlich, daß einer der beiden Befehle vor dem TB-Befehl ausgeführt wird.
- Existiert die festgelegte Programmzeilennummer nicht, tritt bei der Sprungausführung eine Fehlermeldung auf.

### Programmbeispiel (MOVEMASTER-Befehle)

10	ID		Daten vom externen Eingabeport holen
20	TB	+1,100	Sprung zur Zeile 100, wenn das Bit Nr.1 der Eingabedaten eingeschaltet ist
30	MS	1	Position 1 mittels Linear-Interpolation anfahren
40	ED		Programmende
100	MO	10	Position 10 anfahren

## 5.2.85 TBD (Test Bit Direct)

### Funktion: Bitstatus direkt überprüfen

Bewirkt einen Programmsprung in Abhängigkeit vom Bitstatus des festgelegten Bits der externen Eingabedaten.

### Eingabeformat

```
TBD [<+/->] <Eingangs-Bitnummer>,
      <Zeilennummer des Sprungziels>
```

<b>&lt;+/-&gt;</b>	Legt die Sprungbedingung für das Vergleichsbit fest. + : Bit ist eingeschaltet (1) - : Bit ist ausgeschaltet (0)
<b>&lt;Bitnummer&gt;</b>	Legt die Bitnummer der externen Eingabedaten fest. $0 \leq \text{Bitnummer} \leq 32\,767$
<b>&lt;Zeilennummer des Sprungziels&gt;</b>	Legt die Zeilennummer fest, an der das Programm bei Erfüllung der Bedingung springen soll. $1 \leq \text{Zeilennummer} \leq 9999$

### Erläuterung

- Dieser Befehl bewirkt einen Programmsprung, wenn die Sprungbedingung über externe Eingabedaten gegeben ist. Die Eingabedaten werden direkt eingelesen und verglichen (keine Zwischenabspeicherung im internen Register).
- Das Programm springt zur festgelegten Zeilennummer, wenn die Sprungbedingung erfüllt ist. Hierfür muß das festgelegte Bit der externen Eingabedaten entweder eingeschaltet (Sprungbedingung: +) oder ausgeschaltet (Sprungbedingung: -) sein.
- Eine vorherige Ausführung des Eingabebefehls (ID-Befehl) ist nicht erforderlich. Der Wert des internen Registers bleibt während der Ausführung des TBD-Befehls unverändert erhalten.
- Es tritt bei der Sprungausführung eine Fehlermeldung auf, wenn die festgelegte Programmzeilennummer nicht existiert.

### Programmbeispiel (MOVEMASTER-Befehle)

10	TBD	+19,100	Sprung zur Zeile 100, wenn das Bit Nr.19 der Eingabedaten eingeschaltet ist
20	MS	1	Position 1 mittels Linear-Interpolation anfahren
30	ED		Programmende
100	MO	10	Position 10 anfahren

## 5.2.86 TI (Timer)

### Funktion: Zeitglied

Stoppt die Roboterbewegung für die festgelegte Zeitdauer.

### Eingabeformat

TI   <Zählerwert für Timer>
-----------------------------

<Zählerwert für Timer>

Legt die Zeitdauer für den Timer fest.

$0 \leq \text{Zählerwert für Timer} \leq 32\,767$  (in Einheiten von 0,1s)

### Erläuterung

- Dieser Befehl stoppt die Roboterbewegung für die folgende Zeitdauer:  
festgelegter Zählerwert x 0,1 s = Zeitdauer [s]      (max. 32 76,7 s)
- Dieser Befehl kann zum Einfügen einer Zeitverzögerung benutzt werden (z. B. vor dem Schließen oder Öffnen der Hand zum Greifen eines Arbeitsgegenstands).
- Der Standardwert ist 0.

### Programmbeispiel (MOVEMASTER-Befehle)

10	MO	1,0	Position 1 anfahren
20	TI	5	0,5 Sekunden warten
30	GC		Hand schließen
40	TI	10	1,0 Sekunden warten
50	MO	2	Position 2 anfahren
60	ED		Programmende

## 5.2.87 TL (Tool)

### Funktion: Werkzeuglänge einstellen

Legt den Abstand zwischen der Befestigungsoberfläche für die Hand und der Handspitze fest.

### Eingabeformat

TL    <Werkzeuglänge>
-----------------------

<Werkzeuglänge>

Legt den Abstand zwischen der Befestigungsoberfläche für die Hand und der Handspitze fest.

$0 \leq \text{Werkzeuglänge} \leq 300,00 \text{ [mm]}$       (0 für Standardwert)

### Erläuterung

- Die Werkzeuglänge kann in Schrittweiten von 0,01 mm angegeben werden. Geben Sie beispielsweise für 200,05 mm den Wert 200 05 an.
- Die einmal festgelegte Werkzeuglänge bleibt so lange gültig, bis ein neuer Wert eingestellt wird. Bei Änderung der Werkzeuglänge wird dementsprechend auch die aktuelle Position geändert. Die Positionsänderung löst aber keine Roboterbewegung aus.
- Die Grundeinstellung für die Werkzeuglänge ist 107 mm.
- Weil der über den TL-Befehl definierte Punkt als Basis für die Berechnung der aktuellen Position im XYZ-JOG-Betrieb und bei Befehlen, die XYZ-Koordinaten enthalten, eingesetzt wird, ist es erforderlich, daß die genaue Länge des verwendeten Werkzeuges angegeben wird.
- Legen Sie vor dem Starten eines Programms am Programmstart genau die gleiche Werkzeuglänge fest, die vorher auch beim Anfahren der Positionen mit der Teaching Box verwendet wurde.

### Programmbeispiel (MOVEMASTER-Befehle)

10	TL	120	Werkzeuglänge auf den Wert 120 mm einstellen
20	HE	1	Aktuelle Position als Position 1 definieren
30	TL	100	Werkzeuglänge auf den neuen Wert 100 mm einstellen
40	MO	1	Position 1 anfahren (20 mm in Werkzeuglängsrichtung verfahren)
50	ED		Programmende

## 5.2.88 VR (Version Read)

### Funktion: Software-Version lesen

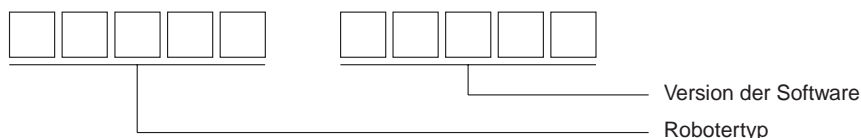
Liest die Software-Versionsnummer des System-ROMs (über die RS232C-Schnittstelle).

### Eingabeformat

VR

### Erläuterung

- Dieser Befehl gibt die Software-Versionsnummer des im Steuergerät eingebauten System-ROMs über die RS232C-Schnittstelle aus.
- Die Daten werden im ASCII-Code ausgegeben. Die ausgegebenen Daten haben folgende Bedeutung:



- Das Abschlußzeichen der Ausgabedaten ist ein „Carriage Return“ (CR, hex. 0D). Deshalb ist es beim Einsatz eines PCs erforderlich, daß die serielle Datenkette bis zur Stelle „hex. 0D“ empfangen wird. Der Befehl „LINE INPUT #“ ist der entsprechende BASIC-Befehl zum Einlesen der Daten.

### 5 Achsen

#### Programmbeispiel (BASIC-Befehle)

```
10 OPEN "COM1 :E83" AS#1
```

RS232C-Kommunikationsdatei wird über den PC mittels BASIC-Befehl geöffnet

```
20 PRINT #1,"VR"
```

VR-Befehl zum Steuergerät übertragen

```
30 LINE INPUT #1,A$
```

Übertragene Daten in A\$ speichern

```
40 PRINT "Software-Version = ":A$
```

Inhalte von A\$ auf dem Bildschirm anzeigen

```
50 END
```

Ende des BASIC-Programms

```
RUN
```

BASIC-Programm starten

```
Software-Version = RV-E5NJM Ver. A1
```

Ausgabe des Robotertyps und der Software-Version

### 6 Achsen

#### Programmbeispiel (BASIC-Befehle)

```
10 OPEN "COM1 :E83" AS#1
```

RS232C-Kommunikationsdatei wird über den PC mittels BASIC-Befehl geöffnet

```
20 PRINT #1,"VR"
```

VR-Befehl zum Steuergerät übertragen

```
30 LINE INPUT #1,A$
```

Übertragene Daten in A\$ speichern

```
40 PRINT "Software-Version = ":A$
```

Inhalte von A\$ auf dem Bildschirm anzeigen

```
50 END
```

Ende des BASIC-Programms

```
RUN
```

BASIC-Programm starten

```
Software-Version = RV-E4NM Ver. A1
```

Ausgabe des Robotertyps und der Software-Version



## 5.2.89 WH (Where)

### Funktion: aktuelle Positionskoordinaten lesen

Liest die Koordinaten der aktuellen Position und den Handgreiferzustand.

### Eingabeformat

WH
----

### Erläuterung

- Dieser Befehl gibt die Koordinaten der aktuellen Handspitzenposition, welche über die Werkzeuglänge (siehe TL-Befehl) festgelegt wurde, und den Handgreiferzustand über die RS232C-Schnittstelle aus.
- Die Daten werden im ASCII-Code ausgegeben.  
Ausgabeformat:  
 RV-E5NJM : XYZ-Koordinaten, Drehwinkel (A, B), R/L, A/B, O/C  
 RV-E4NM/E4NC : XYZ-Koordinaten, Drehwinkel (A, B, C), R/L, A/B, N/F, O/C  
 Die kleinste Schrittweite für die Ausgabewerte ist 0,01 mm oder 0,01°. Es wird z. B. 20.01 für 20,01 mm ausgegeben.
- Das Abschlußzeichen der Ausgabedaten ist ein „Carriage Return“ (CR, hex. 0D). Deshalb ist es beim Einsatz eines PCs erforderlich, daß die serielle Datenkette bis zur Stelle „hex. 0D“ empfangen wird. Der Befehl „LINE INPUT #“ ist der entsprechende BASIC-Befehl zum Einlesen der Daten.

### 5 Achsen

### Programmbeispiel (BASIC-Befehle)

10 OPEN "COM1 :E83" AS#1	RS232C-Kommunikationsdatei wird über den PC mittels BASIC-Befehl geöffnet
20 PRINT #1,"WH"	WH-Befehl zum Steuergerät übertragen
30 LINE INPUT #1,A\$	Übertragene Daten in A\$ speichern
40 PRINT "Aktuelle Koordinaten = "A\$	Inhalte von A\$ auf dem Bildschirm anzeigen
50 END	Ende des BASIC-Programms
RUN	BASIC-Programm starten
Aktuelle Koordinaten = 10.00, 380.00,300.00,50.00,40.00,R,A,C	Ausgabe der Koordinatenwerte der aktuellen Position

**6 Achsen****Programmbeispiel (BASIC-Befehle)**

10	OPEN "COM1 :E83" AS#1	RS232C-Kommunikationsdatei wird über den PC mittels BASIC-Befehl geöffnet
20	PRINT #1,"WH"	WH-Befehl zum Steuergerät übertragen
30	LINE INPUT #1,A\$	Übertragene Daten in A\$ speichern
40	PRINT "Aktuelle Koordinaten = "A\$	Inhalte von A\$ auf dem Bildschirm anzeigen
50	END	Ende des BASIC-Programms
RUN		BASIC-Programm starten
Aktuelle Koordinaten = 10.00, 380.00,300.00,-70.00,50.00,40.00,R,A,N,C		Ausgabe der Koordinatenwerte der aktuellen Position

## 5.2.90 WT (What Tool)

### Funktion: Werkzeuglänge lesen

Liest die aktuell eingerichtete Werkzeuglänge (über die RS232C-Schnittstelle).

### Eingabeformat

WT
----

### Erläuterung

- Dieser Befehl gibt die aktuell eingerichtete Werkzeuglänge über die RS232C-Schnittstelle aus.
- Die Daten werden im ASCII-Code ausgegeben. Die kleinste Schrittweite für die Ausgabewerte ist 0,01 mm. Es wird z. B. 105.07 für 105,07 mm ausgegeben.
- Alle Roboterbewegungen basieren auf der eingerichteten Werkzeuglänge. Es kann zu Kollisionen mit den umliegenden Gegenständen kommen, wenn eine falsche Werkzeuglänge definiert wurde. Überprüfen Sie mit dem WT-Befehl die aktuell eingerichtete Werkzeuglänge.
- Das Abschlußzeichen der Ausgabedaten ist ein „Carriage Return“ (CR, hex. 0D). Deshalb ist es beim Einsatz eines PCs erforderlich, daß die serielle Datenkette bis zur Stelle „hex. 0D“ empfangen wird. Der Befehl „LINE INPUT #“ ist der entsprechende BASIC-Befehl zum Einlesen der Daten.

### Programmbeispiel (BASIC-Befehle)

10 OPEN "COM1 :E83" AS#1	RS232C-Kommunikationsdatei wird über den PC mittels BASIC-Befehl geöffnet
20 PRINT #1,"WT"	WT-Befehl zum Steuergerät übertragen
30 LINE INPUT #1,A\$	Übertragene Daten in A\$ speichern
40 PRINT "Werkzeuglänge = "A\$	Inhalte von A\$ auf dem Bildschirm anzeigen
50 END	Ende des BASIC-Programms
 RUN	 BASIC-Programm starten
Werkzeuglänge = 105.7	Ausgabe der Werkzeuglänge

## 5.2.91 XO (Exclusive Or)

### Funktion: Exklusiv-ODER-Verknüpfung

Exklusiv-ODER-Verknüpfung des festgelegten Datenwertes mit dem Wert des internen Registers.

### Eingabeformat

XO    <Datenwert>
-------------------

<Datenwert>

Legt den Datenwert fest  
 $-32\,768 \leq \text{Datenwert (dezimal)} \leq 32\,767$   
 $\&8000 \leq \text{Datenwert (hexadezimal)} \leq \&7FFF$   
 @ ≤ Zählernummer ≤ @99

### Erläuterung

- Der festgelegte Datenwert kann dezimal oder hexadezimal definiert werden. Bei einem hexadezimalen Wert muß vor der Zahl das Zeichen „&“ stehen.
- Das Ergebnis wird im internen Register abgespeichert und kann mit den entsprechenden Befehlen verändert, verglichen oder gelesen werden (siehe Befehle EQ, NE, LG, SM, CL, DR, AN und OR).
- Durch Ausführung des XO-Befehls nach einem Eingabebefehl (ID oder IN) besteht die Möglichkeit, von einem externen Gerät nur die erforderlichen Bits der parallelen Eingabedaten zu empfangen.

### Programmbeispiel (MOVEMASTER-Befehle)

10	ID		Daten vom externen Eingabeport holen
20	AN	&000F	Empfängt nur die vier unteren Bits
30	OR	&0007	Daten der vier unteren Bits invertieren
40	CL	21	Obere Daten in den Zähler 12 laden
50	EQ	10,200	Sprung zur Zeile 200, wenn die oberen Daten gleich 10 sind
60	ED		Programm beenden
200	MO	99	Position 99 anfahren

## 5.2.92 ' (Comment)

### Funktion: Kommentar

Ermöglicht dem Programmierer das Schreiben eines Kommentars.

### Eingabeformat

```
' [<Datenkette mit bis zu 120 alphanumerischen Zeichen,
inklusive Zeilennummer und ' (Apostroph)>]
```

### Erläuterung

- Mit diesem Befehl können Sie in das Roboterprogramm einen Kommentar von bis zu 120 alphanumerische Zeichen schreiben (inklusive Zeilennummer und ').
- Verwenden Sie diesen Befehl zur Beschreibung des Programmnamens und des Datums der Programmerstellung oder zur Kennzeichnung eines Unterprogramms. Kommentare sind sehr hilfreich bei einer späteren Programmüberprüfung mit dem LR-Befehl (Programmzeilen lesen).
- Das System ignoriert alle Kommentare bei der Ausführung der Befehle.
- Beinhaltet ein Kommentar mehr als 120 Zeichen, werden die überzähligen Zeichen ignoriert.

### Programmbeispiel (MOVEMASTER-Befehle)

10'	*****	
20'	Beispielprogramm	Gibt die Inhalte des Programms, das
30'	Datum: 10-1-95	Datum der Implementierung und den
40'	Programmiert von MITSUBISHI ELECTRIC	Namen des Programmierers an.
50'	*****	

## 5.3 Beispiele mit MOVEMASTER-Befehlen

### 5.3.1 Allgemeine Hinweise

In den nächsten Beispielen wird erläutert, wie Sie ein Programm mittels Verwendung der MOVEMASTER-Befehle erstellen können.

#### Allgemeine Vorgehensweise bei der Programmerstellung

Nr.	Merkmal	Beschreibung der erforderlichen Arbeitsschritte
1	Arbeitsablauf erstellen	① Erstellen Sie ein Flußdiagramm für den Arbeitsablauf. ② Legen Sie die Roboterarbeiten fest.
2	Programmablauf vorbereiten	① Teilen Sie den gesamten Arbeitsablauf in mehrere Arbeitsabschnitte auf. ② Falls Verzweigungen im Arbeitsablauf auftreten, muß dieser in verschiedene Programme oder Programmabschnitte aufgeteilt werden. ③ Legen Sie die Positionsnummer für jede Arbeitsposition fest. ④ Legen Sie fest, welche Ein-/Ausgangssignale eingesetzt werden sollen. ⑤ Erstellen Sie mit Hilfe der festgelegten Daten ein Flußdiagramm für den Arbeitsablauf.
3	Programmierung	Erstellen Sie mit Hilfe des Flußdiagramms das Programm.

**Tab. 5-7:** Allgemeine Vorgehensweise bei der Programmerstellung

### 5.3.2 1. Beispiel: Arbeitsgegenstand ergreifen und platzieren

#### Beschreibung des Arbeitsablaufs

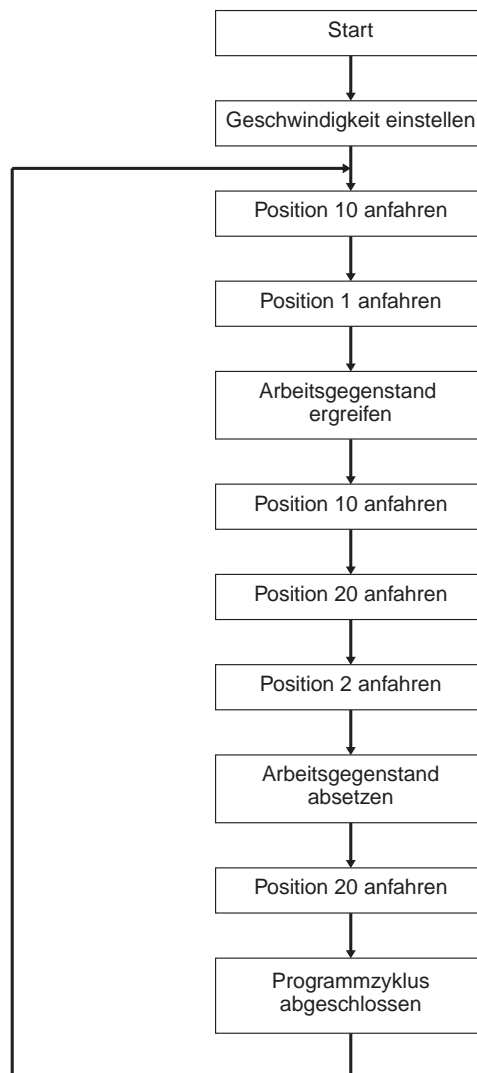
Der Roboter ergreift einen Arbeitsgegenstand und transportiert diesen zu einer anderen Position. Dort wird der Arbeitsgegenstand wieder abgesetzt.

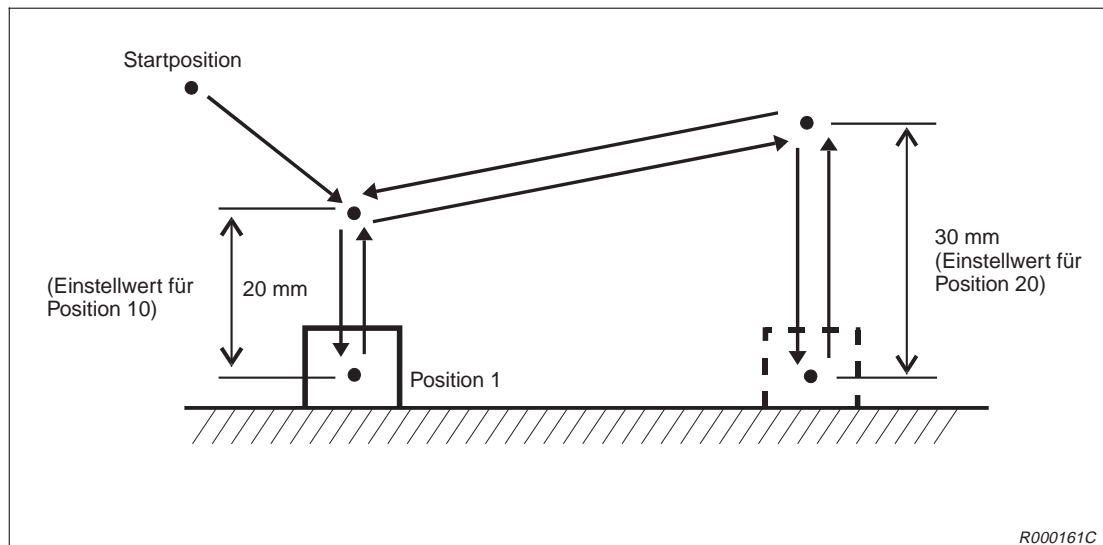
#### Arbeitspositionen

Position	Beschreibung	Eingabe/Festlegung der Position
Position 1	Position zur Ergreifung des Arbeitsgegenstands	über Teaching Box
Position 2	Position zur Platzierung des Arbeitsgegenstands	
Position 10	Position oberhalb von Position 1	über Angabe numerischer Werte
Position 20	Position oberhalb von Position 2	

**Tab. 5-8:** Arbeitspositionen

#### Flußdiagramm des Arbeitsablaufs



**Skizze des Arbeitsablaufs****Abb. 5-24:** Arbeitsgegenstand ergreifen und plazieren**Programmbeispiel (MOVEMASTER-Befehle)**

10	PD	10,0,0,20,0,0,0	Position (Z = 20 mm), die in direkter Linie oberhalb von Position 1 liegt, als Position 10 definieren
20	PD	20,0,0,30,0,0,0	Position (Z = 30 mm), die in direkter Linie oberhalb von Position 2 liegt, als Position 20 definieren
30	SP	17	Geschwindigkeit auf den Wert 17 einstellen
40	MA	1,10,O	Hand öffnen und zur Position 10 fahren, die 20 mm oberhalb von Position 1 liegt
50	MO	1,O	Position 1 zum Ergreifen des Arbeitsgegenstands anfahren
60	GC		Hand schließen und Arbeitsgegenstand ergreifen
70	MA	1,10,C	Eine Position mit geschlossener Hand anfahren, die 20 mm oberhalb von Position 1 liegt
80	MA	2,20,C	Eine Position mit geschlossener Hand anfahren, die 30 mm oberhalb von Position 2 liegt
90	MO	2,C	Position 2 mit geschlossener Hand anfahren
100	GO		Hand zum Absetzen des Werkzeugs öffnen
110	MA	2,20,O	Eine Position mit offener Hand anfahren, die 30 mm oberhalb von Position 2 liegt
120	GT	40	Sprung zur Zeile 40 (Programmzyklus wiederholen)



### 5.3.3 2. Beispiel: Verfahrbewegung über externes Signal unterbrechen

#### Beschreibung des Arbeitsablaufs

Der Roboter ergreift unterschiedlich hohe Arbeitsgegenstände. Deshalb kann die jeweilige Höhe der Greifposition vorher im Roboterprogramm nicht festgelegt werden.

Die aktuelle Höhe der Greifposition wird über die Betätigung eines an der Roboterhand montierten Grenztastschalters bestimmt. Die Robotersteuerung empfängt das Interrupt-Signal des Grenztastschalters über den Handkontrolleingang.

#### Arbeitsposition

Position	Beschreibung	Eingabe/Festlegung der Position
Position 1	Position oberhalb des Arbeitsgegenstands	über Teaching Box

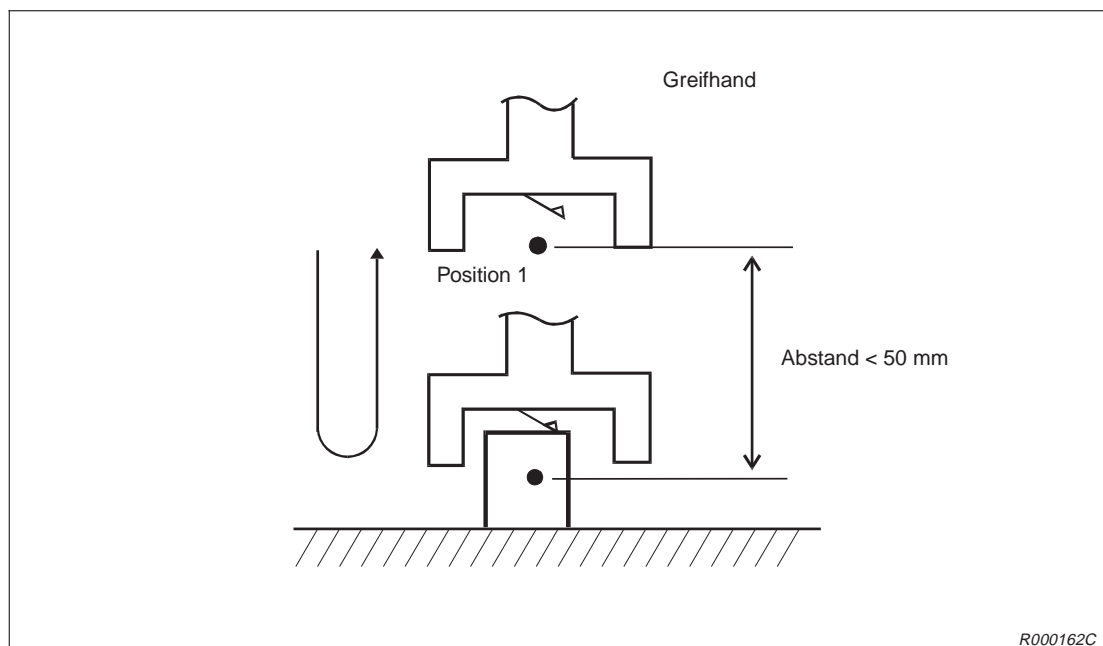
**Tab. 5-9:** Arbeitsposition

#### Eingangssignal

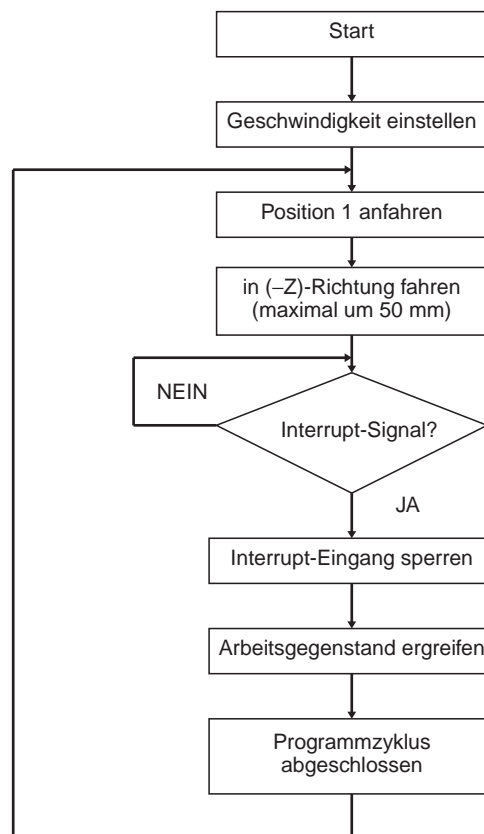
Ein-/Ausgang	Beschreibung	Bitnummer
Eingang	Signal bei Kontakt der Hand mit Arbeitsgegenstand	Bit 900

**Tab. 5-10:** E-/A-Signale

#### Skizze des Arbeitsablaufs



**Abb. 5-25:** Verfahrbewegung über externes Signal unterbrechen

**Flußdiagramm des Arbeitsablaufs****Programmbeispiel (MOVEMASTER-Befehle)**

90	SP	20	Geschwindigkeit auf den Wert 20 einstellen
100	EA	+900, 140	Freigabe der Interrupt-Möglichkeit über Eingangsbit 900
110	MO	1,0	Position 1 anfahren (oberhalb des Arbeitsgegenstands)
120	DS	0,0,-50	In (-Z)-Richtung verfahren (maximal um 50 mm)
130	GT	110	Sprung zur Zeile 110 und damit Rückkehr zu Position 1, wenn kein Arbeitsgegenstand zum Ergreifen vorhanden ist
140	DA	900	Interrupt-Möglichkeit über Eingangsbit 900 sperren
150	GC		Hand schließen und Arbeitsgegenstand ergreifen
160	MO	1,C	Position 1 mit geschlossener Hand anfahren

...

Im obigen Programmbeispiel bewirkt Zeile 120 ein Verfahren des Roboters in (-Z)-Richtung (maximal um 50 mm).

Wenn ein Arbeitsgegenstand vorhanden ist, wird der Grenztastschalter das Interrupt-Signal auslösen, welches über das Eingangsbit 900 von der Robotersteuerung empfangen wird. Das Programm springt zur Zeile 140. Jetzt wird die Interrupt-Möglichkeit wieder gesperrt und die Hand zum Ergreifen des Arbeitsgegenstands geschlossen.

Wenn jedoch kein Arbeitsgegenstand vorhanden ist, d.h. der Grenztastschalter löst das Interrupt-Signal nicht aus, erfolgt in Zeile 130 ein Sprung zur Zeile 110. Der Roboter fährt zur Position 1 zurück, und der Arbeitsablauf wird wiederholt.

### 5.3.4 3. Beispiel: Arbeitsgegenstände palettieren

#### Beschreibung des Arbeitsablaufs

Der Roboter ergreift nacheinander von einer Palette Arbeitsgegenstände und setzt sie in ein Prüfgerät ein. Nach der Überprüfung werden die Arbeitsgegenstände auf einer anderen Palette abgelegt.

#### Arbeitspositionen

Position	Beschreibung	Eingabe/Festlegung der Position
Position 1	Palette 1, Einstellposition	über PT-Befehle
Position 2	Palette 2, Einstellposition	
Position 10	Palette 1, Referenzposition	
Position 11	Palette 1, Endposition in Spaltenrichtung	über Teaching Box
Position 12	Palette 1, Endposition in Zeilenrichtung	
Position 13	Palette 1, Eckposition gegenüber Referenzposition	
Position 20	Palette 2, Referenzposition	
Position 21	Palette 2, Endposition in Spaltenrichtung	
Position 22	Palette 2, Endposition in Zeilenrichtung	
Position 23	Palette 2, Eckposition gegenüber Referenzposition	
Position 30	Position zum Einsetzen des Arbeitsgegenstands in das Prüfgerät	
Position 50	Position oberhalb der aktuellen Palettenposition	

**Tab. 5-11:** Arbeitspositionen

#### Palettenzähler

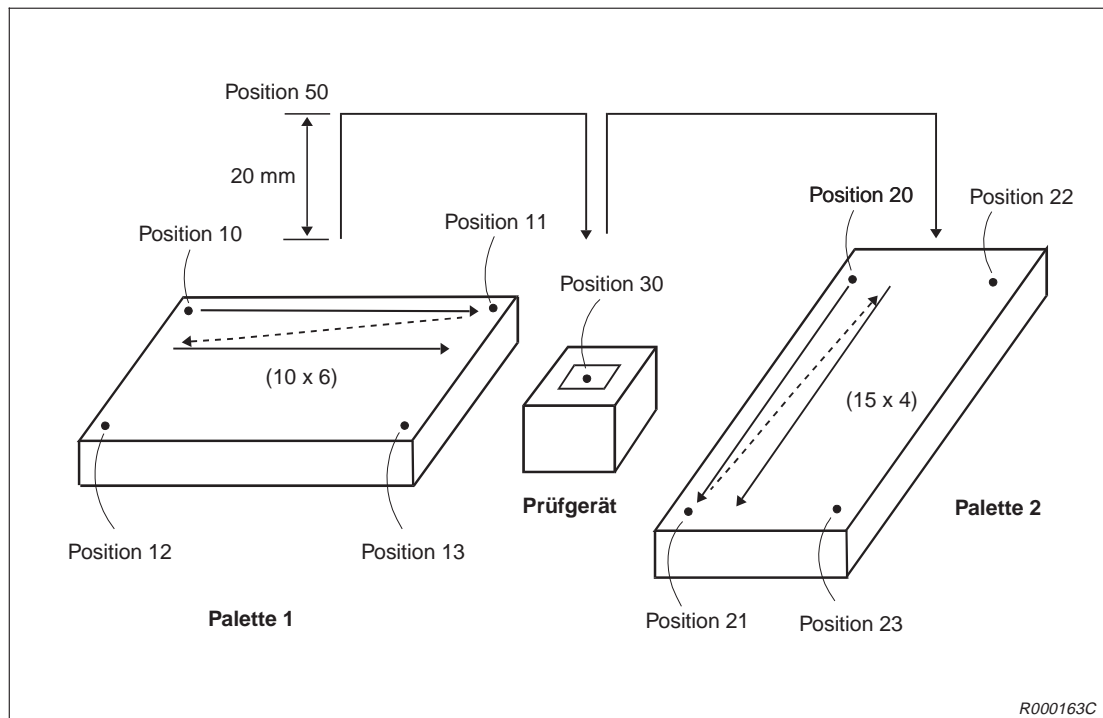
Position	Beschreibung
Zähler 11	Palette 1, Spaltenzähler
Zähler 12	Palette 1, Zeilenzähler
Zähler 21	Palette 2, Spaltenzähler
Zähler 22	Palette 2, Zeilenzähler

**Tab. 5-12:** Palettenzähler

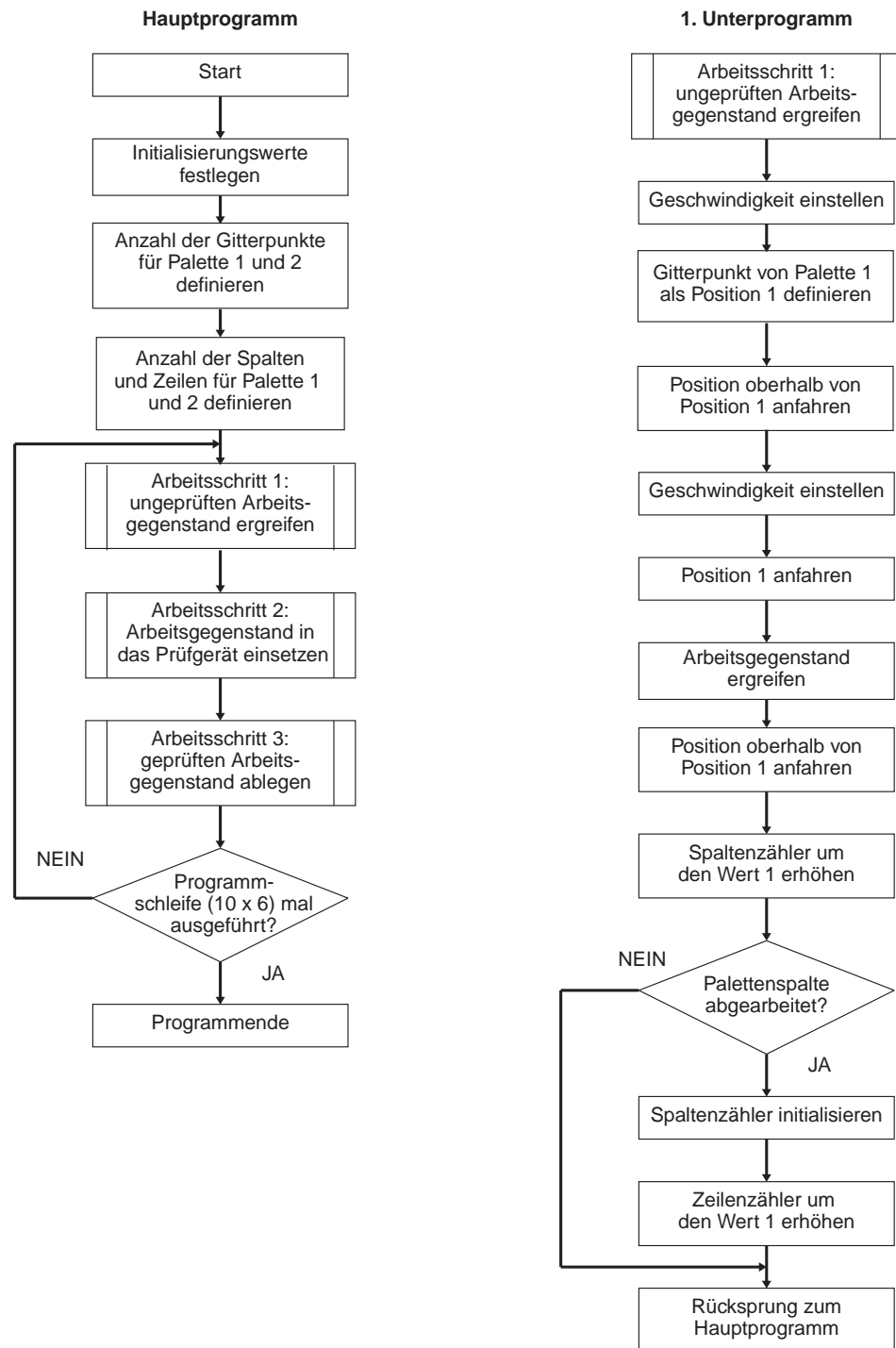
#### Eingangssignal

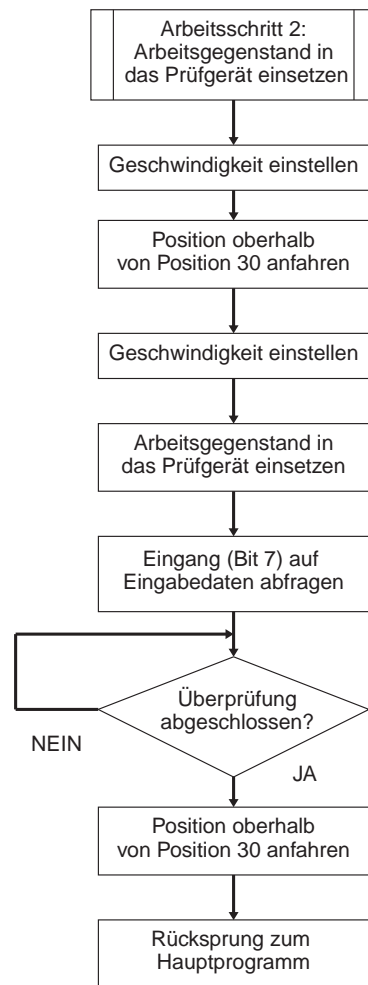
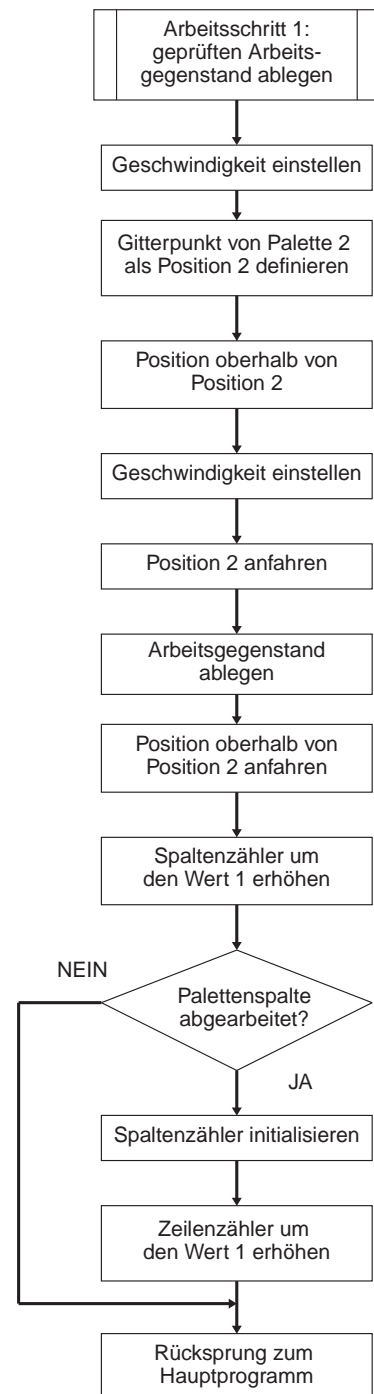
Ein-/Ausgang	Beschreibung	Bitnummer
Eingang	Signal für abgeschlossene Überprüfung	Bit 7

**Tab. 5-13:** E-/A-Signale

**Skizze des Arbeitsablaufs****Abb. 5-26:** Arbeitsgegenstände palettieren

# Flußdiagramm des Arbeitsablaufs



**2. Unterprogramm****3. Unterprogramm**

**Programmbeispiel (MOVEMASTER-Befehle)**

## Initialisierung

10	PD	50,0,0,20,0,0,0	Position (Z = 20 mm), die in direkter Linie oberhalb der aktuellen Palettenposition liegt, als Position 50 definieren
15	TL	145	Werkzeuglänge auf 145 mm einstellen
20	GP	10,8,10	Parameter für die zum Schließen der Hand benötigte Greifkraft einstellen
25	PA	1,10,6	Definition der Gitterpunkte für Palette 1 (10 Spalten x 6 Zeilen)
30	PA	2,15,4	Definition der Gitterpunkte für Palette 2 (15 Spalten x 4 Zeilen)
35	SC	11,1	Spaltenzähler von Palette 1 (Zähler 11) initialisieren (Zählerwert auf 1 setzen)
40	SC	12,1	Zeilenzähler von Palette 1 (Zähler 12) initialisieren (Zählerwert auf 1 setzen)
45	SC	21,1	Spaltenzähler von Palette 2 (Zähler 21) initialisieren (Zählerwert auf 1 setzen)
50	SC	22,1	Zeilenzähler von Palette 2 (Zähler 22) initialisieren (Zählerwert auf 1 setzen)

## Hauptprogramm

100	RC	60	Beginn einer Programmschleife mit 60 Wiederholzyklen (60 Arbeitsgegenstände)
110	GS	200	Sprung zum 1. Unterprogramm ab Zeile 200 (Arbeitsgegenstand von Palette 1 ergreifen)
120	GS	300	Sprung zum 2. Unterprogramm ab Zeile 300 (Arbeitsgegenstand in das Prüfgerät einsetzen)
130	GS	400	Sprung zum 3. Unterprogramm ab Zeile 400 (Arbeitsgegenstand auf Palette 2 ablegen)
140	NX		Ende der Programmschleife (Rücksprung zur Zeile 100)
150	ED		Programmende

## 1. Unterprogramm (Arbeitsgegenstand von Palette 1 ergreifen)

200	SP	25	Geschwindigkeit auf den Wert 25 einstellen
202	PT	1	Gitterpunkt von Palette 1 als Position 1 definieren
204	MA	1,50,O	Eine Position mit offener Hand anfahren, die 20 mm oberhalb von Position 1 liegt
206	SP	8	Geschwindigkeit auf den Wert 8 einstellen
208	MO	1,O	Position 1 anfahren
210	GC		Hand schließen und Arbeitsgegenstand ergreifen
212	MA	1,50,C	Eine Position mit geschlossener Hand anfahren, die 20 mm oberhalb von Position 1 liegt
214	IC	11	Zählerwert des Spaltenzählers von Palette 1 (Zähler 11) um den Wert 1 erhöhen
216	CP	11	Wert des Zählers 11 in das interne Register laden
218	EQ	11,230	Sprung zur Zeile 230, wenn eine Palettenzeile abgeräumt ist (Wertgleichheit)
220	RT		Unterprogramm wird bei Wertgleichheit beendet

230	SC	11,1	Spaltenzähler von Palette 1 (Zähler 11) initialisieren (Zählerwert auf 1 setzen)
232	IC	12	Zeilenzähler von Palette 1 (Zähler 12) um den Wert 1 erhöhen
234	RT		Unterprogrammende (Rücksprung zum Hauptprogramm)

#### 2. Unterprogramm (Arbeitsgegenstand in das Prüfgerät einsetzen)

300	SP	25	Geschwindigkeit auf den Wert 25 einstellen
302	MT	30,-50,C	Eine Position anfahren, die 50 mm oberhalb des Prüfgerätes liegt (50 mm oberhalb von Position 30)
304	SP	8	Geschwindigkeit auf den Wert 8 einstellen
306	MO	30,C	Arbeitsgegenstand in das Prüfgerät einsetzen (Hand geschlossen)
308	ID		Eingabedaten empfangen
310	TB	-7,308	Roboter wartet, bis die Überprüfung des Arbeitsgegenstands abgeschlossen ist
312	MT	30,-50,C	Eine Position anfahren, die 50 mm oberhalb des Prüfgerätes liegt (50 mm oberhalb von Position 30)
314	RT		Unterprogrammende (Rücksprung zum Hauptprogramm)

#### 3. Unterprogramm (Arbeitsgegenstand auf Palette 2 ablegen)

400	SP	25	Geschwindigkeit auf den Wert 25 einstellen
402	PT	2	Gitterpunkt von Palette 2 als Position 2 definieren
404	MA	2,50,C	Position 2 anfahren (20 mm oberhalb von Position 2)
406	SP	8	Geschwindigkeit auf den Wert 8 einstellen
408	MO	2,C	Position 2 anfahren
410	GO		Hand öffnen und Arbeitsgegenstand ablegen
412	MA	2,50,O	Position 2 mit offener Hand anfahren (20 mm oberhalb von Position 2)
414	IC	21	Zählerwert des Spaltenzählers von Palette 2 (Zähler 21) um den Wert 1 erhöhen
416	CP	21	Wert des Zählers 21 in das interne Register laden
418	EQ	16,430	Sprung zur Zeile 430, wenn eine Palettenzeile abgeräumt ist (Wertgleichheit)
420	RT		Unterprogramm wird bei Wertgleichheit beendet
430	SC	21,1	Spaltenzähler von Palette 2 (Zähler 21) initialisieren (Zählerwert auf 1 setzen)
432	IC	22	Zeilenzähler von Palette 2 (Zähler 22) um den Wert 1 erhöhen
434	RT		Unterprogrammende (Rücksprung zum Hauptprogramm)

#### Erläuterungen zum Programmbeispiel:

- Im obigen Programmbeispiel wird der Spaltenzähler der jeweiligen Palette nach dem Ergreifen bzw. Ablegen des Arbeitsgegenstands um den Wert 1 erhöht. Nach Erreichen des Spaltenendes wird der Spaltenzähler wieder zurückgesetzt. Anschließend wird der Wert des Zeilenzählers der jeweiligen Palette um 1 erhöht, damit die nächste Palettenzeile angefahren werden kann (siehe Programmzeile 214 bis 232 und 414 bis 432).



- Der Roboter bleibt so lange in Warteposition, bis das Bestätigungssignal vom Prüfgerät empfangen wird (Programmzeile 310).
- Die Gesamtanzahl der Arbeitsschritte ist von der in der Programmzeile 100 festgelegten Anzahl der Arbeitsgegenstände (60) abhängig.

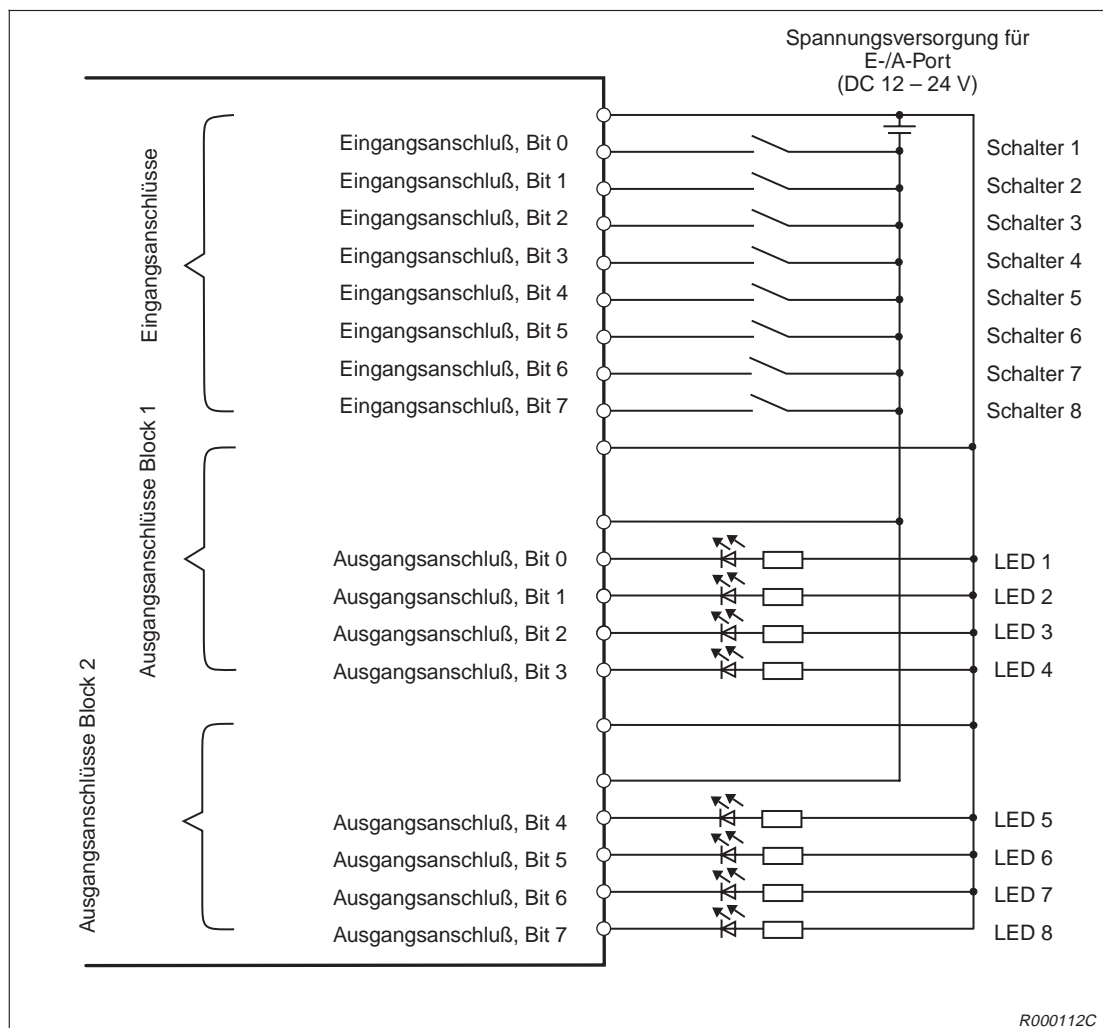
## 5.3.5

## 4. Beispiel: Anschluß externer Ein-/Ausgabegeräte

## Beschreibung des Arbeitsablaufs

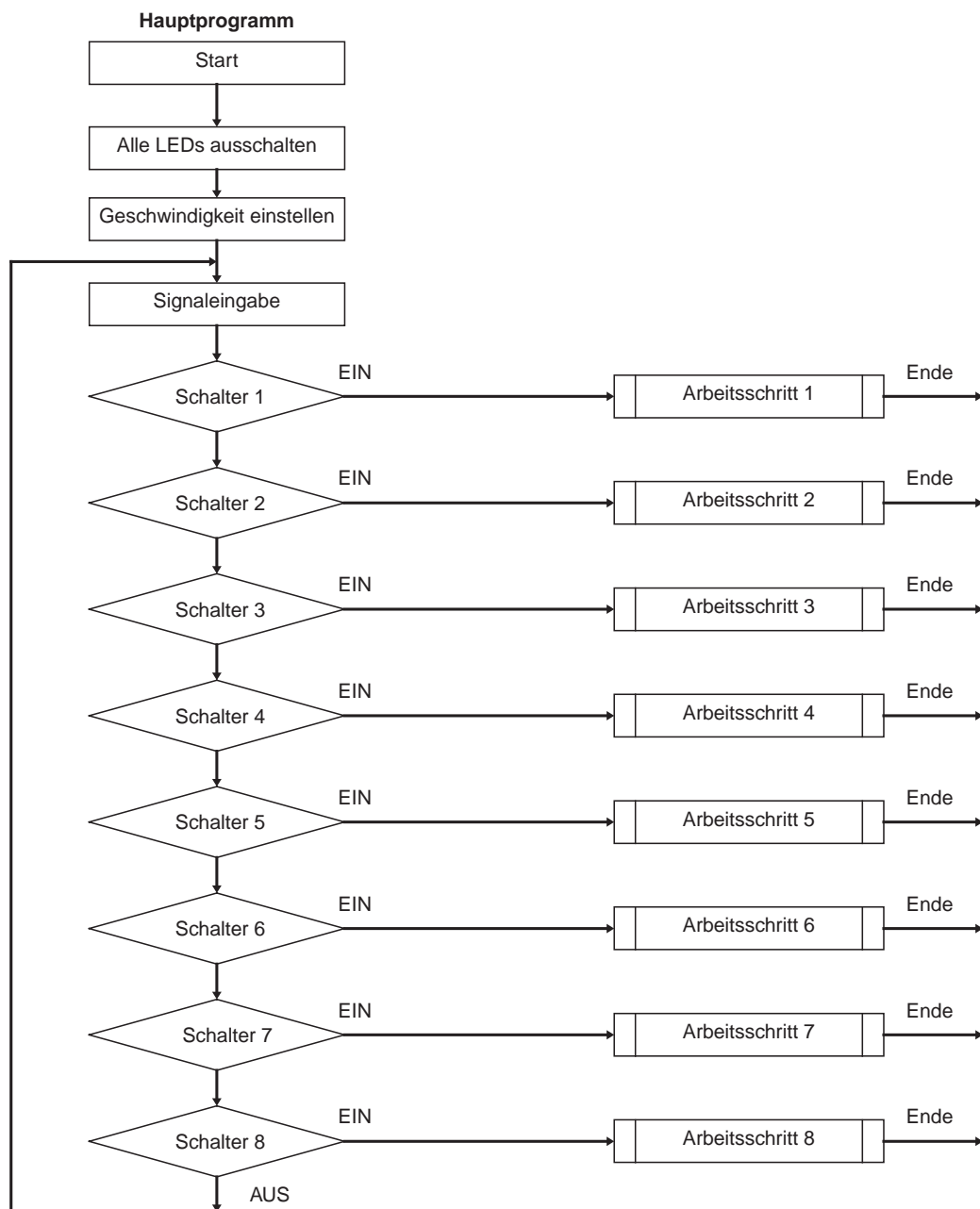
An die Eingänge der Robotersteuerung werden 8 Schalter angeschlossen, über die jeweils ein externes Eingangssignal ein- oder ausgeschaltet werden kann. Mit den Schaltern können verschiedene Arbeitsschritte ausgewählt werden. Der jeweils vom Roboter ausgeführte Arbeitsschritt wird mittels am externen Ausgang angeschlossener Leuchtdioden angezeigt.

## Anschlußplan

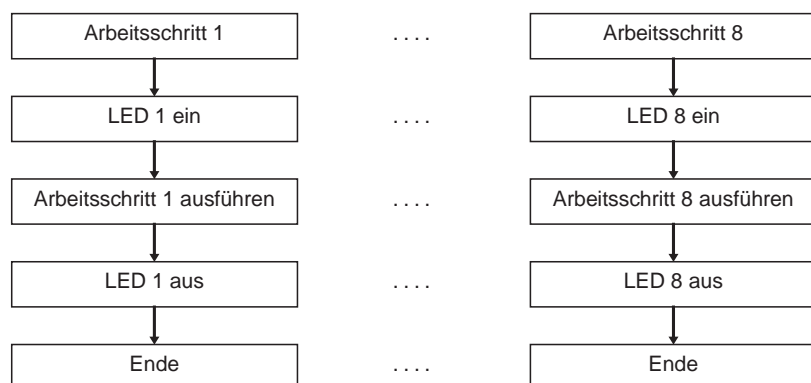


**Abb. 5-27:** Anschlußbeispiel mit externen Ein-/Ausgabegeräten

## Flußdiagramm des Arbeitsablaufs



## 8 Unterprogramme für die einzelnen Arbeitsschritte



**Programmbeispiel (MOVEMASTER-Befehle)**

## Hauptprogramm

15	OD	0	Alle LEDs ausschalten
20	SP	10	Geschwindigkeit auf den Wert 10 einstellen
25	ID		Eingabedaten empfangen
30	TB	+0,100	Sprung zur Zeile 100, wenn der 1. Schalter eingeschaltet wird.
31	TB	+1,200	Sprung zur Zeile 200, wenn der 2. Schalter eingeschaltet wird.
32	TB	+2,300	Sprung zur Zeile 300, wenn der 3. Schalter eingeschaltet wird.
33	TB	+3,400	Sprung zur Zeile 400, wenn der 4. Schalter eingeschaltet wird.
34	TB	+4,500	Sprung zur Zeile 500, wenn der 5. Schalter eingeschaltet wird.
35	TB	+5,600	Sprung zur Zeile 600, wenn der 6. Schalter eingeschaltet wird.
36	TB	+6,700	Sprung zur Zeile 700, wenn der 7. Schalter eingeschaltet wird.
37	TB	+7,800	Sprung zur Zeile 800, wenn der 8. Schalter eingeschaltet wird.
38	GT	25	Sprung zur Zeile 25, wenn alle Schalter eingeschaltet sind.

## Unterprogramme

100	OB	+0	LED 1 einschalten (Start des Arbeitsschritts)
105	MO	10	Arbeitsschritt 1 ausführen
...			
198	OB	-0	LED 1 ausschalten (Arbeitsschritt ist beendet)
199	GT	25	Sprung zur Zeile 25
...			
800	OB	+7	LED 8 einschalten (Start des Arbeitsschritts)
805	MO	80	Arbeitsschritt 8 ausführen
...			
898	OB	-7	LED 8 ausschalten (Arbeitsschritt ist beendet)
899	GT	25	Sprung zur Zeile 25



## 6 MELFA-BASIC III-Programmierung

### 6.1 Programmierung mit der Teaching Box

Sie können ein Roboterprogramm unter Einsatz der MELFA-BASIC III-Befehle mit einem PC und der Teaching Box erstellen. Mit der MELFA-BASIC III-Programmierungsmethode lassen sich komplexere Programme erstellen als mit der MOVEMASTER COMMAND-Programmierungsmethode. In diesem Abschnitt wird die MELFA-BASIC III-Programmierungsmethode unter Verwendung der Teaching Box beschrieben.

Detaillierte Beschreibungen der Programmiersprache MELFA-BASIC III und der verwendeten Befehle finden Sie in den Kapiteln 7 und 8 in diesem Handbuch.

Die nachfolgende Tabelle zeigt die Möglichkeiten der Weiterverarbeitung der Daten einer Zeile.

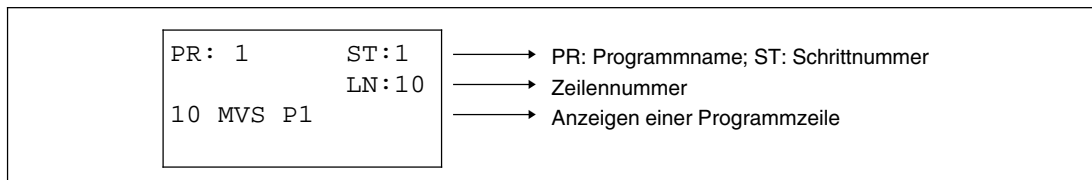
Eingabeformat	Verarbeitung
Zeilennummer und Befehl	Eingabe wird als Zeile des Roboterprogramms verarbeitet
Nur Zeilennummer	Löscht die angegebene Zeile aus dem Programm (durch Überschreiben)
Nur Befehl	Führt den Befehl sofort aus (Direkt-Modus)

**Tab. 6-1:** Weiterverarbeitung der übertragenen Daten einer Zeile

## 6.1.1 Roboterprogramm erstellen

In diesem Abschnitt wird die Erstellung eines neuen Programms beschrieben. Dabei werden die Befehle der Reihe nach eingegeben, während die Positionen später definiert werden (siehe Abschnitt 6.1.3).

### Display-Darstellung



**Abb. 6-1:** Display-Darstellung







### Ausführung

Folgendes Beispielprogramm Nr. 3 soll erstellt werden.


















#### Beispielprogramm

```









10 OVRD 50
20 MVS P10
30 DLY 0.3
40 GOTO 20
    
```

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<div>                     &lt;MENU&gt;                      1. TEACH    2. RUN                      3. FILE    4. MONI.                      5. MAINT.   6. SET                 </div>	 	Das Menü TEACH zur Programmauswahl wird durch Eingabe der Ziffer 1 aufgerufen.
②	<div>                     &lt;TEACH&gt;                      ( 3 )                      SELECT PROGRAM                 </div>	 	Die Programmnummer (3) wird eingegeben. Anschließend wird die Auswahl bestätigt.
③	<div>                     PR:3            ST:1                      INTP:---                      SPD :--                      TIME:0.0 sec                 </div>	  <p>Weitere mögliche Tastenbetätigungen:                      Taste [COND] + [RPL ↓] oder                      Taste [COND] + [DEL ←]</p>	Nach der Tastenbetätigung wird das Menü zur Programmeditierung aufgerufen. <b>HINWEIS:</b> Nach der Tastenbetätigung erfolgt eine Umschaltung in den MELFA-BASIC-Befehls-Modus!

**Tab. 6-2:** Erstellung eines neuen Programms (1)

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
④	PR: 3            ST: 1 LN: 0 -NO DATA-	3 x 	Der Cursor wird zum Eingabefeld für die Befehle bewegt.
⑤	PR: 3            ST: 1 LN: 0 █ CODE EDIT █	 →  → 	Die Zeilennummer (10) wird eingegeben.
⑥	PR: 3            ST: 1 LN: 0 10 █ █ CODE EDIT █	 → 3 x 	Das Zeichen „O“ wird eingegeben.
⑦	PR: 3            ST: 1 LN: 0 10 O █ █ CODE EDIT █		Wenn Sie die Taste 2mal gedrückt haben, werden 4 Befehle angezeigt, die mit dem Zeichen „O“ beginnen.
⑧	1. OVRD    2. ON 3. OFF    4. OUT 10 O █ █ CODE EDIT █	 → 	Der Befehl „OVRD“ wird eingegeben. <b>HINWEIS:</b> Betätigen Sie bei der Befehlseingabe die [POS/CHAR]-Taste nach der Eingabe des ersten Zeichens. Anschließend kann die Nummer des gewünschten Befehls gewählt werden. Bei jeder erneuten Betätigung der [POS/CHAR]-Taste ändert sich das eingegebene Zeichen und die Befehlsliste.
⑨	PR: 3            ST: 1 LN: 0 10 OVRD █ █ CODE EDIT █	 → 	Es wird die Programmzeile „10 OVRD 50“ eingegeben.
⑩	PR: 3            ST: 1 LN: 0 10 OVRD 50 █ █ CODE EDIT █		Jetzt ist die neue Programmzeile „10 OVRD 50“ wirksam. Die Schrittnummer (2) wird angezeigt.
⑪	PR: 3            ST: 2 LN: 0 █ CODE EDIT █	 →  →  →  → 	Die Zeilennummer (20) und das erste Befehlszeichen (M) werden eingegeben.

Tab. 6-2: Erstellung eines neuen Programms (2)

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
⑫	PR: 3            ST: 2 LN: 0 20 M █ █ CODE EDIT █		Wenn Sie die Taste gedrückt halten, werden 4 Befehle, die mit „M“ beginnen angezeigt.
⑬	1. MOV    2. MVS 3. MVC    4. MVR 20 M █ █ CODE EDIT █	 → 	Der Befehl MVS wird ausgewählt.
⑭	PR: 3            ST: 2 LN: 0 20 MVS █ █ CODE EDIT █	 →  →  → 	Es wird die Programmzeile „20 MVS P10“ eingegeben.
⑮	PR: 3            ST: 2 LN: 0 20 MVS P10 █ CODE EDIT █		Jetzt ist die neue Programmzeile „20 MVS P10“ wirksam. Die nächste Schrittnummer (3) wird angezeigt.
⑯	PR: 3            ST: 3 LN: 0 █ █ CODE EDIT █		Die übrigen Befehle werden in der gleichen Weise eingegeben.

**Tab. 6-2:** Erstellung eines neuen Programms (3)



## 6.1.2 Roboterprogramm editieren

Mit der Teaching Box kann ein zuvor erstelltes Roboterprogramm editiert werden. In diesem Abschnitt wird die Editierung eines Roboterprogramms beschrieben.

### ① Menü zur Programmeditierung öffnen

Wählen Sie das gewünschte Programm aus, und öffnen Sie anschließend das Menü zur Programmeditierung (siehe nachfolgendes Beispiel). Es wird automatisch das letzte Programm im Editiermenü angezeigt, wenn Sie keine Programmauswahl vornehmen.

#### Display-Darstellung

<pre>PR: 1      ST:1           LN:10 10 MVS P1</pre>	<p>→ PR: Programmname; ST: Schrittnummer</p> <p>→ Zeilennummer</p> <p>→ Anzeigen einer Programmzeile</p>
--	--


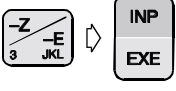
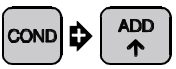
**Abb. 6-2:** Display-Darstellung im Menü zur Programmeditierung

#### HINWEIS

Beachten Sie, daß nach Eingabe der Programmnummer mit anschließender Eingabebestätigung der MELFA-BASIC-Befehlsmodus aufgerufen wird.

#### Ausführung

In der unteren Tabelle wird das Programm Nr. 3 ausgewählt und anschließend das Menü zur Programmeditierung geöffnet.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre>&lt;MENU&gt; 1. TEACH  2. RUN 3. FILE   4. MONI. 5. MAINT. 6. SET</pre>		Das Menü TEACH zur Programmauswahl wird durch Eingabe der Ziffer 1 aufgerufen.
②	<pre>&lt;TEACH&gt; ( 3 )  SELECT PROGRAM</pre>		Die Programmnummer (3) wird eingegeben. Anschließend wird die Auswahl bestätigt.
③	<pre>PR:3      ST:1 INTP:--- SPD :-- TIME:0.0 sec</pre>	 <p>Weitere mögliche Tastenbetätigungen: Taste [COND] + [RPL ↓] oder Taste [COND] + [DEL ←]</p>	Nach der Tastenbetätigung wird das Menü zur Programmeditierung aufgerufen. <b>HINWEIS:</b> Nach der Tastenbetätigung erfolgt eine Umschaltung in den MELFA-BASIC-Befehls-Modus!
④	<pre>PR:3      ST:1           LN:10 10 MVS P1</pre>		Nach der Tastenbetätigung wird das Menü zur Programmeditierung aufgerufen. Es wird die erste Programmzeile angezeigt.

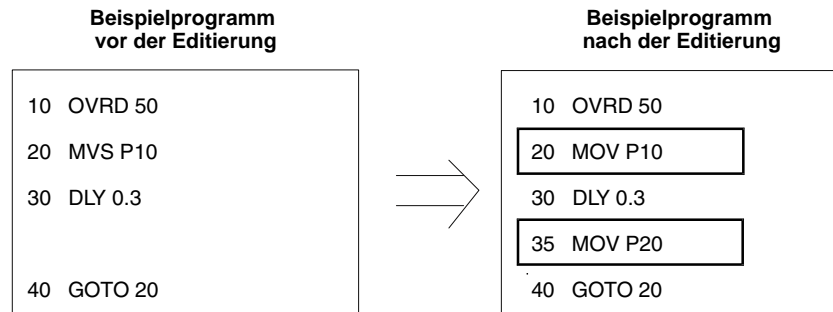
**Tab. 6-3:** Beispiel zum Öffnen des Menüs zur Programmeditierung












## ② Programm editieren

Nehmen Sie die Programmeditierungen vor.















### Ausführung

Im nachfolgenden Beispiel wird die Editierung eines Programmes gezeigt.



Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<div>PR:3      ST:1</div> <div>LN:10</div> <div>10 OVRD</div>	2 x 	Der Cursor wird zum Eingabefeld für die Zeilennummer bewegt.
②	<div>PR:3      ST:1</div> <div>LN:(10)</div> <div>10 OVRD 50</div> <div>LINE NUMBER</div>	   	Die Zeilennummer (20) wird eingegeben. Danach wird Eingabe zur Editierung der Programmzeile aufgerufen (CODE EDIT).
③	<div>PR:3      ST:1</div> <div>LN:20</div> <div>20 MVS P10</div> <div>CODE EDIT</div>	6 x 	Der Cursor wird nach rechts bewegt und eine Stelle hinter „S“ plaziert.
④	<div>PR:3      ST:1</div> <div>LN:20</div> <div>20 MVS P10</div> <div>CODE EDIT</div>	 	Durch zweimaliges Betätigen der Tastenkombination werden die Zeichen „S“ und „V“ gelöscht.
⑤	<div>PR:3      ST:1</div> <div>LN:20</div> <div>20 M P10</div> <div>CODE EDIT</div>		Es werden 4 Befehle, die mit „M“ beginnen angezeigt.
⑥	<div>1.MOV    2.MVS</div> <div>3.MVC    4.MVR</div> <div>20 M P10</div> <div>CODE EDIT</div>	 	Der Befehl „MOV“ wird ausgewählt. Die Programmzeile „20 MVS P10“ wird eingegeben.

Tab. 6-4: Beispiel zum Editieren eines Programms (1)

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
⑦	PR:3      ST:1 LN:20 20 MOV P10 CODE EDIT	 	Jetzt ist die neue Programmzeile wirksam. Es wird die nächste Zeile „30 DLY 0.3“ angezeigt.
⑧	PR:3      ST:1 LN:20 20 MOV P10 CODE EDIT	 2 x 	Die Zeile (20) wird erneut aufgerufen. Der Cursor wird nach rechts bewegt und eine Stelle hinter „0“ platziert.
⑨	PR:3      ST:1 LN:20 20 MOV P10 CODE EDIT	 2 x 	Die Zeilennummer „20“ wird gelöscht.
⑩	PR:3      ST:1 LN:20 MOV P10 Code Edit	 	Die Zeilennummer „35“ wird eingegeben.
⑪	PR:3      ST:1 LN:20 35 MOV P10 CODE EDIT	7 x   	Der Cursor wird auf „0“ platziert. Anschließend wird das Zeichen „1“ gelöscht.
⑫	PR:3      ST:1 LN:20 35 MOV P0 CODE EDIT		Das neue Zeichen „2“ wird eingegeben.
⑬	PR:3      ST:1 LN:20 35 MOV P20 CODE EDIT	 	Jetzt ist die neue Programmzeile „35 MOV P20“ wirksam.
⑭	PR:3      ST:1 LN:30 40 GOTO 20 CODE EDIT		Die nächste Programmzeile wird angezeigt.

Tab. 6-4: Beispiel zum Editieren eines Programms (2)

### Beschreibung

- 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.
- 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.
- Geben Sie am Eingabefeld „LN“ die gewünschte Programmzeile an.
- Mit den Tasten [+ /FORWD] (vorwärts) und [- /BACKWD] (rückwärts) können Sie im Programm „blättern“.
- Nach erfolgter Editierung einer Zeile wird dies auf dem Display angezeigt. Bei einer Überschreibung wird die überschriebene Zeile angezeigt. Nach Einfügen einer neuen Zeile wird die nächste Zeile angezeigt.
- Es besteht die Möglichkeit, ein neues Programm ohne Einsatz eines PCs zu erstellen. Fügen Sie hierzu eine neue Zeile in ein noch nicht belegtes Programm ein.
- Das Menü zur Programmeditierung läßt sich, durch Betätigung der Tasten [COND] + [ADD ↑], von jedem Menüfenster aus öffnen.
- Die nachfolgende Tabelle zeigt die Möglichkeiten der Weiterverarbeitung der mit einem PC übertragenen Daten einer Zeile.

Eingabeformat	Verarbeitung
Zeilennummer und Befehl	Eingabe wird als Zeile des Roboterprogramms verarbeitet
Nur Zeilennummer	Löscht die angegeben Zeile aus dem Programm (durch Überschreiben)
Nur Befehl	Führt den Befehl sofort aus (Direkt-Modus)

**Tab. 6-5:** Weiterverarbeitung der übertragenen Daten einer Zeile

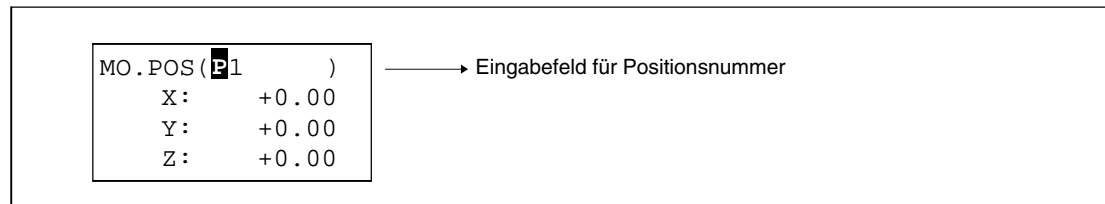
### 6.1.3 Positionsdaten eingeben, anfahren, ersetzen, ändern und löschen

Die über MELFA-BASIC III-Befehle im Programm definierten Positionen können mit der Teaching Box editiert werden (siehe nachfolgendes Beispiel).

#### ① Eingabe der aktuellen Positionsdaten

Es kann die aktuelle Position (Koordinatenwerte) als neue Position (mit Positionsnummer) abgespeichert werden.

#### Display-Darstellung



**Abb. 6-3:** Display-Darstellung zu den Positionsdaten

#### Ausführung

Im nachfolgenden Beispiel wird die aktuelle Position als Position Nr. 3 definiert.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<pre>PR:3   ST:1           LN:10 10 OVRD 50</pre>		Nach der Tastenbetätigung wird das Menü für die Editierung der Positionsdaten angezeigt.
②	<pre>MO.POS(P1) 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 aktuellen Position.
③	<pre>MO.POS(P3) X: +200.00 Y: + 40.00 Z: +500.00</pre>		Bewegen Sie den Roboter im Jog-Betrieb zu der Position, die als neue Position gespeichert werden soll.
④	<pre>MO.POS(P3) X: +200.00 Y: + 40.00 Z: +500.00</pre>		Es ertönt ein Summton, und eine Bestätigungsabfrage wird angezeigt.
⑤	<pre>MO.POS(P3) ADDITION?</pre>		Die Definition der neuen Position wird bestätigt. Nach der Anzeige von „EXECUTING“ ist die neu definierte Position wirksam.

**Tab. 6-6:** Beispiel zur Eingabe der aktuellen Positionsdaten (1)

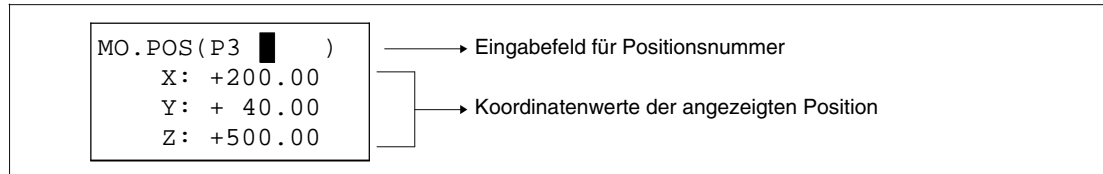
Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
⑥	<div>MO.POS ( P3 <span style="background-color: black; color: black;">    </span> )</div> <div>X: +200.00</div> <div>Y: + 40.00</div> <div>Z: +500.00</div>	<div>COND</div> <div>⇒</div> <div>ADD ↑</div>	Die Koordinatenwerte der neuen Position sind jetzt registriert. Nach der Tastenbetätigung wird wieder das Menü zur Programmeditierung angezeigt.

Tab. 6-6: Beispiel zur Eingabe der aktuellen Positionsdaten (2)

## ② Position anfahren

Es kann die Handspitze (TCP: Tool Center Point) des Roboters zu einer bereits definierten Position gefahren werden.

### Display-Darstellung



**Abb. 6-4:** Display-Darstellung zu den Positionsdaten

### Ausführung

Im Beispiel wird die Position Nr. 3 angefahren.

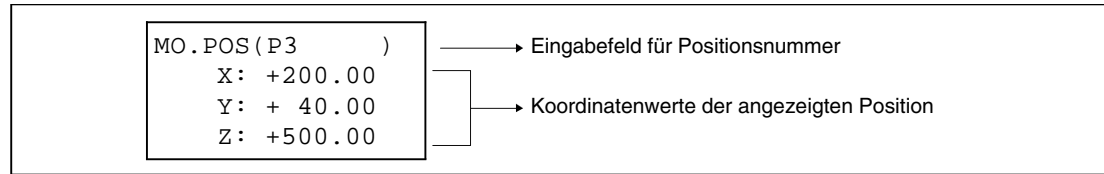
Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<div>           PR: 3    ST: 1                  LN: 10            10 OVRD 50         </div>		Nach der Tastenbetätigung wird das Menü für die Editierung der Positionsdaten angezeigt.
②	<div>           MO.POS ( P1 [ ] )            X:    +0.00            Y:    +0.00            Z:    +0.00         </div>		Die Positionsnummer „3“ wird eingegeben und deren Koordinatenwerte werden angezeigt.
③	<div>           MO.POS ( P3 [ ] )            X:    +200.00            Y:    + 40.00            Z:    +500.00         </div>		Halten Sie den Totmannschalter auf der Rückseite der Teaching Box gedrückt. Nach der Tastenbetätigung fährt der Roboter mittels Gelenk-Interpolation die Position Nr. 3 an. Der Roboter stoppt, sobald die [INP/EXE]-Taste losgelassen wird.
④	<div>           MO.POS ( P3 [ ] )            X:    +200.00            Y:    + 40.00            Z:    +500.00         </div>		Die Bewegung zur Position Nr. 3 ist beendet. Nach der Tastenbetätigung erfolgt die Rückkehr zum Ausgangsmenü.

**Tab. 6-7:** Beispiel zum Anfahren einer Position

### ③ Position ersetzen

Die Positionsdaten einer definierten Position können durch eine aktuelle Position ersetzt werden.

#### Display-Darstellung



**Abb. 6-5:** Display-Darstellung zu den Positionsdaten

#### Ausführung

Die Position Nr. 3 wird durch eine aktuelle Position ersetzt.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<div>                     PR:3            ST:1                                       LN:10                      10 OVRD 50                 </div>		Nach der Tastenbetätigung wird das Menü für die Editierung der Positionsdaten angezeigt.
②	<div>                     MO.POS(P1) )                        X: +0.00                        Y: +0.00                        Z: +0.00                 </div>		Die Positionsnummer „3“ wird eingegeben. Danach wird die Eingabe bestätigt, und es erscheint die Anzeige der aktuellen Position.
③	<div>                     MO.POS(P3) )                        X: +200.00                        Y: + 40.00                        Z: +500.00                 </div>		Bewegen Sie den Roboter im Jog-Betrieb zu der Position, die angeglichen werden soll.
④	<div>                     MO.POS(P3) )                        X: +200.00                        Y: + 40.00                        Z: +500.00                 </div>	 [STEP/MOVE]-Taste gedrückt halten	Es ertönt ein Summton, und eine Bestätigungsabfrage wird angezeigt.
⑤	<div>                     MO.POS(P3) )                      REPLACE ?                 </div>	Schritt ersetzen? 1.) Abfrage bestätigen  2.) Abbrechen [STEP/MOVE]-Taste loslassen	Das Ersetzen der Position wird bestätigt. Nach der Anzeige von „EXECUTING“ ist die ersetzte Position wirksam.
⑥	<div>                     MO.POS(P3) )                        X: +200.00                        Y: + 40.00                        Z: +550.00                 </div>		Das Ersetzen ist jetzt abgeschlossen. Nach der Tastenbetätigung wird wieder das Menü zur Programmeditierung angezeigt.

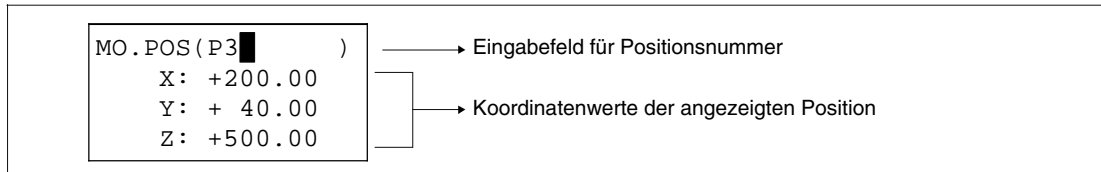
**Tab. 6-8:** Beispiel zum Ersetzen von Positionsdaten



#### ④ Positionsdaten ändern

Es können die Positionsdaten einer definierten Position geändert werden.

##### Display-Darstellung



**Abb. 6-6:** Display-Darstellung zu den Positionsdaten

##### Ausführung

Im Beispiel wird der Y-Koordinatenwert der Position Nr. 3 von +40.00 auf +50.00 geändert.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	MO.POS ( P1 ) X: +0.00 Y: +0.00 Z: +0.00		Es wird die Positionsnummer eingegeben. Nach der Tastenbetätigung werden die Koordinaten von Position Nr. 3 angezeigt.
②	MO.POS ( P3 ) X: +200.00 Y: + 40.00 Z: +500.00	2 x	Der Cursor wird zum Y-Eingabefeld bewegt.
③	MO.POS ( P3 ) X: +200.00 Y ( + 40.00 ) Z: +500.00	4 x	Der Cursor wird auf „4“ plaziert.
④	MO.POS ( P3 ) X: +200.00 Y ( + 40.00 ) Z: +500.00		Der Y-Koordinatenwert „40“ wird mit „50“ überschrieben.
⑤	MO.POS ( P3 ) X: +200.00 Y ( + 50.00 ) Z: +500.00		Es wird der neue Y-Koordinatenwert angezeigt.

**Tab. 6-9:** Beispiel zum Ändern von Positionsdaten

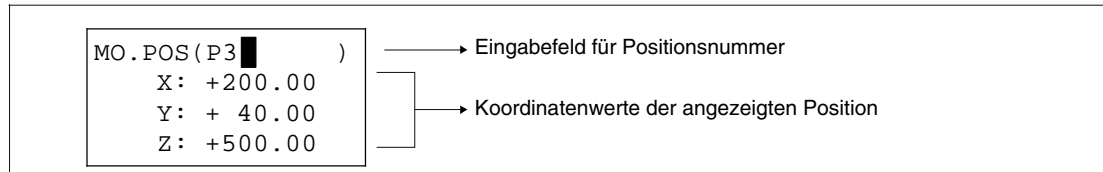
##### Beschreibung

- Den Cursor können Sie mit den Tasten [ADD ↑], [RPL ↓], [DEL ←] und [HAND →] bewegen.
- Betätigen Sie zum Löschen eines Zeichens die Taste [POS/CHAR] mit der Taste [DEL ←].

### ⑤ Positionsdaten löschen

Es können die Positionsdaten einer definierten Position gelöscht werden.

#### Display-Darstellung



**Abb. 6-7:** Display-Darstellung zu den Positionsdaten

#### Ausführung

Im Beispiel werden die Koordinatenwerte der Position Nr. 3 gelöscht.

Nr.	Display-Darstellung	Tastenbetätigungen	Beschreibung
①	<div>                     PR:3            ST:1                                       LN:10                      10 OVRD 50                 </div>		Nach der Tastenbetätigung wird das Menü für die Editierung der Positionsdaten angezeigt.
②	<div>                     MO.POS(P1 )                      X: +0.00                      Y: +0.00                      Z: +0.00                 </div>		Die Positionsnummer „3“ wird eingegeben und deren Koordinatenwerte werden angezeigt.
③	<div>                     MO.POS(P3 )                      X: +200.00                      Y: + 40.00                      Z: +500.00                 </div>		Es ertönt ein Summton, und eine Bestätigungsabfrage wird angezeigt.
④	<div>                     MO.POS(P3 )                      DELETE?                 </div>		Die Koordinaten der Position Nr. 3 werden gelöscht. Nach der Anzeige von „EXECUTING“ ist der Löschvorgang wirksam.
⑤	<div>                     MO.POS(P3 )                      X: +0.00                      Y: +0.00                      Z: +0.00                 </div>		Die Koordinatenwerte von Position Nr. 3 werden angezeigt. Nach der Tastenbetätigung erfolgt die Rückkehr zum Ausgangsmenü.

**Tab. 6-10:** Beispiel zum Löschen von Positionsdaten

# 7 MELFA-BASIC III

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

## 7.1 Begriffserklärung

### 7.1.1 Anweisung

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

**Beispiel** ▾

MOV P1 = Anweisung

MOV = Befehl

P1 = Befehlsparameter



### 7.1.2 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 die Roboterstatusvariable M\_HND(1) auf 1 gesetzt wird.

MOV P1 WTH M\_HND(1) = 1



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

### 7.1.3 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.

### 7.1.4 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 Asterisks-Zeichens (\*) hinter der Zeilennummer und einer alphanumerischen Zeichenkette oder dem Unterstrich (\_). Hierbei wird auch zwischen Groß- und Kleinbuchstaben unterschieden. Das erste Zeichen muß ein Buchstabe sein. Die Länge der Sprungmarke darf maximal 8 Zeichen betragen.

**Beispiel ▽**

120 \*ABLAG; 170 \*BAND\_1

△

**HINWEISE**

Für Markennamen dürfen keine reservierten Wörter benutzt werden (siehe Abschnitt 7.2.9).

Die Zeichenkette, die einen Markennamen beschreibt, darf nicht mit den Zeichen FN beginnen.

### 7.1.5 Zeichentypen

Die in MELFA-BASIC III verwendbaren Zeichentypen sind in Tabelle 7-1 aufgeführt.

Verwendbare Zeichen in MELFA-BASIC III	
Alphanumerische Zeichen	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
Zahlen	0 1 2 3 4 5 6 7 8 9
Symbole	! " ' # \$ % & ( ) * + - . , / : ; = < > ? @ ' [ ¥ ] ^ { } ~
Leerstellen	Leerzeichen
Unterstrich	—

**Tab. 7-1:** In MELFA-BASIC III verwendbare Zeichen

#### 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.

Die Symbole „[“ und „]“ sind auf der Tastatur der Teaching Box nicht vorhanden. Sie können durch Eingabe folgender Zeichen erzeugt werden:

Symbol	Eingabesymbol
[	??{
]	??}

**Tab. 7-2:** Erzeugung der Symbole „[“ und „]“

### 7.1.6 Zeichen mit besonderer Bedeutung

#### Unterstrich (\_)

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

**Beispiel ▾**

```
M_19  
P_100 ()  
M_OVRD  
M_IN
```



#### Apostroph (')

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

**Beispiel ▾**

```
150 'TESTZEILE
```



#### Asterisks (\*)

Das Asterisks-Zeichen wird vor alle Sprungmarken gesetzt.

**Beispiel ▾**

```
200 *SPEICHERN
```



#### Komma (,)

Das Komma dient bei Angabe mehrerer Parameter zur Trennung.

**Beispiel ▾**

```
P1 = (100, 150, ...)
```



#### Punkt (.)

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

**Beispiel ▾**

```
M1 = P2. X
```



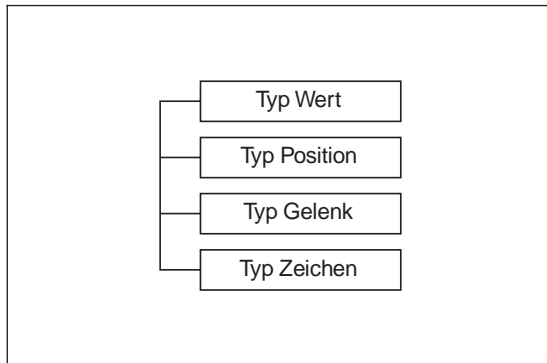
#### Leerzeichen

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

### 7.1.7 Datentypen

Datentypen umfassen Werte, Positionensdaten, Gelenkdaten und Zeichen. Um den Datentyp eindeutig festzulegen, muß er deklariert werden.

Bei Zahlen unterscheidet man zwischen reellen und ganzen Zahlen. Für Rechen- und Interpolationsfunktionen müssen die Daten nicht deklariert werden, da sie bei der Konvertierung automatisch in die passenden Daten für den Roboter umgewandelt werden.



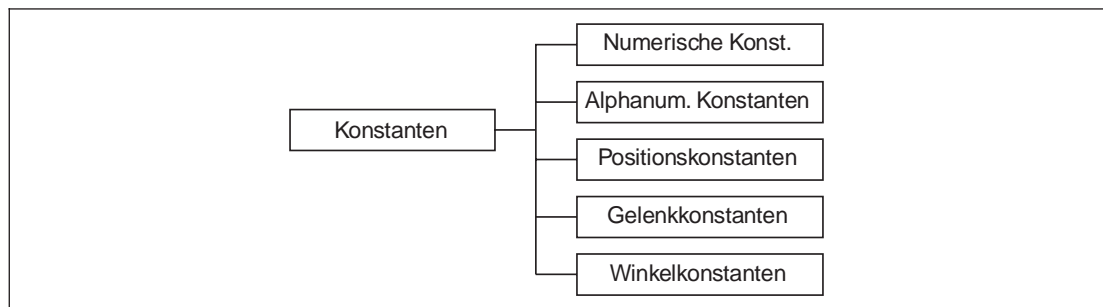
**Abb. 7-1:**  
*Datentypen*

R000395C

### 7.1.8 Konstanten

Man unterscheidet fünf Arten von Konstanten:

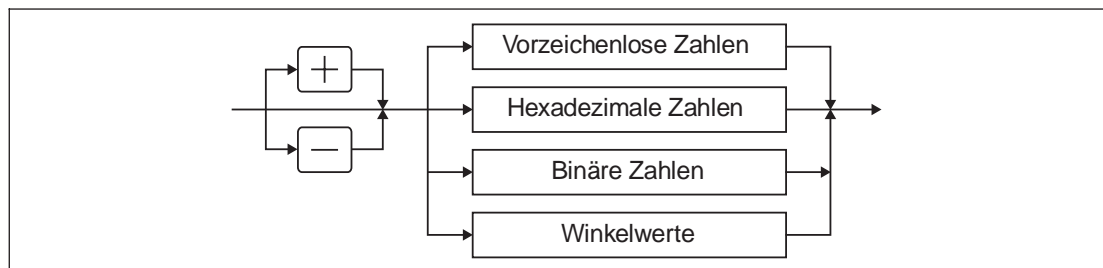
- Numerische Konstanten
- Alphanumerische Konstanten
- Positionskonstanten
- Gelenkkonstanten
- Winkelkonstanten



**Abb. 7-2:** Konstanten

#### Numerische Konstanten

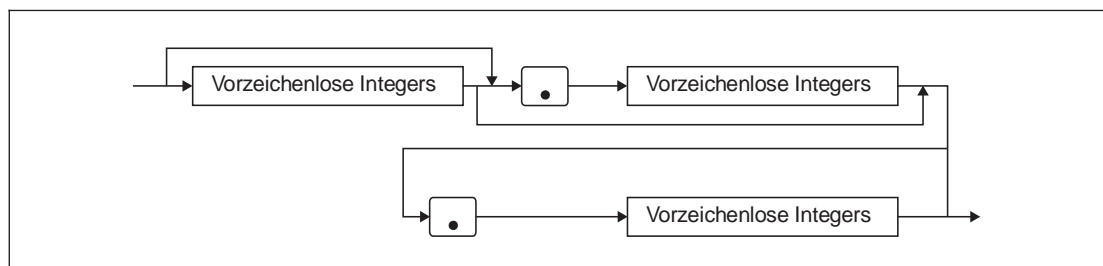
Abbildung 7-3 zeigt die Syntax numerischer Konstanten.



**Abb. 7-3:** Numerische Konstanten

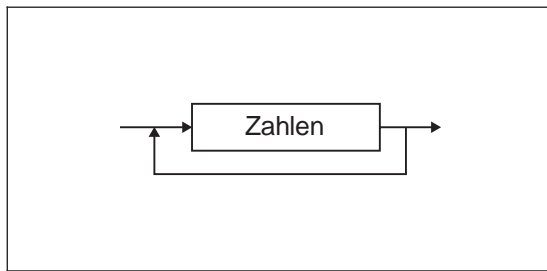
#### Vorzeichenlose Zahlen

Abbildung 7-4 zeigt die Syntax vorzeichenloser Integer-Zahlen.



**Abb. 7-4:** Vorzeichenlose Integer-Zahlen





**Abb. 7-5:**  
Vorzeichenlose Zahlen

Der Wertebereich für Zahlen ist  $-1.701411834604692E+38$  bis  $1.701411834604692E+38$ .

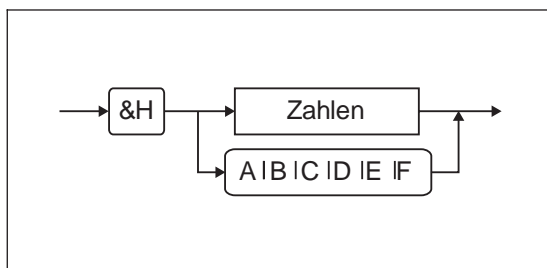
**Beispiel** ▾

1; 17; 1256



Hexadezimale Zahlen

Abbildung 7-6 zeigt die Syntax hexadezimaler Zahlen.



**Abb. 7-6:**  
Hexadezimale Zahlen

Der Wertebereich hexadezimaler Zahlen ist &H0000 bis &HFFFF.

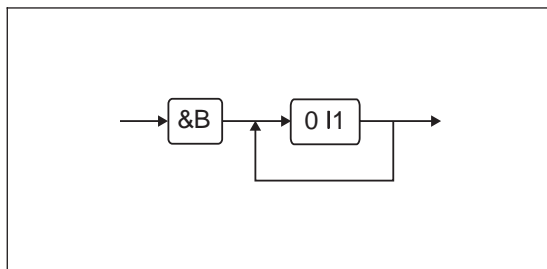
**Beispiel** ▾

&H132; &HC011; &H1AC4



Binäre Zahlen

Abbildung 7-7 zeigt die Syntax binärer Zahlen.



**Abb. 7-7:**  
Binäre Zahlen

Der Wertebereich binärer Zahlen ist &B0000000000000000 bis &B1111111111111111.

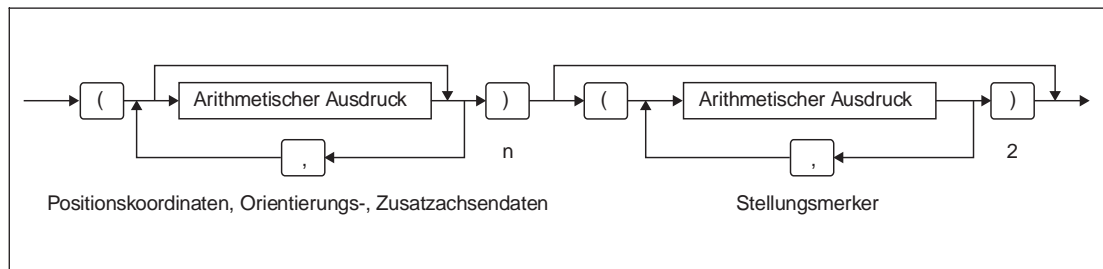
**Beispiel** ▾

&B010011; &B1101



## Positionskonstanten

Abbildung 7-8 zeigt die Syntax der Positionskonstanten.



**Abb. 7-8: Positionskonstanten**

Die Koordinaten, Stellsungsdaten und die zustzlichen Achsdaten 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. Ihre Einheit ist Grad.
- L1, L2 sind zusätzliche Achsendaten. Sie geben die Koordinaten für die zusätzlichen Achsen 1 und 2 an. Ihre Einheit ist mm oder Grad.

Geben Sie immer alle sechs Daten für die Position des Roboters an. Werden die Zusatzachsen verwendet, müssen auch diese Daten angegeben werden.

Bedeutung der Stellungen (fehlen die Stellungen, werden die Standardeinstellungen verwendet).

Struktur: FL1, FL2

- FL1: Stellungsmerker
- FL2: Geschwindigkeitsinformation

Variablen, die für die Positionskonstanten festgelegt sind, dürfen nicht in arithmetischen Operationen benutzt werden.

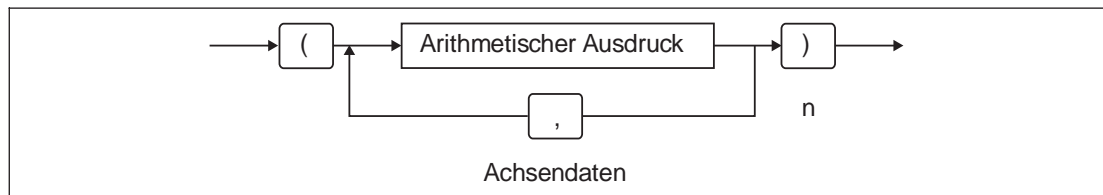
Bei unvollständigen Angaben der Positionskonstanten wird ein Syntax-Fehler gemeldet.

### Beispiel ▾

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


## Gelenkkonstanten

Abbildung 7-9 zeigt die Syntax der Gelenkkonstanten.



**Abb. 7-9: Gelenkkonstanten**

### Struktur und Bedeutung der Achsendaten:

Struktur: W, S, E, T, P, R, L1, L2

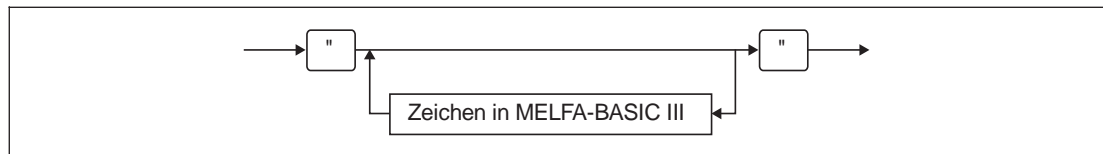
- W, S, E, T, P, R: Achsendaten (in mm oder Grad)
- L1, L2: Daten der Zusatzachsen (in mm oder Grad)

Variablen, die für die Gelenkkonstanten festgelegt sind, dürfen nicht in arithmetischen Operationen benutzt werden.

Bei unvollständigen Angaben der Gelenkkonstanten wird ein Syntax-Fehler gemeldet.

## Zeichenkettenkonstanten

Abbildung 7-10 zeigt die Syntax der Zeichenkettenkonstanten.



**Abb. 7-10:** Zeichenkettenkonstanten

Die maximale Länge einer Zeichenkette darf 127 Zeichen betragen. Bei mehr als 127 Zeichen wird ein Syntax-Fehler gemeldet.

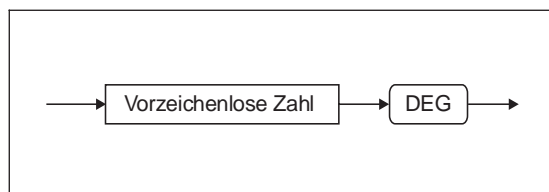
### Beispiel ▾

“Gefertigte Teile =”



### Winkelbetrag

Der Winkelbetrag wird in Grad (nicht in Radiant) angegeben. Abbildung 7-11 zeigt die Syntax des Winkelbetrages.



**Abb. 7-11:**  
Winkelbetrag

### Beispiel ▾

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

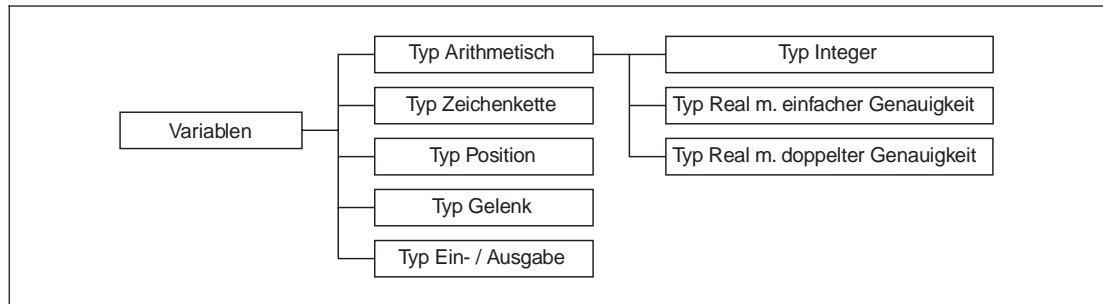


### 7.1.9 Variablen

Die folgenden Variablenwerte können eingesetzt werden. Sie unterscheiden sich anhand der Werte, die in ihnen abgelegt werden.

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

Abbildung 7-12 zeigt die Syntax von Variablen.



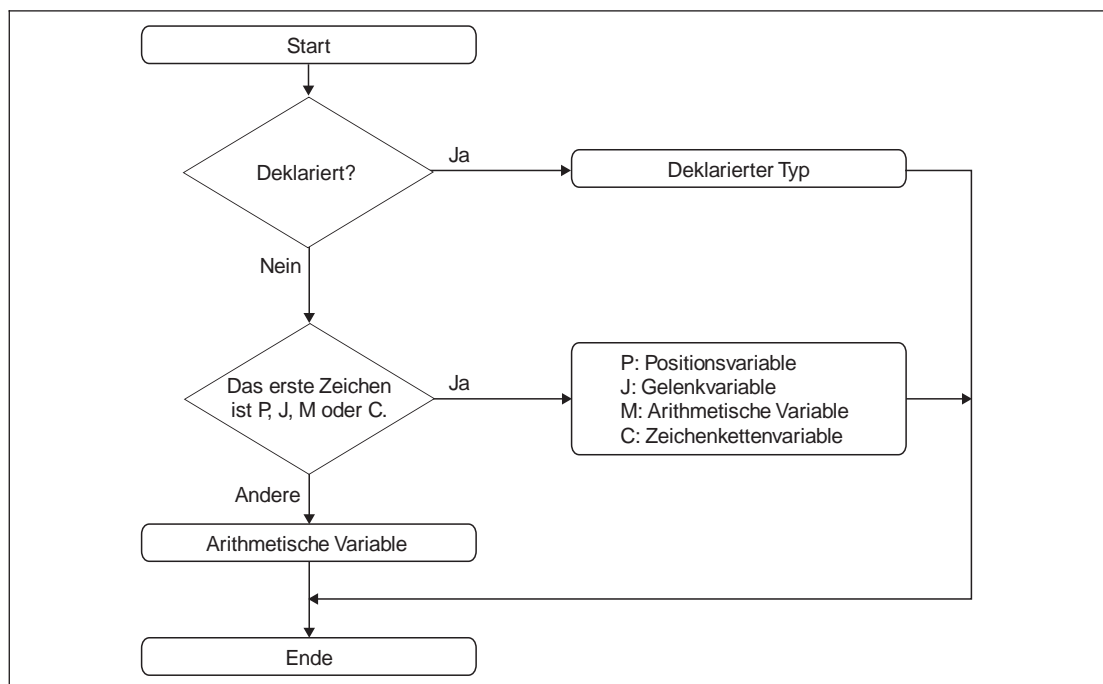
**Abb. 7-12:** Variablen

#### HINWEIS

Von der Software des Personalcomputers und der Teaching Box können nur numerierte Variablenwerte eingelesen und geschrieben werden (z. B. P1, P200, M1, M3 u.s.w.). Nicht numerierte Variablenwerte (z. B. PA, PWORK, MA, MB u.s.w.) können zwar in einem Programm verarbeitet, aber weder eingelesen noch geschrieben werden.

#### Implizite Typendeklaration

MELFA-BASIC III verwendet die implizite Typendeklaration, d.h. nur festgelegte Variablentypen können verwendet werden. Das Flußdiagramm in Abbildung 7-13 zeigt, wie ein Datentyp festgelegt wird.



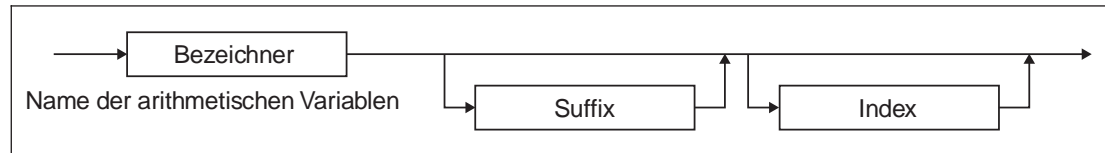
**Abb. 7-13:** Implizite Typendeklaration

### Arithmetische Variablen

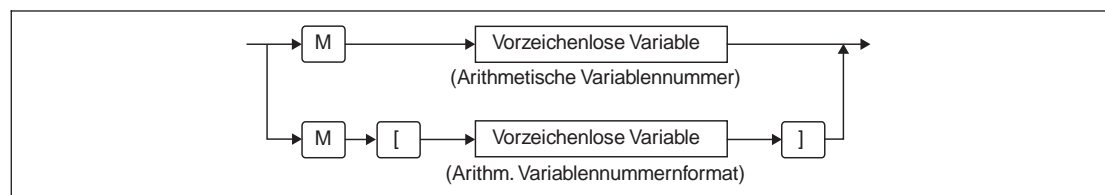
Es gibt zwei Arten von arithmetischen Variablen:

- Variablen ohne Ziffern
- Variablen mit Ziffern

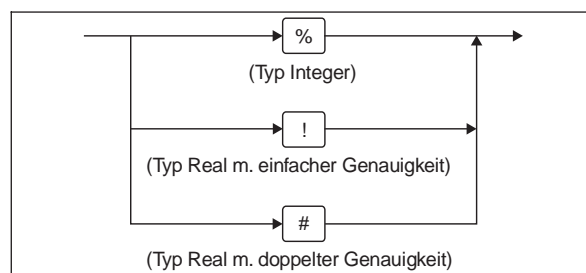
Nachfolgende Abbildungen zeigen die verschiedenen Variablentypen.



**Abb. 7-14:** Arithmetische Variablen ohne Ziffern



**Abb. 7-15:** Arithmetische Variablen mit Ziffern



**Abb. 7-16:**  
Suffix

#### HINWEISE

Fehlt das Suffix in der Variablendeklaration, so wird die Variable als reelle Zahl mit einfacher Genauigkeit interpretiert.

Werden Zahlen vom Typ Integer, reelle Zahlen mit einfacher Genauigkeit und reelle Zahlen mit doppelter Genauigkeit gemeinsam verarbeitet oder ersetzt, tritt ein Konvertierungsfehler in den Stellen auf, die auf die gültige Ziffer folgen. Der Fehler ist bei der Weiterverarbeitung der Rechenergebnisse in Vergleichsoperationen oder bedingten Verzweigungen unbedingt zu beachten.

Tabelle 7-3 zeigt den Wertebereich der verschiedenen Datentypen.

Typ	Bereich
Integer	–32 768 bis 32 767
Reelle Zahlen mit einfacher Genauigkeit	–1,70141E+38 bis 1,70141E+38
Reelle Zahlen mit doppelter Genauigkeit	–1,701411834604692E+38 bis 1,701411834604692E+38

**Tab. 7-3:** Wertebereich

Tabelle 7-4 zeigt die gültigen Variablennummern. Andere Nummern erzeugen eine Fehlermeldung.

Variablennummern	Bedeutung
1 bis 90	Lokale Programmvariablen

**Tab. 7-4:** Variablennummern

Der Standardwert arithmetischer Variablen ist 0.

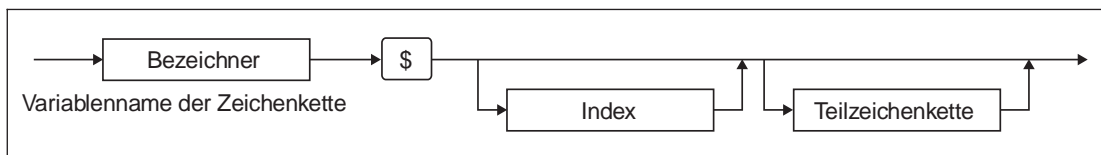
Informationen über die Indizierung finden Sie im Abschnitt 7.1.10 „Feldvariablen“ in diesem Handbuch. In Feldern dürfen keine numerierten Variablen verwendet werden.

### Zeichenketten-Variablen

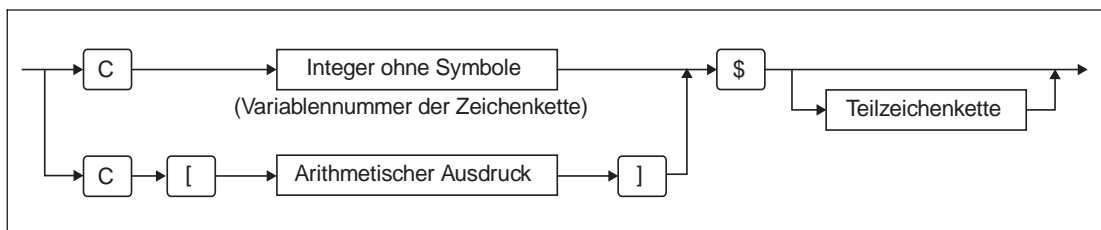
Es gibt zwei Formate für Zeichenketten-Variablen:

- Ein Format für Zeichenketten-Variablen mit Ziffern.
- Ein Format für Zeichenketten-Variablen ohne Ziffern.

Nachfolgende Abbildungen zeigen die Syntax von Zeichenketten-Variablen.



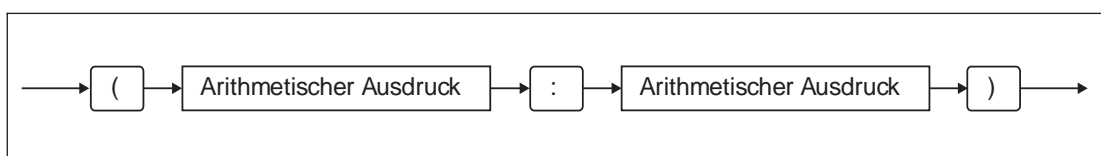
**Abb. 7-17:** Zeichenketten-Variablen



**Abb. 7-18:** Zeichenketten-Variablen mit Ziffern

#### Beispiel ▾

C1\$



**Abb. 7-19:** Teilzeichenkette

Der Standardwert für Zeichenketten-Variablen ist das Null-Zeichen ("").

Es können Zeichenketten mit maximal 127 Zeichen gespeichert werden. Jedes weitere Zeichen wird ignoriert.

Tabelle 7-5 zeigt die zulässigen Nummern der Zeichenketten-Variablen. Andere Nummern erzeugen eine Fehlermeldung.

Zeichenkettennummer	Bedeutung
1 bis 90	Lokale Programmvariablen

**Tab. 7-5:** Nummern von Zeichenketten-Variablen

Die Angabe  $a$(m:n)$  bezeichnet einen Teil einer Zeichenkette  $a$$  vom  $m$ -ten bis zum  $n$ -ten Zeichen. Von links nach rechts sind die Zeichen in aufsteigender Reihenfolge, mit 1 beginnend, numeriert. Liegen  $m$  oder  $n$  nicht im Bereich von 1 bis  $LEN(a$)$ , wird  $m$  als  $MAX(m,1)$  und  $n$  als  $MIN[n,LEN(a$)]$  festgelegt.

Wenn  $n < m < LEN(a$)$ , bezieht sich  $a$(m:n)$  auf das letzte leere Zeichen unmittelbar vor dem  $m$ -ten Zeichen.

Wenn  $LEN(a$) < m$ , bezieht sich  $a$(m:n)$  auf das letzte leere Zeichen unmittelbar vor  $a$$ .

#### HINWEIS

Informationen über die Indizierung finden Sie im Abschnitt 7.1.10 „Feldvariablen“ in diesem Handbuch. In Feldern dürfen keine numerierten Zeichenketten-Variablen verwendet werden.

### Positionsvariablen

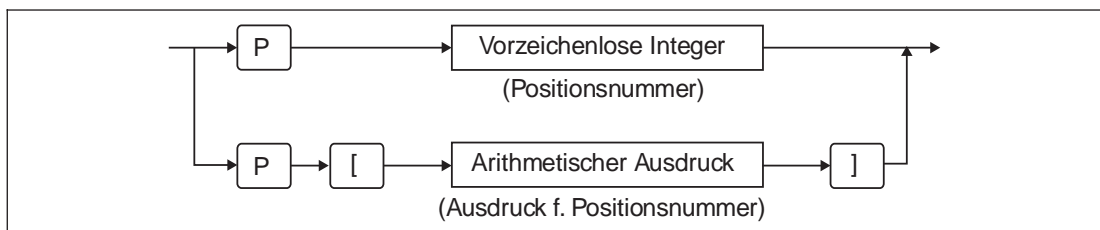
Es gibt zwei Formate für Positionsvariablen:

- Ein Format für Positionsvariablen mit Ziffern.
- Ein Format für Positionsvariablen ohne Ziffern.

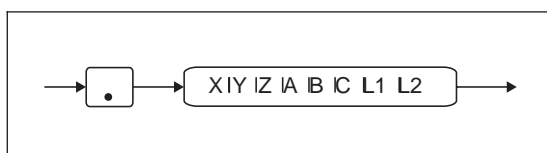
Nachfolgende Abbildungen zeigen die Syntax von Positionsvariablen.



**Abb. 7-20:** Positionsvariablen ohne Ziffern



**Abb. 7-21:** Positionsvariablen mit Ziffern



**Abb. 7-22:**  
Positionskomponenten

Eine Positionsvariable mit Ziffern beginnt mit dem Buchstaben „P“ und enthält Ziffern, die mit dem zweiten Zeichen beginnen.

Das Lesen einer Positionsvariable mit Ziffern beschreibt der formale Ausdruck „P[1-9]+“.

In Feldern dürfen keine numerierten Positionsvariablen verwendet werden.

Tabelle 7-6 zeigt die zulässigen Nummern der Positionsvariablen. Andere Nummern erzeugen eine Fehlermeldung.

Positionsnummern	Bedeutung
1 bis 900	Lokale Programmvariablen

**Tab. 7-6:** Positionsnummern

Der Standardwert der Positionsvariablen ist abhängig vom Modell des Roboters und den zusätzlichen Achsen.

#### HINWEISE

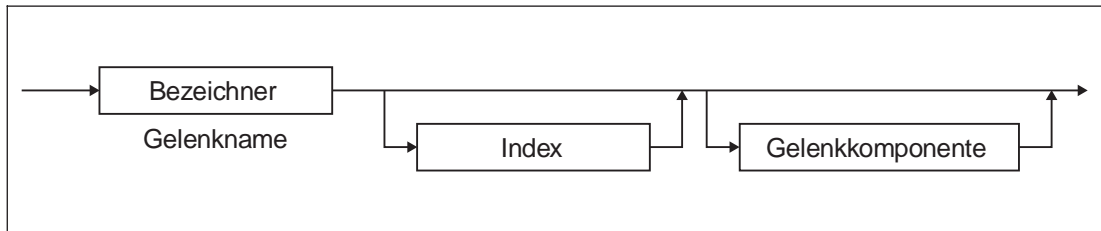
Informationen über die Indizierung finden Sie im Abschnitt 7.1.10 „Feldvariablen“ in diesem Handbuch. In Feldern dürfen keine numerierten Positionsvariablen verwendet werden.

Weitere Informationen über Positionskomponenten finden Sie in Abschnitt 7.1.8 unter „Positionskonstanten“.

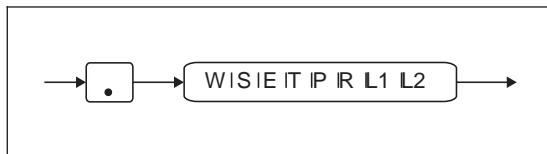


### Gelenkvariablen

Abbildungen 7-23 und 7-24 zeigen die Syntax von Gelenkvariablen.



**Abb. 7-23:** Gelenkvariablen



**Abb. 7-24:**  
Gelenkkomponenten

- Der Standardwert der Gelenkvariablen ist abhängig vom Modell des Roboters und den zusätzlichen Achsen.
- In Feldern dürfen keine numerierten Variablen verwendet werden.
- Es existieren keine numerierten Gelenkvariablen.

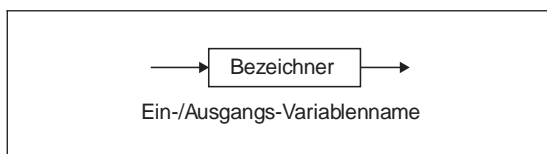
#### HINWEISE

Weitere Informationen über Gelenkkomponenten finden Sie in Abschnitt 7.1.8 unter „Gelenkkonstanten“.

Informationen über die Indizierung finden Sie im Abschnitt 7.1.10 „Feldvariablen“ in diesem Handbuch.

### E/A-Variablen

Abbildung 7-25 zeigt die Syntax von E/A-Variablen.

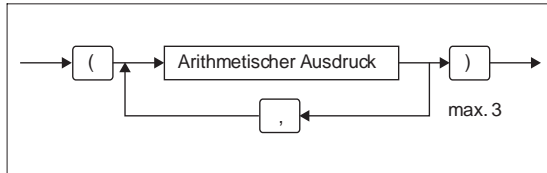


**Abb. 7-25:**  
E/A-Variablen

- Wird eine Variable als E/A-Variable deklariert, ist die Variable der entsprechende Adresse einer E/A-Schnittstelle zugewiesen.

### 7.1.10 Feldvariablen

Arithmetische Variablen ohne Ziffern, Zeichenketten-Variablen ohne Ziffern, Positionsvariablen ohne Ziffern und Gelenkvariablen können in Feldvariablen verwendet werden. Die Festlegung der Feldvariablenelemente finden Sie im Abschnitt für die Indizierung der Variablen. Die Syntax für die Indizierung zeigt Abbildung 7-26.



**Abb. 7-26:**  
Indizierung

#### HINWEISE

- | Deklarieren Sie die Feldvariablen bevor Sie sie verwenden.
- | Eine Feldvariable darf aus maximal 3 Dimensionen bestehen.
- | Wenn Sie die Indizierung nach dem Namen der Feldvariablen in Klammern setzen, können Sie mit einem Variablennamen mehrdimensionale Daten darstellen.
- | Feldvariablen werden wie andere Variablen verwendet.
- | Die Elemente der Feldvariablen können ein- oder zweidimensional sein.
- | Vor Ausführung des Programms müssen Sie die Anzahl der Elemente mit dem DIM-Befehl festlegen. Der Wertebereich ist dabei 1 bis 999.
- | Ist die Anzahl der Elemente nicht festgelegt, wird sie, für ein- oder zweidimensionale Elemente, auf 10 festgesetzt und dementsprechend weiterverarbeitet.
- | Die Position der einzelnen Elemente ist durch den Index festgelegt. Der kleinste Indexwert ist dabei 1. Es ist möglich einen variablen arithmetischen Ausdruck als Index zu verwenden.
- | Ein Feldvariablenname kann aus alphanumerischen Zeichen und dem Unterstrich bestehen. Der Feldname besteht aus einer Zeichenkette mit maximal 8 Zeichen, inklusive des führenden Zeichens, das zur Identifizierung des Variablentypes dient. (Das Dollarzeichen (\$) wird nicht mitgezählt.)

#### Numerische Feldvariablen

Numerische Feldvariablen können ganze oder reelle Zahlen sein.

#### Positions-Feldvariablen

Positions-Feldvariablen beinhalten die Daten für die Position und die Stellung des Roboters im kartesischen Koordinatensystem. Die Variablen können Daten für maximal 8 Achsen und den Konstruktionsmerker enthalten.

#### Gelenk-Feldvariablen

Gelenk-Feldvariablen beinhalten die Daten für die Position und die Stellung des Roboters (Gelenk-Achsendaten). Die Variablen können Daten für maximal 8 Achsen enthalten.

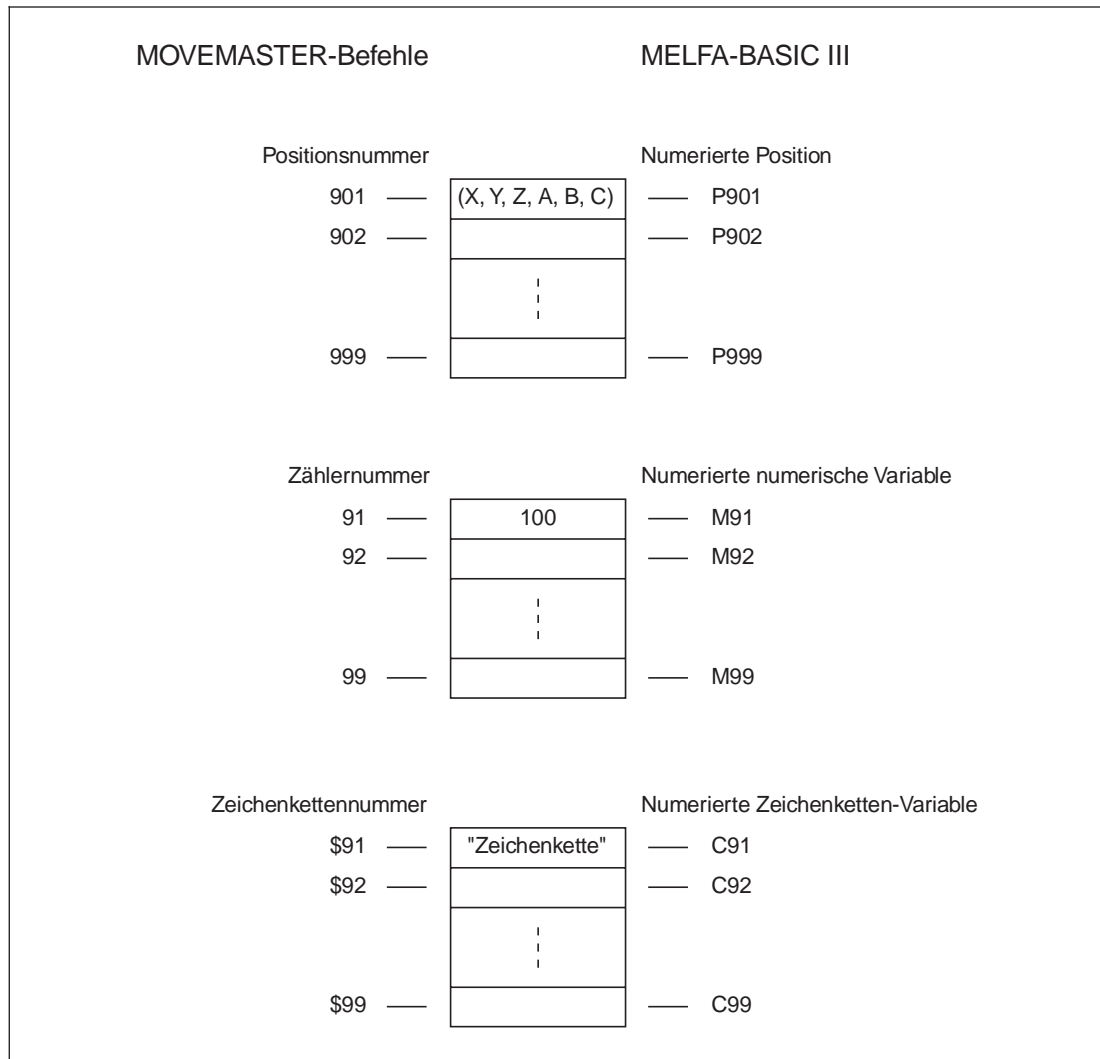
#### Zeichenketten-Variablen

Zeichenketten-Variablen enthalten Zeichenketten.

### 7.1.11 Programmexterne Variablen

MELFA-BASIC III ist so aufgebaut, daß die Daten aus dem globalen Bereich des MOVE-MASTER COMMAND-Mode weiterverwendet werden können.

Abbildung 7-27 zeigt die Beziehung zwischen den globalen Variablen im MOVEMASTER COMMAND-Format und den Variablen im MELFA-BASIC III-Format. Werden gemeinsame Daten im MOVEMASTER-Format von MELFA-BASIC III in die MOVEMASTER COMMAND-Befehlssprache übertragen, bleiben die Daten erhalten.



**Abb. 7-27:** Gemeinsame Variablen

Typ	Variablenname
Positionen	P901 bis P999
Zähler	M91 bis M99
Zeichenketten	C91 bis C99

**Tab. 7-7:** Gemeinsame Variablen MELFA-BASIC III

**Benutzer-Variablen**

Benutzer-Variablen sind numerische und Positionsvariablen, die von verschiedenen Programmen verwendet werden können. Innerhalb der Programme sind sie frei verwendbar.

**Roboterstatus-Variablen**

Roboterstatus-Variablen werden verwendet, um einen schnellen Zugriff auf den Roboterzustand zu ermöglichen oder um ihn zu ändern. Im einzelnen werden diese unterschieden in:

- Systemmanagement-Variablen  
Eine Systemmanagement-Variable enthält Daten, die in direkter Beziehung zum Roboterstatus stehen.
- Benutzermanagement-Variablen  
Eine Benutzermanagement-Variable enthält Daten, die nicht in direkter Beziehung zum Roboterstatus stehen.

Einteilung	Variablenname	Typ	Inhalt	Einheit	Reset	Bemerkung
Position	P_CURR	P	Augenblicksposition (kartesisch)	mm, deg		
	J_CURR	J	Augenblicksposition (Gelenk)	deg		
	P_TOOL	P	Zuletzt festgelegte Werkzeugkonvertierungsdaten	mm, deg	②	Wert ändert sich bei Ausführung des TOOL-Befehls
	P_BASE	P	Zuletzt festgelegte Basiskonvertierungsdaten	mm, deg	②	Wert ändert sich bei Ausführung des BASE-Befehls
	P_NTOOL	P	Standardwert der Werkzeugkonvertierungsdaten	mm, deg		Standardwert = (0,0,107,0,0,0)(0,0)
	P_NBASE	P	Standardwert der Basiskonvertierungsdaten	mm, deg		Standardwert = (0,0,0,0,0,0)(0,0)
Geschwindigkeit	M_OVRD	M	Zuletzt festgelegter Übersteuerungswert (Gültig für das gesamte Programm)	%	①	Wert ändert sich bei Ausführung des OVRD-Befehls
	M_JOVRD	M	Zuletzt festgelegter Übersteuerungswert (Gültig nur bei Gelenk-Interpolation)	%	①	Wert ändert sich bei Ausführung des JOVRD-Befehls
	M_SPD	M	Zuletzt festgelegter Geschwindigkeitswert (Gültig für Linear- und Kreisinterpolation)	mm/s	①	Wert ändert sich bei Ausführung des SPD-Befehls
	M_ACL	M	Zuletzt festgelegte Beschleunigungszeit	s	①	Wert ändert sich bei Ausführung des ACL-Befehls
	M_DACL	M	Zuletzt festgelegte Abbremszeit	s	①	Wert ändert sich bei Ausführung des DACL-Befehls
	M_NOVRD	M	Systemstandardwert (M_OVRD Standardwert)	%		Standardwert = 100
	M_NJOVRD	M	Systemstandardwert (M_JOVRD Standardwert)	%		Parameter: SP1
	M_NSPD	M	Systemstandardwert (M_SPD Standardwert)	mm/s		Parameter: SP1
	M_NACL	M	Systemstandardwert (M_ACL Standardwert)	s		Standardwert = 0.2
	M_NDACL	M	Systemstandardwert (M_DACL Standardwert)	s		Standardwert = 0.2
	M_RSPD	M	Aktuelle Geschwindigkeit (Gültig für Linear- und Kreis-Interpolation)	mm/s		
	M_ACLSTS	M	Aktueller Status der Beschleunigung-/Abbremsung 0 = gestoppt; 1 = beschleunigt; 2 = konstante Geschwindigkeit; 3 = bremst			
Konstanten	M_PI	M	Kreiszahl (3.1415...)			
	M_EXP	M	Basis des natürlichen Logarithmus (2.71828...)			
	M_G	M	Erdbeschleunigung (9.80665)			

**Tab. 7-8:** Roboterstatusvariablen (Systemmanagementvariablen) (1)

Einteilung	Variablenname	Typ	Inhalt	Einheit	Reset	Bemerkung
Andere	C_MAKER\$	C	Herstellerinformation (max. 64 Zeichen)			
	C_USER\$	C	Benutzerinformation			
	DATE\$	C	Aktuelles Datum Jahr/Monat/Datum			
	TIME\$	C	Aktuelle Zeit Stunde/Minute/Sekunde			
	M_BTIME	M	Default-Wert des Batteriezählers	Stunden		
	TIMER	M	Zeitdauer ab Bezugszeit	Sekunden		

**Tab. 7-8:** Roboterstatusvariablen (Systemmanagementvariablen) (2)

**HINWEIS**

Die in Tabelle 7-8 aufgeführten Variablen können nur gelesen werden.

Einteilung	Variablenname	Typ	Inhalt	Einheit	Reset	Bemerkung
Eingang	M_IN( )	M	Allgemeine Bit-Schnittstelle: Bit-Eingang 0 = AUS, 1 = EIN		⑥	Die Anzahl der Feldelemente beträgt 32 767
	M_INB( )	M	Allgemeine Bit-Schnittstelle: Byte-Eingang			Die Anzahl der Feldelemente beträgt 32 767
	M_INW( )	M	Allgemeine Bit-Schnittstelle: Wort-Eingang			Die Anzahl der Feldelemente beträgt 32 767
	M_DIN( )	M	Allgemeine Wort-Schnittstelle: Wort-Eingang			Die Anzahl der Feldelemente beträgt 10
Hand-eingang	M_HNDCQ( )	M	Eingang Handgreiferzustand 0 = AUS, 1 = EIN			Die Anzahl der Feldelemente beträgt 3 (1,2,3)
	M_HIN( )	M	Hand-Bit-Schnittstelle: Bit-Eingang			Die Anzahl der Feldelemente beträgt 32 767
	M_HINB( )	M	Hand-Bit-Schnittstelle: Byte-Eingang			Die Anzahl der Feldelemente beträgt 32 767
	M_HINW( )	M	Hand-Bit-Schnittstelle: Wort-Eingang			Die Anzahl der Feldelemente beträgt 32 767
Ausgang	M_OUT( )	M	Allgemeine Bit-Schnittstelle: Bit-Ausgang 0 = AUS, 1 = EIN		④	Die Anzahl der Feldelemente beträgt 32 767
	M_OUTB( )	M	Allgemeine Bit-Schnittstelle: Byte-Ausgang		④	Die Anzahl der Feldelemente beträgt 32 767
	M_OUTW( )	M	Allgemeine Bit-Schnittstelle: Wort-Ausgang		④	Die Anzahl der Feldelemente beträgt 32 767
	M_DOUT	M	Allgemeine Wort-Schnittstelle: Wort-Ausgang		④	Die Anzahl der Feldelemente beträgt 32 767
Hand-ausgang	M_HND( )	M	Befehlsausgang Handgreifer öffnen/schließen 0 = schließen, 1 = öffnen		③	Die Anzahl der Feldelemente beträgt 3 (1,2,3)
	M_HOUT( )	M	Hand-Bit-Schnittstelle: Bit-Ausgang			Die Anzahl der Feldelemente beträgt 32 767
	M_HOUTB( )	M	Hand-Bit-Schnittstelle: Byte-Ausgang			Die Anzahl der Feldelemente beträgt 32 767
	M_HOUW( )	M	Hand-Bit-Schnittstelle: Wort-Ausgang			Die Anzahl der Feldelemente beträgt 32 767
Position	P_SAFE	P	Position des Rückzugpunktes	mm, deg	⑤	

**Tab. 7-9: Roboterstatus-Variablen (Benutzermanagement-Variablen)**

#### HINWEIS

Die Eingangs- und Handeingangsvariablen können nur gelesen werden. Alle anderen Variablen können übergeben und ersetzt werden.

**Hinweise zur Spalte Reset in Tabelle 7-9:**

- ① Der Wert wird auf den Standardwert (Wert wird mittels Parameter festgelegt) zurückgesetzt, sobald die Spannungsversorgung ausgeschaltet oder das Programm zurückgesetzt wird. Bei Ausführung des entsprechenden Befehls (siehe letzte Spalte) ändert sich der Wert. (Programmabhängig.)
- ② Der Wert bleibt durch die Batterie auch beim Abschalten der Spannungsversorgung erhalten. Bei Ausführung des entsprechenden Befehls (siehe letzte Spalte) ändert sich der Wert.
- ③ Wird zurückgesetzt, wenn die aktuelle Variable durch eine neue ersetzt wird, oder bei Ausführung des HND-Befehls.
- ④ Wird zurückgesetzt, wenn die aktuelle Variable durch eine neue ersetzt wird, oder bei Ausführung des OUT-Befehls.
- ⑤ Wird zurückgesetzt, wenn die aktuelle Variable durch eine neue ersetzt wird.
- ⑥ Das erste Feldelement der Variablen vom Typ M wird auf „0“ gesetzt.



### 7.1.12 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. Tabelle 7-10 zeigt die logischen Werte und deren Bedeutung.

Durch eine 1 dargestellte Zustände	Durch eine 0 dargestellte Zustände
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. 7-10:** Logische Werte und deren Bedeutung

### 7.1.13 Komponentendaten

Komponentendaten sind die Werte der einzelnen Komponenten einer Variablen, die aus mehreren Komponenten besteht.

Komponentendaten werden wie numerische Datentypen verarbeitet. Die Daten können sich bei Positions- oder Gelenkvariablen auf verschiedene Komponenten einer Variablen beziehen.

Jede Komponente kann vom vorher beschriebenen Datentyp (siehe Abschnitt 7.1.7) sein. (Positions- und Gelenkkomponentendaten sind vom numerischen Datentyp.)

Hinter dem Variablennamen steht ein Punkt, dann folgt der Komponentename.

Variablennamen können Positionsvariablennamen, Positions-Feldvariablennamen, Gelenkvariablen oder Gelenk-Feldvariablennamen sein.

#### Beispiel ▽

P10=P5.X Der X-Wert der Position 5 wird in Position 10 abgelegt.



#### Positionsachsensdaten

Positionsachsensdaten bezeichnen den Wert der Positionsvariablendaten (Feldvariablen) der ersten Achse.

#### Stellungsmerkerdaten

Stellungsmerkerdaten beinhalten Informationen für die Positionsvariablen (Feldvariablen).

#### Gelenkachsensdaten

Gelenkachsensdaten bezeichnen den Wert der Gelenkvariablendaten für die erste Achse.

Komponentendaten	Name der Variablen	Name der Komponente
Positionsachsensdaten	Positionsvariablenname Positions-Feldvariablenname	X-Achse ... X, Y-Achse ... Y, Z-Achse ... Z, A-Achse ... A, B-Achse ... B, C-Achse ... C, Zusatzachse 1 ... L1, Zusatzachse 2 ... L2
Stellungsmerker	Positionsvariablenname Positions-Feldvariablenname	Stellungsmerker 1 ... FL1, Stellungsmerker 2 ... FL2
Gelenkachsensdaten	Gelenkvariablenname Gelenk-Feldvariablenname	Achse 1 ... W, Achse 2 ... S, Achse 3 ... E, Achse 4 ... T, Achse 5 ... P, Achse 6 ... R, Achse 7 ... L1, Achse 8 ... L2

**Tab. 7-11:** Komponentendaten

## 7.2 Ausdrücke und Operationen

### 7.2.1 Gruppeneinteilung der Ausdrücke und Operationen

Ausdrücke beinhalten Operationen mit Konstanten und Variablen. In Tabelle 7-12 sind 4 Arten von Ausdrücken und Operationen dargestellt.

Ausdruck	Zugelassene Daten	Zugelassene Operationen
Numerische Operationsausdrücke	Typ Numerisch, Typ Zeichenkette (einschließlich Komponentendaten)	Arithmetische Operationen, Vergleichsoperationen, logische Operationen, Funktionen
Operationsausdrücke mit Positionen	Typ Position	Arithmetische Operationen <sup>①</sup>
Operationsausdrücke mit Gelenken	Typ Gelenk	Arithmetische Operationen <sup>①</sup>
Operationsausdrücke mit Zeichenketten	Typ Zeichenkette	Arithmetische Operationen, Vergleichsoperationen <sup>①</sup>

**Tab. 7-12:** Arten von Ausdrücken und Operationen

<sup>①</sup> Die Operationen in MELFA-BASIC III unterscheiden sich von denen in herkömmlichen Basic-Dialekten.

Auch mit Datentypen, die nicht in Tabelle 7-12 erwähnt sind, lassen sich Operationen definieren und Substitutionen, Additionen und Subtraktionen durchführen.

### 7.2.2 Arithmetische Operationen

Folgende arithmetische Operationen sind verfügbar:

^ ... Exponenten	¥ ... Integer-Division
– ... Vorzeichenumkehr	MOD ... Modulo-Arithmetik
* ... Multiplikation	+ ... Addition
/ ... Division	– ... Subtraktion

#### Numerische arithmetische Operationen

Bei Befehlen, die nur ganzzahlige Werte verarbeiten, wird das Ergebnis gerundet, falls das Ergebnis eine reelle Zahl ist (4: abrunden, 5: aufrunden).

#### Arithmetische Operationen mit Zeichenketten

Zeichenketten lassen sich durch die Verwendung des Additionszeichens verbinden.  
 "ABC"+"XYZ" → "ABCXYZ"

### Arithmetische Operationen mit Positionstypen

Bei allen Operationen wird der Stellungsmerker des linken Operanden übernommen.

- Vorzeichenumkehr:  $\neg \langle \text{Position} \rangle$   
Das Vorzeichen wird für jeden Wert der Position umgekehrt.  
Der Konstruktionsmerker wird nicht von der Vorzeichenumkehr beeinflusst.

#### Beispiel ▾

$\neg(280, -100, 312.5, -17.23, 30, 120)(1, 0) \rightarrow (-280, 100, -312.5, 17.23, -30, -120)(1, 0)$

△

- Addition von Positionstypen:  $\langle \text{Position} \rangle + \langle \text{Position} \rangle$   
Für die Addition von Positionsdaten gelten die Regeln der Vektoraddition. Die Addition wird für jede Achse durchgeführt.

#### Beispiel ▾

$\text{PDATA1} = (300, 300, 100, 10, 20, 110)(1, 0)$   
 $\text{PDATA2} = (20, -30, -20, 30, 10, 30)(0, 1)$   
 $\text{PDATA1} + \text{PDATA2} = (320, 270, 80, 40, 30, 140)(1, 0)$

△

- Subtraktion von Positionstypen:  $\langle \text{Position} \rangle - \langle \text{Position} \rangle$   
Für die Subtraktion von Positionsdaten gelten die Regeln der Vektorsubtraktion. Die Subtraktion wird für jede Achse durchgeführt
- Multiplikation von Positionstypen:  $\langle \text{Position P1} \rangle * \langle \text{Position P2} \rangle$   
Es wird eine relative Konvertierung der Positionsdaten durchgeführt.  
„Relative Konvertierung“ bedeutet eine Änderung der Position und Stellung von P2 im Koordinatensystem von P1.  
Die Reihenfolge der Operanden darf nicht vertauscht werden, d. h.  $P1 * P2 \neq P2 * P1$
- Division von Positionstypen:  $\langle \text{Position} \rangle / \langle \text{Position} \rangle$   
Es wird eine umgekehrte relative Konvertierung der Positionsdaten durchgeführt.  
Wenn durch die Multiplikation zweier Positionstypen (siehe Beispiel oben) ein Wert  $P3 = P1 * P2$  mit einer neuen Position und Stellung entsteht, wird die Berechnung von P1 aus den Daten von P2 und P3 „umgekehrte relative Konvertierung“ genannt ( $P1 = P3 / P2$ ).  
Es ist nicht möglich, P2 aus den Daten P1 und P3 zu berechnen.
- Substitution von Positionstypen:  $\langle \text{Position} \rangle = \langle \text{Position} \rangle$   
Die Daten der linken Position werden durch die Daten der rechten Position ersetzt. Die Substitution wird für jede Achse durchgeführt.

### Arithmetische Operationen mit Gelenktypen

- Addition von Gelenktypen:  $\langle \text{Gelenk} \rangle + \langle \text{Gelenk} \rangle$   
Für die Addition von Gelenkdaten gelten die Regeln der Vektoraddition. Die Addition wird für jede Achse durchgeführt.
- Subtraktion von Gelenktypen:  $\langle \text{Gelenk} \rangle - \langle \text{Gelenk} \rangle$   
Für die Subtraktion von Gelenkdaten gelten die Regeln der Vektorsubtraktion. Die Subtraktion wird für jede Achse durchgeführt.
- Substitution von Gelenktypen:  $\langle \text{Gelenk} \rangle = \langle \text{Gelenk} \rangle$   
Die Daten der linken Seite werden durch die Daten der rechten Seite ersetzt. Die Substitution wird für jede Achse durchgeführt.

### Division durch 0 und negative Exponenten

Eine Division durch 0 und negative Exponenten erzeugen bei der Ausführung der Operation einen Fehler.

### Überlauf und Unterlauf

Auch wenn das Ergebnis einer Rechenoperation den für diese Berechnung zulässigen Wertebereich verläßt, wird das Ergebnis akzeptiert, solange es innerhalb des zulässigen Wertebereiches des Systemes liegt.

## 7.2.3 Vergleichsoperationen

Vergleichsoperationen beziehen sich auf den Vergleich von zwei numerischen Variablen oder von Feldvariablen. Das Ergebnis von Vergleichsoperationen ist entweder „wahr“ (1) oder „unwahr“ (0).

Folgende Vergleichsoperationen sind möglich:

= ... gleich	> ... größer als
<> >< ... ungleich	<= =< ... kleiner oder gleich
< ... kleiner als	>= => ... größer oder gleich

### Vergleichsoperationen mit Zeichenketten

Tabelle 7-13 zeigt Besonderheiten von Vergleichsoperationen mit Zeichenketten.

Beim Vergleich von Zeichenketten werden die Codes (z. B. ASCII-Code) jedes einzelnen Zeichens, beginnend mit dem ersten Zeichen, miteinander verglichen.

Vergleichsoperation	Ergebnis
Wenn die Zeichencodes für alle Zeichen übereinstimmen.	Die Zeichenketten sind gleich.
Einige Zeichencodes stimmen nicht überein.	Die Zeichenkette, deren Zeichencodewert für das erste abweichende Zeichen kleiner ist, ist die kleinere Zeichenkette.
Wenn das Ende einer der Zeichenketten erreicht wird.	Die kürzere Zeichenkette ist die kleinere.

**Tab. 7-13:** Vergleichsoperationen mit Zeichenketten

Das führende Zeichen und das abschließende leere Zeichen gehen mit in die Vergleichsoperation ein.

## 7.2.4 Logische Operationen

Logische Operationen umfassen Bit-Operationen und Boolesche Operationen, deren Ergebnis „wahr“ (1) oder „unwahr“ (0) ist. Es können zwei oder mehrere Bedingungen verknüpft werden. Somit ermöglichen logische Operationen die Steuerung des Programmflusses.

Folgende logische Operationen sind möglich:

<< ... Logische Linksverschiebung	AND ... Logisches UND
>> ... Logische Rechtsverschiebung	OR ... Logisches ODER
NOT ... Negation	XOR ... Exklusives ODER

Bei der Ausführung logischer Operationen werden numerische Werte in ganzzahlige Werte (–32 768 bis 32 767) konvertiert und bitweise weiterverarbeitet. Wenn beide numerischen Werte 0 oder 1 sind, ist das Ergebnis der logischen Operationen ebenfalls 0 oder 1.

Folgende logische Operationen sind möglich:

- <<: Logische Linksverschiebung  
Der numerische Wert X wird durch das Bit Y nach links verschoben. Das verschobene Bit wird 0.
- >>: Logische Rechtsverschiebung  
Der numerische Wert X wird durch das Bit Y nach rechts verschoben. Das verschobene Bit wird 0.
- NOT: Negation  
Dient zur Erzeugung des Zweier-Komplements.
- AND: Logisches UND  
Dient zur UND-Verknüpfung numerischer Variablen.
- OR: Logisches ODER  
Dient zur ODER-Verknüpfung numerischer Variablen.
- XOR: Exklusives ODER  
Dient zur exklusiven ODER-Verknüpfung (Antivalenz) numerischer Variablen.

## 7.2.5 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 Funktionsnamen

Funktionsnamen können aus alphanumerischen Zeichen und dem Unterstrich bestehen. Funktionen beginnen mit den Zeichen „FN“. Das dritte Zeichen dient zur Beschreibung des Datentyps.

(Zeichenkette: C, Numerischer Wert: M, Position: P, Gelenk: J)

Bei einer Zeichenkette wird hinter dem Namen ein „\$“-Zeichen gesetzt.

Es können bis zu 8 Zeichen verwendet werden. Das „\$“-Zeichen wird nicht mitgezählt.

### Verschachtelte Funktionen

Es können bis zu 16 Ebenen in einer FN-Definition beschrieben sein.

Dabei können fest definierte oder benutzerdefinierte Funktionen verwendet werden.

Tabelle 7-14 zeigt die fest definierten Funktionen.

Funktionsart	Funktionsname (Format)	Bedeutung	Ergebnis
Numerische Funktionen	ABS (<Numerischer Ausdruck>)	Bildet den Betrag.	Numerischer Wert
	CINT (<Numerischer Ausdruck>)	Rundet den dezimalen Wert zu einer Integer-Zahl.	
	DEG (<Numerischer Ausdruck: radian>)	Wandelt die Einheit des Winkels von Radiant (rad) in Grad (deg) um.	
	DEGRAD (<Numerischer Ausdruck: degree>)	Wandelt die Einheit des Winkels von Grad (deg) in Radiant (rad) um.	
	EXP (<Numerischer Ausdruck>)	Berechnet den Wert der Exponentialfunktion.	
	FIX (<Numerischer Ausdruck>)	Erzeugt einen Integer-Sektor.	
	INT (<Numerischer Ausdruck>)	Erzeugt die größtmögliche Integer-Zahl, die kleiner als der Wert des numerischen Ausdrucks ist.	
	LEN (<Ausdruck für eine Zeichenkette>)	Berechnet die Länge der Zeichenkette.	
	LN (<Numerischer Ausdruck>)	Berechnet den natürlichen Logarithmus.	
	LOG (<Numerischer Ausdruck>)	Berechnet den dekadischen Logarithmus.	
	MAX (<Numerischer Ausdruck>...)	Berechnet den Maximalwert der numerischen Ausdrücke.	
	MIN (<Numerischer Ausdruck>)	Berechnet den Minimalwert der numerischen Ausdrücke.	
	RAD (<Numerischer Ausdruck: deg>)	Wandelt die Einheit des Winkels von Grad (deg) in Radiant (rad) um.	
	RADDEG (<Numerischer Ausdruck: rad>)	Wandelt die Einheit des Winkels von Radiant (rad) in Grad (deg) um.	
	SGN (<Numerischer Ausdruck>)	Prüft das Vorzeichen des numerischen Ausdrucks.	
	SQR (<Numerischer Ausdruck>)	Berechnet die Quadratwurzel.	
	STRPOS (<Ausdruck für eine Zeichenkette>, <Ausdruck für eine Zeichenkette>)	Gibt die Position der zweiten Zeichenkette innerhalb der ersten Zeichenkette an.	
	RND (<Numerischer Ausdruck>)	Rundet den numerischen Ausdruck.	

**Tab. 7-14:** Fest definierte Funktionen (1)

Funktionsart	Funktionsname (Format)	Bedeutung	Ergebnis
Numerische Funktionen	ASC (<Typ Zeichenkette>)	Erzeugt den ASCII-Code für das erste Zeichen in der Zeichenkette. (MELFA-BASIC)	Numerischer Wert
	ORD (<Typ Zeichenkette>)	Erzeugt einen Code für das erste Zeichen in der Zeichenkette. (SLIM)	
	CVI (<Typ Zeichenkette>)	Wandelt eine 2-byte Zeichenkette in einen Integer-Wert um.	
	CVS(<Typ Zeichenkette>)	Wandelt eine 4-byte Zeichenkette in einen Real-Wert um.	
	VAL (<Typ Zeichenkette>)	Wandelt eine Zeichenkette in einen numerischen Wert um.	
Trigonometrische Funktionen	ATN (<Numerischer Ausdruck>)	Berechnet den Arcus Tangens. Definitionsbereich: $-180^\circ$ bis $+180^\circ$ (außer 0)	Numerischer Wert
	ATN2 (<Numerischer Ausdruck>, <Numerischer Ausdruck>)	Berechnet den Arcus Tangens. ( $ATN2 = ATN(X/Y)$ ) Definitionsbereich: $-180^\circ$ bis $+180^\circ$ (außer 0)	
	COS (<Numerischer Ausdruck>)	Berechnet den Kosinus. Definitionsbereich: $-1$ bis $+1$ Einheit: Grad	
	SIN (<Numerischer Ausdruck>)	Berechnet den Sinus. Definitionsbereich: $-1$ bis $+1$ Einheit: Grad	
	TAN (<Numerischer Ausdruck>)	Berechnet den Tangens. Definitionsbereich: numerischer Wertebereich Einheit: Grad	
Zeichenkettenfunktionen	BIN\$ (<Numerischer Ausdruck>)	Wandelt den Wert des numerischen Ausdrucks in eine binäre Zeichenkette um.	Zeichenkette
	CHR\$ (<Numerischer Ausdruck>)	Erzeugt ein Zeichen, das dem Wert des numerischen Ausdrucks entspricht.	
	HEX\$ (<Numerischer Ausdruck>)	Wandelt den Wert des numerischen Ausdrucks in eine hexadezimale Zeichenkette um.	
	LEFT\$ (<Zeichenkette>, <Numerischer Ausdruck>)	Erzeugt einen Teil der Zeichenkette. Die Länge der erzeugten Zeichenkette, beginnend mit dem linken Zeichen, ist im zweiten Argument festgelegt.	
	MID\$ (<Zeichenkette>, <Numerischer Ausdruck>, >Numerischer Ausdruck>)	Erzeugt einen Teile der Zeichenkette. Die Länge der erzeugten Zeichenkette ist im dritten, die Position von links im zweiten Argument festgelegt.	
	MIRROR\$ (<Typ Zeichenkette>)	Spiegelung der binären Bits der Zeichenkette.	
	MKI\$ (<Numerischer Ausdruck>)	Wandelt den Wert des numerischen Ausdrucks in eine binäre Zeichenkette um.	
	MKS\$ (<Numerischer Ausdruck>)	Wandelt den Wert des numerischen Ausdrucks in eine hexadezimale Zeichenkette um.	
	RIGHT\$ (<Zeichenkette>, <Numerischer Ausdruck>)	Erzeugt einen Teil der Zeichenkette. Die Länge der erzeugten Zeichenkette, beginnend mit dem rechten Zeichen, ist im zweiten Argument festgelegt.	
	STR\$ (<Numerischer Ausdruck>)	Wandelt den Wert des numerischen Ausdrucks in eine dezimale Zeichenkette um.	
Positionsvariablen	POSX (<Position>)	Laden der X-Komponente.	Numerischer Wert
	POSY (<Position>)	Laden der Y-Komponente.	
	POSZ (<Position>)	Laden der Z-Komponente.	
	DIST (<Position>, <Position>)	Berechnet den Abstand zwischen zwei Punkten.	

Tab. 7-14: Fest definierte Funktionen (2)



### Anwendung von Funktionen

#### HINWEISE

- | Für leere Zeilen oder das Zeichen (-) sind Funktionen nicht definiert.
- | Bei Multiplikationen ganzzahliger Werte ist das Ergebnis ebenfalls ganzzahlig. Bei der Division kann das Ergebnis eine reelle Zahl sein.
- | Einige Funktionen können mit dem Argument 0 nicht berechnet werden. Eine Division durch 0 ist nicht erlaubt.
- | Bei der Ausführung exponentieller, modulo und logischer Operationen werden reelle Zahlen vor der Verarbeitung in ganzzahlige Werte umgewandelt und abgerundet.
- | Neu deklarierte Datentypen, die im Zweig „Andere“ liegen (siehe Abbildung 7-13), enthalten automatisch den kleinsten möglichen Wert.

## 7.2.6 Konvertierte Datentypen

Numerische Variablen müssen in MELFA-BASIC III nicht als Integer- oder reelle Zahl 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. Tabelle 7-15 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. 7-15:** Operationsergebnisse in Abhängigkeit der Datenreihenfolge

### Konvertierung der Datentypen in Abhängigkeit der Operation

Tabelle 7-16 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
				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	—
		Modul	—	Integer	Integer	—
		Exponential	—	Integer	Real	—
		Substitution	—	Integer	Integer	—
		Vergleich	—	Integer	Integer	—
	Real	Logisch	—	Integer	Integer	—
		Addition	—	Real	Real	—
		Subtraktion	—	Real	Real	—
		Multiplikation	—	Real	Real	—
		Division	—	Real	Real	—
		Integer-Division	—	Integer	Integer	—
		Modul	—	Integer	Integer	—
		Exponential	—	Integer	Real	—
		Substitution	—	Integer	Real	—
		Vergleich	—	Integer	Integer	—
		Logisch	—	Integer	Integer	—


**Tab. 7-16:** 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	—	—	—	—	—
	Multiplikation	—	Gelenk	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. 7-16:** Konvertierung der Datentypen (2)

### 7.2.7 Rangfolge von Operationen

Werden in einem Ausdruck mehrere Operationen ausgeführt, gilt die in Tabelle 7-17 dargestellte Rangfolge.

Operation (Operator)	Typ der Operation	Priorität
Operation in Klammern ( )	—	<div style="text-align: center;"> Hoch    Niedrig </div>
Funktion	Funktion	
Exponential (^)	Arithmetische Operation	
Operation mit einem Argument (+, -)	Arithmetische Operation	
* /	Arithmetische Operation	
¥	Arithmetische Operation	
MOD	Arithmetische Operation	
+ -	Arithmetische Operation	
<< >>	Logische Operation	
Vergleichsoperation (=, <>, ><, <, <=, =<, >, >=, =>)	Vergleichsoperation	
NOT	Logische Operation	
AND	Logische Operation	
OR	Logische Operation	
XOR	Logische Operation	

**Tab. 7-17:** Rangfolge von Operationen

### 7.2.8 Programmebenen

Beim Entwurf eines Programmes muß die Anzahl der Ebenen und die Struktur festgelegt werden. Werden die in Tabelle 7-18 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)
7 Schritte	Funktionsaufruf (CALLP)
Unbegrenzt	Unterprogrammaufruf (GOSUB) <sup>①</sup>

**Tab. 7-18:** Programmebenen

<sup>①</sup> Bei jedem Rücksprung aus einem Unterprogramm, wird die Programmstruktur eine Ebene flacher.

### 7.2.9 Reservierte Wörter

Reservierte Wörter haben im System eine bestimmte, festliegende Bedeutung. Sie dürfen zum Beispiel nicht als Programmname etc. vergeben werden. Tabelle 7-19 zeigt eine Übersicht der reservierten Wörter.

Anfangsbuchstabe	Reservierte Wörter
A	ABS, ACL, ACT, ALIGN, AND, APPEND, AS, ATN2
B	BASE
C	CALLP, CHR, CINT, CNT, COM, COS, CVI, CVS
D	DEF, DIM, DLY, DACL
E	END, ELSE, EXP
F	FINE, FIX, FOR, FPRM
G	GOSUB, GOTO
H	HLT, HND, HRE
I	IF, IN, INPUT, INT
J	JOVRD
L	LOG
M	MKI, MKS, MOD, MOV, MSP, MVC, MVR, MVR2, MVS
N	NEXT, NOT
O	OFF, ON, OPEN, OR, OUT, OUTPUT, OVRD
P	PLT, PRINT
R	RETURN, RND, REM
S	SGN, SIN, SKIP, SPD, SQR, STEP, STOP, STR
T	TAN, THEN, TO, TOOL
V	VAR
X	XOR
W	WHILE, WEND, WTH, WTHIF

**Tab. 7-19:** Reservierte Wörter



## 8 BASIC-Befehle

### 8.1 Allgemeine Hinweise

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

#### Beschreibung des verwendeten Formats

- Großbuchstaben müssen als solche eingegeben werden.
- Der Inhalt der Klammer ( ) kennzeichnet die notwendigen Befehlsparameter. Sie sind vom Benutzer einzugeben. Dabei sind bei der Eingabe die Erläuterungen zu den jeweiligen Befehlen zu beachten.
- Die eckige Klammer ( [ ] ) kennzeichnet die wahlfreien Befehlsparameter.
- Die unterbrochene vertikale Linie ( | ) bedeutet eine wahlweise Eingabe einer der beiden angegebenen Parameter. Die Angabe <Zeilennummer | Marke> bedeutet, daß entweder eine Zeilennummer oder eine Marke eingegeben werden kann.
- Zeichen wie Kommas ( , ), Punkte ( . ), Semikolons ( ; ), und Klammern ( ) müssen mit eingegeben werden.
- Befehle, die durch das Zeichen ♦ gekennzeichnet sind, können nicht einzeln ausgeführt werden. Sie werden immer in Verbindung mit anderen Befehlen ausgeführt.
- Befehle, die durch das Zeichen ♦♦ gekennzeichnet sind, können sowohl in Verbindung mit anderen Befehlen als auch eigenständig ausgeführt werden.

## 8.2 Übersicht der MELFA-BASIC III-Befehle

Befehl		Funktion	Abschnitt	Seite
ACL	(Accelerate)	Beschleunigung einstellen	8.2.1	8-4
ACT	(Act)	Interrupt freigeben	8.2.2	8-6
ALIGN	(Align)	Hand ausrichten	8.2.3	8-8
BASE	(Base)	Basis	8.2.4	8-9
CALLP	(Call P)	Programm aufrufen	8.2.5	8-11
CLOSE	(Close)	Datei schließen	8.2.6	8-12
CNT	(Control)	Roboterbewegung steuern	8.2.7	8-13
COM OFF	(Communication OFF)	Kommunikations-Interrupt sperren	8.2.8	8-16
COM ON	(Communication ON)	Kommunikations-Interrupt freigeben	8.2.9	8-17
COM STOP	(Communication Stop)	Kommunikations-Interrupt stoppen	8.2.10	8-18
DACL	(Deceleration)	Abbremszeit einstellen	8.2.11	8-19
DEF ACT	(Define act)	Interrupt-Prozeß definieren	8.2.12	8-21
DEF FN	(Define function)	Funktion definieren	8.2.13	8-22
DEF PLT	(Define pallet)	Palette definieren	8.2.14	8-24
DIM	(Dim)	Dimension definieren	8.2.15	8-26
DLY	(Delay)	Verzögerung einstellen	8.2.16	8-27
END	(End)	Programmende	8.2.17	8-28
FINE	(Fine)	Feinpositionierung	8.2.18	8-29
FOR-NEXT	(For-next)	Programmschleife	8.2.19	8-31
FPRM	(FPRM)	Parameter definieren	8.2.20	8-33
GOSUB	(Go Subroutine)	Sprung zu einem Unterprogramm	8.2.21	8-34
GOTO	(Go To)	Sprung zu einer Programmzeile oder Marke	8.2.22	8-35
HLT	(Halt)	Programmablauf stoppen	8.2.23	8-36
HND ♦♦	(Hand)	Handgreiferzustand festlegen	8.2.24	8-37
HRE ♦	(Here)	Aktuelle Position finden	8.2.25	8-39
IF THEN ELSE	(If Then Else)	WENN ... DANN ... SONST-Schleife	8.2.26	8-40
IN ♦	(In)	Bitstatus überprüfen	8.2.27	8-41
INPUT #	(Input)	Eingabe	8.2.28	8-42
JOVRD	(J override)	Übersteuerung	8.2.29	8-43
Label ♦	(Label)	Sprungmarke	8.2.30	8-44
MOV	(Move)	Bewegung mit Gelenk-Interpolation	8.2.31	8-45
Movement Position ♦	(Movement Position)	Koordinatenposition anfahren	8.2.32	8-46
MVC	(Move C)	Kreis-Interpolation	8.2.33	8-47
MVR	(Move R)	Kreis-Interpolation	8.2.34	8-49
MVR2	(Move R2)	Kreis-Interpolation	8.2.35	8-51
MVS	(Move S)	Geradlinige Bewegung	8.2.36	8-53
OADL	(Optimum Acceleration/Deceleration)	Optimale Beschleunigung/Abbremsung	8.2.37	8-55
ON COM GOSUB	(ON Communication Go Subroutine)	Sprung zu einem Unterprogramm	8.2.38	8-57
ON-GOSUB	(ON GOSUB)	Sprung zu einem Unterprogramm	8.2.39	8-58
ON GOTO	(On go to)	Programmverzweigung	8.2.40	8-59

Tab. 8-1: Übersicht der Befehle (1)



Befehl		Funktion	Abschnitt	Seite
<b>OPEN</b>	(Open)	Datei öffnen	8.2.41	8-60
<b>ORG</b>	(Origin)	Nullpunkt einstellen	8.2.42	8-62
<b>OUT ♦♦</b>	(Out)	Ausgabe	8.2.43	8-63
<b>OVRD</b>	(Override)	Übersteuerung	8.2.44	8-65
<b>PLT ♦</b>	(Pallet)	Koordinaten für Palette berechnen	8.2.45	8-67
<b>PRINT #</b>	(Print)	Daten übertragen	8.2.46	8-68
<b>REM</b>	(Remarks)	Kommentar	8.2.47	8-70
<b>RETURN</b>	(Return)	Rücksprung zum Hauptprogramm	8.2.48	8-71
<b>SKIP ♦♦</b>	(Skip)	Sprung in die nächste Zeile	8.2.49	8-72
<b>SPD</b>	(Speed)	Geschwindigkeit festlegen	8.2.50	8-73
<b>STOP ♦♦</b>	(Stop)	Programmablauf stoppen	8.2.51	8-74
<b>SV</b>	(Servo)	Servo ein-/ausschalten	8.2.52	8-75
<b>TOOL</b>	(Tool)	Werkzeug-Konvertierungsdaten	8.2.53	8-77
<b>WHILE~ WEND</b>	While End	Programmschleife	8.2.54	8-78
<b>WTH ♦</b>	(With)	Anweisung hinzufügen	8.2.55	8-80
<b>WTHIF ♦</b>	(With If)	Anweisung hinzufügen, wenn ...	8.2.56	8-81
<b>Substitute</b>	(Substitute)	Daten ersetzen	8.2.57	8-82

**Tab. 8-1:** Übersicht der Befehle (2)

## 8.2.1 ACL (Accelerate)

### Funktion: Beschleunigung einstellen

Legt den Wert für die Beschleunigung fest.

### Eingabeformat

ACL <Beschleunigungszeit>
---------------------------

<Beschleunigungszeit> Legt die Zeit vom Stillstand bis zur maximalen Geschwindigkeit fest.  
 $0,05 \leq \text{Beschleunigungszeit} \leq 2,00$  [s]  
Hinter dem ACL-Befehl muß ein Leerzeichen stehen.  
Die Schreibweise ACL1 wird als Anweisung zur Deklaration einer Variablen gewertet.

### Erläuterung

- Bei Eingabe der Beschleunigungszeit sollten kurze Zeiten für kleine Lasten und lange Zeiten für große Lasten gewählt werden.
- Ist die Beschleunigungszeit nicht eingestellt, wird der Standardwert M\_NACL verwendet. Der Standardwert wird in Abhängigkeit des verwendeten Robotertyps für kleine Lasten gesetzt.
- Sobald der ACL-Befehl ausgeführt wird, bleibt der geänderte Beschleunigungswert solange gesetzt, bis das Programm beendet ist. Ist das Programm abgearbeitet, wird die Beschleunigungszeit auf den Standardwert M\_NACL des Systems zurückgesetzt.
- Nach Freigeben der CNT-Einstellung wird die Roboterbewegung so lange kontinuierlich und gleichmäßig verlaufen (ohne Beschleunigung oder Abbremsung), bis die CNT-Einstellung wieder gesperrt wird. An den Startpunkten wird der Roboter jedoch beschleunigen und an den Endpunkten abbremsen. Das gleiche gilt auch für durch Timer ausgelöste Stopp-Vorgänge oder Eingabebefehle während des Bewegungsverlaufs.
- Wird eine Beschleunigungszeit eingestellt, die kleiner als der M\_NACL-Wert für die jeweilige Last ist, kann eine Fehlermeldung für Geschwindigkeitsüberschreitung entstehen. Zusätzlich kann sich eine Verkürzung der Lebensdauer der mechanischen Teile ergeben, wenn die Beschleunigungszeit herabgesetzt wird. Die Beschleunigungszeit sollte deshalb über dem M\_NACL-Wert liegen.
- Die Abbremszeit wird über den DACL-Befehl gesetzt.
- 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 freigegeben.

**ACHTUNG**

*In der Grundeinstellung ist die CNT-Einstellung freigegeben. Wird die Verfahrgeschwindigkeit, die Beschleunigungs- oder die Abbremszeit geändert, verändert sich auch die Verfahrkurve. Achten Sie darauf, daß es zu keinen Zusammenstößen mit umliegenden Einrichtungen kommt. Kommt es zu Zusammenstößen, sperren Sie die CNT-Einstellung oder verkürzen Sie die Beschleunigungs-/Abbremszeit.*

**Programmbeispiel (MELFA-BASIC III-Befehle)**

10	ACL 0.2	Beschleunigungszeit 0.2 s für kleine Last einstellen
20	MOV P1	Position P1 mittels Gelenk-Interpolation anfahren
30	ACL M_NACL	Beschleunigungszeit für Standardlast einstellen
40	MOV P2	Position P2 mittels Gelenk-Interpolation anfahren

Siehe auch DACL, CNT, Befehl für die Geschwindigkeitseinstellung und Stellungsvariablen des Roboters.

## 8.2.2 ACT (Act)

### Funktion: Interrupt freigeben

Freigeben oder sperren des Interrupts.

### Eingabeformat

ACT <Priorität>=<freigeben/sperren>

<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.

<freigeben/sperren>

freigeben = 1, sperren = 0

### 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 wird 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.
- 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

**Programmbeispiel (MELFA-BASIC III-Befehle)**

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
110	HLT	der Roboter gestoppt
120	RETURN 0	Rücksprung aus der Interruptroutine

Siehe auch DEF ACT und RETURN

### 8.2.3 ALIGN (Align)

**Funktion: Hand ausrichten**

Bewegt die Hand mittels Linear-Interpolation zu der Position, die den kleinstmöglichen Weg zur senkrechten oder waagerechten Stellung der Achsen A, B, C hat.

**Eingabeformat**

ALIGN
-------

**Erläuterung**

- Durch Ausführung des ALIGN-Befehls läßt sich die Stellung verändern, ohne daß die Handspitze bewegt wird.
- Durch Ausführung des ALIGN-Befehls nimmt die Hand die Stellung 0°, 90° oder 180° zur Senkrechten ein.

**Programmbeispiel (MELFA-BASIC III-Befehle)**

10 ALIGN

Siehe auch MOV.

## 8.2.4 BASE (Base)

### Funktion: Basis

Legt die Basis-Transformationsdaten fest.

### Eingabeformat

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

<Basis-Transformationsdaten>

Gibt die Basis-Transformationsdaten in Form von Positionskoordinaten an.

### 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 Basiskoordinatensystem in Bezug auf das Weltkoordinatensystem an.  
X = Parallele Abstand zur X-Achse  
Y = Parallele Abstand zur Y-Achse  
Z = Parallele Abstand zur Z-Achse  
A = Drehwinkel um die X-Achse  
B = Drehwinkel um die Y-Achse  
C = Drehwinkel um die Z-Achse  
Aus Sicht des Koordinatenursprungs werden Winkel in Uhrzeigerrichtung positiv gewertet.
- Die Werte der Stellungsmerker sind ohne Bedeutung.
- Die Änderungen des mit dem BASE-Befehl geänderten Basiskoordinatensystem bleiben auch nach Ausschalten der Spannungsversorgung erhalten.
- Der Standardwert ist P\_NBASE = (0,0,0,0,0,0)(0,0)
- Für den 5-achsigen Roboter können die Werte A, B und C nicht geändert werden. Geben Sie eine „0“ ein.



### ACHTUNG

**Achten Sie auf eine korrekte Eingabe der Basis-Transformationsdaten. Nach der Transformation ändert sich die Roboterstellung. Achten Sie darauf, daß es zu keinen Zusammenstößen mit der umliegenden Einrichtung kommt. Auch wenn die Basis-Transformationsdaten korrekt angegeben wurden, kann es, in Abhängigkeit von der aktuellen Dimensionierungs- und Stellungsgenauigkeit etc. zu Toleranzschwankungen kommen. Setzen Sie beim Einsatz eines 5-achsigen Roboters alle Werte mit Ausnahme der X-, Y- und Z-Werte auf „0“. Andernfalls kann es zu unerwünschten Vibrationen oder einer Fehlermeldung kommen.**

**Programmbeispiel (MELFA-BASIC III-Befehle)**

10	BASE (50,100,0,0,0,0)	Eingabe der Basis-Transformationsdaten als Konstante
20	MVS P1	Position P1 mittels Linear-Interpolation anfahren
30	BASE P1	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

Siehe auch TOOL und Roboterstellungsvariablen.



### 8.2.5 CALLP (Call P)

#### Funktion: Programm aufrufen

Führt das aufgerufene Programm aus (siehe auch GOSUB-Befehl für Unterprogrammaufrufe).

#### Eingabeformat

CALLP	"<Programmname>" [ , <Parameter>[ , <Parameter>] ...]
-------	---

<Programmname>      Legt den Programmnamen als Zeichenkettenkonstante oder Zeichenkettenvariable fest.

<Parameter>      Legt die Variablen fest, die beim Aufruf des Programmes übergeben werden.  
Es können maximal 16 Variablen übergeben werden.

#### Erläuterung

- 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 Hauptprogrammes zurück.
- Das aufgerufene Programm hat keinen Einfluß auf die Anweisungen DEF, ACT, DEF FN, DEF PLT, DIM im aufrufenden Programm. Sobald das aufgerufene Programm zurückspringt, werden sie wieder gültig.
- Die Geschwindigkeits- und Werkzeugdaten bleiben gültig.
- Es können bis zu 7 Programme von einem Programm aus aufgerufen werden. Dieser Wert beinhaltet auch die vom aufgerufenen Programm ausgeführten Unterprogramme.

#### Programmbeispiel (MELFA-BASIC III-Befehle)

10	CALLP "P10",M1,P1,P2	Aufruf des Programms P10 und Übergabe der Parameter M1, P1, P2
----	----------------------	--

Siehe auch END und FPRM.

## 8.2.6 CLOSE (Close)

### Funktion: Datei schließen

Schließt die festgelegte Datei.

### Eingabeformat

<code>CLOSE [[#]&lt;Dateinummer&gt;[, [[#]&lt;Dateinummer&gt; ...]</code>
---

<Dateinummer>                      Legt die Dateinummer der zu schließenden Datei fest.

### Erläuterung

- Legt die Dateinummer der zu schließenden Datei fest. Es können mehrere Dateien angegeben werden.
- Wird die Dateinummer nicht angegeben, werden alle geöffneten Dateien geschlossen.
- Sobald eine Datei geschlossen ist, kann mit derselben Dateinummer eine andere Datei geöffnet werden. Eine geöffnete Ein-/Ausgabedatei lagert die Daten in den Zwischenspeicher aus, sobald die CLOSE-Anweisung ausgeführt wird. Der unterbrochene Prozeß kann somit korrekt zum Abschluß gebracht werden.
- Sobald eine Datei geschlossen ist, können keine Ein- und Ausgaben mehr durchgeführt werden.
- Durch die Ausführung der END-Anweisung wird eine Datei ebenfalls geschlossen.

### Programmbeispiel (MELFA-BASIC III-Befehle)

10 OPEN"COM1:" AS#1	„COM1:“ wird als Datei Nummer 1 geöffnet
20 PRINT 1,M1	Übertrage 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

Siehe auch END.

### 8.2.7 CNT (Control)

#### Funktion: Roboterbewegung steuern

Legt die Steuerung für eine kontinuierliche und gleichmäßige Bewegung fest.

#### Eingabeformat

CNT <freigeben/sperren>
-------------------------

<freigeben/sperren>	Legt den Anfang und das Ende einer kontinuierlichen und gleichmäßigen Roboterbewegung fest. freigeben = 1, gesperrt = 0
---------------------	--

#### Erläuterung

- Standardmäßig ist die CNT-Einstellung freigegeben. Zwischen den Roboterbewegungen findet somit weder eine Beschleunigung noch eine Verzögerung statt (kontinuierliche und gleichmäßige Bewegung). An den Startpunkten wird der Roboter jedoch beschleunigen und an den Endpunkten abbremsen. Das gleiche gilt auch für durch Timer ausgelöste Stopp-Vorgänge oder Eingabebefehle während des Bewegungsverlaufs und für Verzweigungen. Um die Standardeinstellung zu verändern, muß die CNT-Einstellung auf „sperren“ gesetzt werden. Weitere Informationen sind im Anhang A.2 (Übersicht der Parameter) zu finden.
- Bei freigegebener CNT-Einstellung sind die FINE- und OADL-Einstellungen gesperrt.
- Ist die CNT-Einstellung freigegeben, ist es möglich, daß die festgelegte Geschwindigkeit nicht erreicht wird, da Geschwindigkeitsänderungen verhindert werden. In Abhängigkeit von den angefahrenen Positionen kann es zu einer Fehlermeldung kommen. In diesem Fall ist die Geschwindigkeit zu verringern.
- Die Verfahrkurve für eine kontinuierliche gleichmäßige Bewegung (CNT freigegeben) kann von der Verfahrkurve mit Beschleunigung/Verzögerung abweichen. Die Größe der Abweichung ist abhängig vom Wert der eingestellten Geschwindigkeit sowie der Beschleunigungs- und der Abbremszeit. Achten Sie daher darauf, daß es zu keinen Zusammenstößen mit umliegenden Einrichtungen kommt, wenn Sie die Verfahrgeschwindigkeit verändert haben.
- Für eine gleichmäßige Bewegung mit einer konstanten Geschwindigkeit sollten die Beschleunigungs- und die Abbremszeit gleich sein. Sind die Werte unterschiedlich, ist die Geschwindigkeit nicht konstant.
- Wird eine kurze Verfahrbewegung mit freigegebener CNT-Einstellung durchgeführt, und die Geschwindigkeit ist nicht konstant, ist der Verfahrweg zu vergrößern, die Geschwindigkeit und die Abbrems-/Beschleunigungszeit zu verringern.
- Abbildung 8-1 zeigt die Verfahrswege für verschiedene Einstellungen.

Verfahrweg	CNT-Einstellung	SPD [mm/s]	ACL/DACL [s]	Bewegungsbedingungen
A	gesperrt	500	0.2	CNT gesperrt
B	freigegeben	500	0.2	CNT freigegeben
C	freigegeben	250	0.2	Geschwindigkeit auf 1/2 heruntergesetzt
D	freigegeben	100	0.2	Geschwindigkeit auf 1/5 heruntergesetzt
E	freigegeben	500	0.1	Beschleunigungs-/Abbremszeit auf 1/2 heruntergesetzt
F	freigegeben	500	0.4	Beschleunigungs-/Abbremszeit verdoppelt

Tab. 8-2: Einstellungen

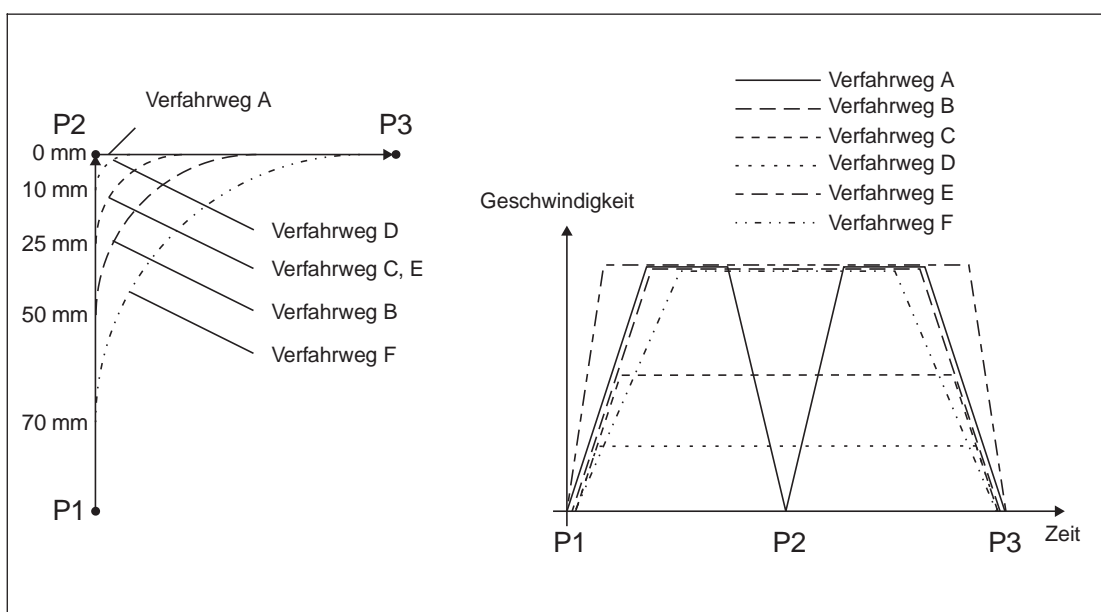


Abb. 8-1: Verfahrwege und Beschleunigungen für verschieden Einstellungen

**ACHTUNG**

In der Grundeinstellung ist die CNT-Einstellung freigegeben. 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 Geschwindigkeit und der Beschleunigungs-/Abbremszeit. Achten Sie daher darauf, daß es zu keinen Zusammenstößen mit umliegenden Einrichtungen kommt. Sollte es zu Zusammenstößen kommen, sperren Sie die CNT-Einstellung oder verkürzen Sie die Beschleunigungs-/Abbremszeit.

**Programmbeispiel (MELFA-BASIC III-Befehle)**

10	CNT 0	Sperren der CNT-Einstellung
20	MOV P1	Position P1 mittels Gelenk-Interpolation und Beschleunigung/Verzögerung anfahren
30	CNT 1	Freigeben der CNT-Einstellung
40	MVS P2	Position P2 mittels kontinuierlicher gleichmäßiger Geschwindigkeit anfahren
50	MVS P3	Position P3 mittels kontinuierlicher gleichmäßiger Geschwindigkeit anfahren
60	CNT 0	Sperren der CNT-Einstellung

Siehe auch Bewegungs-, Geschwindigkeits-, Beschleunigungs-, Abbremsbefehle, FINE und OADL.

## 8.2.8 COM OFF (Communication OFF)

### Funktion: Kommunikations-Interrupt sperren

Sperrt die Interrupts von Kommunikationsleitungen.

### Eingabeformat

COM [( <Nummer der Kommunikationsleitung> )] OFF
--

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

### Erläuterung

- Fehlt die Nummernangabe der Kommunikationsleitung, wird Leitung 1 ausgewählt.
- Nach Ausführung des COM OFF-Befehls bleibt der Interrupt auch bei anstehenden Nachrichten gesperrt.
- Informationen über Kommunikationsleitungen, sind unter dem Befehl OPEN zu finden.

### Programmbeispiel (MELFA-BASIC III-Befehle)

10 COM(1) OFF                    Sperrt den Kommunikations-Interrupt der Leitung 1

Siehe auch OPEN, COM ON, COM STOP und ON COM GOSUB.

## 8.2.9 COM ON (Communication ON)

### Funktion: Kommunikations-Interrupt freigeben

Gibt die Interrupts von Kommunikationsleitungen frei.

### Eingabeformat

COM [( <Nummer der Kommunikationsleitung> )] ON
---

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

### Erläuterung

- Fehlt die Nummernangabe der Kommunikationsleitung, wird Leitung 1 ausgewählt.
- Informationen über Kommunikationsleitungen, sind unter dem Befehl OPEN zu finden.

### Programmbeispiel (MELFA-BASIC III-Befehle)

10 COM(1) ON                      Gibt den Kommunikations-Interrupt der Leitung 1 frei

Siehe auch OPEN, COM OFF, COM STOP und ON COM GOSUB.

## 8.2.10 COM STOP (Communication STOP)

### Funktion: Kommunikations-Interrupt stoppen

Stoppt die Interrupts von Kommunikationsleitungen.

### Eingabeformat

```
COM [( <Nummer der Kommunikationsleitung> )] STOP
```

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

### Erläuterung

- Fehlt die Nummernangabe der Kommunikationsleitung, wird Leitung 1 ausgewählt.
- 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 der Leitung abgearbeitet.
- Für ein erneutes Öffnen, ist der Befehl COM ON zu verwenden.
- Liegt während eines Stopps ein Interrupt an, wird der Interrupt nach Ausführung des COM ON-Befehls sofort gesendet.
- Informationen über Kommunikationsleitungen sind unter dem Befehl OPEN zu finden.

### Programmbeispiel (MELFA-BASIC III-Befehle)

```
10 COM(1) STOP                    Gibt den Kommunikations-Interrupt der Leitung frei 1
```

Siehe auch OPEN, COM ON, COM OFF und ON COM GOSUB.



## 8.2.11 DACL (Deceleration)

### Funktion: Abbremszeit einstellen

Legt den Wert für die Abbremszeit fest.

### Eingabeformat

DACL <Abbremszeit>

<Abbremszeit>

Legt die Zeit von der maximalen Geschwindigkeit bis zum Stillstand fest.

$0,05 \leq \text{Abbremszeit} \leq 2,00 \text{ [s]}$

### Erläuterung

- Bei Eingabe der Abbremszeit sollten kurze Zeiten für kleine Lasten und lange Zeiten für große Lasten gewählt werden.
- Ist die Abbremszeit nicht eingestellt, wird der Standardwert  $M\_NDACL = 0.2$  verwendet. Der Standardwert wird in Abhängigkeit des verwendeten Robotertyps für kleine Lasten gesetzt.
- Sobald der DACL-Befehl ausgeführt wird, bleibt die geänderte Abbremszeit solange gesetzt, bis das Programm beendet ist oder zurückgesetzt wird.
- Nach Freigeben der CNT-Einstellung wird die Roboterbewegung so lange kontinuierlich und gleichmäßig verlaufen (ohne Beschleunigung oder Abbremsung), bis die CNT-Einstellung wieder gesperrt wird. An den Startpunkten wird der Roboter jedoch beschleunigen und an den Endpunkten abbremsen. Das gleiche gilt auch für durch Timer ausgelöste Stopp-Vorgänge oder Eingabebefehle während des Bewegungsverlaufs.
- Wird eine Abbremszeit eingestellt, die kleiner als der  $M\_NDACL$ -Wert für die jeweilige Last ist, kann eine Fehlermeldung für Geschwindigkeitsüberschreitung entstehen. Zusätzlich kann sich eine Verkürzung der Lebensdauer der mechanischen Teile ergeben, wenn die Abbremszeit herabgesetzt wird. Die Abbremszeit sollte deshalb über dem  $M\_NDACL$ -Wert liegen.
- Die Beschleunigungszeit wird über den ACL-Befehl gesetzt.
- 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 Abbremszeit. Für eine gleichmäßige Bewegung mit einer konstanten Geschwindigkeit sollten die Beschleunigungs- und die Abbremszeit gleich sein. In der Grundeinstellung ist die CNT-Einstellung freigegeben.



### ACHTUNG

*In der Grundeinstellung ist die CNT-Einstellung freigegeben. Wird die Verfahrgeschwindigkeit, die Beschleunigung oder die Abbremszeit geändert, verändert sich auch die Verfahrkurve. Achten Sie darauf, daß es zu keinen Zusammenstößen mit umliegenden Einrichtungen kommt. Kommt es zu Zusammenstößen, sperren Sie die CNT-Einstellung oder verkürzen Sie die Beschleunigungs-/Abbremszeit. Wird der Roboter, bei einer Abbremszeit oberhalb des zulässigen Wertebereiches, während einer Bewegung gestoppt (durch das Steuergerät, die Teaching Box oder ein externes Signal etc.), wird der Roboter nicht sofort stoppen, da er noch die Abbremsrampe herunterfährt. Verkürzen Sie in diesem Fall die Abbremszeit, oder betätigen Sie den NOT-AUS-Schalter.*

**Programmbeispiel (MELFA-BASIC III-Befehle)**

10	HND 1=1	Hand öffnen
20	DACL 0.2	Abbremszeit für kleine Last einstellen
30	MOV P1	Position P1 mittels Gelenk-Interpolation anfahren
40	HND 1=0	Hand schließen
50	DACL M_NDACL	Abbremszeit für Standardlast einstellen
60	MOV P2	Position P2 mittels Gelenk-Interpolation anfahren

Siehe auch ACL, CNT, Befehl für die Geschwindigkeitseinstellung und Stellungsvariablen des Roboters.

## 8.2.12 DEF ACT (Define act)

### Funktion: Interrupt-Prozeß definieren

Legt den Status und die Ausführung des Interrupts fest.

### Eingabeformat

```
DEF ACT <Priorität>, <Ausdruck>, <Prozeß>
```

<b>&lt;Priorität&gt;</b>	Gibt die Priorität des Interrupts an. $1 \leq \text{Priorität} \leq 8$
<b>&lt;Ausdruck&gt;</b>	Folgende Formate können für den Interrupt-Status verwendet werden (siehe auch Syntaxdiagramme): <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>
<b>&lt;Prozeß&gt;</b>	Legt eine GOTO- oder GOSUB-Anweisung fest, die bei einem Interrupt ausgeführt wird.

### Erläuterung

- Die Prioritäten der Interrupts sind in aufsteigender Reihenfolge von 1 bis 8 festgelegt.
- Über die Priorität können bis zu 8 Interrupts unterschieden werden.
- 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 Programmmteils erhalten, und es werden nur Interrupts höherer Priorität akzeptiert. Der Interrupt kann durch die Ausführung der END-Anweisung deaktiviert werden.

### Programmbeispiel (MELFA-BASIC III-Befehle)

10 DEF ACT 1,M_IN(17)=1 GOSUB 100	Definiert einen Unterprogrammsprung 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.

Siehe auch ACT, GOSUB, RETURN und Roboterstellungsveränderungen.

### 8.2.13 DEF FN (Define function)

#### Funktion: Funktion definieren

Definiert eine Funktion und legt den Namen fest.

#### Eingabeformat

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

<Name>	Besteht aus einem Zeichen zur Identifizierung und einer Zeichenkette.
<Formalparameter>	Legt die Variablen der Funktion fest. Es können maximal 16 Variablen verwendet werden.
<Funktionsausdruck>	Legt die Rechenoperation fest.

#### 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.
- Eine benutzerdefinierte Funktion, die nicht mit dem DEF FN-Befehl definiert wurde, erzeugt vor ihrer Ausführung eine Fehlermeldung.
- 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 Formelparametern aufgeführt wurden, werden für die Variablen die augenblicklichen Werte verarbeitet. 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 der sie definiert worden ist. Sie kann von einem anderen Programm nicht durch einen CALL P-Befehl aufgerufen werden.

**Programmbeispiel (MELFA-BASIC III-Befehle)**

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	MAVE=FNMAVE(MDATA1,MDATA2)	Der Durchschnitt von 20 und 30 (= 25) wird der numerischen Variablen MAVE zugewiesen

Siehe auch Variablen, Feldvariablen und Funktionen.

## 8.2.14 DEF PLT (Define pallet)

### Funktion: Palette definieren

Definiert eine Palette.

### Eingabeformat

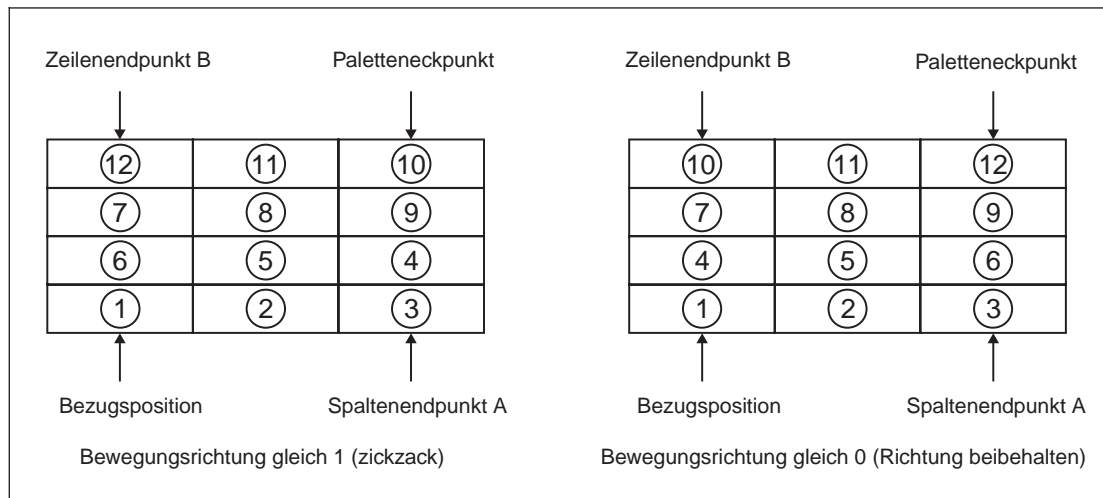
```
DEF PLT  <Palettennummer>, <Bezugsposition>,
          <Spaltenendpunkt A>, <Zeilenendpunkt B>,
          [<Paletteneckpunkt, der gegenüber der
          Bezugsposition liegt>],
          <Anzahl der Spaltengitterpunkte A>,
          <Anzahl der Zeilengitterpunkte B>,
          <Bewegungsrichtung>
```

<Palettennummer>	Legt die Nummer der eingesetzten Palette fest. $1 \leq \text{Palettennummer} \leq 8$
<Bezugsposition>	Legt den Anfangspunkt der Palette fest. Es können Ausdrücke mit Positionsoperationen verwendet werden.
<Spaltenendpunkt A>	Legt einen der Endpunkte der Palette fest. Es können Ausdrücke mit Positionsoperationen verwendet werden.
<Zeilenendpunkt B>	Legt einen der Endpunkte der Palette fest. Es können Ausdrücke mit Positionsoperationen verwendet werden.
<Paletteneckpunkt, der gegenüber der Bezugsposition liegt>	Legt den Punkt fest, der gegenüber der Bezugsposition liegt. Es können Ausdrücke mit Positionsoperationen verwendet werden.
<Anzahl der Spaltengitterpunkte>	Legt die Gitterpunkte der Palette in Spaltenrichtung fest. Es können Ausdrücke mit numerischen Operationen verwendet werden.
<Anzahl der Zeilengitterpunkte>	Legt die Gitterpunkte der Palette in Zeilenrichtung fest. Es können Ausdrücke mit numerischen Operationen verwendet werden.
<Bewegungsrichtung>	Die Eingabe von „1“ oder „2“ legt die Bewegungsrichtung fest. 1 = Zickzack 2 = Bewegungsrichtung beibehalten (z. B. von links nach rechts)

### 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 Programmes 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 <Anzahl der Spaltengitterpunkte> x <Anzahl der Zeilengitterpunkte> den Wert 32 767 überschreitet, erfolgt eine Fehlermeldung.



**Abb. 8-2:** Beispiel zur Palettendefinition

#### Programmbeispiel (MELFA-BASIC III-Befehle)

```

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

```

Siehe auch PLT und Befehle für Roboterbewegungen.

## 8.2.15 DIM (Dim)

### Funktion: Dimension definieren

Legt die Anzahl der Elemente bei Feldvariablen fest.

### 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 Konstanten enthalten. Ausdrücke mit numerischen Operationen sind nicht erlaubt.

### Erläuterung

- Es sind ein- und zweidimensionale Felder erlaubt.
- Bei Überschreitung des Wertebereiches für den maximalen Indexwert erfolgt bei der Ausführung der DIM-Anweisung eine Fehlermeldung.
- Bei Ausführung der DIM-Anweisung sind die Standardwerte der Feldvariablen:  
 0 für numerische Variablen.  
 Für Zeichenketten-Feldvariablen sind die Elemente alle Null-Zeichen.  
 Für Positions-Feldvariablen sind die Elemente nicht definiert.
- Werden Felder ohne DIM-Anweisung verwendet, wird automatisch Speicherplatz reserviert:  
 Für eindimensionale Felder, DIM Variablenname (10).  
 Für zweidimensionale Felder, DIM Variablenname (10, 10).
- Ist die Anzahl der Elemente eine reelle Zahl, wird die Zahl automatisch auf eine Integer-Zahl gerundet.
- Der DIM-Befehl steht nur innerhalb des ausgeführten Programmes zur Verfügung. Er kann nicht von einem anderen Programm aufgerufen werden. Bei Verwendung in einem Unterprogramm muß er erneut definiert werden.
- Soll in einem externen Speicher (z. B. einem PC) eine Sicherungskopie einer mit dem DIM-Befehl deklarierten Positions-Feldvariablen angelegt werden, müssen die Daten einmal in die üblichen Variablen kopiert werden. Die Daten können nicht als gesamte Positions-Feldvariable gesichert werden.

### Programmbeispiel (MELFA-BASIC III-Befehle)

10	DIM PDATA(10)	Deklariert die Positions-Feldvariable als eine Variable mit 10 Elementen
20	DIM MDATA(2,3)	Deklariert die zweidimensionale numerische Feldvariable MDATA als eine Variable mit 2x3 Elementen

Siehe auch Feldvariablen.



## 8.2.16 DLY (Delay)

### Funktion: Verzögerung einstellen

Als einzelner Befehl wird zur festgelegten Zeit der Wartestatus erzeugt. Wird der DLY-Befehl für einen zusätzlichen Impulsausgang genutzt, wird die Impulsdauer festgelegt.

### Eingabeformat

DLY <Zeit>
------------

<Zeit>                      Legt die Dauer des Wartestatus oder die Impulsdauer in Sekunden fest.

### Erläuterung

- Der DLY-Befehl wird verwendet, um in Programmen Verzögerungszeiten zu erzeugen. Ebenso läßt sich die Impulsdauer eines Ausgangssignals in der OUT-Anweisung festlegen.
- Der Impulsausgang wird gleichzeitig mit Ausführung des in der nächsten Zeile stehenden Befehls gesetzt.
- Es können bis zu 4 Impulsausgänge gleichzeitig gesteuert werden. Wird dieser Wert überschritten, kommt es bei Ausführung des Befehls zu einer Fehlermeldung.
- Nach Ablauf der festgesetzten Zeit wird wieder der Zustand vor Ausführung des Befehls angenommen.
- Wird während der festgesetzten Zeit eine END-Anweisung, die letzte Zeile des Programmes oder ein NOT-HALT ausgeführt, behält der Impulsausgang seinen gegenwärtigen Zustand bei.
- Die Reihenfolge der Prioritäten ist:  
COM > ACT > WTHIF (WTH) > Impulsausgang (Zeitintervall aktiv)

### Programmbeispiel (MELFA-BASIC III-Befehle)

10	DLY 30	Wartezeit von 30 Sekunden
20	M_OUT(17)=1 DLY 10.0	Sende den Signalausgang für 10 Sekunden zum allgemeinen Ausgangssignal 17

Siehe auch Roboterstellungsvariablen und Substitution.

## 8.2.17 END (End)

### Funktion: Programmende

Beendet das Programm.

### Eingabeformat

END
-----

### Erläuterung

- Es können mehrere END-Anweisungen in einem Programm ausgeführt werden.
- Eine END-Anweisung, die durch einen CALL P-Befehl aufgerufen wird, übergibt die Kontrolle an das Programm, in dem der CALL P-Befehl ausgeführt wurde.
- Eine END-Anweisung im Hauptprogramm schließt alle geöffneten Dateien.

### Programmbeispiel (MELFA-BASIC III-Befehle)

```
100 END
```

Siehe auch CALL P, FPRM, GOSUB und RETURN.

## 8.2.18 FINE (Fine)

### Funktion: Feinpositionierung

Legt den Status bei der Beendigung eines Interpolationsbefehls fest.

### Eingabeformat

```
FINE <freigeben/sperrn>
```

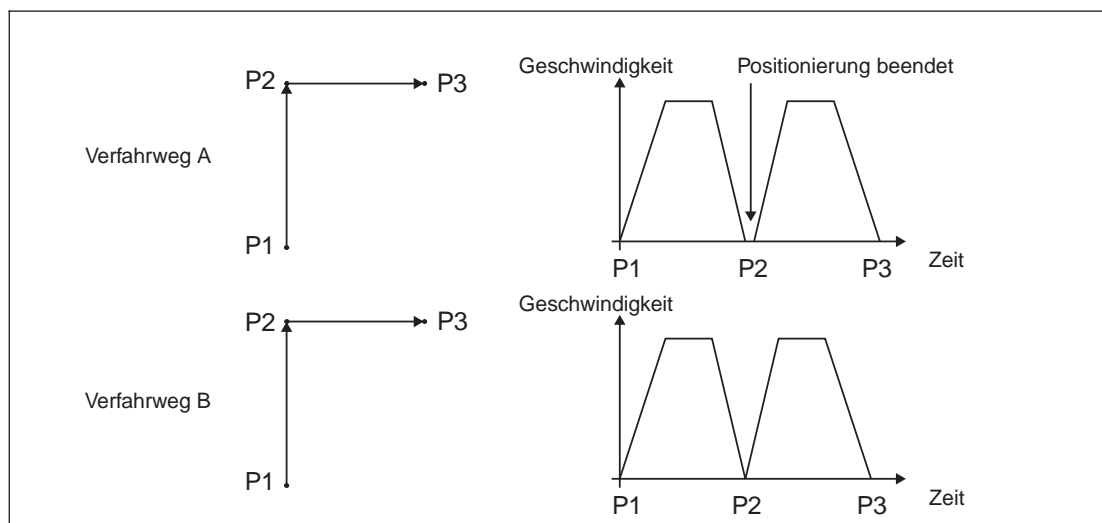
<freigeben/sperrn>      freigeben = 1, sperren = 0, Standardwert = 0

### Erläuterung

- Bei freigegebener FINE-Einstellung wird das Ende der Abarbeitung eines Interpolationsbefehls durch die Servoantriebe gemeldet.
- Bei freigegebener FINE-Einstellung wird der Interpolationsbefehl zu Ende geführt, nachdem der Positioniervorgang durch die Servoantriebe abgeschlossen ist.
- 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 freigegeben, bis sie erneut gesperrt wird.
- Nach Abarbeitung des Programms wird die FINE-Einstellung gesperrt.
- 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).
- Abbildung 8-3 zeigt den Einfluß der Kombination des CNT- und FINE-Befehls auf die Roboterbewegung.

Verfahrweg	CNT freigeben/sperrn	FINE freigeben/sperrn
Verfahrweg A	freigegeben	freigegeben
Verfahrweg B		gesperrt

**Tab. 8-3:** Kombination von CNT- und FINE-Befehl



**Abb. 8-3:** Roboterbewegung bei Kombination von CNT- und FINE-Befehl

**Programmbeispiel (MELFA-BASIC III-Befehle)**

10	FINE 1	FINE-Einstellung freigeben
20	MOV P1	Position P1 mittels Gelenk-Interpolation anfahren
30	FINE 0	FINE-Einstellung sperren
40	MOV P2	Position P2 mittels Gelenk-Interpolation anfahren

Siehe auch CNT und Roboterbewegungsbefehle.

## 8.2.19 FOR-NEXT (For-Next)

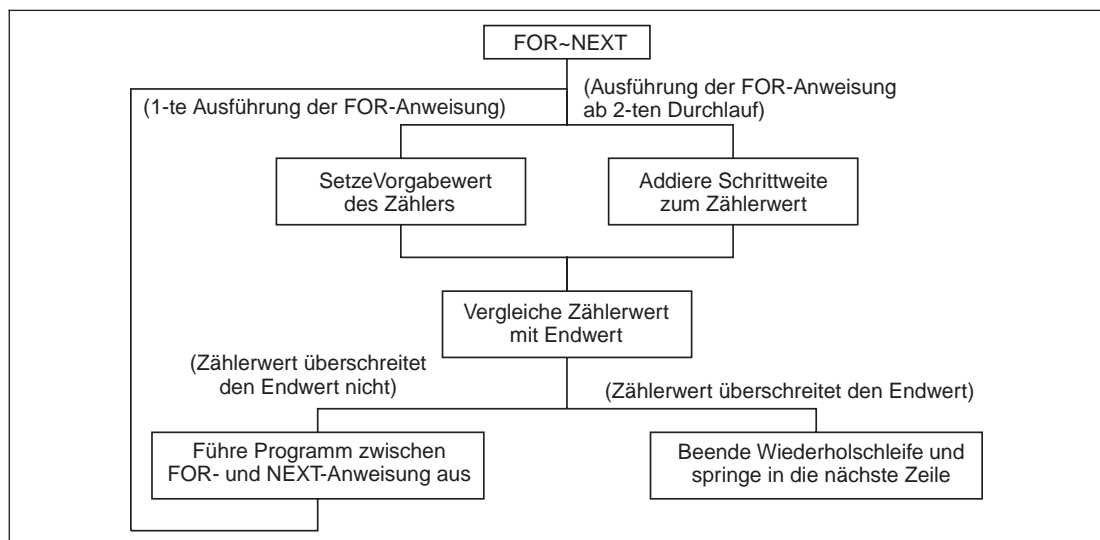
### Funktion: Programmschleife

Dieser Befehl bewirkt eine Wiederholung des Programnteils, der zwischen der FOR-Anweisung und der NEXT-Anweisung steht. Der Programnteil wird solange wiederholt, bis die Abbruchbedingungen erfüllt sind.

### Eingabeformat

```
FOR <Zähler> = <Vorgabewert> TO <Endwert>[STEP<Schrittweite>]
NEXT [<Zähler 1> [, <Zähler 2> ...]
```

<Zähler>	Der numerische Datentyp gibt die Anzahl der Wiederholungen der Programmschleife an.
<Vorgabewert>	Die Angabe Zähler 1, Zähler 2 kann auch verwendet werden. Gibt den Startwert des Zählers vor.
<Endwert>	Gibt den Endwert des Zählers vor.
<Schrittweite>	Legt die Schrittweite des Zählers fest.



**Abb. 8-4:** Programmschleife

### Erläuterung

- Die Programmschleife wird nicht ausgeführt, wenn:  
Der <Vorgabewert> größer als der <Endwert> und der <Schrittweite> positiv ist.  
Der <Vorgabewert> kleiner als der <Endwert> und der <Schrittweite> negativ ist.
- Widersprechen sich die FOR- und die NEXT-Anweisung, erfolgt eine Fehlermeldung. Werden FOR-NEXT-Programmschleifen verschachtelt verwendet, und sie haben den gleichen Endwert, ist es möglich nur eine NEXT-Anweisung zu verwenden. Die Zeilen 50 und 60 im Programmbeispiel können zu einer Zeile zusammengefaßt werden: NEXT MY,MX.

- Steht die NEXT-Anweisung in unmittelbarer Beziehung zur nächsten FOR-Anweisung, können die Variablennamen in der NEXT-Anweisung weggelassen werden. „MY“ in Zeile 50 und „MX“ in Zeile 60 im Programmbeispiel können weggelassen werden.
- **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.

### Programmbeispiel (MELFA-BASIC III-Befehle)

Programm zur Addition der Zahlen 1 bis 10

10	MSUM=0	Weist MSUM den Wert 0 zu
20	FOR MDATA=1 TO 10 STEP 1	Setze den Vorgabewert, den Endwert und die Schrittweite der Variablen MDATA
30	MSUM=MSUM+MDATA	Addiere MDATA zu der numerischen Variablen MSUM
40	NEXT MDATA	Sprung zu Zeile 20
50	END	Programmende

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 10x10 Feldvariable
20	FOR MX=1 TO 10 STEP 1	Erhöhe den Zähler der numerischen Variablen MX von 1 bis 10 um 1 und springe zu Zeile 70, sobald der Wert 10 überschritten ist („STEP 1“ kann weggelassen werden)
30	FOR MY=1 TO 10 STEP 1	Erhöhe den Zähler der numerischen Variablen MY von 1 bis 10 um 1 und springe zu Zeile 60, sobald der Wert 10 überschritten ist („STEP 1“ kann weggelassen werden)
40	MBOX(MX,MY)=MX*MY	Ersetze die Elemente der Feldvariablen MBOX (MX,MY) durch das Produkt MX*MY
50	NEXT MY	Sprung zu Zeile 30
60	NEXT MX	Sprung zu Zeile 20
70	END	Programmende

Siehe auch GOSUB, WHILE, END und Ausdrücke mit numerischen Operationen (Syntaxdiagramm).

## 8.2.20 FPRM (FPRM)

### Funktion: Parameter definieren

Legt im Hauptprogramm die Reihenfolge, den Typ und die Anzahl von Parametern fest, die in einem Unterprogramm verwendet werden (z. B. bei Aufruf eines Unterprogramms von einem Hauptprogramm mit dem CALL P-Befehl).

### Eingabeformat

```
FPRM <Formalparameter>[ , <Formalparameter>] ...
```

<Formalparameter>      Parameter in einem Unterprogramm, die bei Rücksprung in das Hauptprogramm übergeben werden.  
Es können alle Variablen verwendet werden.  
Es dürfen maximal 16 Variablen verwendet werden.

### Erläuterung

- Der FPRM-Befehl wird nicht benötigt, wenn im aufgerufenen Unterprogramm keine Parameter verwendet werden.
- Eine Variable, die nicht als Formalparameter aufgeführt ist, behält ihren aktuellen Wert bei.
- Weicht der mit dem FPRM-Befehl festgelegte Datentyp oder die Anzahl der Formalparameter von denen im mit CALL P-Befehl aufgerufenen Programm ab, erfolgt eine Fehlermeldung.
- Der FPRM-Befehl kann in einem Unterprogramm mehrmals verwendet werden. Bei jeder Ausführung wird der Parameter von der entsprechenden Variablen überschrieben (Formalparameter).
- Programmebenen  
Der Aufruf von Unterprogrammen erlaubt eine Programmstruktur mit mehreren Ebenen. Jeder Programmaufruf bedeutet dabei eine zusätzliche Ebene. Es dürfen maximal 7 Ebenen verwendet werden. Bei mehr als 7 Ebenen erfolgt eine Fehlermeldung.

### Programmbeispiel (MELFA-BASIC III-Befehle)

10 FPRM M1,P1,P2      Festlegung der Datentypen, der Reihenfolge und der Anzahl

Siehe auch CALL P, Variablen, Feldvariablen und Funktionen.

## 8.2.21 GOSUB (Go Subroutine)

### Funktion: Sprung zu einem Unterprogramm

Bewirkt einen Sprung zu einem Unterprogramm, das mit einer festgelegten Zeilennummer oder einer Marke beginnt.

### Eingabeformat

GOSUB     <Sprungziel>
------------------------

<Sprungziel>                      Legt eine Zeilennummer oder eine Marke fest.

### Erläuterung

- Eine RETURN-Anweisung bewirkt einen Rücksprung vom Unterprogramm ins Hauptprogramm.
- Wenn die festgelegte Zeilennummer oder Marke nicht existiert oder nicht definiert ist, oder es existieren zwei Definitionen, erfolgt eine Fehlermeldung.
- Es kann auch eine aktuelle Variable zur Angabe des Sprungziels verwendet werden.
- Mit der Ausführung der letzten Zeile oder der END-Anweisung in einem Unterprogramm wird die Programmabarbeitung abgeschlossen.

Sprungziel	Anweisung	Steuerung
Unterprogramm	RETURN	Steuerung springt in nächste Zeile hinter der Zeile in der die GOSUB-Anweisung ausgeführt wurde
	END (oder letzte Programmzeile)	Programm wird beendet

**Tab. 8-4:** Wirkung der RETURN- und END-Anweisung

### Programmbeispiel (MELFA-BASIC III-Befehle)

100 GOSUB 1000	Sprung zum Unterprogramm (Zeile 1000)
110 END	Hauptprogrammende
1000 MOV P1	Position P1 anfahren
1010 MOV P2	Position P2 anfahren
1020 RETURN	Unterprogrammende

Siehe auch RETURN, END, GOTO, DEF ACT und ACT.



## 8.2.22 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.

### Erläuterung

- Wenn die festgelegte Zeilennummer oder Marke nicht definiert ist, oder es existieren zwei Definitionen, erfolgt eine Fehlermeldung.
- Die Ausführung des GOTO-Befehls hat keinen Einfluß auf die Ebenen der Programmstruktur.

### Programmbeispiel (MELFA-BASIC III-Befehle)

10	GOTO 100	Unbedingter Programmsprung zur Zeile 100
20	GOTO*JUMP	Unbedingter Programmsprung zur Marke *JUMP
100	*JUMP	
110	MOV P1	Position P1 anfahren
120	END	Programmende

Siehe auch IF ... THEN ... ELSE, GOSUB, DEF ACT und ACT.

### 8.2.23 HLT (Halt)

**Funktion: Programmablauf stoppen**

Stoppt die Roboterbewegung und den Programmablauf.

**Eingabeformat**

HLT
-----

**Erläuterung**

- Unterbricht den Programmablauf und stoppt den Roboter mit der definierten Abbremszeit.
- 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.

**Programmbeispiel (MELFA-BASIC III-Befehle)**

10	MOV P1	Position P1 anfahren
20	HLT	Programmablauf und Roboterbewegung stoppen
30	MOV P2	Position P2 anfahren
100	IF M_IN(1)=1 THEN HLT	Programmablauf und Roboterbewegung stoppen, wenn der Eingang 1=1 ist

Siehe auch WTHIF, WTH, STOP und SKIP.

## 8.2.24 HND ♦♦ (Hand)

### Funktion: Handgreiferzustand festlegen

Legt den Handgreiferzustand (offen/geschlossen) fest.

### Eingabeformat

HND <Handnummer >=<offen/geschlossen>

<Handnummer>	<p>Legt fest, welche Hand betätigt werden soll.            Handnummer = 1, 2 oder 3            Hinter dem HND-Befehl muß ein Leerzeichen stehen. HND1 wird als Anweisung für eine Variablendeklaration interpretiert.</p>
<offen/geschlossen>	<p>Legt den Handgreiferzustand fest.            1 = offen            0 = geschlossen            In Abhängigkeit der Schlauchanschlüsse an den Magnetventilen können sich die Handgreiferzustände umkehren.</p>

### Erläuterung

- Die Hand wird durch die HND-Anweisung oder durch Änderung der Roboterstatusvariablen M\_HND() geöffnet oder geschlossen.
- Zum Öffnen und Schließen des Handgreifers ist eine bestimmte Zeitdauer erforderlich. Wird der gewünschte Handgreiferzustand nicht mittels HND-Anweisung oder M\_HND vor Erreichen der nächsten Position angenommen, bestehen folgende Möglichkeiten:
  - Zeitverzögerung mittels DLY-Anweisung
 

```
10 HND 1=0
20 DLY 0.5
30 MOV P1
```
  - Überprüfung des Handgreiferzustandes mittels Roboterstatusvariable M\_HNDCQ()
 

```
10 HND = 0
20 IF M_HNDCQ(1) <> 1 GOTO 20
30 MOV P1
```

Die Beziehung zwischen den in Klammern gesetzten Wert bei M\_HNDCQ() und dem Wert der Handnummer ist in Tabelle 8-5 gezeigt.
- Der Impuls Ausgang kann nicht benutzt werden, wenn die Roboterhand mittels Roboterstatusvariable M\_HND() geöffnet oder geschlossen wird.

Beispiel:  
 M\_HND(1)=1 DLY0.5

In diesem Fall sollte das nachfolgende Programmbeispiel verwendet werden:

```
10 M_HND(1)=1
20 DLY 0.5
30 M_HND(1)=0
```

Statusvariable	Bezeichnung	Prüfsignal (2-Bit-Status)			
		0	1	2	3
M_HNDCQ (1)	Handprüfsignal 1 =	0	1	0	1
	Handprüfsignal 2 =	0	0	1	1
M_HNDCQ (2)	Handprüfsignal 3 =	0	1	0	1
	Handprüfsignal 4 =	0	0	1	1
M_HNDCQ (3)	Handprüfsignal 5 =	0	1	0	1
	Handprüfsignal 6 =	0	0	1	1

**Tab. 8-5:** Parametereinstellungen**Parameter**

Mit dem Parameter GCD kann der Handgreiferzustand (offen/geschlossen) beim Ausführen eines Handbefehls und beim Einschalten der Spannungsversorgung geändert werden.

Parametername GCD: Hand 1 vorwärts/rückwärts, Standardwert Hand 1  
 Hand 2 vorwärts/rückwärts, Standardwert Hand 2  
 Hand 3 vorwärts/rückwärts, Standardwert Hand 3

Die Vorwärts-/Rückwärtseinstellung legt fest, welche Handrichtung bei einer Befehlsausführung gewählt werden soll (0: vorwärts, 1: rückwärts)

Die Standardeinstellungen legen fest, welcher Handgreiferzustand beim Einschalten der Spannungsversorgung gewählt werden soll.

Standardwert Hand 1 (2, 3)	0	1	2	3
Ausgangsbit 900 (902, 904, 906)	0	1	0	1
Ausgangsbit 901 (903, 905, 907)	0	0	1	1

**Tab. 8-6:** Parametereinstellungen für den Handgreiferzustand der pneumatisch angetriebenen Hand**Programmbeispiel (MELFA-BASIC III-Befehle)**

```

10  HND 1=1           Öffnet Hand 1
20  HND 1=0           Schließt Hand 1
30  MOV P1 WTH HND 1=1 Position P1 anfahren und Hand 1 öffnen
                        (Handbefehl in Verknüpfung mit einem
                        Bewegungsbefehl)

```

Siehe auch Roboterstatusvariablen (siehe Seite 7-19 ff.).

## 8.2.25 HRE ♦ (Here)

### Funktion: aktuelle Position

Liefert die aktuelle Position in kartesischen Koordinaten.

### Eingabeformat

HRE
-----

### Erläuterung

- Die HRE-Anweisung kann verwendet werden, um die aktuelle Position des Roboters zu speichern, wenn während einer Roboterbewegung ein ACT- oder COM-Interrupt generiert wurde (z. B. bei Aktivierung eines Berührungssensors an der Handspitze).
- Wird die HRE-Anweisung einzeln ausgeführt, erfolgt eine Fehlermeldung:  
10 HRE → Syntax error
- Auch wenn die HRE-Anweisung an derselben Position ausgeführt wird, sind die Positionsangaben nach einer Änderung des Basiskoordinatensystems (BASE-Anweisung) oder der Werkzeugdaten (TOOL-Anweisung) unterschiedlich. Wird die HRE-Anweisung verwendet, um eine Position anzufahren, ist die Position nach Änderung des Basiskoordinatensystems oder der Werkzeugdaten eine andere, als die, die eingegeben wurde.

### Programmbeispiel (MELFA-BASIC III-Befehle)

10	P1=HRE	Weist P1 die aktuelle Position zu
20	MOV HRE+P2	Aktuelle Position+P2 mittels Gelenk-Interpolation anfahren

Siehe auch Roboterstatusvariablen (siehe Seite 7-19 ff.), BASE und TOOL.

## 8.2.26 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>]
```

<Ausdruck> Beschreibt einen Booleschen Ausdruck.

<Anweisung> Beschreibt die in MELFA-BASIC III verwendeten Ausdrücke (mit Ausnahme von bedingten Verzweigungen und Schleifen), Zeilennummern und Marken.

### Erläuterung

- Ist das Ergebnis des Booleschen Ausdrucks wahr, wird die THEN-Anweisung ausgeführt. Ist das Ergebnis des Booleschen Ausdrucks unwahr, wird die ELSE-Anweisung ausgeführt.
- Die ELSE-Anweisung kann weggelassen werden.
- Die in Tabelle 8-7 aufgeführten Anweisungen dürfen nicht verwendet werden.

Art der Anweisung	MELFA-BASIC III-Anweisung
Bedingte Verzweigung	ON ... GOTO, ON ... GOSUB, IF ... THEN ... ELSE
Wiederholschleifen	FOR~NEXT, WHILE~WEND
Anweisungen, die nicht ohne zusätzliche Angabe ausgeführt werden dürfen	Befehle mit einem Asterisks (*) z. B. WTH, WTHIF, IN, PLT
Anweisungen zur Definition und Deklaration	DIM, DEF, FN, ON COM GOSUB
Anweisungen für Kommentare	REM

**Tab. 8-7:** Anweisungen, die nicht verwendet werden dürfen

### Programmbeispiel (MELFA-BASIC III-Befehle)

100 IF MDATA > 10 THEN 1000	Sprung zu Zeile 1000, falls MDATA größer 10
110 IF MDATA > 10 GOTO 1000 ELSE GOTO 2000	Sprung zu Zeile 1000, falls MDATA größer 10, sonst Sprung zu Zeile 2000
120 IF MDATA > 10 THEN GOSUB 1000 ELSE GOTO 2000	Sprung zum Unterprogramm in Zeile 1000, falls MDATA größer 10, sonst Sprung zu Zeile 2000
130 IF MDATA > 10 GOTO 1000 ELSE GOTO *WORK	Sprung zu Zeile 1000, falls MDATA größer 10, sonst Sprung zur Marke *WORK

Siehe auch ON, GOSUB, GOTO und numerische Operationsausdrücke (Syntaxdiagramm).

## 8.2.27 IN ♦ (In)

### Funktion: Bitstatus überprüfen

Überprüft einen Bitstatus der allgemeinen Eingabeschnittstelle.

### Eingabeformat

IN <Eingangs-Bitnummer>

<Eingangs-Bitnummer> Legt die Bitnummer der allgemeinen Eingabeschnittstelle fest.  
 $0 \leq \text{Eingangs-Bitnummer} \leq 32767$   
 Hinter der IN-Anweisung muß ein Leerzeichen stehen.  
 IN1 wird als Anweisung zur Variablennamendeclaration interpretiert.

### Erläuterung

- Das Bit wird über den allgemeinen Eingabeport eingegeben. Es kann in Booleschen Ausdrücken oder für Substitutionen verwendet werden.
- Der Befehl darf nicht ohne zusätzliche Anweisung verwendet werden:  
 100 IN 17=0 → Syntax error
- Es gelten dieselben Bedingungen wie für Roboterstatusvariablen M\_IN(). Abbildung 8-5 zeigt zusätzlich die Beschreibung der M\_INB()- und der M\_INW-Anweisung.

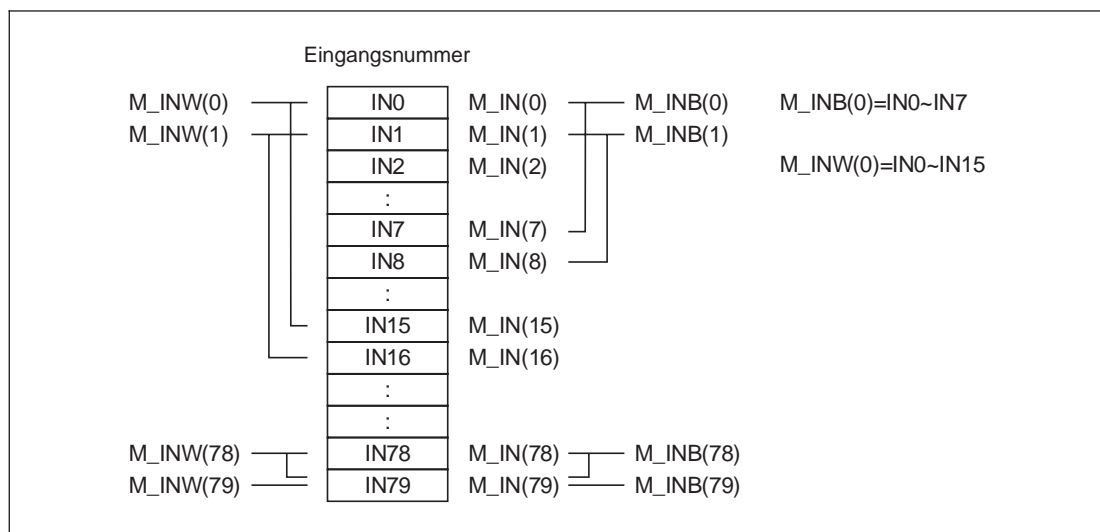


Abb. 8-5: Eingangsadressen

### Programmbeispiel (MELFA-BASIC III-Befehle)

10	*TOP	Marke
20	IF IN 10=0 GOTO 20	Wartet bis das Eingangsbit Nr.10 eingeschaltet ist
30	IF M_IN(1)=0 GOTO 30	Wartet bis das Eingangsbit Nr.1 eingeschaltet ist
40	IF M1<>0 GOTO *TOP	Springt zur Marke *TOP, wenn die arithmetische Variable ungleich 0 ist

Siehe auch OUT und Roboterstatusvariablen (siehe Seite 7-19 ff.).

## 8.2.28 INPUT# (Input)

### Funktion: Eingabe

Überträgt Daten aus Dateien oder Eingabegeräten.

### Eingabeformat

INPUT#    <Dateinummer>, <Datenname> [ , <Datenname> ] ...
--

<Dateinummer>            Legt die Dateinummer fest.  
 $1 \leq \text{Dateinummer} \leq 8$

<Datenname>            Name der Variablen in die die Daten übertragen werden.  
 Es können alle Variablen verwendet werden.

### Erläuterung

- Überträgt Eingangsdaten aus Dateien (oder von Eingabegeräten), die mittels OPEN-Anweisung geöffnet worden sind, in eine Variable.
- Der übertragene Datentyp und der Variablentyp müssen übereinstimmen.
- Werden mehrere Variablennamen 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.
- Wird nur die Eingabetaste (CR = Carriage Return) betätigt, erfolgt eine Fehlermeldung.
- Es erfolgt eine Fehlermeldung, wenn der Variablentyp oder die Anzahl der Variablen nicht übereinstimmen.

Beispiel:

Bei Eingabe von Zeichenketten, numerischen Werten und Positionen.

10 INPUT#1,C1\$,M1,P1

"MELFA",125.75,"(460,280,150,0,180)(1,0)"

"MELFA" wird in C1 übertragen, 125.75 in M1 und (460,280,150,0,180)(1,0) in P1

### Programmbeispiel (MELFA-BASIC III-Befehle)

10 OPEN "COM:" AS 1	Weist dem Personalcomputer die Datei Nummer 1 zu
20 INPUT# 1,MDATA	Erfolgt eine Eingabe von der Tastatur, wird dieser Wert in die numerische Variable MDATA übertragen

Siehe auch OPEN und PRINT#.



## 8.2.29 JOVRD (J override)

### Funktion: Übersteuerung

Legt die Geschwindigkeits-Übersteuerung für die Gelenk-Interpolation fest.

### Eingabeformat

JOVRD	<Übersteuerungswert>
-------	----------------------

<Übersteuerungswert>    Legt den prozentualen Übersteuerungswert fest  
 $1 \leq \text{Übersteuerungswert} \leq 200.0$

### Erläuterung

- Legt den prozentualen Übersteuerungswert für die Arbeitsgeschwindigkeit des Roboters fest.
- Der JOVRD-Befehl ist nur bei der Gelenk-Interpolation wirksam.
- Die aktuelle Arbeitsgeschwindigkeit ergibt sich folgendermaßen:

$$\text{Gelenk-Interpolation} = \text{Playback-Übersteuerungswert} \times \text{Einstellwert des OVRD-Befehls} \times \text{Einstellwert des JOVRD-Befehls}$$

- Der Maximalwert der Arbeitsgeschwindigkeit ist 100%. Der Standardwert der Arbeitsgeschwindigkeit beträgt 100% der Standardeinstellung (M\_NOVRD).
- Der Standardwert bleibt so lange wirksam, bis der JOVRD-Befehl ausgeführt wird. Die so festgesetzte Arbeitsgeschwindigkeit kann durch einen weiteren JOVRD-Befehl geändert werden.
- Bei kurzen Verfahrwegen ist es möglich, daß der festgelegte Übersteuerungswert nicht erreicht wird.
- Liegt der Übersteuerungswert außerhalb des Wertebereiches des Roboters, erfolgt eine Fehlermeldung. Der Wert muß zwischen 0 und 100% liegen.



#### ACHTUNG

*In der Grundeinstellung ist die CNT-Einstellung freigegeben. Wird die Verfahrgeschwindigkeit, die Beschleunigungs- oder die Abbremszeit geändert, verändert sich auch die Verfahrkurve. Achten Sie darauf, daß es zu keinen Zusammenstößen mit umliegenden Einrichtungen kommt. Kommt es zu Zusammenstößen, sperren Sie die CNT-Einstellung oder verkürzen Sie die Beschleunigungs-/Abbremszeit.*

### Programmbeispiel (MELFA-BASIC III-Befehle)

```
10  JOVRD 50           Übersteuerung auf den Wert 50% einstellen
20  MOV P1            Position P1 anfahren
30  JOVRD M_NJOVRD    Standardwert einstellen
```

Siehe auch OVRD, SPD, Roboterbewegungsbefehle und Roboterstatusvariablen (siehe Seite 7-19 ff.).

### 8.2.30 LABEL ♦ (Label)

**Funktion: Sprungmarke**

Legt ein Sprungziel fest.

**Eingabeformat**

`*<Name der Marke>`

<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).

**Erläuterung**

- Es erfolgt keine Fehlermeldung, wenn die Marke während eines Programmlaufes nicht aufgerufen wird.
- Ist die gleiche Marke in einem Programm mehrmals definiert, erfolgt eine Fehlermeldung.

**Programmbeispiel (MELFA-BASIC III-Befehle)**

```
10 *SUB1      Name der Marke ist *SUB1
```

Siehe auch GOTO und GOSUB.

## 8.2.31 MOV (Move)

### Funktion: Bewegung mit Gelenk-Interpolation

Bewegt die Handspitze mittels Gelenk-Interpolation zu einer festgelegten Position.

### Eingabeformat

MOV <Zielposition> [ , <Abstand>] [<Verknüpfungsbedingung>]

<Zielposition>	Legt die Zielposition fest.
<Abstand>	Legt den Verfahrwegbetrag in Werkzeugrichtung auf der Z-Achse fest (Abstand zur Zielposition).
<Verknüpfungsbedingung>	Es können die Verknüpfungen WTH und WTHIF verwendet werden.

### Erläuterung

- Bei einer Bewegung mittels Gelenk-Interpolation wird die Zielposition durch gleichzeitiges Starten und Stoppen der Roboterelenke angefahren.
- Die Arbeitsgeschwindigkeit wird dabei durch den Playback-Übersteuerungswert, den OVRD- und den JOVRD-Befehl festgelegt. Der SPD-Befehl ist wirkungslos.
- Liegt die Zielposition außerhalb des zulässigen Roboterarbeitsbereichs, oder wird die Geschwindigkeit überschritten, erfolgt eine Fehlermeldung.
- Wird die Ausführung des MOV-Befehls unterbrochen und nach einem JOG-Betrieb neu gestartet, bewegt sich die Roboterhand, von dieser Position ausgehend, um den Verfahrweg.

### Programmbeispiel (MELFA-BASIC III-Befehle)

10	MOV P1	Position 1 anfahren
20	MOV J1	Position 1 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 (4+5) anfahren, wenn Eingangsbit 18 gleich 1 ist, setze Ausgangsbit 20 auf 1

Siehe auch Roboterbewegungsbeefhle, WTH, WTHIF, FINE, CNT und Bewegungspositionen.

## 8.2.32 Movement Position ♦ (Movement Position)

### Funktion: Koordinatenposition anfahren

Legt die Koordinaten der Endposition für eine Interpolation fest.

#### Eingabeformat 1

```
<Temporäre Zielkoordinaten>[, <Verfahrbetrag>]
```

#### Eingabeformat 2

```
<Abstand>
```

<Temporäre Zielkoordinaten>

Legt die Zielposition fest, wenn kein Verfahrbetrag angegeben ist.

<Verfahrbetrag>

Modifiziert die temporären Zielkoordinaten und legt den Abstand zur Zielposition in Richtung der Werkzeuglängsachse (Z-Achse) fest. Die Position, die durch Angabe der temporären Zielkoordinaten und des Verfahrbetrags dargestellt wird, ist die Zielposition. Der Fahrweg muß nicht angegeben werden.

<Abstand>

Modifiziert die momentane Position und legt den Abstand zur Zielposition in Richtung der Werkzeuglängsachse (Z-Achse) fest. Die Position, die durch die aktuellen Koordinaten und dem Abstand dargestellt wird, ist die Zielposition.

### Erläuterung

- Eingabeformat 1 ist das Standardeingabeformat.
- Die momentanen Koordinaten müssen nicht angegeben werden, wenn das Eingabeformat 2 verwendet wird. In diesem Fall werden die momentanen Koordinaten als temporäre Koordinaten verwendet. Es wird die Position beschrieben, die man durch Addition der Z-Koordinate in Werkzeuglängsrichtung erhält.
- Eingabeformat 2 kann für die MOV- und die MVS-Anweisung verwendet werden.

### Programmbeispiel (MELFA-BASIC III-Befehle)

10	MVS P1	Position P1 mittels Linear-Interpolation anfahren
20	MVS P2+P10,100.0+2	Position anfahren, die 100.0+2 mm in Werkzeuglängsrichtung von den Koordinaten (P2+P10) entfernt ist
30	MVS ,10*2	Position anfahren, die 10*2 mm in Werkzeuglängsrichtung von der aktuellen Position entfernt ist

Siehe auch MOV, MVS und Syntaxdiagramme.

### 8.2.33 MVC (Move C)

#### Funktion: Kreis-Interpolation

Bewegt die Handspitze mittels 3D-Kreis-Interpolation entlang eines durch Startposition, Zwischenposition 1, Zwischenposition 2 und Startposition festgelegten Kreises.

#### Eingabeformat

```
MVC <Startposition>,<Zwischenposition 1>,<Zwischenposition 2>  
[<Verknüpfungsbedingung>]
```

<Startposition>	Legt den Start und Endpunkt des Kreises fest.
<Zwischenposition 1>	Legt die erste Zwischenposition auf dem Kreisumfang fest.
<Zwischenposition 2>	Legt die zweite Zwischenposition auf dem Kreisumfang fest.
<Verknüpfungsbedingung>	Es können die Verknüpfungen WTH und WTHIF verwendet werden.

#### Erläuterung

- Mittels Kreis-Interpolation bewegt sich die Handspitze des Roboters auf dem Kreisumfang des durch die 3 Punkte festgelegten Kreises (360°).
- Die Arbeitsgeschwindigkeit wird durch den Playback-Übersteuerungswert, den OVRD- und den SPD-Befehl festgelegt. Der JOVRD-Befehl ist wirkungslos.
- Wird der zulässige Arbeitsbereich oder die Geschwindigkeit überschritten, erfolgt eine Fehlermeldung.
- Während der Kreis-Interpolation bleibt die Orientierung des Roboters unverändert.
- Die Arbeitsgeschwindigkeit ist von der Größe der Positionsänderungen abhängig.
- Entspricht die momentane Position nicht der Startposition, fährt der Roboter die Startposition mittels Linear-Interpolation an.
- Wird die Kreisbogenbewegung fortgesetzt, wenn die Kreis-Interpolation unterbrochen und nach einem JOG-Betrieb neu gestartet wurde, bewegt sich der Roboter mittels Linear-Interpolation zu der Stopposition und setzt dort die Kreisbogenbewegung fort.

**Programmbeispiel (MELFA-BASIC III-Befehle)**

10	MVC P1,P2,P3	Bewegung entlang des durch P1, P2 und P3 festgelegten Kreises
20	MVC P1,J2,P3	Bewegung entlang des durch P1, J2 und P3 festgelegten Kreises
30	MVC P1,P2,P3 WTH M_OUT(17)=1	Bewegung entlang des durch P1, P2 und P3 festgelegten Kreises und Setzen des Ausgangsbits 17 auf 1
40	MVC P3,(PLT 1,5),P4 WTHIF M_IN(20)=1,M_OUT(21)=1	Bewegung entlang des durch P3, (PLT 1,5) und P4 festgelegten Kreises und Setzen des Ausgangsbits 21, falls Eingangsbit 20 gleich 1

Siehe auch Roboterbewegungsbefehle, WTH, WTHIF, FINE, CNT und Bewegungspositionen.

## 8.2.34 MVR (Move R)

### Funktion: Kreis-Interpolation

Bewegt die Handspitze mittels 3D-Kreis-Interpolation entlang eines durch Startposition, Zwischenposition und Endposition festgelegten Kreisbogens.

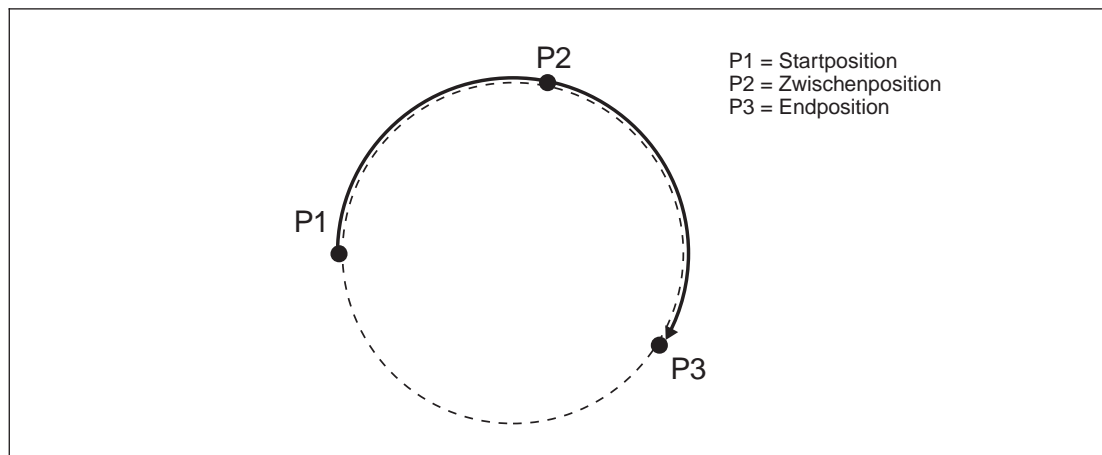
### Eingabeformat

```
MVR <Startposition>,<Zwischenposition>,<Endposition>
    [<Verknüpfungsbedingung>]
```

<Startposition>	Legt den Startpunkt des Kreises fest.
<Zwischenposition>	Legt die Zwischenposition auf dem Kreisumfang fest.
<Endposition>	Legt die Endposition auf dem Kreisumfang fest.
<Verknüpfungsbedingung>	Es können die Verknüpfungen WTH und WTHIF verwendet werden.

### Erläuterung

- Mittels Kreis-Interpolation bewegt sich der Roboterarm auf dem Kreisbogen, der durch die 3 Punkte festgelegt ist.



**Abb. 8-6:** Beispiel zur Kreis-Interpolation über eine Zwischenposition

- Die Arbeitsgeschwindigkeit wird durch den Playback-Übersteuerungswert, den OVRD- und den SPD-Befehl festgelegt. Der JOVRD-Befehl ist wirkungslos.
- Wird der zulässige Arbeitsbereich oder die Geschwindigkeit überschritten, erfolgt eine Fehlermeldung.
- Die Roboterstellung wird vom Startpunkt zum Endpunkt interpoliert. Während der Kreis-Interpolation ändert sich die Stellung nicht.
- Die Arbeitsgeschwindigkeit ist von der Größe der Positions- und Stellungsänderungen abhängig.
- Entspricht die aktuelle Position nicht der Startposition, fährt der Roboter die Startposition mittels Linear-Interpolation an.
- Der Roboter verfährt mit Linear-Interpolation, wenn zwei der drei Positionen gleich sind oder alle Positionen auf einer Geraden liegen. Es erfolgt keine Fehlermeldung.

**Programmbeispiel (MELFA-BASIC III-Befehle)**

10	MVR P1,P2,P3	Bewegung entlang des durch P1, P2 und P3 festgelegten Kreisbogens
20	MVR P1,J2,P3	Bewegung entlang des durch P1, J2 und P3 festgelegten Kreisbogens
30	MVR P1,P2,P3 WTH M_OUT(17)=1	Bewegung entlang des durch P1, P2 und P3 festgelegten Kreisbogens und Setzen des Ausgangsbits 17 auf 1
40	MVR P3,PLT (1,5),P4 WTHIF M_IN(20)=1,M_OUT(21)=1	Bewegung entlang des durch P3, PLT1,5 und P4 festgelegten Kreisbogens und Setzen des Ausgangsbits 21 auf 1, wenn Eingangsbit 20 gleich 1 ist

Siehe auch Roboterbewegungsbefehle, WTH, WTHIF, FINE, CNT und Bewegungspositionen.



## 8.2.35 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.

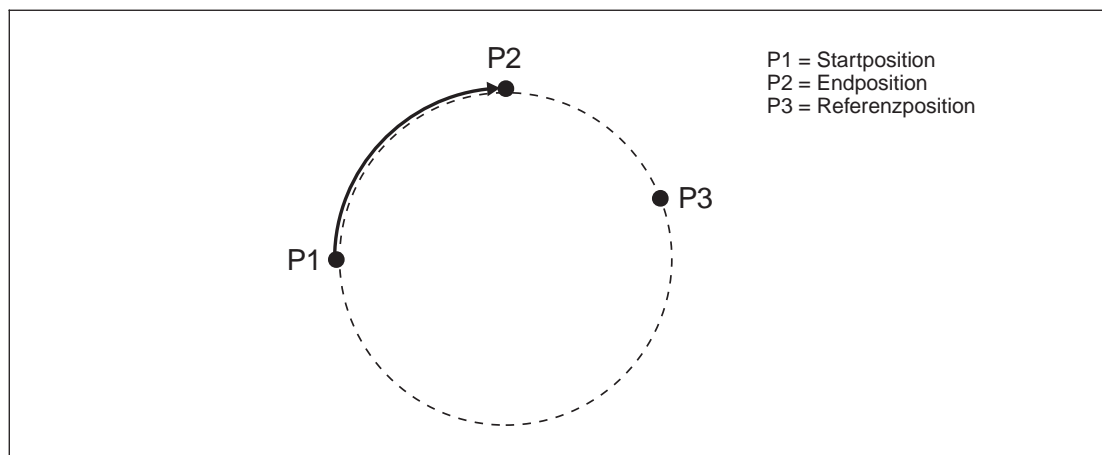
### Eingabeformat

```
MVR2 <Startposition>,<Endposition>,<Referenzposition>
      [<Verknüpfungsbedingung>]
```

<Startposition>	Legt den Startpunkt des Kreises fest.
<Endposition>	Legt die Endposition auf dem Kreisumfang fest.
<Referenzposition>	Legt die Referenzposition auf dem Kreisumfang fest.
<Verknüpfungsbedingung>	Es können die Verknüpfungen WTH und WTHIF verwendet werden.

### 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. 8-7:** Beispiel zur Kreis-Interpolation über einen Referenzpunkt

- Die Arbeitsgeschwindigkeit wird durch den Playback-Übersteuerungswert, den OVRD- und den SPD-Befehl festgelegt. Der JOVRD-Befehl ist wirkungslos.
- Wird der zulässige Arbeitsbereich oder die Geschwindigkeit überschritten, erfolgt eine Fehlermeldung.
- Die Roboterstellung wird vom Startpunkt zum Endpunkt interpoliert. Die Stellung des Referenzpunktes hat keinen Einfluß.
- Die Arbeitsgeschwindigkeit ist von der Größe der Positions- und Stellungsänderungen abhängig.
- Entspricht die aktuelle Position nicht der Startposition, fährt der Roboter automatisch die Startposition mittels Linear-Interpolation an.

- Wird die Kreisbogenbewegung fortgesetzt, wenn die Kreis-Interpolation unterbrochen und nach einem JOG-Betrieb neu gestartet wurde, bewegt sich der Roboter mittels Linear-Interpolation zu der Stopposition und setzt dort die Kreisbogenbewegung fort.
- Der Roboter verfährt mit Linear-Interpolation, wenn zwei der drei Positionen gleich sind oder alle Positionen auf einer Geraden liegen. Es erfolgt keine Fehlermeldung.

#### Programmbeispiel (MELFA-BASIC III-Befehle)

10	MVR2 P1,P2,P3	Bewegung auf dem durch P1, P2 und P3 festgelegten Kreisbogen
20	MVR P1,J2,P3	Bewegung auf dem durch P1, J2 und P3 festgelegten Kreisbogen
30	MVR 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	MVR 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

Siehe auch Roboterbewegungsbefehle, WTH, WTHIF, FINE, CNT und Bewegungspositionen.

## 8.2.36 MVS (Move S)

### Funktion: geradlinige Bewegung

Bewegt die Handspitze mittels Linear-Interpolation zur festgesetzten Position.

#### Eingabeformat 1

```
MVS <Zielposition> [, <Abstand>] [<Verknüpfungsbedingung>]
```

#### Eingabeformat 2

```
MVS , <Verfahrbetrag>
```

<Zielposition>	Legt die Zielposition fest.
<Abstand>	Legt den Verfahrbetrag in Werkzeugrichtung auf der Z-Achse fest. (Abstand zur Zielposition). Bei einem positiven Betrag bewegt sich die Handspitze in Werkzeuglängsrichtung nach vorne. Bei Angabe eines negativen Verfahrbetrags wird die Handspitze in Werkzeuglängsrichtung zurückgefahren.
<Verknüpfungsbedingung>	Es können die Verknüpfungen WTH und WTHIF verwendet werden.
<Verfahrbetrag>	Legt den Verfahrbetrag von der Augenblicksposition in Werkzeuglängsrichtung auf der Z-Achse fest (Abstand zur Zielposition). Bei einem positiven Betrag bewegt sich die Handspitze in Werkzeuglängsrichtung nach vorne. Bei Angabe eines negativen Verfahrbetrags wird die Handspitze in Werkzeuglängsrichtung zurückgefahren.

### Erläuterung

- Dieser Befehl verfährt die Handspitze entlang einer geraden Linie zur festgelegten Position.
- Die Arbeitsgeschwindigkeit wird durch den Playback-Übersteuerungswert, den OVRD- und den SPD-Befehl festgelegt. Der JOVRD-Befehl ist wirkungslos.
- Wird der zulässige Arbeitsbereich oder die Geschwindigkeit überschritten, erfolgt eine Fehlermeldung.
- Die Roboterstellung wird vom Startpunkt zum Endpunkt interpoliert.
- Die Arbeitsgeschwindigkeit ist von der Größe der Positions- und Stellungsänderungen abhängig.
- Wird die geradlinige Bewegung fortgesetzt, wenn die Kreis-Interpolation unterbrochen und nach einem JOG-Betrieb neu gestartet wurde, bewegt sich der Roboter zu der Stopposition und setzt dort die geradlinige Bewegung fort.
- Beginnt ein Neustart mit Befehlen, die sich auf die aktuelle Position beziehen (z. B. MVS, 100), bewegt sich die Handspitze um den verbleibenden Verfahrbetrag.

**Programmbeispiel (MELFA-BASIC III-Befehle)**

10	MVS P1	Position P1 mittels Linear-Interpolation anfahren
20	MVS J1	Position J1 mittels Linear-Interpolation anfahren
30	MVS (PLT1,10),-100.0 WTH M_OUT(17)=1	Position 100 mm über dem Gitterpunkt 10 der Palette 1 mittels Linear-Interpolation anfahren und Ausgangsbit 17 auf 1 setzen
40	MVS P4+P5,-50.0 WTHIF M_IN(18)=1,M_OUT(20)=1	Position anfahren, die um 50 mm in Werkzeuglängs- richtung entfernt von (P4+P5) liegt und Ausgangsbit 20 auf 1 setzen, falls Eingangsbit 18 gleich 1 ist
50	MVS ,-50	Position anfahren, die 50 mm in Werkzeuglängsrichtung von der aktuellen Position entfernt ist

Siehe auch Roboterbewegungsbeefehle, WTH, WTHIF, FINE, CNT und Bewegungspositionen.

## 8.2.37 OADL (Optimum Acceleration/Deceleration)

### Funktion: Optimale Beschleunigung/Abbremsung

Legt die optimale Beschleunigungs-/Abbremszeit in Abhängigkeit von der Last fest.

### Eingabeformat

```
OADL <Schalter> [, <Standardlast> [, <Maximallast> ,  
      <Handnummer>]]
```

<Schalter>	Legt fest, ob die Einstellung für die optimale Beschleunigung/Abbremsung freigegeben oder gesperrt ist.
<Standardlast>	Legt die Last bei geöffnetem Handgreifer in kg fest (Gewicht der Hand). Bei fehlender Angabe wird die optimale Beschleunigungs-/Abbremszeit für die Last berechnet, die beim HNBD-Parameter angegeben wurde. Standardwert: 2 kg.
<Maximallast>	Legt die Last bei geschlossenem Handgreifer in kg fest. Bei fehlender Angabe bleibt die Beschleunigungs-/Abbremszeit unabhängig vom Handgreiferzustand gleich.
<Handnummer>	Legt die Handnummer für die Einstellung der optimalen Beschleunigungs-/Abbremszeit fest. $1 \leq \text{Handnummer} \leq 3$ Bei fehlender Angabe bleibt die Beschleunigungs-/Abbremszeit unabhängig vom Handgreiferzustand gleich.

### Erläuterung

- Es kann in einem Programm eine Handnummer angegeben werden.
- Vor Anwendung des OADL-Befehls sollten folgende Parametereinstellungen überprüft werden:
  - GDIR (Mechanische Einbaurichtung)  
Legt die mechanische Einbaurichtung des Roboters fest. Sie braucht bei Verwendung der Standardeinbaurichtung (Bodenmontage) nicht verändert zu werden.
  - HNDM (Last für Handgreifer)  
Legt die Handgreiferlast fest. Dieser Wert wird bei fehlender Angabe der Standardlast im OADL-Befehl verwendet.
  - HNDG (Handspezifischer Schwerpunkt)  
Der handspezifische Schwerpunkt wird im Werkzeugkoordinatensystem festgelegt.
- In der Regel gilt die Standardlast für den geöffneten Handgreifer und die Maximallast für den geschlossenen Handgreifer. Die Einstellungen können mit dem GCD-Befehl umgekehrt werden.
- Ist die OADL-Einstellung freigegeben, ist der Befehl für die Einstellung der Beschleunigungs-/Abbremszeit unwirksam, auch wenn er ausgeführt wird. Der OADL-Befehl hat die höhere Priorität.
- Der OADL-Befehl ist so lange aktiv, bis er auf 0 gesetzt wird (OADL 0), das Programm zurückgesetzt oder die Spannungsversorgung abgeschaltet wird.

**Programmbeispiel (MELFA-BASIC III-Befehle)**

10	OADL 1,2,3,1	Berechnet die optimale Beschleunigung/Abbremsung für die geöffnete/geschlossene Hand 1 mit einer Standardlast von 2 kg und einer Maximallast von 3 kg
15	HND 1=1	Öffnet Hand 1
20	MOV P1	Position P1 anfahren
30	MOV P2	Position P2 anfahren
35	HND 1=0	Schließt Hand 1
40	MOV P3	Position P3 anfahren
50	OADL 0	Sperrt optimale Beschleunigung/Abbremsung
60	END	Programmende

## 8.2.38 ON COM GOSUB (ON Communication Go Subroutine)

### Funktion: Sprung zu einem Unterprogramm

Legt den Sprung in ein Unterprogramm fest, wenn ein Interrupt von einer Kommunikationsleitung anliegt.

### Eingabeformat

```
ON COM [( <Nummer der Kommunikationsleitung> ) ] GOSUB
      <Sprungziel>
```

<Nummer der Kommunikationsleitung>    Legt die Nummer der Kommunikationsleitung (1 oder 2) fest.

<Sprungziel>    Legt eine Zeilennummer oder eine Marke fest

### Erläuterung

- Bei fehlender Nummer der Kommunikationsleitung wird der Standardwert 1 gesetzt.
- Liegt auf COM (1) und COM (2) gleichzeitig ein Interrupt an, hat COM (1) die höhere Priorität.
- Bei anliegendem Interrupt wird die Roboterbewegung gestoppt. Mit COM STOP kann der Interrupt ignoriert werden, und die Roboterbewegung wird nicht unterbrochen.
- Mit dem COM ON-Befehl wird der Interrupt wieder zugelassen.
- Die Prioritäten der Interrupts sind:  
COM > ACT > WTHIF (WTH) > Impulsausgang

### Programmbeispiel (MELFA-BASIC III-Befehle)

```
10  ON COM(2) GOSUB 1000    Springt zur Zeile 1000, wenn auf der Kommunikations-
                             leitung Nummer 2 ein Interrupt anliegt
20  ON COM GOSUB *RECV    Springt zu Marke *RECV, wenn auf der Kommunikations-
                             leitung Nummer 1 ein Interrupt anliegt
```

Siehe auch COM ON, COM OFF, COM STOP und OPEN.

## 8.2.39 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.

### Erläuterung

- Der Wert des Ausdrucks legt fest, zu welcher Zeilennummer oder Marke das Programm verzweigt wird. Beispiel: Ist der Wert der Variablen 2, wird das zweite Sprungziel aufgerufen.
- Die Länge der Zeilennummer oder Marke darf maximal eine Zeile (maximal 127 Zeichen) betragen.
- Wird eine Zeilennummer oder Marke aufgerufen, die nicht existiert oder zweimal definiert ist, erfolgt eine Fehlermeldung.
- Ist der Wert des Ausdrucks eine reelle Zahl, wird sie vor der Verzweigung in eine Integer-Zahl umgewandelt (gerundet).
- Ist der Wert des Ausdrucks 0, oder größer als die Zahl der Zeilen oder der angegebenen Marken, verzweigt das Programm in die nächste Zeile.
- Ist der Wert des Ausdrucks negativ oder größer als 127, erfolgt eine Fehlermeldung.
- Gibt der Wert des Ausdrucks ein nicht vorhandenes Sprungziel an, 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 127 ist	ERROR
Zeilennummer oder Marke ist nicht angegeben	ERROR

**Tab. 8-8:** Werte des Ausdrucks und deren Verarbeitung

### Programmbeispiel (MELFA-BASIC III-Befehle)

```
10 ON MDATA GOSUB 1000, *SUBPR      Springt zur Zeile 1000, wenn auf der Wert
                                         des Ausdrucks 1 ist und zur Marke *SUBPR,
                                         wenn der Wert 2 ist
```

Siehe auch RETURN, END, ON GOTO, GOSUB und IF THEN ELSE.



## 8.2.40 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.

### 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.
- Die Länge der Zeilennummer oder Marke darf maximal eine Zeile (maximal 127 Zeichen) betragen.
- Wird eine Zeilennummer oder Marke aufgerufen, die nicht existiert oder zweimal definiert ist, erfolgt eine Fehlermeldung.
- Ist der Wert der Variablen eine reelle Zahl, wird sie vor der Verzweigung in eine Integer-Zahl umgewandelt (gerundet).
- Ist der Wert des Ausdrucks 0 oder größer als die Zahl der angegebenen Sprungziele, verzweigt das Programm in die nächste Zeile.
- Ist der Wert des Ausdrucks negativ oder größer als 127, erfolgt eine Fehlermeldung.
- Gibt der Wert des Ausdrucks ein nicht vorhandenes Sprungziel an, 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 127 ist	ERROR
Zeilennummer oder Marke ist nicht angegeben	ERROR

**Tab. 8-9:** Werte des Ausdrucks und deren Verarbeitung

### Programmbeispiel (MELFA-BASIC III-Befehle)

```
10  ON MDATA GOTO 1000, *SUBPR
```

Springt zur Zeile 1000, wenn auf der Wert des Ausdrucks 1 ist und zur Marke \*SUBPR, wenn der Wert 2 ist

Siehe auch GOTO, ON ... GOSUB und IF THEN ELSE.

## 8.2.41 OPEN (Open)

### Funktion: Datei öffnen

Öffnet eine Datei.

### Eingabeformat

```
OPEN "<Dateibezeichnung>" [FOR <Modus>] AS [#] <Dateinummer>
```

- <Dateibezeichnung>      Gibt den Namen der Datei oder des Kommunikationskanals an. Die Dateibezeichnung "<Dateiname>:" 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: OUTPUT = Ausgabemodus (neue Datei)  
Legt eine neue Datei an und schreibt Daten in diese Datei.
  - APPEND: APPEND = Ausgabemodus (vorhandene Datei)  
Hängt Daten an das Ende einer vorhandenen Datei an.

Dateibezeichnung	Dateiname	Zugriffsmethode
Datei	Maximal 8 Zeichen	INPUT, PRINT, APPEND
Kommunikationsleitung	COM 1 :, COM 2 :, COM 3 :	Keine Angabe = Wahlfreier Modus

**Tab. 8-10:** Dateibezeichnung und Zugriffsmethode

- <Dateinummer>             $1 \leq \text{Dateinummer} \leq 8$   
Wird in der INPUT- und PRINT-Anweisung verwendet.

**Erläuterung**

- Die Dateinummer wird mit den Ziffern 1 bis 8 innerhalb der Anführungszeichen (""") festgelegt.
- Ein Kommunikationskanal wird wie eine Datei behandelt.
- Für die Kommunikationskanäle gelten die in Tabelle 8-11 aufgeführten Dateinamen. Werden die freien Steckkartenplätze als Kanal 2 verwendet, gilt die interne Schnittstellenummer.

Kommunikationsdatei	Schnittstelle	Interne Schnittstellenummer
COM 1	Standard RS-232-C	—
COM 2	Standard RS-422	—
COM 3	Freier Steckkartenplatz 1	0: RS-422 1: RS-232-C
COM 4	Freier Steckkartenplatz 2	
COM 5	Freier Steckkartenplatz 3	

**Tab. 8-11:** Zuordnung der Schnittstellen

- Die Kommunikationskanäle und Schnittstellenummern werden wie folgt verwendet:  
COM3:0 RS-422-C Schnittstelle auf dem freien Steckkartenplatz 1  
COM4: RS-232-C Schnittstelle auf dem freien Steckkartenplatz 2  
COM5:1 RS-232-C Schnittstelle auf dem freien Steckkartenplatz 3
- Eine Änderung der Einstellungen, wie Datentransferraten, erfolgt über die Parameter.
- Mit END oder CLOSE wird die Datei geschlossen.

**Programmbeispiel (MELFA-BASIC III-Befehle)**

```

10 OPEN "COM1:" AS#1  Öffnet die RS-232-C-Kommunikationsdatei
20 INPUT #1,C$        Liest die Daten in C$ ein
30 PRINT #1,"HELLO"   Zeichenkette HELLO über Schnittstelle ausgeben
40 CLOSE              Datei schließen
50 END                Programmende

```

Siehe auch INPUT, PRINT und COM ON/OFF/STOP und ON COM GOSUB.

## 8.2.42      **ORG (Origin)**

### **Funktion: Nullpunkt einstellen**

Legt den Nullpunkt fest.

### **Eingabeformat**

ORG [ <Zusatzachse Achsennummer> ]
------------------------------------

<Zusatzachse Achsennummer>

Die Zusatzachse Achsennummer ist 1 oder 2.

### **Erläuterung**

- Fehlt die Achsennummer der Zusatzachse, wird die Ersatzposition (P-SAFE) mittels Gelenk-Interpolation angefahren.
- Ist die Achsennummer der Zusatzachse angegeben, wird der Nullpunkt der Zusatzachse gesetzt.  
Beachten Sie: Ist die Zusatzachse mit einem Absolutencodersystem ausgerüstet, wird der Befehl nicht ausgeführt.

### **Programmbeispiel (MELFA-BASIC III-Befehle)**

10	ORG	Bewegt den Roboterarm zur Ersatzposition (P_SAFE)
20	ORG 1	Setzt den Nullpunkt der Zusatzachse

### 8.2.43 OUT ♦♦ (Out)

### Funktion: Ausgabe

Gibt ein Bit über den Ausgabeport aus.

## Eingabeformat

OUT <Ausgangs-Bitnummer> = <Wert des Eingangs-/Ausgangsbits>  
[<Ergänzungsangabe>]

<Ausgangs-Bitnummer>

$$1 \leq \text{Ausgangs-Bitnummer} \leq 80$$

Hinter der OUT-Befehl muß ein Leerzeichen eingefügt werden.

OUT1 wird als Anweisung zur Variablendeklaration interpretiert.

<Wert des Eingangs-/Ausgangsbits>

Ein 1-Bit-Signal kann 0 oder 1 sein.

<Ergänzungsangabe>

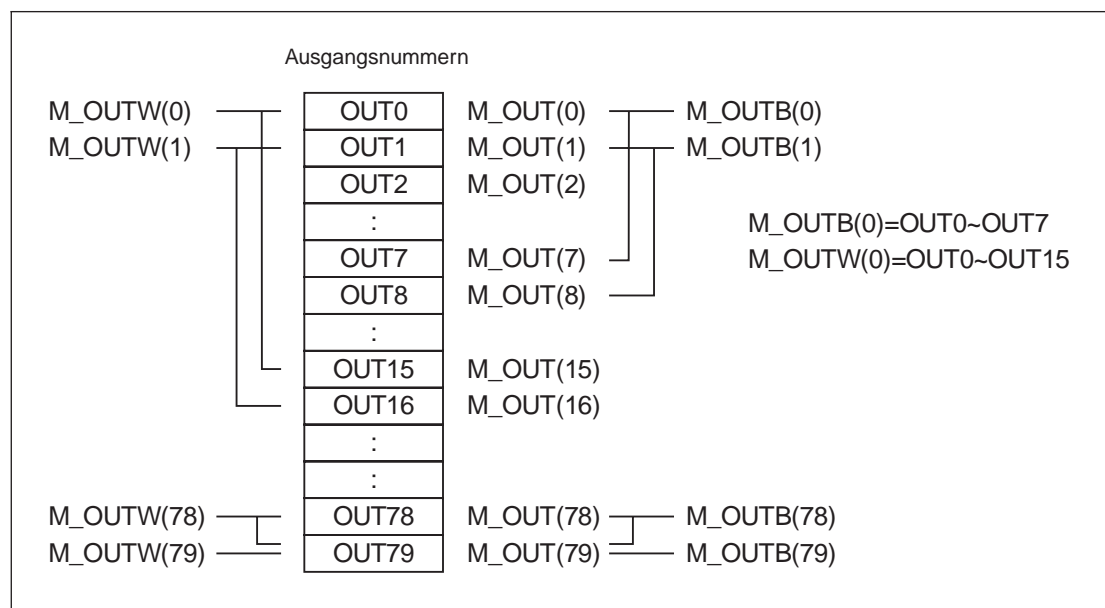
DLY   Sekunden

Durch die DLY-Anweisung kann das Bit für eine festgelegte Zeit ausgegeben werden.

Währenddessen wird das Programm weiter abgearbeitet.

### Erläuterung

- Das Bit wird über den allgemeinen Ausgabeport ausgegeben.
- Auch durch Eingabe des NOT-Halt-Signals, durch Ausführung der END-Anweisung oder durch Zurücksetzen des Programms ändert sich das Ausgangssignal nicht.
- Es gelten die selben Bedingungen wie für Roboterstatusvariablen M\_OUT(). Abbildung 8-8 zeigt zusätzlich die Beschreibung der M\_INB()- und der M\_INW-Anweisung.



**Abb. 8-8: Ausgangsadressen**

**Programmbeispiel (MELFA-BASIC III-Befehle)**

10 OUT 10=0                   Setzt das Ausgangsbit Nummer 10 auf 0  
20 OUT 10=1 DLY3.0       Setzt das Ausgangsbit Nummer 10 für 3 Sekunden auf 1

Siehe auch IN und Roboterstatusvariablen (siehe Seite 7-19 ff.).

## 8.2.44 OVRD (Override)

### Funktion: Übersteuerung

Legt den Programmwert für die Geschwindigkeits-Übersteuerung fest.

### Eingabeformat

```
OVRD <Übersteuerungswert>
```

<Übersteuerungswert>      Legt den prozentualen Übersteuerungswert fest.  
 $1 \leq \text{Übersteuerungswert} \leq 200.0$

### Erläuterung

- 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:

Gelenk-Interpolation = Playback-Übersteuerungswert  $\times$  Einstellwert des OVRD-Befehls  $\times$  Einstellwert des JOVRD-Befehls

Linear-Interpolation = Playback-Übersteuerungswert  $\times$  Einstellwert des OVRD-Befehls  $\times$  Einstellwert des SPD-Befehls

- 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.
- Bei kurzen Verfahrwegen ist es möglich, daß der festgelegte Übersteuerungswert nicht erreicht wird.
- Liegt der Übersteuerungswert außerhalb des Wertebereiches des Roboters, erfolgt eine Fehlermeldung. Der Wert muß zwischen 0 und 100% liegen.



#### ACHTUNG

*In der Grundeinstellung ist die CNT-Einstellung freigegeben. Wird die Verfahrgeschwindigkeit, die Beschleunigungs- oder die Abbremszeit geändert, verändert sich auch die Verfahrkurve. Achten Sie darauf, daß es zu keinen Zusammenstößen mit umliegenden Einrichtungen kommt. Kommt es zu Zusammenstößen, sperren Sie die CNT-Einstellung oder verkürzen Sie die Beschleunigungs-/Abbremszeit.*

**Programmbeispiel (MELFA-BASIC III-Befehle)**

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

Siehe auch JOVRD, SPD, Roboterbewegungsbefehle und Roboterstatusvariablen.



## 8.2.45 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>, <numerischer Operationsausdruck>

<Palettennummer>

Wählt eine vorher mit dem DEF PLT-Befehl definierte Palette aus.

$1 \leq \text{Palettennummer} \leq 8$

<numerischer Operationsausdruck>

Legt die Positionsnummer für die berechneten Koordinaten fest.

### 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.

### Programmbeispiel (MELFA-BASIC III-Befehle)

10	M1=0	Setzt M1 auf 0
20	DEF PLT 1,P10,P11,P12,P13,8,8,1	Definiert Palette Nummer 1
30	P1=PLT 1,5	Weist P1 die Koordinaten des Gitterpunktes 5 zu
40	M1=M1+1	Erhöht den Wert von M1 um 1
50	MOV (PLT 1,M1),-50	Position anfahren, die um 50 mm in Werkzeugzeuglängsrichtung von Position (PLT 1,M1) entfernt liegt

Siehe auch DEF PLT und Roboterbewegungsbefehle.

## 8.2.46 PRINT# (Print)

### Funktion: Daten übertragen

Überträgt Daten in eine Datei oder an ein Ausgabegerät.

### 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.

### Erläuterung

- Fehlt eine Angabe für <Ausdruck>, wird ein „Carriage Return“ ausgegeben.
- Bei fehlender <Dateinummer> wird der Standardwert 1 verwendet.
- Ausgabeformat der Daten:  
 Der Platz für die Ausgabe von <Ausdruck> ist in Einheiten von 10. 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.

### Beispiel ▽

```
10 M1=123.5
20 P1=(460,280,150,0,180)(1,0)
```

nach Eingabe von

```
30 PRINT# 1,"OUTPUT TEST",M1,P1
```

OUTPUT TEST      123.5      (460,280,150,0,180)(1,0) ausgegeben

nach Eingabe von

```
30 PRINT# 1,"OUTPUT TEST";M1;P1
```

OUTPUT TEST 123.5(460,280,150,0,180)(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(460,280,150,0,180)(1,0) ausgegeben



### Programmbeispiel (MELFA-BASIC III-Befehle)

10	MDATA=150	Setzt MDATA auf 150
20	PRINT# 1,"***PRINT TEST***"	Gibt die Zeichenkette ***PRINT TEST*** aus
30	PRINT#	Gibt eine Leerzeile aus
40	PRINT# 1,"MDATA=",MDATA	Gibt die Zeichenkette MDATA= und den Wert von MDATA aus, (150)
50	PRINT#	Gibt eine Leerzeile aus
60	PRINT# 1,"*****"	Gibt die Zeichenkette ***** aus
70	END	Programmende

Folgendes Ergebnis wird ausgegeben:

\*\*\*PRINT TEST\*\*\*

MDATA=150

\*\*\*\*\*

Siehe auch OPEN und INPUT.

## 8.2.47 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.

### Erläuterung

- Die REM-Anweisung kann durch ein halbes Anführungszeichen (') oder durch den Schrägstrich (/) abgekürzt werden.

### Programmbeispiel (MELFA-BASIC III-Befehle)

```
10 REM ***Hauptprogramm***      Legt die Zeichenkette ***Hauptprogramm***  
20 '***Hauptprogramm***        als Kommentar fest
```

## 8.2.48 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.

### Erläuterung

- Dieser Befehl schließt das über den GOSUB-Befehl aufgerufene Unterprogramm ab und bewirkt einen Rücksprung zum Hauptprogramm.
- 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.
- Es erfolgt eine Fehlermeldung, wenn der RETURN-Befehl ausgeführt wird, ohne daß vorher ein GOSUB-Befehl ausgeführt wurde.
- Es erfolgt eine Fehlermeldung, wenn die Ausführung des GOSUB- und des RETURN-Befehls zu Konflikten in der Struktur der Programmebenen führt.

### Programmbeispiel (MELFA-BASIC III-Befehle)

```
100 RETURN          Rücksprung zum Hauptprogramm
   :
200 RETURN 0        Rücksprung zu der Zeile, in der der Interrupt aufgetreten ist
   :
300 RETURN 1        Rücksprung in die nächste Zeile hinter der Zeile, in der der
                    Interrupt aufgetreten ist
```

Siehe auch GOSUB, ON ... GOSUB und END.

## 8.2.49 SKIP ♦♦ (Skip)

### Funktion: Sprung in die nächste Zeile

Die Programmsteuerung springt in die nächste Zeile.

### Eingabeformat

SKIP
------

### Erläuterung

- Dieser Befehl kann ohne weitere Angaben, oder mit anderen Befehlen in Verbindung mit WTH und WTHIF verwendet werden.
- Wird der Befehl ohne weitere Angaben verwendet, springt die Programmsteuerung bei seiner Ausführung in die nächste Zeile.
- 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. (Es muß kein Neustart durchgeführt werden.)

### Programmbeispiel (MELFA-BASIC III-Befehle)

- |    |                              |   |
|----|------------------------------|---|
| 10 | SKIP                         | Programmsteuerung springt in die nächste Zeile<br>(Die Verarbeitung ist die gleiche, wie bei der<br>Beschreibung von Kommentaren)   |
| 20 | MOV P1 WTHIF M_IN(17)=0,SKIP | Fährt Position P1 mittels Gelenk-Interpolation an,<br>und unterbricht die Roboterbewegung, wenn das<br>Eingangsbit Nummer 17 gleich 0 wird<br>Die Programmsteuerung springt in die nächste<br>Zeile |

Siehe auch STOP, HLT, WTH und WTHIF.

## 8.2.50 SPD (Speed)

### Funktion: Geschwindigkeit festlegen

Legt die Geschwindigkeit für lineare und kreisförmige Bewegungen fest.

### Eingabeformat

```
SPD <Geschwindigkeitswert>
```

<Geschwindigkeitswert>

$1 \leq \text{Geschwindigkeitswert} \leq 650 \text{ [mm/s]}$

### Erläuterung

- Der SPD-Befehl ist nur bei linearen und kreisförmigen Bewegungen des Roboters wirksam.
- Der aktuelle Übersteuerungswert ergibt sich aus:

$$\begin{array}{lclclcl} \text{Aktueller} & & & & & & \\ \text{Übersteuerungs-} & = & \text{Playback-} & \times & \text{Einstellwert} & \times & \text{Einstellwert} \\ \text{wert} & & \text{Übersteuerungswert} & & \text{des} & & \text{des} \\ & & & & \text{OVRD-Befehls} & & \text{JOVRD-Befehls} \\ \\ \text{Aktueller} & = & \text{Playback-} & \times & \text{Einstellwert} & \times & \text{Einstellwert} \\ \text{Übersteuerungs-} & & \text{Übersteuerungswert} & & \text{des} & & \text{des} \\ \text{wert} & & & & \text{OVRD-Befehls} & & \text{SPD-Befehls} \end{array}$$

- Der Standardwert (M\_NSPD) beträgt 63.3 mm/s.
- 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.
- Bei zu kurzen Verfahrbewegungen, kann es sein, daß die festgelegte Geschwindigkeit nicht erreicht wird.
- Der Geschwindigkeitswert darf außerhalb des festgelegten Bereiches liegen. Wird jedoch die maximale Geschwindigkeit des Roboters überschritten, erfolgt eine Fehlermeldung.



#### ACHTUNG

*In der Grundeinstellung ist die CNT-Einstellung freigegeben. Wird die Verfahrgeschwindigkeit, die Beschleunigungs- oder die Abbremszeit geändert, verändert sich auch die Verfahrkurve. Achten Sie darauf, daß es zu keinen Zusammenstößen mit umliegenden Einrichtungen kommt. Kommt es zu Zusammenstößen, sperren Sie die CNT-Einstellung oder verkürzen Sie die Beschleunigungs-/Abbremszeit.*

### Programmbeispiel (MELFA-BASIC III-Befehle)

```
10 SPD 100           Legt die Geschwindigkeit auf 100 mm/s fest
20 MVS P1            Führt Position P1 mittels Linear-Interpolation an
30 SPD M_NSPD        Setzt Geschwindigkeit auf den Standardwert von 63,3 mm/s
```

Siehe auch ACL, DACL, CNT, OVRD und JOVRD.

## 8.2.51 STOP ♦♦ (Stop)

### Funktion: Programmablauf stoppen

Stoppt die Roboterbewegung und den Programmablauf. Hintergrundprozesse werden nicht unterbrochen.

### Eingabeformat

STOP
------

### Erläuterung

- Dieser Befehl stoppt die Abarbeitung des Programms. Die Roboterbewegung wird bis zum Stillstand abgebremst.
- Ein Neustart des Programms ist über die Teaching Box oder durch ein externes Startsignal möglich. Der Programmneustart beginnt eine Zeile nach dem STOP-Befehl. Wird der STOP-Befehl mit einer Verknüpfung verwendet, beginnt der Neustart in der Zeile, in der das Programm unterbrochen wurde.

### Programmbeispiel (MELFA-BASIC III-Befehle)

10	MOV P1	Position P1 anfahren
20	STOP	Programmablauf und Roboterbewegung stoppen
30	MOV P2	Position P2 anfahren

Siehe auch END, HLT, WTH und WTHIF.



## 8.2.52 SV (Servo)

### Funktion: Servo ein-/ausschalten

Schaltet den Servoantrieb oder die Bremse der Handgelenkdrehachse ein und aus.

### Eingabeformat

```
SV <Servostatus> [ , <Achse>]
```

<Servostatus>

Legt den Zustand des Servoantriebes (EIN/AUS) und der Bremse fest.

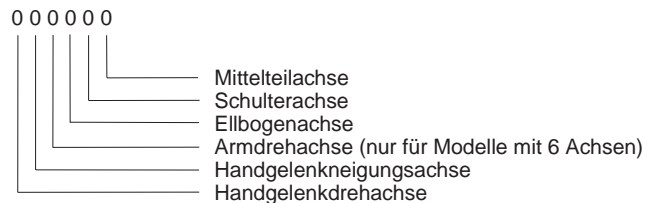
0: Grundzustand Servo AUS (Servo AUS, Bremse EIN)

1: Grundzustand Servo EIN (Servo EIN, Bremse AUS)

2: Freizustand (Servo AUS, Bremse AUS)

<Achsen>

Legt durch ein Bitmuster die Achse fest, deren Servomotor angesprochen wird. Bei den Robotern der RV-EN-Serie kann jedoch nur die Handgelenkdrehachse freigeschaltet werden.



Bei fehlender Achsenangabe werden alle Achsen ausgewählt.

### Erläuterung

#### Beispiel ▽

Der Befehl SV 2,&B100000 schaltet die Handgelenkdrehachse in den Freizustand.



- Ist für eine Achse der Servo-EIN-Zustand eingestellt, kann ein Roboterbewegungsbefehl ausgeführt werden. In diesem Fall wird das Servo-EIN-Signal SVA ausgegeben.
- Der Betrieb im Servo-Freizustand wird beendet, indem der Befehl SV 0 ausgeführt und der Servo für alle Achsen ausgeschaltet wird.



#### ACHTUNG

*Tritt während einer Interpolation im Servo-Freizustand oder im Servo-AUS-Zustand ein Interrupt auf, und der Servo-EIN Zustand wird eingestellt (durch das Steuergerät, die Teaching Box oder ein externes Eingangssignal), kann sich beim Einschalten des Servoantriebes die Achse, die sich im Freizustand befunden hatte (Handgelenkdrehachse) plötzlich zu einem Teaching-Punkt bewegen oder eine Fehlermeldung auftreten.*

*Schalten Sie bei einem Interrupt immer für alle Achsen den Servoantrieb aus, setzen Sie das Programm zurück und fahren Sie den Roboter im JOG-Betrieb zurück.*

**Programmbeispiel (MELFA-BASIC III-Befehle)**

10	MOV P1	Position P1 anfahren
20	SV 2,&B100000	Setzt den Servo der Handgelenkdrehachse auf Freizustand
30	MVS P1,-20	Position anfahren, die 20 mm in Werkzeugrichtung von P1 entfernt ist
40	SV 0,&B100000	Setzt den Servo der Handgelenkdrehachse auf AUS 1 (siehe auch Hinweis)
50	MVS P1	Position P1 anfahren
60	END	Programmende

**HINWEIS**

Da die Handgelenkdrehachse im Freizustand ist, endet die Interpolation bevor die Position 1 erreicht wird. Die Gelenkdaten (Handgelenkdrehachse), die auf der Teaching Box angezeigt werden, weichen von denen der aktuellen Roboterposition ab, bis der Betrieb im Freizustand (alle Achsen auf Servo AUS) beendet ist. Es wird für alle kartesischen Daten \*\*\*\*\* angezeigt. Das gilt auch, wenn während einer Interpolation eine Achse im Servo-AUS-Zustand ist.

### 8.2.53 TOOL (Tool)

#### Funktion: Werkzeug-Konvertierungsdaten

Legt die Werkzeug-Konvertierungsdaten fest.

#### Eingabeformat

```
TOOL <Werkzeug-Konvertierungsdaten>
```

#### Erläuterung

- Die Werkzeug-Konvertierungsdaten können nur mit dem TOOL-Befehl geändert werden.
- 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.
- Für den 5-achsigen Roboter kann nur die Z-Achse verändert werden. Für die anderen Achsen muß „0“ eingegeben werden.



#### ACHTUNG

**Achten Sie auf eine richtige Eingabe, wenn Sie die Werkzeugdaten ändern.**

**Bewegt sich der Roboter zur eingestellten Position, nachdem die Werkzeugdaten geändert wurden, ändert sich auch die Stellung des Roboters. Achten Sie darauf, daß es zu keinen Zusammenstößen mit umliegenden Einrichtungen kommt.**

**Auch wenn die Werkzeug-Konvertierungsdaten korrekt eingegeben wurden, ist es möglich, daß die gewünschte Genauigkeit aufgrund von Maßtoleranzen und Installationstoleranzen u.s.w. nicht erreicht wird.**

**Geben Sie bei 5-achsigen Robotern für alle Achsen bis auf die Z-Achse „0“ ein. Andere Werte können zu Vibrationen oder einer Fehlermeldung führen.**

#### Programmbeispiel (MELFA-BASIC III-Befehle)

10	TOOL P1	Setzt Werkzeug-Konvertierungsdaten auf P1
20	MVS P2	Position P2 anfahren
30	TOOL P_NTOOL	Setzt Werkzeug-Konvertierungsdaten auf den Standardwert P_NTOOL

Siehe auch BASE und Roboterstatusvariablen.

## 8.2.54 WHILE ~ WEND (While End)

### Funktion: Programmschleife

Solange die Schleifenbedingung wahr ist, wird das Programm zwischen der WHILE- und der WEND-Anweisung wiederholt.

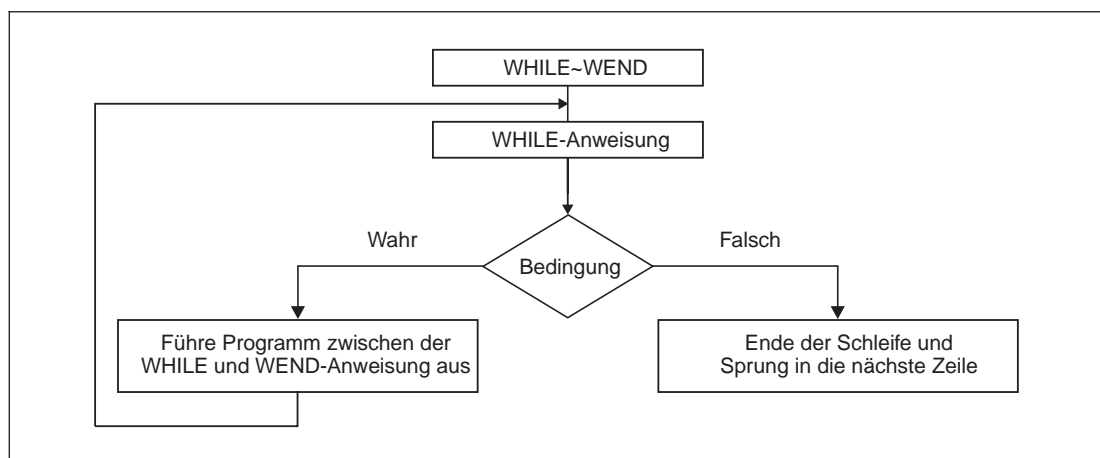
### Eingabeformat

```
WHILE    <Schleifenbedingung>
:
WEND
```

<Schleifenbedingung>    Legt die Abarbeitung der Schleife über eine Vergleichsbedingung fest (siehe Abbildung 8-9).

### Erläuterung

- Der Programmblock zwischen WHILE und WEND wird wiederholt, solange die Schleifenbedingung wahr ist. Ist die Schleifenbedingung unwahr, springt das Programm eine Zeile hinter die WEND-Anweisung.
- Es erfolgt eine Fehlermeldung, wenn eine WHILE-WEND-Schleife nicht mit einer WEND-Anweisung abgeschlossen wird. Steht zwischen der WHILE- und der WEND-Anweisung eine END-Anweisung oder wird die letzte Zeile des Programms ausgeführt, endet das Programm in dieser Zeile.



**Abb. 8-9:** Flußdiagramm WHILE ~ WEND-Anweisung

- **Programmebenen**
  - Durch die WHILE ~ WEND-Schleife wird die Programmstruktur um eine Ebene erweitert.
  - In einer WHILE ~ WEND-Schleife kann eine weitere WHILE ~ WEND- oder eine FOR ~ NEXT-Anweisung ausgeführt werden.
  - Die Programmstruktur darf maximal 16 Ebenen enthalten. Bei mehr als 16 Ebenen erfolgt eine Fehlermeldung.

**Programmbeispiel (MELFA-BASIC III-Befehle)**

20	WHILE (MDATA>=-5) AND (MDATA<=5)	Wiederholt den Programmblock, solange MDATA zwischen -5 und +5 liegt und springt zu Zeile 60, wenn MDATA außerhalb des Wertebereichs liegt
30	MDATA=-(MDATA+1)	Addiert 1 zu MDATA und kehrt das Vorzeichen um
40	PRINT# 1,MDATA	Gibt den Wert von MDATA aus
50	WEND	Springt zurück zur WHILE-Anweisung (Zeile 20)
60	END	Programmende

## 8.2.55 WTH ♦ (With)

### Funktion: Anweisung hinzufügen

Während einer Interpolationsbewegung wird eine zusätzliche Anweisung ausgeführt.

### Eingabeformat

```
WTH <Anweisung>
```

#### <Anweisung>

Legt die zusätzlich ausgeführte Anweisung fest. Es dürfen folgende Operationen ausgeführt werden:

- <num. Datentyp B><Substitutionsoperator><num. Datentyp A> [Substitutionen, Signal-Anweisungen (siehe entsprechendes Syntaxdiagramm)]
- HLT-Anweisung
- STOP-Anweisung
- SKIP-Anweisung
- HND-Anweisung
- OUT-Anweisung

### Erläuterung

- Dieser Befehl wird dazu verwendet, während einer Interpolationsbewegung eine zusätzliche Anweisung auszuführen.
- Es erfolgt eine Fehlermeldung, wenn die Anweisung nicht angegeben wird.
- Die Anweisung wird mit Beginn Roboterbewegung ausgeführt.
- Die Prioritäten der Interrupts sind:  
COM > ACT > WTHIF (WTH) > Impulsausgang

### Programmbeispiel (MELFA-BASIC III-Befehle)

10	MOV P1 WTH M_OUT(17)=1 DLY M1+M2	Position P1 anfahren und Ausgangsbit 17 für die Zeit von (M1+M2) Sekunden auf 1 setzen
20	MOV P2 WTH M_HND(1)=0	Position P2 anfahren und Hand Nummer 1 schließen

Siehe auch Roboterbewegungsbeefhle, WTHIF, Substitutionen und Syntaxdiagramme.

## 8.2.56 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 DEF ACT).
<Anweisung>	Legt die zusätzlich ausgeführte Anweisung fest (siehe auch WTH). Es dürfen folgende Operationen ausgeführt werden: <ul style="list-style-type: none"> <li>– &lt;num. Datentyp B&gt;&lt;Substitutionsoperator&gt;&lt;num. Datentyp A&gt;                [Substitutionen, Signal-Anweisungen (siehe entsprechendes Syntaxdiagramm)]</li> <li>– HLT-Anweisung</li> <li>– STOP-Anweisung</li> <li>– SKIP-Anweisung</li> <li>– HND-Anweisung</li> <li>– OUT-Anweisung</li> </ul>

### Erläuterung

- Dieser Befehl wird dazu verwendet, während einer Interpolationsbewegung eine zusätzliche bedingte Anweisung auszuführen.
- Es erfolgt eine Fehlermeldung, wenn die Anweisung nicht angegeben wird.
- Die Anweisung wird mit Beginn der Roboterbewegung ausgeführt.
- Die Prioritäten der Interrupts sind:  
COM > ACT > WTHIF (WTH) > Impulsausgang

### Programmbeispiel (MELFA-BASIC III-Befehle)

10	MOV P1 WTHIF M_IN(17)=1,HLT	Position P1 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 P2 anfahren und das Ausgangsbit Nummer 17 für die Zeit von (M1+2) Sekunden auf 1 setzen, falls M_RSPD>200

Siehe auch Roboterbewegungsbeefehle, WTH, Substitutionen und Syntaxdiagramme.

## 8.2.57 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.

### Erläuterung

- Wird der Befehl für einen Impulsausgang verwendet, wird der Impuls parallel zu den Befehlen der nachfolgenden Zeilen geschaltet.
- Nach Ablauf der eingestellten Verzögerungszeit, nimmt der Impulsausgang den Wert an, den er vor Ausführung des Befehles hatte.
- Wird während der festgesetzten Zeit eine END-Anweisung, die letzte Zeile des Programmes oder ein NOT-HALT ausgeführt, behält der Impulsausgang seinen gegenwärtigen Zustand bei. Der Ausgangszustand während der Unterbrechung kann über die Systemparameter eingestellt werden, wenn der Ausgang den Zustand annehmen soll, den er vor der Impulsausgabe oder nach abgelaufener Verzögerungszeit haben soll.
- Die Prioritäten der Interrupts sind:  
COM > ACT > WTHIF (WTH) > Impulsausgang

### Programmbeispiel (MELFA-BASIC III-Befehle)

```
10  P100=P1+P2*2           Übertrage das Ergebnis der Operation P1+P2*2 in die
                             Variable P100
20  M_OUT(17)=1 DLY 10.0    Setze das Ausgangsbit Nummer 17 für 10 Sekunden auf 1
```

Siehe auch DLY und Syntaxdiagramme.



## 8.3 Übersicht der SLIM-Befehle

### Einführung

In den nachfolgenden Abschnitten finden Sie eine Beschreibung der SLIM-Programmiersprache (JISB8439-1992), einer Programmiersprache für Industrieroboter. (Siehe auch: Industrial Robot Language „SLIM“, Japan Industrial Standards Committee.)

Die SLIM-Programmiersprache (Standard Language for Industrial Manipulators) ist über einen Zeitraum von 10 Jahren von Experten, Mitarbeitern von Universitäten und Ingenieuren entwickelt worden. Diese Fachleute, die durch den nahen Umgang mit Industrierobotern mit den auftretenden Problematiken vertraut sind, trafen sich im Juni 1994 und entwickelten die Roboterprogrammiersprache SLIM.

SLIM wurde auf der Basis der Programmiersprache BASIC entwickelt. Sie wurde speziell im Hinblick auf die Bewegungen und die Eingangs- und Ausgangsanweisungen für Roboter modifiziert und erweitert. Weitere Details zu den einzelnen Befehlen finden Sie in JISB8439-1992.

Tabelle 8-12 zeigt eine Übersicht der SLIM-Befehle

Befehl		Funktion	Abschnitt	Seite
<b>ACCEL</b>	(Accelerate)	Beschleunigung/Abbremsung einstellen	8.3.1	8-84
<b>CHANGE</b>	(Change)	Hand wechseln	8.3.2	8-85
<b>DEFINT</b>	(Define Integer)	Integer definieren	8.3.3	8-86
<b>DEFIO</b>	(Define I/O)	Ein-/Ausgänge definieren	8.3.4	8-87
<b>DEFJNT</b>	(Define Joint)	Gelenk definieren	8.3.5	8-88
<b>DEFPOS</b>	(Define Position)	Position definieren	8.3.6	8-89
<b>DELAY</b>	(Delay)	Verzögerung einstellen	8.3.7	8-90
<b>DRIVE</b>	(Drive)	Achsenbewegung	8.3.8	8-91
<b>GOHOME</b>	(Go Home)	Nullpunkt anfahren	8.3.9	8-92
<b>GRASP</b>	(Grasp)	Hand schließen	8.3.10	8-93
<b>HALT</b>	(Halt)	Programmablauf stoppen	8.3.11	8-94
<b>HAND</b>	(Hand)	Hand zuweisen	8.3.12	8-95
<b>HOLD</b>	(Hold)	Programm unterbrechen	8.3.13	8-96
<b>IN</b>	(Input)	Daten einlesen	8.3.14	8-97
<b>INPUT</b>	(Input)	Daten vom Eingabegerät einlesen	8.3.15	8-98
<b>IOBLOCK</b>	(I/O Block)	E/A-Block	8.3.16	8-99
<b>JSPEED</b>	(J Speed)	Achsengeschwindigkeit einstellen	8.3.17	8-100
<b>MOVE</b>	(Move)	Position anfahren	8.3.18	8-101
<b>OUT</b>	(Output)	Daten ausgeben	8.3.19	8-104
<b>PRINT</b>	(Print)	Daten ausgeben	8.3.20	8-105
<b>RELEASE</b>	(Release)	Handgreifer öffnen	8.3.21	8-106
<b>RESET</b>	(Reset)	E/A-Variable zurücksetzen	8.3.22	8-107
<b>SET</b>	(Set)	E/A-Variable setzen	8.3.23	8-108
<b>SPEED</b>	(Speed)	Verfahrgeschwindigkeit einstellen	8.3.24	8-109
<b>WAIT</b>	(Wait)	Programmunterbrechung	8.3.25	8-110

**Tab. 8-12:** Übersicht der SLIM-Befehle

### 8.3.1 ACCEL (Accelerate)

#### Funktion: Beschleunigung/Abbremsung einstellen

Legt den Wert für die Beschleunigung/Abbremsung fest.

#### Eingabeformat

```
ACCEL    <Beschleunigungszeit> [ , <Abbremszeit> ]
```

<Beschleunigungszeit> <Abbremszeit>    Legt die Beschleunigungs-/Abbremszeit fest.  
 $50 \leq \text{Beschleunigungszeit/Abbremszeit} \leq 2000$  [ms]

#### Erläuterung

- Die minimale Wert der Beschleunigungs-/Abbremszeit beträgt 50 ms. Der Standardwert beim Einschalten der Spannungsversorgung ist 200 ms.
- Der einmal eingestellte Wert ist so lange gültig, bis ein neuer Wert eingestellt wird.
- Die Beschleunigungszeit ist die Zeit, die zum Erreichen der maximalen Geschwindigkeit während einer Linear-Interpolation gebraucht wird. Wird die maximale Geschwindigkeit nicht erreicht, muß die Beschleunigungs-/Abbremszeit verkürzt werden.
- Der Beschleunigungs-/Abbremsweg ist abhängig von der Einstellung der Beschleunigungs-/Abbremszeit. Bei zu kurzen Wegen, wird die eingestellte Geschwindigkeit nicht erreicht.
- Ist die eingestellte Beschleunigungs-/Abbremszeit kürzer als 200 ms, kann eine Fehlermeldung für Geschwindigkeitsüberschreitung erfolgen. In Abhängigkeit von der Belastung kann sich zusätzlich die Lebensdauer der mechanischen Komponenten verkürzen. Die Beschleunigungs-/Abbremszeiten sollten daher über 200 ms und unter 2000 ms liegen.



#### ACHTUNG

*Wird die Roboterbewegung gestoppt (durch das Steuergerät, die Teaching Box oder ein externes Stopp-Signal) und die Beschleunigungs-/Abbremszeit liegt nicht im angegebenen Bereich, ist es möglich, daß der Roboter nicht sofort gestoppt wird. Verkürzen Sie in diesem Fall die Abbremszeit oder betätigen Sie die NOT-STOP-Taste.*

#### Programmbeispiel (SLIM-Befehle)

10 ACCEL 400,400            Setze die Beschleunigungszeit und die Abbremszeit auf 0,4 s

### 8.3.2 CHANGE (Change)

**Funktion: Hand wechseln**

Wechselt zu der festgelegten Hand.

**Eingabeformat**

CHANGE    <Handvariablenname>
-------------------------------

<Handvariablenname>      Legt den Handvariablennamen für die mit der  
HAND-Anweisung deklarierten Hand fest.

**Erläuterung**

- Dieser Befehl wird dazu verwendet, einen Handwechsel zu melden. Es ist kein Befehl, mit dem die Hand automatisch gewechselt werden kann. Soll die Hand automatisch gewechselt werden, muß ein Unterprogramm für die entsprechenden Ein- und Ausgaben geschrieben werden.

**Programmbeispiel (SLIM-Befehle)**

```
10  HAND HANDW = 2    Vergibt den Handvariablennamen HANDW für Hand 2  
                        (standardmäßig wird Hand 1 verwendet)  
100 CHANGE HANDW      Wechselt zur Hand mit dem Variablennamen HANDW (Hand 2)
```

Siehe auch HAND, RELEASE und GRASP.

### 8.3.3 DEFINT (Define Integer)

**Funktion: Integer definieren**

Deklariert eine arithmetische Variable.

**Eingabeformat**

```
DEFINT  <Name einer arithmetischen Variablen>  
        [, <Name einer arithmetischen Variablen>] ...
```

<Name einer arithmetischen Variablen>

Legt den Variablennamen fest.  
(Siehe auch Abschnitt 7.1.9  
„Arithmetische Variablen“.)

**Erläuterung**

- Der Variablenname darf aus maximal 8 Zeichen bestehen. Siehe auch Abschnitt 7.1.5 „Zeichentypen“ für die zugelassenen Zeichen.
- Bei Deklaration mehrerer Variablennamen, dürfen maximal 123 Zeichen (inklusive des Befehls) in einer Zeile verwendet werden.
- Für die Variablen gelten die in Abschnitt 7.1.9 festgelegten Regeln.
- Die mit DEFINT deklarierten Variablen können wie die mit „M“ festgelegten Variablen verwendet werden.
- Beim Zurücksetzen des Programms wird die Variable auf 0 zurückgesetzt.
- Der DEFINT-Befehl ist ein zusätzlicher MELFA-BASIC III-Befehl.

**Programmbeispiel (SLIM-Befehle)**

```
10 DEFINT WORK      Deklariert WORK als Namen einer arithmetischen Variablen  
20 WORK = 100       Setzt den Wert der arithmetischen Variablen WORK auf 100
```

### 8.3.4 DEFIO (Define I/O)

#### Funktion: Ein-/Ausgänge definieren

Deklariert eine Ein- bzw. Ausgangsvariable.

#### Eingabeformat

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

<E/A-Variablenname>	Legt den Variablennamen fest.
<Variablentyp>	Legt fest, ob die Variable vom Typ BIT, BYTE, WORD oder INTEGER ist.
<E/A-Bitnummer>	Legt die Nummer des Eingangs-/Ausgangsbits fest.
<Maskeninformation>	Legt fest, ob ein bestimmtes Signal zugelassen wird.

#### Erläuterung

- Der Variablenname darf aus maximal 8 Zeichen bestehen. Siehe auch Abschnitt 7.1.5 „Zeichentypen“ für erlaubte Zeichen.
- Ist eine Maskeninformation angegeben, wird nur ein bestimmtes Signal zugelassen.

#### Beispiel ▾

In Zeile 20 wird ein Maskierungsprozeß mit dem hexadezimalen Wert 0F ausgeführt, so daß die Bits Nummer 107 bis 110 zugelassen und die Bits Nummer 111 bis 114 gesperrt werden.

```
gesperrt  zugelassen
0 0 0 0 1 1 1 1
114                107 (E/-A-Bitnummer)
```

△

#### Programmbeispiel (SLIM-Befehle)

```
10  DEFIO PORT1 = BIT,0      Deklariert die Variable Port 1 als Bit-Typ und weist
                             ihr den Wert des E/A-Bits Nummer 0 zu
20  DEFIO PORT2 = BYTE,107,&H0F Deklariert die Variable Port 2 als BYTE-Typ,
                             weist ihr den Wert des E/A-Bits Nummer 107 zu
                             und legt die Maskeninformation auf 0F fest
```

Siehe auch SET und RESET.

### 8.3.5 DEFJNT (Define Joint)

**Funktion: Gelenk definieren**

Definiert eine Gelenkvariable.

**Eingabeformat**

```
DEFJNT <Gelenkvariablenname> [, <Gelenkvariablenname>] ...
```

<Gelenkvariablenname>

Legt den Namen der Gelenkvariablen fest.  
Siehe auch Abschnitt 7.1.9 „Gelenkvariablen“.

**Erläuterung**

- Der Variablenname darf aus maximal 8 Zeichen bestehen. Siehe auch Abschnitt 7.1.5 „Zeichentypen“ für erlaubte Zeichen.
- Bei Deklaration mehrerer Variablenamen, dürfen maximal 123 Zeichen (inklusive des Befehls) in einer Zeile verwendet werden
- Für die Variablen gelten die in Abschnitt 7.1.9 festgelegten Regeln.
- Die mit DEFJNT deklarierten Variablen können wie die mit „J“ festgelegten Variablen verwendet werden.
- Der DEFJNT-Befehl ist ein zusätzlicher MELFA-BASIC III-Befehl.

**Programmbeispiel (SLIM-Befehle)**

10	DEFJNT SAFE	Deklariert SAFE als Namen einer Gelenkvariablen
20	MOV SAFE	Fährt die Position SAFE an

### 8.3.6 DEFPOS (Define Position)

**Funktion: Position definieren**

Definiert eine Positionsvariable.

**Eingabeformat**

```
DEFPOS  <Positionsvariablenname>  
        [, <Positionsvariablenname>] ...
```

<Positionsvariablenname>

Legt den Namen der Positionsvariablen fest.  
Siehe auch Abschnitt 7.1.9 „Positionsvariablen“.

**Erläuterung**

- Der Variablenname darf aus maximal 8 Zeichen bestehen. Siehe auch Abschnitt 7.1.5 „Zeichentypen“ für die zugelassenen Zeichen.
- Bei Deklaration mehrerer Variablenamen, dürfen maximal 123 Zeichen (inklusive des Befehls) in einer Zeile verwendet werden
- Für die Variablen gelten die in Abschnitt 7.1.9 festgelegten Regeln.
- Die mit DEFPOS deklarierten Variablen können wie die mit „P“ festgelegten Variablen verwendet werden.
- Der DEFPOS-Befehl ist ein zusätzlicher MELFA-BASIC III-Befehl.

**Programmbeispiel (SLIM-Befehle)**

10	DEFPOS WORKSET	Deklariert WORKSET als Namen einer Positionsvariablen
20	MOV WORKSET	Fährt die Position WORKSET an

### 8.3.7 DELAY (Delay)

**Funktion: Verzögerung einstellen**

Die Abarbeitung des Programms wird um die eingestellte Zeit verzögert.

**Eingabeformat**

DELAY     <Zeit>
------------------

&lt;Zeit&gt;

Legt die Länge der Wartezeit in Millisekunden fest.

**Erläuterung**

- Der DELAY-Befehl wird verwendet, um in Programmen Verzögerungszeiten zu erzeugen. Ebenso lassen sich Zeiten für die Ein- und Ausgänge einstellen.
- Der DELAY-Befehl hat dieselbe Funktion wie der DLY-Befehl in MELFA-BASIC III. Zu beachten sind jedoch die unterschiedlichen Einheiten (DLY-Einheit: s).

**Programmbeispiel (SLIM-Befehle)**

100 OUT IO1 = 1

Setzt Ausgangsvariable IO1 auf 1

110 DELAY 500

Wartet 0,5 s (z. B. Wartezeit für ein Peripheriegerät bis zum Beginn einer Bewegung)



### 8.3.8 DRIVE (Drive)

**Funktion: Achsenbewegung**

Direkte Steuerung der Achsen.

**Eingabeformat**

```
DRIVE    (<Achsennummer>, <Rotationswinkel oder Translationsweg>)  
         [, (<Achsennummer>, <Rotationswinkel oder  
         Translationsweg>)] ...
```

<Achsennummer>

Legt die Nummer der zu steuernden Achse fest.

<Rotationswinkel oder Translationsweg>

Legt den Rotationswinkel (Einheit: Grad) oder den Translationsweg fest (Einheit: mm).

**Erläuterung**

- Bei Verwendung einer Rotationsachse muß die Nummer der Achse und der Rotationswinkel in Klammern angegeben werden.
- Bei Verwendung einer Translationsachse muß die Nummer der Achse und der Translationsweg in Klammern angegeben werden.
- Die Reihenfolge der Achsen kann frei gewählt werden. Es dürfen nur die verwendeten Achsen angegeben werden. Siehe auch MOVEMASTER-Befehl DJ.

**Programmbeispiel (SLIM-Befehle)**

100 DRIVE (5,10DEG),(3,10DEG)      Dreht die Achsen 3 und 5 um jeweils 10 Grad

### 8.3.9 GOHOME (Go Home)

**Funktion: Nullpunkt anfahren**

Der Roboter fährt von der Augenblicksposition zum benutzerdefinierten Nullpunkt.

**Eingabeformat**

GOHOME
--------

**Erläuterung**

- Der Roboter fährt mittels Gelenk-Interpolation zum benutzerdefinierten Nullpunkt. Siehe auch MOVEMASTER-Befehl OG.

**Programmbeispiel (SLIM-Befehle)**

```
10  MOVE P,P1   Führt Position P1 mittels Gelenk-Interpolation an
:
100 *RESET      Markenname *RESET
110 GOHOME      Führt benutzerdefinierten Nullpunkt an
```

### 8.3.10 GRASP (Grasp)

**Funktion: Hand schließen**

Schließt den Greifer der Hand.

**Eingabeformat**

GRASP    [<Handvariablenname>]
--------------------------------

<Handvariablenname>

Legt den Handvariablennamen fest, der mit dem HAND-Befehl definiert wurde.

**Erläuterung**

- Fehlt die Angabe des Handvariablennamens, wird Hand 1 geschlossen.

**Programmbeispiel (SLIM-Befehle)**

```
10  HAND WHAND = 1    Weist dem Handvariablennamen WHAND die Hand 1 zu
   :
100 GRASP WHAND       Schließt die Hand 1
```

Siehe auch HAND, RELEASE und CHANGE.

### 8.3.11 HALT (Halt)

**Funktion: Programmablauf stoppen**

Stoppt den Programmablauf.

**Eingabeformat**

`HALT [ <Ausdruck> ]`

<Ausdruck>

Legt den Ausdruck oder die Zeichenkette fest, die vom Steuergerät ausgegeben wird.

**Erläuterung**

- Der Ausdruck oder die Zeichenkette wird über die RS-232-C-Schnittstelle auf der Frontseite des Steuergerätes ausgegeben.
- Der Befehl hat die gleiche Wirkung wie der END-Befehl in MELFA-BASIC III.

**Programmbeispiel (SLIM-Befehle)**

100 HALT

Stoppt den Programmablauf

:

200 HALT "ENDE DER BEARBEITUNG"

Gibt den Grund für den  
Programmstopp aus

### 8.3.12 HAND (Hand)

**Funktion: Hand zuweisen**

Deklariert die Handvariable.

**Eingabeformat**

```
HAND <Handvariablenname> = <Handnummer>
```

<Handvariablenname>

Legt den Variablennamen der Hand fest.

<Handnummer>

Handnummer 1 = 1

Handnummer 2 = 2

**Erläuterung**

- Es dürfen bis zu 8 Zeichen für den Variablennamen verwendet werden. Siehe auch Abschnitt 7.1.5 für verwendbare Zeichen.

**Programmbeispiel (SLIM-Befehle)**

```
10  HAND WHAND = 1      Weist dem Handvariablennamen WHAND die Hand 1 zu
:
100 RELEASE WHAND      Öffnet Hand 1
```

Siehe auch RELEASE, GRASP und CHANGE.

### 8.3.13 HOLD (Hold)

**Funktion: Programm unterbrechen**

Zeitweise Unterbrechung des Programmablaufes.

**Eingabeformat**

`HOLD [ <Ausdruck> ]`

<Ausdruck>

Legt den Ausdruck oder die Zeichenkette fest, die vom Steuergerät ausgegeben wird.

**Erläuterung**

- Bei einem Neustart, wird das Programm von der Anweisung an weiterverarbeitet, die hinter dem HOLD-Befehl steht.
- Der Ausdruck oder die Zeichenkette wird über die RS232C-Schnittstelle auf der Frontseite des Steuergerätes ausgegeben.
- Der Befehl hat die gleich Wirkung wie der HLT-Befehl in MELFA-BASIC III.

**Programmbeispiel (SLIM-Befehle)**

100 HOLD 4\*IO2+2\*IO1+IO0

Berechnet die E/A-Variable und gibt sie aus

200 HOLD "BEARBEITUNGSFEHLER"

Gibt den Grund für die Unterbrechung aus

### 8.3.14 IN (Input)

#### Funktion: Daten einlesen

Liest die Daten einer Schnittstellenadresse, die mit dem E/A-Variablenamen festgelegt wurde.

#### Eingabeformat

```
IN  <Arithmetische Variable> = <E/A-Variable>
```

<Arithmetische Variable>

Legt einen Variablennamen fest, der mit dem DEFINT-Befehl deklariert wurde.

<E/A-Variable>

Legt einen Variablennamen fest, der mit dem DEFIO-Befehl deklariert wurde.

#### Erläuterung

- E/A-Variablen können nicht in Vergleichsoperationen verarbeitet werden. Der Wert muß zuerst in eine arithmetische Variable übertragen werden. Die arithmetische Variable kann dann in einer Vergleichsoperation verwendet werden (z. B. in einer IF-Anweisung).

#### Programmbeispiel (SLIM-Befehle)

```
10 DEFIO LIMITSW1 = BIT,&H0FF4
```

Deklariert die E/A-Variable LIMITSW1

```
20 DEFINT ILSW1
```

Deklariert die arithmetische Variable ILSW1

```
:
```

```
100 IN ILSW1 = LIMITSW1
```

Liest den Wert der E/A-Variable LIMITSW1 von der externen Quelle und überträgt sie in die arithmetische Variable ILSW1

```
110 IF ILSW1 = 0 THEN GOTO 200
```

Bedingte Verzweigung in Abhängigkeit vom Wert der arithmetischen Variablen ILSW1

Siehe auch DEFIO.

### 8.3.15 INPUT (Input)

#### Funktion: Daten vom Eingabegerät einlesen

Überträgt die Daten vom Eingabegerät.

#### Eingabeformat

```
INPUT    <Arithmetische Variable>
         [, <Arithmetische Variable>] ...
         <Zeichenkettenvariable> [, <Zeichenkettenvariable>] ...
         <Positionsvariable> [, <Positionsvariable>] ...
         <Numerierte Positionsvariable> [, <Numerierte
         Positionsvariable>] ...
```

<Arithmetische Variable>

Legt eine arithmetische Variable fest, in die die Daten vom Eingabegerät übertragen werden.

<Zeichenkettenvariable>

Legt eine Zeichenkettenvariable fest, in die die Zeichenkette vom Eingabegerät übertragen wird (die Eingabe der Anführungszeichen ist nicht erforderlich).

<Positionsvariable>

<Numerierte Positionsvariable> Legt eine Positionsvariable fest, in die die Daten vom Eingabegerät übertragen werden. Die Variable muß im folgenden Format eingegeben werden:  
(X, Y, Z, A, B, C) (FL1, FL2)  
(Siehe auch Abschnitt 7.1.9, „Positionsvariablen“)

#### Erläuterung

- Werden mehrere Variablenamen festgelegt, dürfen maximal 127 Zeichen in einer Zeile verwendet werden.
- Die Daten werden während der Programmausführung von einer externen Quelle über die RS232C-Schnittstelle an der Frontseite des Steuergerätes in die festgelegte Variable übertragen.
- Wird die INPUT-Anweisung ausgeführt, wartet das Programm auf eine Eingabe vom Bediengerät. Sind die Daten eingegeben, fährt das Programm mit der nächsten Zeile fort.
- Stimmt die Zahl der eingegebenen Daten nicht mit der Zahl der Variablen überein, erfolgt nach Ausführung der INPUT-Anweisung eine Eingabeaufforderung.
- Bei Angabe mehrerer Variablen, muß ein Komma zwischen den einzelnen Variablen verwendet werden.

#### Programmbeispiel (SLIM-Befehle)

10 DEFPOS WORKSET	Deklariert WORKSET als Positionsvariable
:	
100 INPUT WORKSET	Überträgt die Positionsdaten vom Eingabegerät in WORKSET
110 MOVE P,WORKSET	Führt Position WORKSET an



### 8.3.16 IOBLOCK (I/O Block)

#### Funktion: E/A-Block

Während der Roboter einen Bewegungsbefehl ausführt, wird gleichzeitig der Block mit E/A-Befehlen, der zwischen den Anweisungen BLOCK ON und BLOCK OFF steht, abgearbeitet.

#### Eingabeformat

```
IOBLOCK ON
.
.
IOBLOCK OFF
```

#### Erläuterung

- Die Roboterbewegung und die Ein- Ausgabeprozesse werden parallel ausgeführt.
- Innerhalb des Blocks IOBLOCK ON/OFF werden bei jeder MOVE-Anweisung die E/A-Prozesse parallel ausgeführt (siehe auch Programmbeispiel).
- Programmverschachtelungen haben keinen Einfluß auf die IOBLOCK-Anweisung.

#### Programmbeispiel (SLIM-Befehle)

100	IOBLOCK ON	Schaltet die parallele Ausführung ein
110	MOVE P,P001	Gibt während der Roboterbewegung zur Position P001 eine 1 an IO1 und IO2 aus
120	SET IO1	
130	SET IO2	
140	MOVE P,P002	Gibt nach Erreichen von P001 eine 0 an IO2 aus, während sich der Roboterarm zur Position P002 bewegt
150	RESET IO2	
160	*****	(Parallele Ausführung ist in dieser Zeile beendet)
170	SET IO3	Gibt nach Erreichen von P002 eine 1 an IO3 aus
180	IOBLOCK OFF	Schaltet die parallele Ausführung aus

### 8.3.17 JSPEED (J Speed)

**Funktion: Achsengeschwindigkeit einstellen**

Legt die Achsengeschwindigkeit fest.

**Eingabeformat**

JSPEED    <Übersteuerungswert>
--------------------------------

<Übersteuerungswert>

$1 \leq \text{Übersteuerungswert} \leq 100$  (Einheit: %)

**Erläuterung**

- Legt den prozentualen Anteil der maximalen Geschwindigkeit fest, wenn der Roboter eine mit der DRIVE-Anweisung festgelegte Achsenbewegung ausführt.
- Der Übersteuerungswert bleibt so lange aktiv, bis ein neuer Wert festgelegt wird.
- Der Übersteuerungswert darf außerhalb des angegebenen Wertebereichs liegen. Wird jedoch der Wertebereich des Roboters überschritten, erfolgt eine Fehlermeldung.
- Siehe auch MELFA-BASIC III-Befehl JOVRD.

**Programmbeispiel (SLIM-Befehle)**

10 JSPEED 100                    Legt die Achsengeschwindigkeit auf 100% fest

### 8.3.18 MOVE (Move)

#### Funktion: Position anfahren

Bewegt die Handspitze zu einer festgelegten Position.

#### Eingabeformat

```
MOVE <Interpolationsart> [@ [<Positioniergenauigkeit>]]
    <Position>
    [, [@ [<Positioniergenauigkeit>]] <Position> ...
    [, <Bewegungsoption>] ...
    [, <Übergangsoption>] ...
```

<Interpolationsart>	L: Linear-Interpolation C: Kreis-Interpolation P: Gelenk-Interpolation (PTP-Interpolation, PTP = Point To Point)
<Positioniergenauigkeit>	1 * Positioniergenauigkeit (Klasse) * 10000
<Position>	Für kartesische Koordinaten ist die Einheit mm. Für Gelenkkoordinaten ist die Einheit %.
<Bewegungsoption>	HAND = 1: geöffnet HAND = 0: geschlossen Der Handzustand gilt für die zuletzt ausgewählte Hand. TIME = Einheit: s Die Bewegungsoptionen können folgendermaßen abgekürzt werden: SPEED: S, HAND: H, TIME: T Bei Anforderungen vom Steuergerät, werden die Abkürzungen angezeigt.
<Übergangsoption>	Legt den Übergang zu einem nachfolgenden MOVE-Befehl fest. CONT = kontinuierlich

#### Erläuterung

- Positioniergenauigkeit:  
Die Positioniergenauigkeit stellt sicher, daß die Handspitze die Position mit der eingestellten Genauigkeit anfährt.
- Je größer der Wert ist, der dem @-Zeichen folgt, desto größer ist die Genauigkeit. Fehlt diese Angabe, wird die größte Positioniergenauigkeit festgelegt. Fehlt nur die Angabe des Wertes hinter dem @-Zeichen, wird die größte Positioniergenauigkeit eingestellt. Es kann keine Variable verwendet werden.
- Die Positioniergenauigkeit und die Einteilung in Klassen ist in Tabelle 8-13 gezeigt. (Der Standardwert ist Klasse 1. Für weitere Informationen über Positionierimpulse siehe auch MOVEMASTER-Befehl PW.)

Positioniergenauigkeit	Positionierimpulse
Klasse 1	10 000 (Standardwert)
~	~
Klasse 5 000	5 000
~	~
Klasse 10 000	1

**Tab. 8-13:** Positioniergenauigkeit

- **Bewegungsoption SPEED (Verfahrgeschwindigkeit):**  
Dieser Befehl wird verwendet, um die Verfahrgeschwindigkeit zu verändern. Wird er in einer MOVE-Anweisung verwendet, ist er nur in dieser MOVE-Anweisung gültig. Die mit dem SPEED-Befehl eingestellte Verfahrgeschwindigkeit ist dann unwirksam. Innerhalb einer MOVE-Anweisung wird der SPEED-Befehl zum Beispiel verwendet, wenn eine Verfahrbewegung zur Werkstückaufnahme langsam ausgeführt werden soll.
- **Bewegungsoption HAND (Handoperation):**  
Dieser Befehl wird verwendet, wenn während einer Verfahrbewegung eine Hand geöffnet oder geschlossen werden soll.  
Eine Hand kann dann zum Beispiel schon während der Verfahrbewegung geöffnet werden, um ein Werkstück aufzunehmen.
- **Bewegungsoption TIME (Verfahrzeit):**  
Der TIME-Befehl legt die Zeit bis zum Erreichen einer Position fest. Ist die eingestellte Zeit so kurz, daß die maximale Geschwindigkeit des Roboters überschritten werden würde, wird die Zeit ignoriert, und der Roboter fährt die Position mit der maximalen Geschwindigkeit an.
- **Übergangsoption CONT:**  
Dieser Befehl legt den Übergang von der Ausführung eines MOVE-Befehls zum nächsten fest. Wird die Übergangsoption CONT angegeben, so wird der nachfolgende MOVE-Befehl ausgeführt, ohne daß der Roboter an der Zielposition des vorherigen MOVE-Befehls anhält.

**Programmbeispiel (SLIM-Befehle)**

10	MOVE L,*+(0,0,45.6)	Fährt die Position mittels Linear-Interpolation an, die 45,6 mm in Z-Richtung von der aktuellen Position entfernt ist
20	MOVE L,P1,P2,P3	Fährt mittels Linear-Interpolation von der aktuellen Position zur Position P1 und über P2 nach P3
30	MOVE C,P1,P2	Fährt über den Kreisbogen, der durch die aktuelle Position, P1 und P2 festgelegt ist von der aktuellen Position nach P2
40	MOVE L,@5P001	Die Positioniergenauigkeit an der Position P001 beträgt Klasse 5
50	MOVE L,P001,SPEED = 100	Fährt die Position P001 mittels Linear-Interpolation und einer Geschwindigkeit von 100 mm/s an
(50	MOVE L,P001,S = 100)	
60	MOVE L,P001,HAND = 0	Öffnet die Hand, während die Position P001 mittels Linear-Interpolation angefahren wird
70	MOVE L,P001,TIME = 3	Fährt die Position P001 mittels Linear-Interpolation in 3 s an
100	MOVE C,P1,P2,CONT	Durch die Übergangsoption CONT wird die
110	MOVE C,P3,P4,CONT	Kreis-Interpolation nicht unterbrochen

### 8.3.19 OUT (Output)

#### Funktion: Daten ausgeben

Gibt Daten an die Schnittstellenadresse aus, die durch den E/A-Variablenamen festgelegt ist.

#### Eingabeformat

```
OUT <E/A-Variable> = <Arithmetische Variable>
```

<E/A-Variable>

Legt den mit dem DEFIO-Befehl deklarierten Variablenamen fest.

<Arithmetische Variable>

Siehe Abschnitt 7.2 „Ausdrücke und Variablen“.

#### Erläuterung

- Die Syntax ist dieselbe wie beim IN-Befehl.

#### Programmbeispiel (SLIM-Befehle)

10 DEFIO OPORT = BYTE	Deklariere OPORT als E/A-Variable vom Typ BYTE
:	
100 OUT OPORT = 1	Gibt eine 1 an die E/A-Variable OPORT aus
110 OUT OPORT = WORK+10	Gibt die Summe von 10 und der arithmetischen Variablen WORK an die E/A-Variable OPORT aus

Siehe auch DEFIO.

### 8.3.20 PRINT (Print)

#### Funktion: Daten ausgeben

Gibt Daten an das Bediengerät aus. Komma und Semikolon werden als Trennung zwischen den einzelnen Ausdrücken verwendet.

#### Eingabeformat

```
PRINT  <Ausdruck> [ , <Ausdruck> ] ...
        <Ausdruck> [ ; <Ausdruck> ] ...
```

<Ausdruck>

Legt einen arithmetischen Ausdruck, eine Zeichenkette oder Variable fest.

Bei Trennung der Ausdrücke durch ein Komma (,) werden die Daten mit Zwischenraum ausgegeben.

Bei Trennung der Ausdrücke durch ein Semikolon (;) werden die Daten ohne Zwischenraum ausgegeben.

#### Erläuterung

- Mit dem PRINT-Befehl werden festgelegte Daten über die RS232C-Schnittstelle ausgegeben, während das Programm abgearbeitet wird.
- Bei arithmetischen Ausdrücken wird das Ergebnis ausgegeben.
- Bei Variablennamen wird der Wert ausgegeben.
- Eine Zeichenkette muß in Anführungszeichen ("" ) angegeben werden (siehe auch Beispielprogramm).
- Jeder Print-Befehl wird mit einem „Carriage Return“ abgeschlossen.

#### Programmbeispiel (SLIM-Befehle)

100 PRINT 2*IO1+IO0	Gibt das Ergebnis der nebenstehenden Operation aus
200 PRINT "WORK=";WK	Gibt nach der Zeichenkette WORK den Wert der Variablen WK aus
300 PRINT P001	Gibt die Daten der Positionsvariablen P001 aus

Die Ausgabe sieht folgendermaßen aus:

2	Ergebnis aus Zeile 100 (E/A-Variable)
WORK=10	Ergebnis aus Zeile 200 (Variable WK)
(100,100,50,0,0,0)(0,0)	Ergebnis aus Zeile 300 (Variable P001)

### 8.3.21 RELEASE (Release)

**Funktion: Handgreifer öffnen**

Öffnet den Handgreifer.

**Eingabeformat**

```
RELEASE [<Handvariablenname>]
```

<Handvariablenname>

Legt den mittels HAND-Befehl definierten Handvariablennamen fest.

**Erläuterung**

- Bei fehlendem Handvariablennamen, wird die Hand Nummer 1 geöffnet.

**Programmbeispiel (SLIM-Befehle)**

```
10  HAND WHAND = 1    Weist dem Handvariablennamen WHAND die Hand Nummer 1  
                        zu  
   :  
100 RELEASE WHAND    Öffnet die Hand Nummer 1
```

Siehe auch HAND, GRASP und CHANGE.



### 8.3.22 RESET (Reset)

#### Funktion: E/A-Variable zurücksetzen

Setzt den E/A-Variablen-Ausgang vom Typ BIT auf 0.

#### Eingabeformat

RESET     <E/A-Variable>
--------------------------

<E/A-Variable>

Legt den mit dem DEFIO-Befehl deklarierten Variablennamen fest.  
Wird ALL angegeben, werden alle E/A-Variablen vom Typ BIT auf 0 gesetzt.

#### Erläuterung

- Die Syntax ist dieselbe wie beim SET-Befehl.

#### Programmbeispiel (SLIM-Befehle)

10	DEFIO IO1 = BIT,1	Deklariere die E/A-Variable IO1 als Eingangs-/Ausgangsbit Nummer 1 vom Typ BIT
20	DEFIO IO2 = BIT,2	Deklariere die E/A-Variable IO2 als Eingangs-/Ausgangsbit Nummer 2 vom Typ BIT
	:	
100	RESET IO1	Setze E/A-Variable IO1 auf 0
110	RESET ALL	Setze die E/A-Variablen IO1 und IO2 gleichzeitig auf 0

Siehe auch DEFIO und SET.

### 8.3.23 SET (Set)

#### Funktion: E/A-Variable setzen

Setzt den E/A-Variablen-Ausgang vom Typ BIT auf 1.

#### Eingabeformat

```
SET <E/A-Variable> [, <Ausgangsimpulsdauer>]
```

<E/A-Variable>

Legt den mit dem DEFIO-Befehl deklarierten Variablenamen fest.

Wird ALL angegeben, werden alle E/A-Variablen vom Typ BIT auf 1 gesetzt.

<Ausgangsimpulsdauer>

Legt die Dauer des Ausgangsimpulses fest. (Einheit: ms)

#### Erläuterung

- Der SET-Befehl wird zur Ausgabe eines Bits, z. B. einem Verriegelungsbit, verwendet.
- Die Ausgangsimpulsdauer legt die Zeit für den gesetzten Zustand des Bits fest. Das Bit wird nur während der festgelegten Impulsdauer ausgegeben.
- Um alle mit DEFIO deklarierten E/A-Variablen auf 1 zu setzen, muß anstelle der Variablen ALL eingegeben werden. Dies ist zum Beispiel bei der Initialisierung des Systems der Fall. Die Angabe All kann nur in Verbindung mit den E/A-Bit-Typen verwendet werden.

#### Programmbeispiel (SLIM-Befehle)

10 DEFIO IO1 = BIT,1	Deklariere die E/A-Variable IO1 als Eingangs-/Ausgangsbit Nummer 1 vom Typ BIT
20 DEFIO IO2 = BIT,2	Deklariere die E/A-Variable IO2 als Eingangs-/Ausgangsbit Nummer 2 vom Typ BIT
:	
100 SET IO1	Setze die E/A-Variablen auf 1
110 SET IO2,10	Gibt einen Ausgangsimpuls von 10 ms Dauer aus
120 SET ALL	Setze die mit DEFIO delarierten E/A-Variablen IO1 und IO2 gleichzeitig auf 1

Siehe auch DEFIO und RESET.

### 8.3.24 SPEED (Speed)

#### Funktion: Verfahrgeschwindigkeit einstellen

Legt die Verfahrgeschwindigkeit für die Handspitze des Roboters fest.

#### Eingabeformat

SPEED     <Verfahrgeschwindigkeit>
------------------------------------

<Verfahrgeschwindigkeit>

$1 \leq \text{Verfahrgeschwindigkeit} \leq 650 \text{ [mm/s]}$

#### Erläuterung

- Die Syntax ist dieselbe wie beim SPD-Befehl in MELFA-BASIC III.
- Der festgelegte Wert der Verfahrgeschwindigkeit bleibt so lange gültig, bis ein neuer Wert gesetzt wird.
- Der festgelegte Wert darf außerhalb des Bereiches für die Verfahrgeschwindigkeit liegen. Wird jedoch der Geschwindigkeitsbereich des Roboters überschritten, erfolgt eine Fehlermeldung.

#### Programmbeispiel (SLIM-Befehle)

10	SPEED 100	Legt die Verfahrgeschwindigkeit auf 100 mm/s fest
20	MOVE P,P1	Fährt die Position P1 mit der festgelegten Geschwindigkeit an
30	SPEED M_NSPD	Legt die Verfahrgeschwindigkeit auf den Standardwert M_NSPD fest Mit dem Standardwert des Systems (M_NSPD) wird die maximal mögliche Geschwindigkeit des Roboters festgelegt

### 8.3.25 WAIT (Wait)

#### Funktion: Programmunterbrechung

Überwacht den E/A-Port, der mit einer E/A-Variablen verbunden ist. Der Programmablauf wird während der Sperrzeit unterbrochen und die Vergleichsbedingung für den Port überprüft. Nach Ablauf der Sperrzeit, oder bei Erfüllung der Vergleichsbedingung, wird die Programmarbeitung fortgesetzt.

#### Eingabeformat

```
WAIT <Vergleichsbedingung> [ , <Zeitsperre>]
```

<Zeitsperre>

Einheit: ms

#### Erläuterung

- Der WAIT-Befehl bewirkt eine Programmunterbrechung, bis die festgelegte Vergleichsbedingung erfüllt worden ist. Die Zeitsperre wird gesetzt, um eine unbegrenzte Programmunterbrechung zu vermeiden.
- Wenn eine Zeitsperre festgelegt wurde, lässt sich nicht unterscheiden, ob die Programmunterbrechung beendet wurde, weil die Vergleichsbedingung erfüllt, oder, weil die Zeitsperre abgelaufen war. In einem solchen Fall, kann das nachfolgende Programm 2 verwendet werden.

#### Programmbeispiele (SLIM-Befehle)

##### Beispiel ▾

```
10  DEFIO PORT3 = BYTE,1    Deklariert die Variable Port 3 als BYTE-Type
                             und weist ihr den Wert des E/A-Bits Nummer 1 zu
50  WAIT PORT3 = 2          Wartet, bis Port 3 den Wert 2 annimmt
```

△

##### Beispiel ▾

```
100 WAIT PORT3 = 2,60      Wartet 60 ms, bis Port 3 den
                             Wert 2 annimmt
110 IN SW2 = PORT3          Überträgt den Wert von Port 3 in
                             die arithmetische Variable SW2
120 IF SW2 = 2 THEN *ZOKOO ELSE *TIMEOUT Geht zur nächsten Zeile, falls der Wert
                             der arithmetischen Variablen SW2
                             gleich 2 ist, und springt zur Zeile 300,
                             falls er ungleich 2 ist
130 *ZOKOO                 Setzt die Programmarbeitung fort
    :
300 *TIMEOUT               Bricht das Programm ab
310 HOLD
```

△

# A Anhang

## A.1 Übersicht der Befehle

### A.1.1 MOVEMASTER

#### Befehle für Positionierungen und Verfahrbewegungen

Nr.	Befehle	Eingabeformat	Funktion	programmierbar	Bemerkungen	Seite
1	Stellungsdaten ändern	5 Achsen: CF a [, [R/L] [, [A/B]]]	Es werden die Stellungen des Roboters an der festgelegten Position geändert.	●	5 Achsen: $1 \leq a \leq 999$ R: rechts, L: links A: oben, B: unten	5-13
		6 Achsen: CF a [, [R/L] [, [A/B] [, [N/F]]]			6 Achsen: $1 \leq a \leq 999$ R: rechts, L: links A: oben, B: unten N: nicht kippen, F: kippen	
2	relative Gelenk-Interpolation	DJ a, b	Das Robotergelenk (a) wird um den festgelegten Betrag (b) bewegt.	●	5 Achsen: $1 \leq a \leq 5$ 6 Achsen: $1 \leq a \leq 6$ Gelenk-Interpolation	5-25
3	Positionsnummer dekrementieren	DP	Der Roboter fährt von der aktuellen Position zur vorherigen Position.	●	Gelenk-Interpolation	5-27
4	relative geradlinige Bewegung	DS [x], [y], [z]	Der Roboter fährt von der aktuellen Position um den festgelegten Betrag.	●	Linear-Interpolation	5-30
5	relative Bewegung	DW [x], [y], [z]	Der Roboter fährt von der aktuellen Position um den festgelegten Betrag.	●	Gelenk-Interpolation	5-32
6	aktuelle Position speichern	HE a	Die Robotersteuerung speichert die aktuellen Koordinaten als Position ab.	●	$1 \leq a \leq 999$ (Im Falle von 0 wird die aktuelle Position als benutzerdefinierter Nullpunkt abgespeichert.)	5-48
7	Nullpunkteinstellung	HO [a]	Der Roboter speichert die aktuelle Position als benutzerdefinierten Nullpunkt ab.	●	a = 0: Einstellung über mechanische Stopper a = 1: Einstellung über Kalibrierungsvorrichtung a = 2: Einstellung über benutzerdefinierten Nullpunkt	5-50
8	Positionsnummer inkrementieren	IP	Der Roboter fährt zur nächst folgenden Position mit der Positionsnummer (a).	●	Gelenk-Interpolation	5-55
9	Drehrichtungswechsel	JRC <[+]/-1>	Addiert zur aktuellen Position der R-Achse +/- 360 und überschreibt die Position mit dem neuen Wert.	●	+1 Addition von 360 -1 Subtraktion von 360	5-56
10	relative Koordinatenaddition	MA a1, a2 [, [O/C]	Die Koordinaten von Position (a1) und (a2) werden addiert.	●	$1 \leq a1, a2 \leq 999$ Gelenk-Interpolation O: Hand offen C: Hand geschlossen	5-61
11	kontinuierliche Bewegung	MC a1, a2 [, [O/C]	Der Roboter verfährt mittels Linear-Interpolation von Position (a1) zur Position (a2).	●	$1 \leq a1, a2 \leq 999$ $ a1 - a2  \leq 99$ O: Hand offen C: Hand geschlossen	5-63

**Tab. A-1:** Übersicht der Befehle für Positionierungen und Verfahrbewegungen (1)

Nr.	Befehle	Eingabeformat	Funktion	programmierbar	Bemerkungen	Seite
12	relative Mehrfach-Gelenkbewegung	5 Achsen: MJ [w], [s], [e], [p], [r]	Der Roboter dreht die Gelenke um den festgelegten Betrag in bezug zur aktuellen Position.	●	5 Achsen: MJ [w], [s], [e], [p], [r] Gelenk-Interpolation	5-65
		6 Achsen: MJ [w], [s], [e], [t], [p], [r]			6 Achsen: MJ [w], [s], [e], [t], [p], [r] Gelenk-Interpolation	
13	Zusatzachse bewegen	ML [a],[b]	Bewegt die erste Zusatzachse um den Wegbetrag (a) und die zweite Zusatzachse um den Wegbetrag (b).	●	a: Wegbetrag der ersten Zusatzachse [mm oder deg.] b: Wegbetrag der zweiten Zusatzachse [mm oder deg.]	5-67
14	Position anfahren	MO a [, [O/C]]	Die Handspitze des Roboters wird zur festgelegten Position bewegt.	●	1 ≤ a ≤ 999 Gelenk-Interpolation O: Hand offen C: Hand geschlossen	5-68
15	Koordinatenposition anfahren	5 Achsen: MP [x], [y], [z], [a], [b], [, [R/L] [, [A/B]]]	Der Roboter fährt zur festgelegten Position.	●	5 Achsen: R: rechts, L: links A: oben, B: unten	5-69
		6 Achsen: MP [x], [y], [z], [a], [b], [c], [, [R/L] [, [A/B] [, [N/F]]]			6 Achsen: R: rechts, L: links A: oben, B: unten N: nicht kippen, F: kippen	
16	Parameter für Teaching-Playback-Methode festlegen	5 Achsen: MPB [d], [e], [f], [g], [h], [i], [, [j]], [x], [y], [z], [a], [b] [, [R/L] [, [A/B]]] [, [O/C]]	Der Roboter fährt zur festgelegten Position mit folgenden voreingestellten Parametern: Interpolation, Geschwindigkeit, Timer und Ein-/Ausgangssignale.	●	5 Achsen: 0 ≤ b ≤ 32 767 0 ≤ e ≤ 2455 0 ≤ f, g, h, i ≤ &7FFF j = 0 (Gelenk), 1 = (linear), 2 = (Kreis) R: rechts, L: links A: oben, B: unten O: Hand offen C: Hand geschlossen	5-71
		6 Achsen: MPB [d], [e], [f], [g], [h], [i], [, [j]], [x], [y], [z], [a], [b], [c] [, [R/L] [, [A/B] [, [N/F]]] [, [O/C]]			6 Achsen: 0 ≤ b ≤ 32 767 0 ≤ e ≤ 2455 0 ≤ f, g, h, i ≤ &7FFF j = 0 (Gelenk), 1 = (linear), 2 = (Kreis) R: rechts, L: links A: oben, B: unten N: nicht kippen, F: kippen O: Hand offen C: Hand geschlossen	
17	Interpolationsart für Teaching-Playback-Methode festlegen	5 Achsen: MPC [d] [x], [y], [z] [a], [b] [, [R/L] [, [A/B]]] [, [O/C]]	Der Roboter fährt mittels der festgelegten Interpolationsmethode zur definierten Position.	●	5 Achsen: D = 0 (Gelenk), 1 = (linear), 2 = (Kreis) R: rechts, L: links A: oben, B: unten O: Hand offen C: Hand geschlossen	5-74
		6 Achsen: MPC [d], [x], [y], [z], [a], [b], [c] [, [R/L] [, [A/B] [, [N/F]]] [, [O/C]]			6 Achsen: d = 0 (Gelenk), 1 = (linear), 2 = (Kreis) R: rechts, L: links A: oben, B: unten N: nicht kippen, F: kippen O: Hand offen C: Hand geschlossen	

Tab. A-1: Übersicht der Befehle für Positionierungen und Verfahrbewegungen (2)

Nr.	Befehle	Eingabeformat	Funktion	programmierbar	Bemerkungen	Seite
18	Zwischenposition bei Kreis-Interpolation	MR a1, a2, a3 [, [O/C]]	Der Roboter fährt mittels Kreis-Interpolation den durch die Positionen a1, a2 und a3 definierten Bogen ab.	●	$1 \leq a1, a2, a3 \leq 999$ O: Hand offen C: Hand geschlossen	5-76
19	Kreis-Interpolation	MRA a [, [O/C]]	Die Handspitze des Roboters wird mittels Kreis-Interpolation zur festgelegten Position bewegt.	●	$1 \leq a \leq 999$ O: Hand offen C: Hand geschlossen	5-78
20	geradlinige Bewegung	MS a [, [O/C]]	Die Handspitze des Roboters wird mittels Linear-Interpolation zur festgelegten Position bewegt.	●	$1 \leq a \leq 999$ O: Hand offen C: Hand geschlossen	5-80
21	Werkzeugbewegung mit Gelenk-Interpolation	MT a [, [O/C]]	Bewegt die Handspitze zur einer Position, die um den festgelegten Betrag in Werkzeuglängsrichtung verschoben von der festgelegten Position liegt.	●	$1 \leq a \leq 999$ Gelenk-Interpolation O: Hand offen C: Hand geschlossen	5-82
22	geradlinige Werkzeugbewegung	MTS a [, [O/C]]	Bewegt die Handspitze zur einer Position, die um den festgelegten Betrag in Werkzeuglängsrichtung entfernt von der festgelegten Position liegt.	●	$1 \leq a \leq 999$ Linear-Interpolation O: Hand offen C: Hand geschlossen	5-84
23	Nullpunkt einstellen	NT	Der Roboter fährt zum benutzerdefinierten Nullpunkt.	●		5-90
24	Nullpunkt anfahren	OG	Der Roboter fährt zum benutzerdefinierten Nullpunkt.	●	Gelenk-Interpolation	5-96
25	Übersteuerung	OVR a	Es wird die Geschwindigkeitsübersteuerung eingestellt.	●	$1 \leq a \leq 200$	5-99
26	Gitterpunkte für Palette definieren	PA i, j, k	Definiert die Anzahl der Gitterpunkte in Spalten- und Zeilenrichtung für die festgelegte Palette.	●	$1 \leq i \leq 9$ $1 \leq j, k \leq 32\ 767$	5-100
27	Position löschen	PC a1 [, [a2]]	Löscht die Positionsdaten der festgelegten Positionen zwischen a1 und a2.		$a1 \leq a2$ $1 \leq a1, a2 \leq 999$	5-101
28	Position definieren	5 Achsen: PD a, [x], [y], [z], [a], [b], [, [R/L] [, [A/B] [, [O/C]]]]	Die Position mit der Nummer a wird über die festgelegten Koordinatenwerte definiert.	●	5 Achsen: $1 \leq a \leq 999$ R: rechts, L: links A: oben, B: unten O: Hand offen C: Hand geschlossen	5-102
		6 Achsen: PD a, [x], [y], [z], [a], [b], [c] [, [R/L] [, [A/B] [, [N/F] [, [O/C]]]]			6 Achsen: $1 \leq a \leq 999$ R: rechts, L: links A: oben, B: unten N: nicht kippen, F: kippen O: Hand offen C: Hand geschlossen	
29	Koordinaten für Palette berechnen	PT a	Es werden die Koordinaten eines Gitterpunktes der festgelegten Palette berechnet. Anschließend werden die berechneten Koordinaten der festgelegten Position zugewiesen.	●	$1 \leq a \leq 999$	5-111
30	Impuls-Warteschleife	PW a	Positioniert über die Servomotoren alle Gelenke so lange, bis die festgelegte Positioniertoleranz erreicht ist.	●	$1 \leq a \leq 10\ 000$	5-113
31	Position austauschen	PX a1, a2	Ersetzt die Koordinaten der Position a1 durch die Koordinaten der Position a2.	●	$1 \leq a1, a2 \leq 999$	5-114

Tab. A-1: Übersicht der Befehle für Positionierungen und Verfahrbewegungen (3)

Nr.	Befehle	Eingabeformat	Funktion	programmierbar	Bemerkungen	Seite
32	absolute Geschwindigkeit definieren	SD a [, b [, c, d [, e]]]	Es wird die absolute Verfahrensgeschwindigkeit (a), die Zeitkonstante für den Bewegungsablauf (b), die absolute Beschleunigungszeit (c) oder Abbremszeit (d) und der Einstellmodus für die kontinuierliche gleichmäßige Verfahrbewegung definiert.	●	$0.1 \leq a \leq 650.0$ $1 \leq b \leq 300$ $0 \leq c, d \leq 2000$ E = 0: nicht kontinuierlich, 1: kontinuierlich	5-123
33	Betriebsgeschwindigkeit einstellen	SP a [, H/L [, b]]	Es wird die relative Betriebsgeschwindigkeit (a), die relative Beschleunigungs-/Abbremszeit (H/L) und der Einstellmodus für die kontinuierliche gleichmäßige Verfahrbewegung (b) definiert.	●	$1 \leq a \leq 30$ H: schnelle Bewegung, L: langsame Bewegung b = 0: nicht kontinuierlich, 1: kontinuierlich	5-128
34	Timer (Zeitglied)	TI a	Der Roboter wird für die festgelegte Zeit (a) gestoppt.	●	$0 \leq a \leq 32\,767$ (in Einheiten von 0,1 s)	5-136
35	Werkzeuglänge einstellen	TL [a]	Legt den Abstand zwischen der Befestigungsoberfläche für die Hand und der Handspitze fest.	●		5-137

**Tab. A-1:** Übersicht der Befehle für Positionierungen und Verfahrbewegungen (4)

#### Befehle für die Programmsteuerung

Nr.	Befehle	Eingabeformat	Funktion	programmierbar	Bemerkungen	Seite
36	Zähler laden	CL a	Es wird der Wert des internen Registers in den festgelegten Zähler geladen.	●	$1 \leq a \leq 99$	5-16
37	Zählerwert vergleichen	CP a	Es wird der Wert des festgelegten Zählers in das interne Register geladen.	●	$1 \leq a \leq 99$	5-18
38	Interrupt-Möglichkeit sperren	DA a	Die Interrupt-Möglichkeit für die festgelegten Bits am Eingabeport wird gesperrt.	●	$0 \leq a \leq 32\,767$	5-22
39	Programmzeile löschen	DL [a1] [, [a2] [, [b1] [, b2]]]	Es werden die Programmzeilen zwischen a1 und a2 oder die Programmschritte zwischen b1 und b2 gelöscht.		$a1 \leq a2, b1 \leq b2$ $1 \leq a1, a2$ $b1, b2 \leq 9999$	5-26
40	Interrupt-Eingang festlegen	EA [+/-] a, b [, [c]]	In Abhängigkeit von den Signalzuständen der festgelegten Eingangsbits wird das Programm unterbrochen und ein Programmsprung ausgeführt.	●	$0 \leq a \leq 32\,767$ +: Bit ein, -: Bit aus $1 \leq b \leq 99$ C = 0 oder 1, 0: Sprung, 1: Unterprogrammaufruf	5-34
41	Programmende	ED	Dieser Befehl kennzeichnet das Programmende.	●		5-36

**Tab. A-2:** Übersicht der Befehle für die Programmsteuerung (1)



Nr.	Befehle	Eingabeformat	Funktion	programmierbar	Bemerkungen	Seite
42	Datenwerte auf Gleichheit überprüfen	EQ a (oder &b), c	Es wird ein Programmsprung ausgeführt, wenn der Wert des internen Registers mit dem festgelegten Vergleichswert übereinstimmt.	●	$-32\,768 \leq a \leq 32\,767$ $\&8000 \leq b \leq \&7FFF$ $1 \leq c \leq 9999$	5-37
43	Sprung zu einem Unterprogramm	GS [a] [, [b]]	Es wird ein Sprung zu einem Unterprogramm ausgeführt, welches mit der festgelegten Programmzeile beginnt.	●	$1 \leq a \leq 9999$	5-45
44	Sprung zu einer Programmzeile	GT a	Es wird ein Sprung zur einer Programmzeile ausgeführt.	●	$1 \leq a \leq 999$	5-47
45	Programmablauf stoppen	HLT	Es werden die Roboterbewegung und der Programmablauf gestoppt.	●		5-49
46	Wertevergleich: >	LG a (oder &b) c	Bewirkt einen Programmsprung, wenn der externe Eingabewert oder der Zählerwert größer als der Vergleichswert ist.	●	$-32\,768 \leq a \leq 32\,767$ $\&8000 \leq b \leq \&7FFF$ $1 \leq c \leq 9999$	5-57
47	Wertevergleich: ≠	NE a (oder &b) c	Bewirkt einen Programmsprung, wenn der externe Eingabewert oder der Zählerwert mit dem Vergleichswert nicht übereinstimmt.	●	$-32\,768 \leq a \leq 32\,767$ $\&8000 \leq b \leq \&7FFF$ $1 \leq c \leq 9999$	5-88
48	Programm- und Positionsspeicher löschen	NW	Es werden das aktuelle Programm und die Positionsdaten gelöscht.			5-91
49	Programmschleife beenden	NX	Es wird das Ende einer Programmschleife festgelegt, welche über den RC-Befehl aufgerufen wurde.	●		5-92
50	Programmschleife	RC a	Es wird der mit dem NX-Befehl festgelegte Programmabschnitt sofort wiederholt, bis die festgelegte Anzahl der Wiederholungen erreicht ist.	●	$1 \leq a \leq 32\,767$	5-116
51	Programm starten	RN [a] [, a2 [, b]]	Es wird der mit a1 und a2 festgelegte Programmabschnitt abgearbeitet.	●	$1 \leq a1, a2 \leq 999$ b = Programmname	5-117
52	Rücksprung zum Hauptprogramm	RT [a]	Das mit dem GS-Befehl aufgerufene Unterprogramm wird abgeschlossen und anschließend ein Sprung zum Hauptprogramm ausgeführt.	●	$1 \leq a1 \leq 999$ a1: Sprungziel	5-120
53	Wertevergleich: <	SM a1 (oder &b) a2	Es wird ein Programmsprung ausgeführt, wenn der Eingabewert oder der Zählerwert kleiner als der Vergleichswert ist.	●	$-32\,768 \leq a2 \leq 32\,767$ $\&8000 \leq b \leq \&7FFF$ $1 \leq a2 \leq 9999$	5-126

**Tab. A-2:** Übersicht der Befehle für die Programmsteuerung (2)

**Befehle für die Handsteuerung**

Nr.	Befehle	Eingabeformat	Funktion	program- mierbar	Bemerkungen	Seite
54	Handgreifer schließen	GC [a]	Der Handgreifer wird geschlossen.	●	0: Hand 1 1: Hand 2 2: Hand 3	5-41
55	Handgreiferzustand festlegen	GF a	Es wird der Handgreiferzustand festgelegt (in Verbindung mit dem PD-Befehl).	●	a = 0: offen 1: geschlossen	5-43
56	Handgreifer öffnen	GO [a]	Der Handgreifer wird geöffnet.	●	a: Auswahl der Hand 0: Hand 1 1: Hand 2 2: Hand 3	5-44

**Tab. A-3:** Übersicht der Befehle für die Handsteuerung

**Befehle für die Ein-/Ausgabe**

Nr.	Befehle	Eingabeformat	Funktion	programmierbar	Bemerkungen	Seite
57	Eingänge einlesen	ID [a]	Es werden die Daten vom externen Eingang oder Handkontrolleingang eingelesen.	●	$0 \leq a \leq 32\,767$	5-52
58	Ausgänge ein-/ausschalten	OB [+/-] a	Es wird der Ausgabestatus des festgelegten Bit am externen Ausgang eingestellt.	●	$0 \leq a \leq 32\,767$ +: Bit ein -: Bit aus	5-93
59	Zählerwert ausgeben	OC a [, [a1] [, [a2]]]	Der Wert des festgelegten Zählers wird ohne Ausführungsbedingung über den Ausgabeport ausgegeben.	●	$1 \leq a \leq 99$ $0 \leq a1 \leq 32\,767$ $1 \leq a2 \leq 16$	5-94
60	direkte Ausgabe	OD a [, [a1] [, [a2]]]	Die festgelegten Daten werden ohne Ausführungsbedingung über den Ausgabeport ausgegeben.	●	$-32\,768 \leq a \leq 32\,767$ $\&8000 \leq b \leq \&7FFF$ $0 \leq a1 \leq 32\,767, 1 \leq a2 \leq 16$	5-95
61	Bitstatus überprüfen	TB [+/-] a, b	Es wird ein Programmsprung in Abhängigkeit vom Bitstatus des festgelegten Bit im internen Register ausgeführt.	●	$1 \leq a \leq 15$ +: Bit ein, -: Bit aus $1 \leq b \leq 9999$	5-134
62	Bitstatus direkt überprüfen	TBD [+/-] a, b	Es wird ein Programmsprung in Abhängigkeit vom Bitstatus der externen Eingabedaten ausgeführt.	●	$1 \leq a \leq 32\,767$ +: Bit ein, -: Bit aus $1 \leq b \leq 9999$	5-135

**Tab. A-4:** Übersicht der Befehle für die Ein-/Ausgabe

## Operations-, Substitutions- und Austauschbefehle

Nr.	Befehle	Eingabeformat	Funktion	programmierbar	Bemerkungen	Seite
63	Addition	ADD a	Der Wert (a) wird zum Wert des internen Registers addiert.	●	$-32\,768 \leq a \leq 32\,767$ &8000 ≤ a ≤ &7FFF @1 ≤ a ≤ @99 (Zählernummer)	5-11
64	Subtraktion	SUB a	Der Wert (a) wird vom Wert des internen Registers subtrahiert.	●	$-32\,768 \leq a \leq 32\,767$ &8000 ≤ a ≤ &7FFF @1 ≤ a ≤ @99 (Zählernummer)	5-133
65	Multiplikation	MUL a	Der Wert (a) wird mit dem Wert des internen Registers multipliziert.	●	$-32\,768 \leq a \leq 32\,767$ &8000 ≤ a ≤ &7FFF @1 ≤ a ≤ @99 (Zählernummer)	5-86
66	Division	DIV a	Der Wert des internen Registers wird durch den Wert (a) dividiert.	●	$-32\,768 \leq a \leq 32\,767$ &8000 ≤ a ≤ &7FFF @1 ≤ a ≤ @99 (Zählernummer)	5-24
67	Zählerwert inkrementieren	IC a	Es wird 1 zum Wert des festgelegten Zählers addiert.	●	1 ≤ a ≤ 99 @1 ≤ a ≤ @99 (Indirekte Angabe einer Positionsnummer)	5-51
68	Zählerwert dekrementieren	DC a	Es wird der Wert 1 vom Wert des festgelegten Zählers abgezogen.	●	1 ≤ a ≤ 99 @1 ≤ a ≤ @99 (Indirekte Angabe einer Positionsnummer)	5-23
69	UND-Verknüpfung	AN a	Es erfolgt eine UND-Verknüpfung des Wertes (a) mit dem Wert des Registers.	●	$-32\,768 \leq a \leq 32\,767$ &8000 ≤ a ≤ &7FFF @1 ≤ a ≤ @99 (Zählernummer)	5-12
70	ODER-Verknüpfung	OR a	Es erfolgt eine ODER-Verknüpfung des Wertes (a) mit dem Wert des Registers.	●	$-32\,768 \leq a \leq 32\,767$ &8000 ≤ a ≤ &7FFF @1 ≤ a ≤ @99 (Zählernummer)	5-98
71	Exklusiv-ODER-Verknüpfung	XO a	Es erfolgt eine Exklusiv-ODER-Verknüpfung des Wertes (a) mit dem Wert des Registers.	●	$-32\,768 \leq a \leq 32\,767$ &8000 ≤ a ≤ &7FFF @1 ≤ a ≤ @99 (Zählernummer)	5-142
72	Zählerwert einstellen	SC a,[b]	Es wird der festgelegte Wert (b) in einen festgelegten Zähler (a) oder eine Zeichenkette geladen.	●	1 ≤ a ≤ 99 (Zählernummer) \$1 ≤ a ≤ \$99 (Zeichenkettennummer) $-32\,768 \leq b \leq 32\,767$ (Dezimaler Vergleichswert) &8000 ≤ b ≤ &7FFF (Hexadezimaler Vergleichswert) 1 ≤ b ≤ 122 (Anzahl der Zeichen)	5-121
73	Position kopieren	PL a1,a2	Die Koordinatenwerte der Position (a1) werden mit den Koordinatenwerten der Position a2 überschrieben.	●	1 ≤ a1, a2 ≤ 999	5-104
74	Positionskoordinaten addieren	SF a1,a2	Es werden die Koordinatenwerte der Position (a2) zu den entsprechenden Koordinatenwerten der Position (a1) addiert. Anschließend wird die Position (a1) erneut abgespeichert.	●	1 ≤ a1, a2 ≤ 999	5-125
75	Position austauschen	PX a1,a2	Ersetzt die Koordinaten der Position (a1) durch die Koordinaten der Position (a2).	●	1 ≤ a1,a2 ≤ 999	5-114

Tab. A-5: Übersicht der Operations- Substitutions- und Austauschbefehle

**Kommunikationsbefehle für die RS232C-Schnittstelle**

Nr.	Befehle	Eingabeformat	Funktion	programmierbar	Bemerkungen	Seite
76	Zählerwert lesen	CR a	Der Inhalt des festgelegten Zählers wird über die RS232C-Schnittstelle gelesen.	●	$1 \leq a \leq 99$ (Zählernummer) @ $1 \leq a \leq @99$ (Indirekte Angabe einer Positionsnummer) \$ $1 \leq a \leq \$99$	5-20
77	Daten lesen	DR [a]	Der Wert des internen Registers, die Handkontrollsignale und der allgemeine Status der Ausgänge wird über die RS232C-Schnittstelle gelesen.	●	$0 \leq a \leq 32\,767$ 16-Bit-Daten	5-28
78	Fehler lesen	ER [a]	Der aktuelle Fehlerstatus und die bisher angezeigten Fehlercodes werden über die RS232C-Schnittstelle gelesen.		$1 \leq a \leq 16$ 0: keine Fehler 1: Fehlercode 01 – 19 2: Fehlercode 23 – 89	5-39
79	Programmzeile lesen	LR [a]	Die Programminhalte der festgelegten Zeilennummer werden über die RS232C-Schnittstelle gelesen.		$0 \leq a \leq 9999$	5-59
80	Parameterwerte lesen	PMR [“a”]	Die Inhalte des festgelegten Parameters werden gelesen.	●		5-105
81	Positionsdaten lesen	PR [a]	Die Positionsdaten und der Handgreiferzustand werden über die RS232C-Schnittstelle gelesen.	●	$0 \leq a \leq 999$ @ $1 \leq a \leq @99$ (Indirekte Angabe einer Positionsnummer)	5-107
82	Programminformationen lesen	QN [a]	Der Programmname oder die Programminformationen werden gelesen.	●	$0 \leq a \leq 31$ Programminformation: Anzahl der Programmschritte, Anzahl der Positionen, Anzahl der Zähler	5-115
83	Programmschritt lesen	STR [a]	Es werden die Inhalte der festgelegten Programmschritte gelesen.		$0 \leq a \leq 9999$	5-131
84	Software-Version lesen	VR	Es wird die Software-Versionsnummer des System-ROMs gelesen.	●		5-138
85	aktuelle Positionskoordinaten lesen	WH	Es werden die Koordinaten der aktuellen Position und der Handgreiferzustand gelesen.	●		5-139
86	Werkzeuglänge lesen	WT	Die aktuell eingerichtete Werkzeuglänge wird über die RS232C-Schnittstelle gelesen.	●		5-141

**Tab. A-6:** Kommunikationsbefehle für die RS232C-Schnittstelle

## Zusatzfunktionen

Nr.	Befehle	Eingabeformat	Funktion	programmierbar	Bemerkungen	Seite
87	Zählerwert oder Positionsdaten lesen	INP a, b, [, [c]]	Es werden die Zählerwerte (b) oder die Positionsdaten (a) vom Kanal (c) gelesen.	●	$1 \leq a \leq 4$ $1 \leq b \leq 99$ oder 999 C = 0 oder 1 0: Zählerwerte, 1: Positionsdaten	5-53
88	Programm auswählen	N	Es wird das festgelegte Programm ausgewählt.		$1 \leq a \leq 31$	5-87
89	Kommunikationskanäle öffnen	OPN a, b	Der Kommunikationskanal (a) mit der Schnittstelle (b) wird geöffnet.	●	$1 \leq a \leq 4$ b = 0: RS422 (Standard) 1: RS232C (Standard) 2: RS232C-1 (Option) 3: RS232C-2 (Option)	5-97
90	Parameterwerte schreiben	PMW "a", "b"	Der Parameter (a) wird mit den Inhalten (b) überschrieben.	●		5-106
91	Daten übertragen	PRN a, b oder c	Der Einstellwert für einen Zähler (a) oder die Koordinatenwerte für eine Position (b) oder eine Zeichenkette (c) werden an die RS232C-Schnittstelle übertragen.	●	$-32\,768 \leq a \leq 32\,767$ B: Koordinatenwerte 5 Achsen: X, Y, Z, A, B 6 Achsen: X, Y, Z, A, B, C $1 \leq c \leq 120$	5-109
92	Programm zurücksetzen	RS [a]	Die Fehlermeldung und das Programm werden zurückgesetzt.	●	A = 0: Fehlermeldung abbrechen 1: Zähler zurücksetzen 2: Batterie-Timer zurücksetzen 3: Programme und Positionen löschen 4: Nullpunkteinstellung zurücksetzen	5-119
93	Bemerkung	'	Es kann eine Bemerkung mit bis zu 120 Zeichen in ein Programm geschrieben werden.	●		5-143

Tab. A-7: Zusatzfunktionen

## A.1.2 MELFA-BASIC III

### Befehle für Positionierungen und Verfahrbewegungen

Nr.	Befehle	Eingabeformat	Funktion	programmierbar	Bemerkungen	Seite
1	Koordinatenposition mittels Gelenk-Interpolation anfahren	MOV a[,b][c]	Führt die Verknüpfungsbedingung (c) aus, und bewegt sich mittels Gelenk-Interpolation zu einem Punkt, der in einem Abstand (b) von der Position (a) entfernt liegt.	●	a: Zielposition (Positionskonstante oder -variable) b: Abstand c: Verknüpfungsbedingung (WTH, WTHIF)	8-45
2	Bewegung mittels 3D-Kreis-Interpolation	MVC a1,a2,a3[c]	Führt die Verknüpfungsbedingung (c) aus, und bewegt sich mittels 3D-Kreis-Interpolation auf einem Kreis, der durch den Start- und Endpunkt (a1) und die Zwischenpositionen (a2,a3) festgelegt ist.	●	a1, a2, a3: Positionskonstante oder -variable c: Verknüpfungsbedingung (WTH, WTHIF)	8-47
3	Bewegung mittels 3D-Kreis-Interpolation	MVR a1,a2,a3[c]	Führt die Verknüpfungsbedingung (c) aus, und bewegt sich mittels 3D-Kreis-Interpolation auf einem Kreis, der durch den Startpunkt (a1), die Zwischenposition (a2) und die Endposition (a3) festgelegt ist.	●	a1, a2, a3: Positionskonstante oder -variable c: Verknüpfungsbedingung (WTH, WTHIF)	8-49
4	Bewegung mittels 3D-Kreis-Interpolation	MVR2 a1,a2,a3[c]	Führt die Verknüpfungsbedingung (c) aus, und bewegt sich mittels 3D-Kreis-Interpolation vom Startpunkt (a1) zum Endpunkt (a2) auf einem Kreis, ohne den Punkt (a3) zu durchlaufen.	●	a1, a2, a3: Positionskonstante oder -variable c: Verknüpfungsbedingung (WTH, WTHIF)	8-51
5	Zielposition anfahren	MVS a[,b][c] oder MVS ,b	Führt die Verknüpfungsbedingung (C) aus, und bewegt sich mittels Linear-Interpolation zu einem Punkt, der in einem Abstand (b) von der Position (a) entfernt liegt.	●	a: Positionskonstante oder -variable b: Abstand c: Verknüpfungsbedingung (WTH, WTHIF)	8-53
6	Hand ausrichten	ALIGN	Bewegt die Hand mittels Linear-Interpolation zu der Position, die den kleinstmöglichen senkrechten Abstand zu den Werten der Stellungsachsen (A, B, C) hat.	●	Linear-Interpolation	8-8
7	Optimale Beschleunigung/Abbremsung	OADL a[,b[,c,d]]	Legt die optimale Beschleunigung/Abbremsung in Abhängigkeit der Last (b), (c) der festgelegten Hand (d) fest.	●	a: optimale Beschleunigung/Abbremsung ON/OFF 1: ON, 0: OFF b: Standardlast c: Maximallast d: Handnummer $1 \leq d \leq 3$	8-55
8	Nullpunkt einstellen	ORG [<Zusätzliche Achsennummer>]	Legt den Nullpunkt der Zusatzachse fest. Bei fehlender Angabe bewegt sich der Roboter zur Ersatzposition.	●	Gelenk-Interpolation	8-62
9	Anweisung hinzufügen	WTH b	Während einer Interpolationsbewegung wird eine unbedingte Anweisung (b) ausgeführt.	●	b: Prozesse wie Substitutionen oder Signal-Anweisungen	8-80
10	Bedingte Anweisung hinzufügen	WTHIF a,b	Während einer Interpolationsbewegung wird eine Anweisung (b) unter der Bedingung (a) ausgeführt.	●	a: Bedingung b: Prozesse wie Substitutionen oder Signal-Anweisungen	8-81
11	Übersteuerung einstellen	OVRD a	Legt die Arbeitsgeschwindigkeit (a) des Roboters fest.	●	$1 \leq a \leq 200$ [%]	8-65
12	Übersteuerung einstellen	JOVRD a	Legt die Arbeitsgeschwindigkeit (a) Für Gelenk-Interpolation fest.	●	$1 \leq a \leq 200$ [%]	8-43

**Tab. A-8:** Übersicht der Befehle für Positionierungen und Verfahrbewegungen (1)

Nr.	Befehle	Eingabeformat	Funktion	programmierbar	Bemerkungen	Seite
13	Geschwindigkeit einstellen	SPD a	Legt die Geschwindigkeit (a) für Linear- und Kreis-Interpolation fest.	●	$1 \leq a \leq 650$ [mm/s]	8-73
14	Beschleunigung einstellen	ACL a	Legt die Beschleunigung (a) während einer Bewegung fest.	●	$0,05 \leq a \leq 2,00$ [s]	8-4
15	Abbremsung einstellen	DACL a	Legt die Abbremsung (a) während einer Bewegung fest.	●	$0,05 \leq a \leq 2,00$ [s]	8-19
16	Roboterbewegung steuern	CNT a	Legt die Steuerung (a) für eine kontinuierliche und gleichmäßige Bewegung fest.	●	a: CNT-Einstellung a = 0: gesperrt a = 1: freigegeben	8-13
17	Verzögerung einstellen	DLY a	Erzeugt als einzelner Befehl eine Wartezeit (a). Als zusätzlicher Impulsausgang wird die Impulsdauer (a) festgelegt.	●	a: Zeit [s] Der Minimalwert beträgt 0,05 s	8-27
18	Feinpositionierung	FINE a	Legt den Status (a) bei Beendigung eines Interpolationsbefehls fest.	●	a: Beendigungsbedingung a = 0: Die Servopositionierung wird nicht abgewartet a = 1: Die Servopositionierung wird abgewartet.	8-29
19	aktuelle Position	HRE	Liefert die aktuelle Position in kartesischen Koordinaten.	●	Der Befehl kann nicht einzeln ausgeführt werden. Verwenden Sie ihn wie folgt: P1=HRE	8-39
20	Basis-Transformationsdaten festlegen	BASE a	Legt die Basis-Transformationsdaten (a) fest.	●	a: Positionsdaten	8-9
21	Werkzeugkonvertierungsdaten festlegen	TOOL a	Legt die Werkzeugkonvertierungsdaten fest.	●	a: Positionsdaten	8-77
22	Palette definieren	DEF PLT a,b1,b2,b3,[b4],c1,c2,d	Die Palette wird durch die Palettennummer (a), Startpunkt (b1), Endpunkt A (b2), Endpunkt B (b3), Paletteneckpunkt (b4), Anzahl der Spaltengitterpunkte A (c1), Anzahl der Zeilengitterpunkte B (c2) und die Bewegungsrichtung (d) definiert.	●	$1 \leq a \leq 8$ b1, b2, b3, b4: Positionsdaten c1, c2: numerischer Wert d = 1: zickzack d = 2: Richtung beibehalten	8-24
23	Palettenkoordinaten berechnen	PLT a,b	Berechnet die Koordinaten eines Gitterpunktes (b) der festgelegten Palette (a).	●	$1 \leq a \leq 8$ b: numerischer Wert	8-67
24	Servo ein-/aus-schalten	SV a[,b]	Legt den Zustand (a) des Servoantriebes (EIN/AUS) und der Bremse (EIN/AUS) für die Handdrehung (b) fest.	●	a = 0: Servo AUS, Bremse EIN a = 1: Servo EIN, Bremse AUS a = 2: Servo AUS, Bremse AUS b: Bitmuster für festgelegte Achse (Handdrehachse)	8-75

**Tab. A-8:** Übersicht der Befehle für Positionierungen und Verfahrbewegungen (2)



## Befehle für die Programmsteuerung

Nr.	Befehle	Eingabeformat	Funktion	programmierbar	Bemerkungen	Seite
25	Interrupt-Prozeß definieren	DEF ACT a,b,c	Legt den Prozeß (c), die Priorität (a) und die Interrupt-Bedingung (b) für die Ausführung eines Interrupts fest.	●	$1 \leq a \leq 8$ b: (Numerischer Datentyp, Vergleichsoperator, Numerischer Datentyp) oder (Numerischer Datentyp, Logischer Operator, Numerischer Datentyp) c: GOTO oder GOSUB	8-21
26	Interrupt freigeben/sperrern	ACT a=b	Legt den Interrupt-Status (b) des Interrupts mit der Priorität (a) fest.	●	$0 \leq a \leq 8$ 0 bedeutet alle Interrupts b = 0: sperren b = 1: freigeben	8-6
27	Sprung bei Interrupt	ON COM[(a)] GOSUB b	Legt einen Sprung zur ersten Zeile (b) der Interrupt-Routine fest, wenn ein Interrupt von der Kommunikationsleitung (a) anliegt.	●	a = 1, 2 b: Sprungziel (Zeilennummer oder Marke)	8-57
28	Kommunikations-Interrupt freigeben	COM [(a)] ON	Gibt das Interrupt der festgelegten Kommunikationsleitung (a) frei.	●	a = 1, 2	8-17
29	Kommunikations-Interrupt sperren	COM [(a)] OFF	Sperrt das Interrupt der festgelegten Kommunikationsleitung (a).	●	a = 1, 2	8-16
30	Kommunikations-Interrupt stoppen	COM [(a)] STOP	Stoppt das Interrupt der festgelegten Kommunikationsleitung (a).	●	a = 1, 2	8-18
31	Sprung zu einem Unterprogramm	ON a GOSUB [b1],[b2] ...	Der Wert (a) legt fest, welches Unterprogramm (b1), (b2), ... aufgerufen wird.	●	a: Numerischer Operationsausdruck b: Sprungziel (Zeilennummer oder Marke)	8-58
32	Programmverzweigung	ON a GOTO [b1],[b2] ...	Der Wert (a) legt fest, zu welchem Sprungziel (b1), (b2), ... verzweigt wird.	●	a: Numerischer Operationsausdruck b: Sprungziel (Zeilennummer oder Marke)	8-59
33	Sprung zu einem Unterprogramm	GOSUB a	Das festgelegte Unterprogramm (a) wird aufgerufen.	●	a: Sprungziel (Zeilennummer oder Marke)	8-34
34	Rücksprung zum Hauptprogramm	RETURN für Unterprogramm RETURN a für Interrupt-Routine	Spring beim Rücksprung aus einem Unterprogramm in die Zeile nach dem GOSUB-Befehl und beim Rücksprung aus einer Interrupt-Routine in die Zeile in der das Interrupt aufgetreten ist, oder in die nächste Zeile.	●	a: Rücksprungziel a = 0: Rücksprung zur Zeile, in der das Interrupt aufgetreten ist a = 1: Rücksprung zur Zeile hinter der Zeile, in der das Interrupt aufgetreten ist	8-71
35	Programm aufrufen	CALLP "a"[b1],[b2] ...]	Das aufgerufenen Programm (a) wird aufgerufen. Die festgelegten Variablen (b1), (b2) ... können übergeben werden.	●	a: Programmname b: Formalparameter	8-11
36	Parameter definieren	FPRM b1,[b2] ...]	Die Formalparameter (b1), (b2), ... des mit CALL P aufgerufenen Unterprogramms werden definiert.	●	b: Formalparameter	8-33
37	Sprung zu einer Programmzeile oder Marke	GOTO a	Bewirkt einen unbedingten Sprung zu einer festgelegten Zeilennummer oder Marke.	●	a: Sprungziel (Zeilennummer oder Marke)	8-35
38	Bedingter Prozeß	IF a THEN b1 [ELSE b2]	Ist das Ergebnis des Booleschen Ausdrucks wahr, wird Prozeß (b1) ausgeführt. Ist das Ergebnis falsch, wird Prozeß (b2) oder die nächste Zeile ausgeführt.	●	a: Numerischer Operationsausdruck b1, b2: Prozeß	8-40

Tab. A-9: Übersicht der Befehle für die Programmsteuerung (1)

Nr.	Befehle	Eingabeformat	Funktion	program- mierbar	Bemerkungen	Seite
39	Programmschleife	FOR a=b1 TO b2 [STEP b3] : NEXT [a[,a1] ...]	Der Programmteil zwischen der FOR- und der NEXT-Anweisung wird solange wiederholt, bis der Wert des Zählers (a) sich von (b1) dem Wert (b2) entspricht. Der Schrittwert kann über (b3) festgelegt werden.	●	a: Numerischer Wert b1: Vorgabewert b2: Endwert b3: Schrittwert	8-31
40	Programmschleife	WHILE a : WEND	Der Programmteil zwischen der WHILE- und der WEND-Anweisung wird ausgeführt, solange die Schleifenbedingung (a) erfüllt ist.	●	a: Numerischer Operationsausdruck	8-78
41	Programmablauf stoppen	HLT	Die Ausführung des Programmes wird gestoppt.	●	Verzögerter Stop.	8-36
42	Programmablauf stoppen	STOP	Die Ausführung des Programmes wird gestoppt. Hintergrundprozesse werden nicht unterbrochen.	●	Verzögerter Stop.	8-74
43	Sprung in die nächste Zeile	SKIP	Die Programmsteuerung springt in die nächste Zeile.	●		8-72
44	Programmende	END	Dieser Befehl kennzeichnet das Programmende.	●		8-28

**Tab. A-9:** Übersicht der Befehle für die Programmsteuerung (2)**Befehle für die Handsteuerung**

Nr.	Befehle	Eingabeformat	Funktion	program- mierbar	Bemerkungen	Seite
45	Handgreiferzustand festlegen	HND a=b	Der Handgreiferzustand (b) der gewählten Hand (a) wird festgelegt.	●	a = 1, 2, 3 b = 0: geschlossen b = 1: offen	8-37

**Tab. A-10:** Übersicht der Befehle für die Handsteuerung**Befehle für die Ein-/Ausgabe**

Nr.	Befehle	Eingabeformat	Funktion	program- mierbar	Bemerkungen	Seite
46	Eingabe	IN a	Die Adresse (a) der Eingabeschnittstelle wird überprüft.	●	Für ein 1-Bit-Signal: $0 \leq a \leq 32\,767$	8-41
47	Ausgabe	OUT a=b [c]	Der festgelegte Wert (b) wird über die Ausgangsadresse (a) ausgegeben. Wird eine DLY-Anweisung (c) angehängt, erhält man einen Impulsausgang.	●	Für ein 1-Bit-Signal: $0 \leq a \leq 32\,767$ b: Wert des Ausgangssignals: 1: EIN, 0: AUS c: DLY-Anweisung	8-63

**Tab. A-11:** Übersicht der Befehle für die Ein-/Ausgabe

## Zusatzfunktionen

Nr.	Befehle	Eingabeformat	Funktion	program- mierbar	Bemerkungen	Seite
48	Dimension definieren	DIM a1(b1[,b2]), [,a2(b3[,b4])]	Legt die Anzahl (b) der Elemente einer Feldvariablen (a) fest.	●	a: Variablenname b: Anzahl der Elemente $1 \leq b \leq 999$	8-26
49	Funktion definieren	DEF FN a [(b1[,b2] ... )]=c	Die Funktion (a) wird durch den Funktionsausdruck (c) definiert.	●	a: Identifizierungszeichen + Zeichenkette b: Formalparameter c: Funktionsausdruck	8-22
50	Datei öffnen	OPEN "a" [FOR b] AS [#]c	Die Datei mit Namen (a) wird mit der Zugriffsmethode (b) geöffnet. Ihr wird die Dateinummer (c) zugewiesen.	●	a: Dateiname b: Modus INPUT: Eingabemodus OUTPUT: Ausgabemodus (neue Datei) APPEND: Ausgabemodus (vorhandene Datei) Keine Angabe: Wahlfreier Modus $1 \leq c \leq 8$	8-60
51	Datei schließen	CLOSE [#]a [,[#]a2 ... ]	Die festgelegte Datei Nummer (a) wird geschlossen.	●	$1 \leq a \leq 8$	8-12
52	Eingabe	INPUT# a,b [,b2] ...	Überträgt die Daten von der Datei (a) in die festgelegte Variable (b).	●	$1 \leq a \leq 8$ b: Variablenname	8-42
53	Daten übertragen	PRINT# a [,b2][,b3] ... oder PRINT# a [:b2][:b3] ...	Gibt die Daten (b) einer Datei Nummer (a) aus.	●	$1 \leq a \leq 8$ b: Ausdruck (Daten)	8-68
54	Kommentar	REM [a]	Ermöglicht dem Programmierer einen Kommentar (a) zu schreiben.	●	a: Frei gewählte Zeichenkette	8-70

Tab. A-12: Übersicht der Zusatzfunktionen

### A.1.3 SLIM

#### Befehle für Positionierungen und Verfahrbewegungen

Nr.	Befehle	Eingabeformat	Funktion	programmierbar	Bemerkungen	Seite
1	Position anfahren	MOVE a,[@[b1]]c1 [,[@b2]]c2] ... [,d][,e]	Die festgelegten Positionen (c1), (c2) werden mit der Genauigkeit (b1), (b2), ... und der gewählten Interpolationsart angefahren.	●	a: Interpolationsart (L: Linear-Interpolation C: Kreis-Interpolation P: Gelenk-Interpolation) b: Positioniergenauigkeit $1 \leq b \leq 10000$ c: Position d: Bewegungsoption (SPEED = Bewegungsgeschwindigkeit, HAND = Handgreiferzustand (offen/geschlossen), TIME = Verfahrzeit) e: Übergangsoption (CONT: Kontinuierliche Bewegung)	8-101
2	Achsenbewegung	DRIVE (a1,b1) [, (a2,b2)] ...	Bewegung der festgelegten Achsen (a1), (a2), ... um den Weg (b1), (b2) ...	●	a: Achsennummer b: Weg der Bewegung	8-91
3	Nullpunkt anfahren	GOHOME	Der benutzerdefinierte Nullpunkt wird mittels Gelenk-Interpolation angefahren.	●		8-92
4	Achsen-geschwindigkeit einstellen	JSPEED a	Legt die Bewegungsgeschwindigkeit (a) der Handspitze für die Ausführung des DRIVE-Befehls fest.	●	a: Geschwindigkeit [%]	8-100
5	Verfahr-geschwindigkeit einstellen	SPEED a	Legt die Bewegungsgeschwindigkeit (a) der Handspitze fest.	●	a: Geschwindigkeit [mm/s]	8-109
6	Beschleunigung/Abbremsung einstellen	ACCEL a[,b]	Legt die Beschleunigungszeit (a) und die Abbremszeit (b) fest.	●	$0 \leq a, b \leq 2000$ [ms]	8-84
7	Verzögerung einstellen	DELAY a	Die Abarbeitung des Programms wird um die eingestellte Zeit verzögert.	●	a: Zeit [ms]	8-90

**Tab. A-13:** Übersicht der Befehle für Positionierungen und Verfahrbewegungen

#### Befehle für die Programmsteuerung

Nr.	Befehle	Eingabeformat	Funktion	programmierbar	Bemerkungen	Seite
8	Programmablauf stoppen	HALT [a]	Der festgelegte Ausdruck (a) wird an das Bediengerät ausgegeben und der Programmablauf unterbrochen.	●	a: Ausdruck oder Zeichenkette	8-94
9	Programm unterbrechen	HOLD [a]	Der festgelegte Ausdruck (a) wird an das Bediengerät ausgegeben und der Programmablauf zeitweise unterbrochen.	●	Verzögerter Stop.	8-96
10	Programmunterbrechung	WAIT a[,b]	Der Programmablauf wird während der Sperrzeit (b) unterbrochen, bis die Vergleichsbedingung (a) für den E/A-Port erfüllt ist.	●	a: Vergleichsbedingung für den E/A-Port b: Zeitsperre [s]	8-110

**Tab. A-14:** Übersicht der Befehle für die Programmsteuerung

**Befehle für die Handsteuerung**

Nr.	Befehle	Eingabeformat	Funktion	programmierbar	Bemerkungen	Seite
11	Hand zuweisen	HAND a=b	Die Handnummer (b) wird einem Handvariablennamen (a) zugewiesen.	●	a: Handvariablenname b: 1, 2	8-95
12	Handgreifer schließen	GRASP [a]	Die Hand mit dem Variablennamen (a) wird geschlossen.	●	a: Handvariablenname	8-93
13	Handgreifer öffnen	RELEASE [a]	Die Hand mit dem Variablennamen (a) wird geöffnet.	●	a: Handvariablenname	8-106
14	Hand wechseln	CHANGE a	Wechselt von der momentan benutzten Hand zur Hand mit dem Variablennamen (a).	●	a: Variablenname, der mit dem HAND-Befehl festgelegt wurde	8-85

**Tab. A-15:** Übersicht der Befehle für die Handsteuerung**Befehle für die Ein-/Ausgabe**

Nr.	Befehle	Eingabeformat	Funktion	programmierbar	Bemerkungen	Seite
15	Ein-/Ausgänge definieren	DEFIO a=b,c [,d]	Deklariert eine Variable (a) als Typ (b) mit der Bitnummer (c). Ist eine Maskeninformation (d) angegeben, wird nur ein bestimmtes Signal zugelassen.	●	a: Variablenname b: Typfestlegung c: Bitnummer (numerischer Wert) d: Maskeninformation (numerischer Wert)	8-87
16	Ein-/Ausgabeblock	IOBLOCK ON : IOBLOCK OFF	Die Bewegungsbefehle zwischen den Anweisungen IOBLOCK ON und IOBLOCK OFF und die folgenden Ein-/Ausgabebefehle werden parallel abgearbeitet.	●		8-99
17	Daten einlesen	IN a=b	Überträgt die Daten von der E/A-Variablen (b) einer Schnittstellenadresse in die Variable (a).	●	a: Variablenname, der mit dem DEFINT-Befehl deklariert wurde b: Variablenname, der mit dem DEFIO-Befehl deklariert wurde	8-97
18	Daten ausgeben	OUT a=b	Gibt den Wert (b) an die Schnittstellenadresse aus, die durch den E/A-Variablennamen (a) festgelegt ist.	●	a: Variablenname, der mit dem DEFIO-Befehl deklariert wurde b: Arithmetischer Ausdruck	8-104
19	E/A-Variable setzen	SET a[,b]	Der Ausgang der festgelegten E/A-Variablen (a) vom Typ BIT wird auf 1 gesetzt. Durch Angabe einer Impulsdauer, kann ein Impulsausgang erzeugt werden.	●	a: Variablenname, der mit dem DEFIO-Befehl deklariert wurde b: Impulsdauer	8-108
20	E/A-Variable zurücksetzen	RESET a	Der Ausgang der festgelegten E/A-Variablen (a) vom Typ BIT wird auf 0 gesetzt.	●	a: Variablenname, der mit dem DEFIO-Befehl deklariert wurde	8-107

**Tab. A-16:** Übersicht der Befehle für die Ein-/Ausgabe

**Zusatzfunktionen**

Nr.	Befehle	Eingabeformat	Funktion	program- mierbar	Bemerkungen	Seite
21	Integer definieren	DEFINT a[,a2][,a3] ...	Deklariert eine arithmetische Variable (a).	●	a: Variablenname (maximal 8 Zeichen)	8-86
22	Gelenk definieren	DEFJNT a[,a2][,a3] ...	Deklariert eine Gelenkvariable (a).	●	a: Variablenname (maximal 8 Zeichen)	8-88
23	Position definieren	DEFPOS a[,a2][,a3] ...	Deklariert eine Positionsvariable (a).	●	a: Variablenname (maximal 8 Zeichen)	8-89
24	Daten vom Eingabegerät einlesen	INPUT a[,a2][,a3] ...	Überträgt Daten vom Eingabegerät in eine festgelegte Variable (a)	●	a: Variablenname (arithmetische Variable, Zeichenkettenvariable, Positionsvariable, numerierte Positionsvariable)	8-98
25	Daten ausgeben	PRINT a[,a2][,a3] ... oder PRINT a[,a2][,a3] ...	Die festgelegten Daten (a) werden an das Bediengerät ausgegeben.	●	a: Ausdruck (arithmetischer Ausdruck, Zeichenkette, Variable)	8-105

**Tab. A-17:** Übersicht der Zusatzfunktionen

## A.2 Übersicht der Parameter

Parameter	Parameter-name	Reihen-nummer	Bemerkung	werkseitige Standardwerte	Gültigkeit für numerische Werte
Standardwerkzeugkoordinaten	XTL	Dezimalzahl 6 (Real number 6)	Legt den Initialisierungswert für das Werkzeugkoordinatensystem über den Abstand zwischen Befestigungsflansch und Handspitze fest. Beim 5-Achser kann nur der Z-Koordinatenwert geändert werden. (X, Y, Z, A, B, C) Einheiten: mm, mm, mm, Grad, Grad, Grad	0.0, 0.0, 107.0, 0.0, 0.0, 0.0	
Standardbasiskoordinaten	XBS	Dezimalzahl 6 (Real number 6)	Legt den Initialisierungswert über das Verhältnis zwischen XYZ-Koordinaten und Roboterkoordinaten fest. Es können nur die XYZ-Koordinatenwerte geändert werden. (X, Y, Z, A, B, C) Einheiten: mm, mm, mm, Grad, Grad, Grad	0.0, 0.0, 0.0, 0.0, 0.0, 0.0	
Verfahrensgrenzen für XYZ-Bewegungen	PAR	Dezimalzahl 6 (Real number 6)	Legt die Verfahrensgrenzen für das XYZ-Koordinatensystem fest. (2 Richtungen mit – und +), (–X, +X, –Y, +Y, –Z, +Z), Einheit: mm	–10 000, 10 000, –10 000, 10 000, –10 000, 10 000	siehe Anmerkung ❶
Verfahrensgrenzen für Gelenkbewegungen	JAR	Dezimalzahl 12 (Real number 12)	Legt die Verfahrensgrenzwerte für jedes einzelne Gelenk fest. (2 Richtungen mit – und +) (–J1, +J1, –J2, +J2, –J3, +J3, –J4, +J4, –J5, +J5, –J6, +J6), Einheit: Grad	5 Achsen: –160.0, 160.0, –90.0, 140.0, –130.0, 140.0, –160.0, 160.0, –120.0, 120.0 –200.0, 200.0 6 Achsen: –160.0, 160.0, –90.0, 140.0, 15.0, 169.0, –160.0, 160.0, –120.0, 120.0 –200.0, 200.0	
benutzerdefinierte Verfahrensgrenzen	UAR	Dezimalzahl 6 (Real number 6)	Es wird ein Signal ausgegeben, wenn der festgelegte Bereich überschritten wird. (–X, +X, –Y, +Y, –Z, +Z), Einheit: mm	1.0, 0.0, 1.0, 0.0, 1.0, 0.0	siehe Anmerkung ❷
Programmierungsmethode einstellen	RLNG	Ganzzahl 1 (Integer 1)	Auswahl der Programmierungsmethode. 0: MOVEMASTER 1: MELFA-BASIC III	1	
automatische Programmausführung	ATP	String 1	Legt den Namen eines Programms fest, das bei Einschalten der Spannungsversorgung automatisch ausgeführt wird.		
Dauerfunktion	CTN	Ganzzahl 1 (Integer 1)	Legt fest, ob der letzte Ausführungszustand des Roboters nach Einschalten der Versorgungsspannung wieder eingenommen werden soll (Programmschritt, interne Variablenwerte, E-/A-Zustände usw.) 0: Funktion AUS, 1: Funktion EIN	0	
Summer EIN/AUS	BZR	Ganzzahl 1 (Integer 1)	Schaltet den Summer EIN/AUS 0: AUS, 1: EIN	1	
Geschwindigkeit für Automatikbetrieb	SPI	Ganzzahl 1 (Integer 1)	Legt den Initialisierungswert der Geschwindigkeit für den Automatikbetrieb fest. (Wert des MOVEMASTER-Befehls SP)	12	
Übersteuerungswert für Automatikbetrieb	EOV	Dezimalzahl 2 (Real number 2)	Legt den Übersteuerungswert für den Automatikbetrieb fest. (externe Übersteuerung, Programmübersteuerung)	100.0, 100.0	

**Tab. A-18:** Übersicht der Parameter (1)

Parameter	Parameter-name	Reihen-nummer	Bemerkung	werkseitige Standardwerte	Gültigkeit für numerische Werte
JOG-Betrieb	JMOD	Ganzzahl 1 (Integer 1)	Es wird der JOG-Modus festgelegt, der nach Einschalten der Spannungsversorgung wirksam ist. 0: Gelenk-Modus 1: Kartesische Modus 2: Werkzeug-Modus	0	
Einstellungen für kontinuierliche Bewegung	CNT	Ganzzahl 1 (Integer 1)	Legt zur Erstellung eines kontinuierlichen Verfahrweges die Werte fürs Beschleunigen und Abbremsen für die über die Teaching Box eingegeben Positionen fest.	1	
Beschleunigungs- und Abbremszeit	ADL	Dezimalzahl 2 (Real number 2)	Legt die Beschleunigungs- und Abbremszeit fest. (Beschleunigungszeit, Abbremszeit) Einheit: s	0.2, 0.2	
Zeitkonstante für Steuerung	TSR	Dezimalzahl 1 (Real number 1)	Legt die Zeitkonstante für die Steuerung fest. Einheit: ms	20.0	
Positioniergenauigkeit	PWI	Dezimalzahl 1 (Real number 1)	Legt die Genauigkeit für die Positionierung fest. Einheit: Impulse	10 000	
Einstellungen für Handsteuerung	GCD	Ganzzahl 1 (Integer 1)	Legt den Initialisierungswert für den Handgreiferzustand fest, welcher nach Einschalten der Versorgungsspannung eingenommen werden soll. 1. Hand: vorwärts/rückwärts, Initialisierungsbedingungen, 2. Hand: vorwärts/rückwärts, Initialisierungsbedingungen, 3. Hand: vorwärts/rückwärts, Initialisierungsbedingungen, vorwärts/rückwärts = 0: vorwärts, 1: rückwärts Initialisierungszustand: 0 – 3	0, 1, 0, 1, 0, 1 Befehle: GC, GO	
Festlegen der Zugriffsmöglichkeiten für die Nullpunkteinstellung	HOE	Ganzzahl 1 (Integer 1)	Legt fest, ob eine Nullpunkteinstellung erlaubt oder nicht erlaubt sein soll. 0: erlaubt, 1: nicht erlaubt	0	siehe Anmerkung ③
benutzerdefinierter Nullpunkt	UOG	Dezimalzahl 6 (Real number 6)	Legt den benutzerdefinierten Nullpunkt fest. (W, S, E, T, P, R)	-160.0, -45.00, 0.0 0.0, -120.0, -200.0	
Bewegungsreihenfolge für Nullpunktrückstellung	UNG	Dezimalzahl 6 (Real number 6)	Legt die Bewegungsreihenfolge für die Nullpunktrückstellung fest.	2, 1, 1, 1, 2, 2	siehe Anmerkung ③
Handwinkel (R) Koordinatenauswahl	RCD	Ganzzahl 1 (Integer 1)	Legt die Steuerung und Anzeige des Handwinkels für die allgemeine Winkel- oder Gelenkwinkelmethode fest. 0: allgemeine Winkelmethode, 1: Gelenkwinkelmethode	0	
Kontakttyp für externen NOT-HALT auswählen	INB	Ganzzahl 1 (Integer 1)	Legt die speziellen E/As für den A- oder B-Typ-Kontakt fest.	0	
Einstellungen für parallele Eingabedaten	IN1	String 20	Legt die Einstellungen für externe Eingabedaten fest. Legt die Einstellungen für die 1. Eingabe über die E-/A-Schnittstelle fest.	PI0, , , , PI1, , , , , , , , , , , , , , STA, STP, RST	
	IN2	String 20	Legt die Einstellungen für die 2. Eingabe über die E-/A-Schnittstelle fest.		
	IN3	String 20	Legt die Einstellungen für die 3. Eingabe über die E-/A-Schnittstelle fest.		

Tab. A-18: Übersicht der Parameter (2)



<b>Parameter</b>	<b>Parameter-name</b>	<b>Reihen-nummer</b>	<b>Bemerkung</b>	<b>werkseitige Standardwerte</b>	<b>Gültigkeit für numerische Werte</b>
Einstellungen für parallele Ausgabedaten	OT1  OT2  OT3	String 16  String 16  String 16	Legt die Einstellungen für externe Ausgabedaten fest. Legt die Einstellungen für die 1. Ausgabe über die E-/A-Schnittstelle fest.  Legt die Einstellungen für die 2. Ausgabe über die E-/A-Schnittstelle fest.  Legt die Einstellungen für die 3. Ausgabe über die E-/A-Schnittstelle fest.	, , , , , , , , , , , , RUN, WAI, ERR	
Starten mit Einlesen eines Programms	PST	Ganzzahl 1 (Integer 1)	Der Roboter startet ein bestimmtes Programm über numerischen Eingabewert 0: aktuelles Programm fortsetzen 1: vorgegebenes Programm starten	0	
Kommunikations-Modus	CMO	Ganzzahl 7 (Integer 7)	Legt den Kommunikations-Modus der RS232C-Schnittstelle fest. Baudrate (1:19200, 2:9600, 3:4800, 4:2400, 5:1200, 6:600), Daten-Bit (7, 8), Parity-Bit (E: gleich, O: ungleich, N: kein), Stop-Bit (1, 2), END-Code (0: auto, 1: CR+LF, 2: CR), Kommunikationspfad (M: kein Protokoll)	2, 8, E, 2, 0, M	siehe Anmerkung ④
Montagerichtung	GDIR	Dezimalzahl 4 (Real number 4)	Die Montagerichtung des Roboters wird in Bezug zum Befehl für die optimale Beschleunigungs-/Abbremszeit (OADL) gesetzt. Ist die Montagerichtung „3: Spezielle Montage“, wird das Gewicht in X-, Y- und Z-Richtung in das Basis-Koordinatensystem des Roboters gesetzt. (Montagerichtung, Gewicht in X-Richtung, Gewicht in Y-Richtung, Gewicht in Z-Richtung) Erläuterung: – Montagerichtung 0: Standard (Bodenmontage) 1: Wandmontage 2: Deckenmontage 3: Spezielle Montage – Gewicht in X-Richtung: Gewicht, das in Richtung der X-Achse wirkt [kgf/mm] – Gewicht in Y-Richtung: Gewicht, das in Richtung der Y-Achse wirkt [kgf/mm] – Gewicht in Z-Richtung: Gewicht, das in Richtung der Z-Achse wirkt [kgf/mm]	0, 0.0, 0.0, 0.0	
Last der Handspitze	HNDM	Dezimalzahl 1 (Real number 1)	Die Last der Handspitze wird eingesetzt, wenn bei der Festlegung der optimalen Beschleunigungs-/Abbremszeit (OADL) kein Wert für die Last angegeben wurde. Einheit: kg	2.0	
Schwerpunkt der Handspitze	HNDG	Dezimalzahl 3 (Real number 3)	Der Schwerpunkt der Hand (X-, Y- und Z-Komponente im Werkzeug-Koordinatensystem) wird in Bezug zum Befehl für die optimale Beschleunigungs-/Abbremzeit (OADL) gesetzt. Einheit: mm	0.0, 0.0, 0.1	

**Tab. A-18:** Übersicht der Parameter (3)

## Anmerkungen:

- ❶ Der Initialisierungswert liegt innerhalb des Arbeitsbereichs.
- ❷ Der Initialisierungswert liegt außerhalb des Arbeitsbereichs.
- ❸ Der neue Parameterwert wird direkt nach der Einstellung wirksam.
- ❹ Bei der Einstellung „0: auto“ sendet der Roboter mit dem selben Code, den er vom externen Zubehör empfängt. (Die Werkseinstellung ist „CR“.)

## A.3 Übersicht der Fehlercodes

Fehlercode			Fehlerursachen und Fehlerbehebung	
Anzeige am Steuergerät		Unternummer		
0	1	00 – 20		Ursache: In der Spannungsversorgung ist ein Fehler aufgetreten. Behebung: Schalten Sie die Spannungsversorgung aus und wieder ein.
		3	0	Ursache: Die Versorgungsspannung ist zu niedrig. Behebung: Überprüfen Sie, ob Versorgungsspannung und Betriebsspannung übereinstimmen. Überprüfen Sie, ob die Spannung bei einer Roboterbewegung absinkt.
		4	0	Ursache: Die Versorgungsspannung ist zu hoch. Behebung: Überprüfen Sie, ob Versorgungsspannung und Betriebsspannung übereinstimmen. Überprüfen Sie, ob die Spannung bei einer Roboterbewegung ansteigt.
0	2	01 – 66 (letzte Stelle gibt Gelenknummer an)		Ursache: Die Encodererkennung ist fehlerhaft. Behebung: Schalten Sie die Spannungsversorgung aus und wieder ein. Tritt der Fehler weiterhin auf, setzen Sie den Encoder zurück und definieren Sie den Nullpunkt neu.
		0	1 – 6	Encoderdaten-Datenbereichsüberschreitung
		2	1 – 6	Encoderdaten-Initialisierungsfehler
		3	1 – 6	Encoderdaten-Kommunikationsfehler
		5	1 – 6	Encoderdatenfehler wegen Überhitzung
		7	1 – 6	Ursache: Encoder meldet unzulässige Geschwindigkeit. Behebung: Schalten Sie die Spannungsversorgung aus und wieder ein. Stellen Sie bei einer Abweichung der Position den Nullpunkt erneut ein.
0	3	0	0	Ursache: In der Steuerung ist ein Fehler des Batterie-Backups aufgetreten. Behebung: Überprüfen Sie die Anschlüsse des Batteriekabels zur Steuereinheit. Tauschen Sie die Batterie aus, wenn die Fehlermeldung weiterhin ansteht. Beachten Sie dazu die Hinweise zum Tauschen der Batterie.
		1	1 – 6 (Gelenk)	Ursache: Im Roboter ist ein Fehler des Batterie-Backups aufgetreten. Der Anschluß der Batteriekabel ist fehlerhaft. Behebung: Überprüfen Sie die Anschlüsse des Batteriekabels auf der Platine. Tauschen Sie die Batterie aus, wenn die Fehlermeldung weiterhin ansteht. Beachten Sie dazu die Hinweise zum Tauschen der Batterie.
0	4	00 – 50		Ursache: Die gespeicherten Daten sind unzulässig. Behebung: Löschen Sie die gespeicherten Daten.
		9	0	Ursache: Die Daten werden nicht korrekt gespeichert. Behebung: Laden Sie alle Programme neu und prüfen Sie die Werkzeugparameter und die restaurierten Parameter.
0	5	00 – 66 (letzte Stelle gibt Gelenknummer an)		Ursache: Im Servo-System ist ein Fehler aufgetreten. Behebung: Schalten Sie die Spannungsversorgung aus und wieder ein.
		0	0	2-port memory versagt
		1	0	Servo-Speicher versagt
		2	0	Watch-dog Zeitüberschreitung
		3	0 – 6	Amplifier/Encoder nicht angeschlossen
		4	0	gate array Fehler
		5	0 – 6	A/D-Wandler fehlerhaft
		6	0 – 6	Überstrom

**Tab. A-19:** Fehlercodes (1)

Fehlercode				Fehlerursachen und Fehlerbehebung	
Anzeige am Steuergerät		Unternummer			
0	6	0	0	Ursache: Behebung:	Die Verwendung des Kommunikationsprotokolls ist fehlerhaft. Prüfen Sie das Kommunikationsprotokoll und die Daten des angeschlossenen Zubehörs.
		1	0	Ursache: Behebung:	Die Kommunikationsverbindung ist unterbrochen. Überprüfen Sie die Kabelverbindungen zum Steuergerät, und ob das angeschlossene Zubehör eingeschaltet ist.
		2	0	Ursache: Behebung:	Der Datenempfang ist fehlerhaft. Überprüfen Sie die Kabelverbindung.
		3	0	Ursache: Behebung:	Die Datensendung ist fehlerhaft. Überprüfen Sie die Kabelverbindung.
		4	0	Ursache: Behebung:	Die Kommunikationsverbindung zur Teaching Box ist unterbrochen. Schalten Sie die Spannungsversorgung aus und wieder ein.
		5	0	Ursache: Behebung:	Der Kommunikationskanal ist noch nicht geöffnet. Öffnen Sie den Kommunikationskanal mit dem OPN-Befehl.
0	7	1	0	Ursache: Behebung:	Die Sicherung der pneumatisch angetriebenen Hand ist durchgebrannt. Überprüfen Sie die Kabel und Kabelverbindungen, und wechseln Sie die Sicherung.
		1	1	Ursache: Behebung:	Die Sicherung der CPU-Karte ist durchgebrannt. Überprüfen Sie die Kabel und Kabelverbindungen, und wechseln Sie die Sicherung.
		2	0	Ursache: Behebung:	Die Sicherung für die parallele E-/A-Schnittstelle ist durchgebrannt. Überprüfen Sie die Kabel und Kabelverbindungen, und wechseln Sie die Sicherung.
1	2	0	0	Ursache: Behebung:	Das NOT-HALT-Signal wurde ausgelöst (über externes Signal). Schalten Sie das NOT-HALT-Signal aus und beheben Sie die Fehlerursache.
		1	0	Ursache: Behebung:	Das NOT-HALT-Signal wurde ausgelöst (über Bedienfeld). Schalten Sie das NOT-HALT-Signal aus und beheben Sie die Fehlerursache.
		2	0	Ursache: Behebung:	Das NOT-HALT-Signal wurde ausgelöst (über Teaching Box). Schalten Sie das NOT-HALT-Signal aus und beheben Sie die Fehlerursache.
1	2	3	0	Ursache: Behebung:	Die Teaching Box wurde bei eingeschalteten EMG-CANCEL-Schalter der Steuerung angeschlossen. Lösen Sie die Verbindung der Teaching Box, wenn Sie diese nicht benutzen wollen. Schalten Sie den EMG-CANCEL-Schalter der Steuerung aus, wenn Sie die Teaching Box benutzen wollen.
1	4	01 – 56 (letzte Stelle gibt Gelenknummer an)		Ursache: Behebung:	Eine Überlastung ist aufgetreten. Überprüfen Sie, ob die zulässige Lastkapazität überschritten wurde, die programmierte Bewegung des Roboters möglich ist oder, ob ein Gegenstand die Roboterbewegung stört.
		1	1 – 6	Mechanische Verriegelung/Blockade	
		2	1 – 6	Zu hohe Geschwindigkeit (Befehlsbereich überschritten)	
		3	1 – 6	Überlast	
		4	1 – 6	Übermäßiger Positionierungsfehler	
1	4	6	0	Ursache: Behebung:	Regenerative Überlast des Servo-Verstärkers. Überprüfen Sie die Last und verringern Sie die Bewegungsgeschwindigkeit. Lassen Sie den Servo eine Zeitlang ausgeschaltet.
1	6	0	0	Ursache: Behebung:	Es kann keine Verfahrbewegung im XYZ-Modus ausgeführt werden, weil keine Nullpunkteinstellung erfolgt ist. Nehmen Sie die Nullpunkteinstellung über die Teaching Box vor. Anschließend kann die Verfahrbewegung im XYZ-Modus ausgeführt werden.

Tab. A-20: Fehlercodes (2)

Fehlercode				Fehlerursachen und Fehlerbehebung	
Anzeige am Steuergerät		Unternummer			
1	7	00 – 99		Ursache:	Der Servoparameter ist fehlerhaft.
				Behebung:	Korrigieren Sie den Parameter.
1	8	0	0	Ursache:	Das STOP-Signal oder der STOP-Schalter ist aktiviert.
				Behebung:	Setzen Sie den STOP-Status zurück.
2	3	0	0	Ursache:	Die Lebensdauer der Batterie ist abgelaufen.
				Behebung:	<b>Wechseln Sie die Batterie schnellstens aus !</b>
2	4	0	0	Ursache:	Die Operation kann bei laufendem Programm nicht ausgeführt werden.
				Behebung:	Stoppen Sie den Programmablauf. Führen Sie anschließend die Operation erneut aus.
		1	0	Ursache:	Die Operation kann bei eingeschalteter Servospannung nicht ausgeführt werden.
				Behebung:	Schalten Sie die Spannung aus. Führen Sie anschließend die Operation erneut aus.
		2	0	Ursache:	Die Operation kann bei ausgeschalteter Servospannung nicht ausgeführt werden.
				Behebung:	Schalten Sie die Spannung ein. Führen Sie anschließend die Operation erneut aus.
2	6	0	0	Ursache:	Die angegebene Zeilennummer existiert nicht.
				Behebung:	Überprüfen und Ändern Sie die Angaben zu den Zeilennummern im Programm.
2	7	0	0	Ursache:	Die angegebenen Positionsdaten existieren nicht.
				Behebung:	Überprüfen und Ändern Sie die Angaben zu den Positionsdaten im Programm.
2	8	0	0	Ursache:	Bei der Befehlsausführung ist ein struktureller Programmierfehler aufgetreten.
				Behebung:	Korrigieren Sie die Programmierung des Befehls. Nehmen Sie anschließend die Befehlseingabe erneut vor.
		1	0	Ursache:	Der Wert des Befehlsargumentes liegt außerhalb des zulässigen Bereichs.
				Behebung:	Korrigieren Sie den Wert.
		3	0	Ursache:	Abweichung der Programmiersprache.
				Behebung:	Überprüfen Sie, ob die verwendete und die ausgewählte Programmiersprache übereinstimmen.
2	9	0	0	Ursache:	Der Wert des Befehls ist fehlerhaft programmiert.
				Behebung:	Korrigieren Sie die Programmierung des Befehls. Nehmen Sie anschließend die Befehlseingabe erneut vor.
3	4	0	0	Ursache:	Die Tiefe der Verschachtelungsstruktur zwischen dem RC- und NX-Befehl überschreitet den zulässigen Höchstwert von 9 Schritten.
				Behebung:	Reduzieren Sie die Verschachtelungsstruktur auf maximal 9 Schritte.
3	5	0	0	Ursache:	Die Anzahl der Interrupt-Befehle überschreitet die zulässige Anzahl.
				Behebung:	Überprüfen Sie das Programm. Löschen Sie alle überzähligen Interrupt-Befehle.
3	6	0	0	Ursache:	Es kann nicht korrekt gerechnet werden.
				Behebung:	Überprüfen und korrigieren Sie das Programm.

Tab. A-19: Fehlercodes (3)

Fehlercode				Fehlerursachen und Fehlerbehebung	
Anzeige am Steuergerät		Unternummer			
3	7	0	0	Ursache:	Fehlerhaftes Zusammenspiel zwischen Sprung- (GS) und Rücksprungbefehl (RT).
		1	0	Behebung:	Korrigieren Sie das Programm.
3	8	0	0	Ursache:	Die festgelegte Palette ist vorher nicht definiert worden.
3	9	0	0	Behebung:	Definieren Sie die Palette vor der Ausführung.
3	9	0	0	Ursache:	Der Gerätebetrieb ist nicht möglich, weil das Gerät nicht betriebsbereit ist.
		1	0	Behebung:	Stellen Sie die Betriebsbereitschaft für das angeschlossene Gerät her.
4	5	0	1 – 6 (Gelenk)	Ursache:	Der Winkelbetrag für ein Gelenk überschreitet den zulässigen Wertebereich.
		1	1 – 6 (Gelenk)	Behebung:	Bewegen Sie den Roboter mittels Gelenk-JOG-Modus in den zulässigen Arbeitsbereich. Tritt dieser Fehler während des Programmablaufs auf, ändern Sie die Positionsdaten.
		2	1 – 6 (Gelenk)	Ursache:	Die orthogonale Position überschreitet den zulässigen Wertebereich.
		5	0	Behebung:	Bewegen Sie den Roboter mittels Gelenk-JOG-Modus in den zulässigen Arbeitsbereich. Tritt dieser Fehler während des Programmablaufs auf, ändern Sie die Positionsdaten.
		6	0	Ursache:	Der benutzerdefinierte Arbeitsbereich überschreitet den zulässigen Bereich.
4	6	0	0	Behebung:	Bewegen Sie den Roboter mittels Gelenk-JOG-Modus in den zulässigen Arbeitsbereich. Tritt dieser Fehler während des Programmablaufs auf, ändern Sie die Positionsdaten.
4	7	0	0	Ursache:	Die anzufahrende Position kann nicht berechnet werden.
4	8	01 – 16		Behebung:	Fahren Sie den Roboter im Positions-JOG-Modus zu einer anderen Position.
4	9	0	0	Ursache:	Die geteachte Position kann vom Roboter nicht angefahren werden.
		1	0	Behebung:	Fahren Sie den Roboter in den zulässigen Arbeitsbereich und teachen Sie die Position erneut.
4	6	0	0	Ursache:	Die Zielposition für die Roboterbewegung liegt außerhalb des zulässigen Bereichs, oder Start- und Zielposition weisen in der Linear- und in der Kreisinterpolation unterschiedliche Kennzeichnungsstrukturen auf. ?? structure flag ?
4	7	0	0	Behebung:	Korrigieren Sie die Zielposition für die Roboterbewegung oder die Kennzeichnungsstruktur.
4	7	0	0	Ursache:	Die Halteposition zwischen der Start- und Endposition kann nicht berechnet werden.
4	8	01 – 16		Behebung:	Fügen Sie zwischen der Start- und Endposition eine zusätzliche Halteposition ein.
4	8	01 – 16		Ursache:	Die Verfahrgeschwindigkeit ist zu groß.
4	9	0	0	Behebung:	Reduzieren Sie die festgelegte Verfahrgeschwindigkeit.
4	9	0	0	Ursache:	Die Anzahl der abzuspeichernden Dateien ist zu groß.
		1	0	Behebung:	Überprüfen Sie das Programm. Löschen Sie alle nicht benötigten Dateien.
4	9	0	0	Ursache:	Es können keine Befehle mehr eingegeben werden, weil der Speicherbereich voll ist.
		1	0	Behebung:	Überprüfen Sie das Programm. Löschen Sie alle nicht benötigten Befehle.

Tab. A-19: Fehlercodes (4)

Fehlercode				Fehlerursachen und Fehlerbehebung	
Anzeige am Steuergerät		Unternummer			
5	6	0	0	Ursache:	Es wurde ein Programmname angegeben, dessen zugehöriges Programm nicht existiert.
				Behebung:	Überprüfen Sie den Dateinamen des Programms.
		1	0	Ursache:	Der Dateiname wurde zweimal vergeben.
				Behebung:	Überprüfen Sie den Dateinamen des Programms und löschen Sie das überschüssige Programm.
		2	0	Ursache:	Der Schreibvorgang ist nicht zulässig.
				Behebung:	Geben Sie eine andere Datei an.
		3	0	Ursache:	Diese Datei ist schreibgeschützt.
				Behebung:	Geben Sie eine andere Datei an.
		4	0	Ursache:	Der Dateiname ist nicht richtig angegeben worden.
				Behebung:	Überprüfen Sie, ob bei Eingabe des Programmnamens unzulässige Zeichen angegeben wurden oder die Reihenfolge der Zeichen fehlerhaft ist.
5	7	0	0	Ursache:	Es wurde kein Programm ausgewählt.
				Behebung:	Geben Sie den Namen des gewünschten Programms an.
5	8	0	0	Ursache:	Die Satznummer für die Neunummerierung ist fehlerhaft.
				Behebung:	Überprüfen Sie die Satznummer.
		1	0	Ursache:	Die zur Editierung angegebenen Daten existieren nicht.
				Behebung:	Überprüfen Sie das Programm und wiederholen Sie den Schritt.
6	7	0	0	Ursache:	Es wurde ein spezielles Ausgangssignal für die allgemeine Datenausgabe eingesetzt.
				Behebung:	Ändern Sie die Signalzuordnung für die spezielle Ausgabe oder das Programm.
		0	1	Ursache:	Es wurde eine unzulässige Änderung eines Eingabebefehls vorgenommen.
				Behebung:	Die Befehle STA, STP und RST sind auf Bit Nummer 17, 18 und 19 der Standard E/A-Schnittstelle festgelegt.
6	8	0	0	Ursache:	Die Zuordnung der Signalparameter PI0 bis PI2 ist fehlerhaft.
				Behebung:	Beachten Sie die Angaben zu den Signalparametern in der Bedienungsanleitung.
6	9	0	0	Ursache:	Die Zuordnung der Signalparameter PO0 bis PO2 ist fehlerhaft.
				Behebung:	Beachten Sie die Angaben zu den Signalparametern in der Bedienungsanleitung.
7	0	01 – 06		Ursache:	Die Encoder-Batterie ist leer.
				Behebung:	Tauschen Sie die Batterie. Überprüfen Sie die Kabel und Kontakte.
		11 – 16		Ursache:	Die Multi-Rotationsdaten des Encoders sind fehlerhaft.
				Behebung:	Schalten Sie die Spannungsversorgung aus und wieder ein. Weicht die Position ab, wiederholen Sie die Nullpunkteinstellung. Entfernen Sie eventuelle Störquellen.
		21 – 26		Ursache:	Die Motorposition beim Ausschalten weicht von der Position beim letzten Ausschalten ab.
				Behebung:	Überprüfen Sie, ob der Roboterarm bei manuell gelösten Bremsen bewegt wurde, während die Spannung ausgeschaltet war. Lässt sich keine Ursache für die Abweichung feststellen, setzen Sie die Fehlermeldung zurück und überprüfen Sie, ob die aktuelle Position mit den Positionsdaten übereinstimmt. Bei Übereinstimmung können Sie den Betrieb wieder aufnehmen. Bei einer Abweichung oder bei nicht geklärter Ursache setzen Sie sich mit Ihrem Servicepartner in Verbindung.
7	9	0	0	Ursache:	Die Spannung im Hauptschaltkreis ist abgefallen.
				Behebung:	Überprüfen Sie die Spannungsversorgung.
8	9	00 – 99		Ursache:	Bei der Selbstdiagnose der Systemsoftware ist ein Fehler aufgetreten.
				Behebung:	Wiederholen Sie nach Rücksetzen der Fehlermeldung den Bedienschritt.

Tab. A-19: Fehlercodes (5)

## A.4 Störungssuche

Die nachfolgenden Tabellen beschreiben die Vorgehensweise bei der Störungssuche und deren Behebung.

Störung	Überprüfungen	Maßnahmen zur Störungsbehebung
Störung in der Spannungsversorgung	<ol style="list-style-type: none"> <li>① Überprüfen Sie den Spannungsanschluß.</li> <li>② Überprüfen Sie die Sicherung.</li> <li>③ Überprüfen Sie die Spannungshöhe der Versorgungsspannung.</li> </ol>	<ul style="list-style-type: none"> <li>● Achten Sie auf einen korrekten und festen Sitz des Verbindungssteckers.</li> <li>● Tauschen Sie die Sicherung aus. Gehen Sie wie im technischen Handbuch beschrieben vor.</li> <li>● Das Steuergerät benötigt eine Versorgungsspannung mit AC 230 V.</li> </ul>
Der Servo-Antrieb läßt sich nicht einschalten.	<ol style="list-style-type: none"> <li>① Wird eine Fehlermeldung angezeigt?</li> <li>② Überprüfen Sie, ob der [ENABLE/DISABLE]-Schalter sich beim Einschalten der Spannungsversorgung des Steuergerätes in der Stellung „ENABLE“ befindet und der Totmannschalter betätigt ist.</li> <li>③ Betätigen Sie den Totmannschalter, während Sie den [ENABLE/DISABLE]-Schalter auf „ENABLE“ stellen.</li> <li>④ Ist das externe Signal „SERVO ON/OFF“ aktiviert?</li> </ol>	<ul style="list-style-type: none"> <li>● Stellen Sie den [ENABLE/DISABLE]-Schalter der Teaching Box auf „DISABLE“, betätigen Sie den [RESET]-Schalter auf der Frontseite des Steuergerätes und quittieren Sie die Fehlermeldung.</li> <li>● Schalten Sie den Servo EIN, während Sie den Totmannschalter auf der Rückseite der Teaching Box betätigen. Stellen Sie den [ENABLE/DISABLE]-Schalter auf „DISABLE“ und schalten Sie die Spannungsversorgung des Steuergerätes wieder ein.</li> <li>● Während des JOG-Betriebes über die Teaching Box muß der Totmannschalter betätigt sein.</li> <li>● Deaktivieren Sie das Signal.</li> </ul>
Der Roboter arbeitet nicht.	<ol style="list-style-type: none"> <li>① Überprüfen Sie die Programmierung der Befehle.</li> <li>② Überprüfen Sie die Kabelverbindungen.</li> <li>③ Überprüfen Sie den NOT-HALT-Schalter.</li> <li>④ Überprüfen Sie den Arbeitsbereich.</li> <li>⑤ Überprüfen Sie, ob der Roboterarm mit den mechanischen Stoppern oder mit anderen Maschinenteilen kollidiert.</li> <li>⑥ Überprüfen Sie den Spannungsanschluß für den Motor.</li> <li>⑦ Überprüfen Sie die externen STOP- und NOT-HALT-Schalter.</li> </ol>	<ul style="list-style-type: none"> <li>● Überprüfen Sie den Befehlsnamen und das Format.</li> <li>● Achten Sie auf einen korrekten und festen Sitz der Kabelstecker.</li> <li>● Setzen Sie den NOT-HALT-Schalter zurück.</li> <li>● Führen Sie die Roboterbewegungen nur innerhalb des zulässigen Arbeitsbereichs aus.</li> <li>● Verfahren Sie den Roboter so, daß eine Kollision vermieden wird.</li> <li>● Achten Sie auf einen korrekten und festen Sitz der Kabelstecker.</li> <li>● Setzen Sie die STOP- und NOT-HALT-Schalter zurück.</li> </ul>
Über den Computer kann kein Programm eingegeben werden.	<ol style="list-style-type: none"> <li>① Überprüfen Sie, ob der Schreibschutz-Schalter eingeschaltet ist.</li> <li>② Überprüfen Sie die Programmierung der Befehle.</li> <li>③ Überprüfen Sie die Kabelverbindungen zum Computer.</li> <li>④ Ist der Programmablauf gestoppt?</li> </ol>	<ul style="list-style-type: none"> <li>● Schalten Sie den Schreibschutz-Schalter aus.</li> <li>● Überprüfen Sie den Befehlsnamen und das Format.</li> <li>● Achten Sie auf einen korrekten und festen Sitz der Kabelstecker.</li> <li>● Betätigen Sie zum Rücksetzen des Programms die RESET-Taste.</li> </ul>

**Tab. A-20:** Tabelle für Störungssuche (1)



Störung	Überprüfungen	Maßnahmen zur Störungsbehebung
Die pneumatisch angetriebene Hand arbeitet nicht.	<ol style="list-style-type: none"> <li>① Überprüfen Sie die Druckluftversorgung.</li> <li>② Ist der richtige Magnetventiltyp eingesetzt worden?</li> </ol>	<ul style="list-style-type: none"> <li>● Stellen Sie sicher, daß immer eine ausreichende Druckluftversorgung gegeben ist.</li> <li>● Setzen Sie nur einen kompatiblen Magnetventiltyp ein.</li> </ul>
Der Roboter stoppt während des Betriebes.	<ol style="list-style-type: none"> <li>① Überprüfen Sie die Kabel.</li> <li>② Überprüfen Sie die Belastung.</li> <li>③ Überprüfen Sie die Spannungshöhe der Versorgungsspannung.</li> <li>④ Überprüfen Sie, ob die Spannung kurzzeitig ausgefallen ist.</li> <li>⑤ Überprüfen Sie die Stellung des NOT-HALT-Schalters.</li> <li>⑥ Überprüfen Sie den Motor auf eine ungewöhnliche Geruchsentwicklung.</li> <li>⑦ Überprüfen Sie, ob der Motorantrieb ungewöhnliche Geräusche oder starke Vibrationen erzeugt.</li> </ol>	<ul style="list-style-type: none"> <li>● Tauschen Sie defekte Kabel aus.</li> <li>● Reduzieren Sie die Belastung. Beachten Sie die Angaben zur maximalen Belastung im Technischen Handbuch.</li> <li>● Die Spannungshöhe muß innerhalb des zulässigen Bereichs liegen.</li> <li>● Wiederholen Sie die Bedienschritte.</li> <li>● Setzen Sie den NOT-HALT-Schalter zurück und wiederholen Sie die Bedienschritte.</li> <li>● Möglicherweise ist die Motorwicklung durchgebrannt.</li> <li>● Beachten Sie die Erläuterungen im nachfolgenden Abschnitt „ungewöhnliche Geräusche, starke Vibrationen“.</li> </ul>
Die Wiederholgenauigkeit läßt nach.	<ol style="list-style-type: none"> <li>① Überprüfen Sie, ob eine elektrische Störspannung auftritt.</li> <li>② Überprüfen Sie den Sitz der Befestigungsschrauben am Boden, am Roboterarm und an der Roboterhand.</li> <li>③ Überprüfen Sie die Spannung des Zahnriemens.</li> <li>④ Überprüfen Sie den Abgleich zwischen Steuergerät und Roboter.</li> <li>⑤ Prüfen Sie, ob bei dem bisherigen Betrieb des Roboters eine Kollision aufgetreten ist.</li> </ol>	<ul style="list-style-type: none"> <li>● Beseitigen Sie die Störquelle.</li> <li>● Ziehen Sie alle losen Befestigungsschrauben nach.</li> <li>● Beachten Sie die Angaben zur Spannung des Zahnriemens in der Bedienungsanleitung.</li> <li>● Führen Sie eine Nullpunkteinstellung durch.</li> <li>● Achten Sie darauf, das keine Kollisionen auftreten.</li> </ul>
Es treten ungewöhnliche Geräusche oder starke Vibrationen auf.	<ol style="list-style-type: none"> <li>① Überprüfen Sie den Sitz der Befestigungsschrauben am Roboterarm.</li> <li>② Überprüfen Sie, ob die Geräusche oder Vibrationen durch eines der Getriebe verursacht werden.</li> </ol>	<ul style="list-style-type: none"> <li>● Ziehen Sie alle losen Befestigungsschrauben nach.</li> <li>● Möglicherweise ist das Getriebe beschädigt.</li> </ul>
Die Ausgabe über die E-/A-Schnittstelle funktioniert nicht.	<ol style="list-style-type: none"> <li>① Überprüfen Sie die Spannungshöhe und Polarität der externen Spannungsversorgung.</li> <li>② Überprüfen Sie die Programmierung der OD-, OT- und OB-Befehle.</li> <li>③ Überprüfen Sie, ob ein Transistor aufgrund einer Überlastung beschädigt wurde.</li> </ol>	<ul style="list-style-type: none"> <li>● Schließen Sie den Roboter an eine Spannungsversorgung an, die die korrekte Spannungshöhe und Polarität besitzt.</li> <li>● Korrigieren Sie die Programmierung der aufgeführten Befehle.</li> <li>● Lassen Sie die E/A-Karte reparieren.</li> </ul>

**Tab. A-20:** Tabelle für Störungssuche (2)

## A.5 Stellungsmerker

### Was ist ein Stellungsmerker?

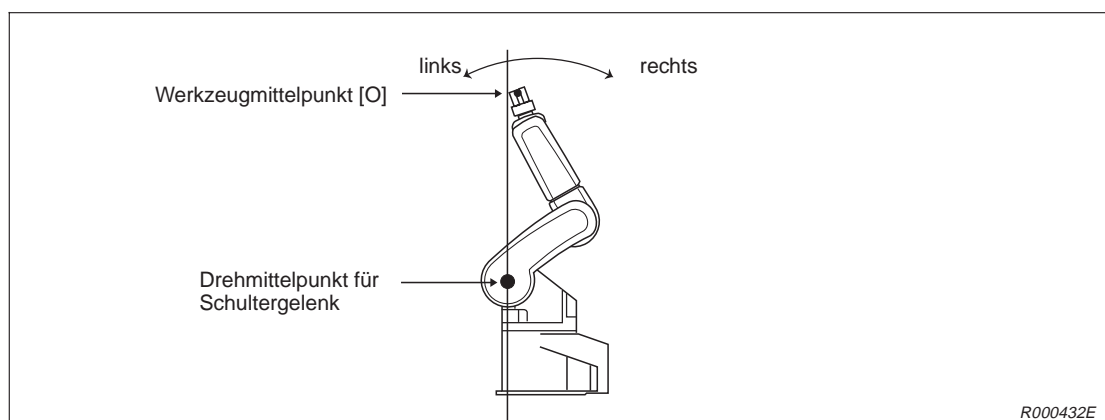
Dies ist ein Kennzeichen am Roboter, welches die Roboterstellung anzeigt. Die Robotersteuerung definiert eine bestimmte Roboterposition (Position der Handspitze) über die Positionsdaten (Achsen X, Y, Z und Winkel A, B, C). Es gibt jedoch komplementäre Positionen mit den gleichen Positionsdaten, aber mit unterschiedlichen Roboterstellungen (Stellung der Roboterelkenke). Diese unterschiedlichen Roboterstellungen werden über die Stellungsmerker eindeutig identifiziert und festgelegt.

### A.5.1 Definition der einzelnen Stellungsmerker

#### 5 Achsen

#### Stellungsmerker für Stellung: links / rechts (L/R)

Dieser Merker zeigt die Position des Werkzeugmittelpunktes [O] in bezug zu einer senkrechten Achsenlinie, die durch den Drehmittelpunkt des Schultergelenks geht.

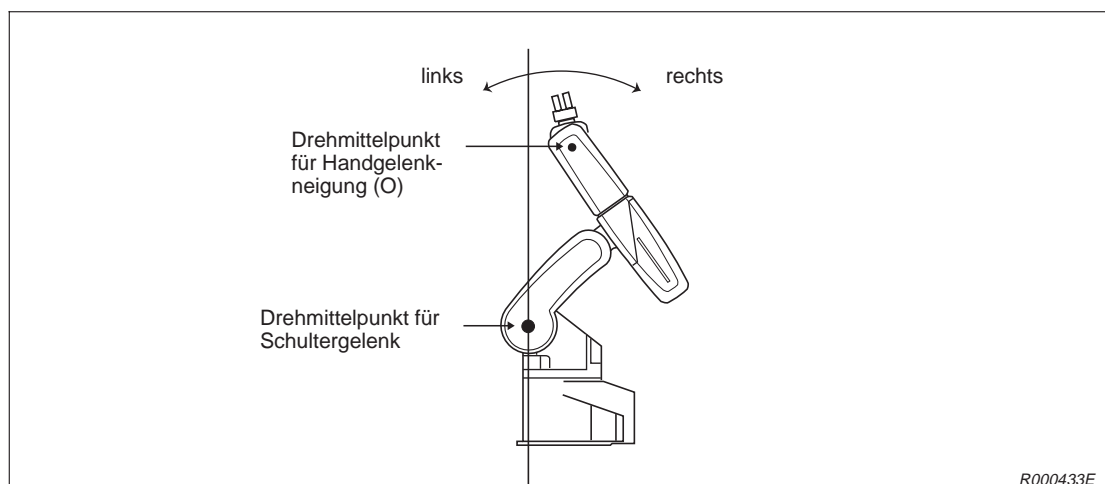


**Abb. A-1:** Kennungen für die Stellung am 5-achsigen Roboter: links / rechts (L/R)

#### 6 Achsen

#### Stellungsmerker für Stellung: links / rechts (L/R)

Dieser Merker zeigt die Position des Drehmittelpunktes [O] für die Handgelenkneigung in bezug zu einer senkrechten Achsenlinie, die durch den Drehmittelpunkt des Schultergelenks und den Drehmittelpunkt für die Handgelenkdrehung geht.

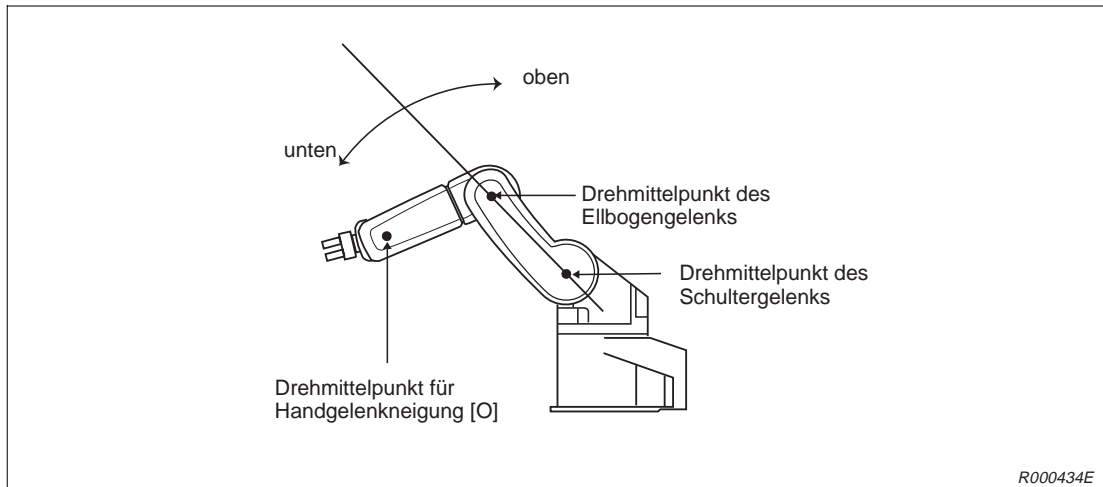


**Abb. A-2:** Kennungen für die Stellung am 6-achsigen Roboter: links / rechts (L/R)

### Stellungsmerker für Stellung: oben / unten (A/B)

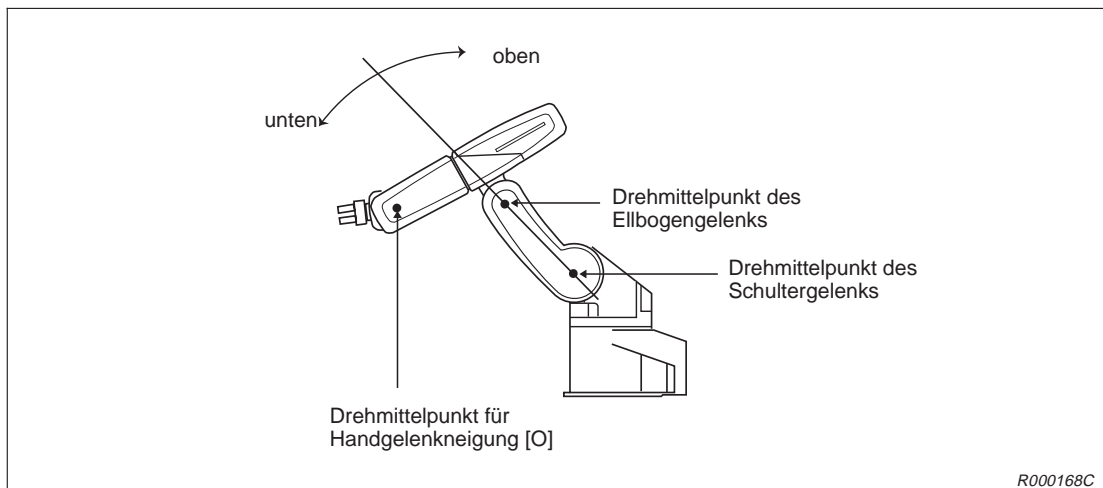
Dieser Merker zeigt die Position des Drehmittelpunktes [O] für die Handgelenkneigung in bezug zu einer Achsenlinie, die durch den Drehmittelpunkt des Ellbogengelenks und Schultergelenks geht.

#### 5 Achsen



**Abb. A-3:** Kennungen für die Stellung am 5-achsigen Roboter: oben / unten (A/B)

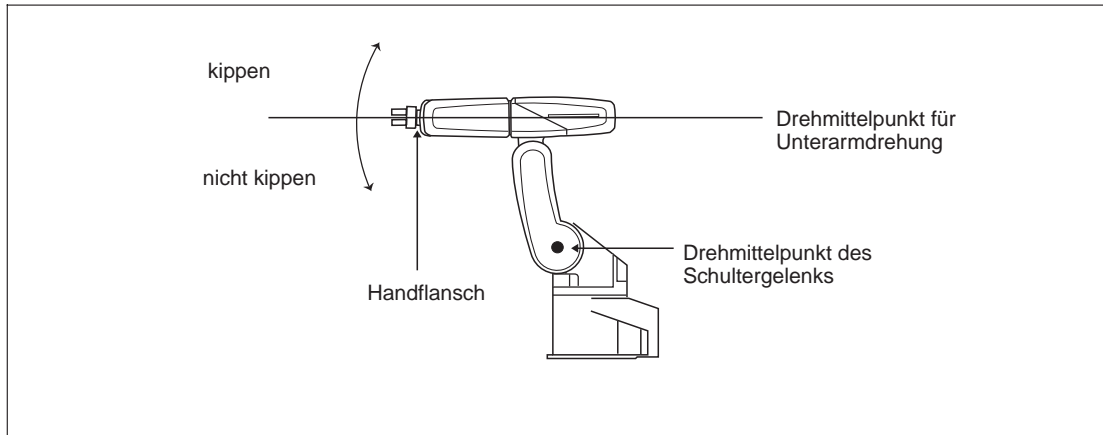
#### 6 Achsen



**Abb. A-4:** Kennungen für die Stellung am 6-achsigen Roboter: oben / unten (A/B)

**6 Achsen****Stellungsmerker für Stellung: kippen / nicht kippen (F/N)**

Dieser Merker zeigt die Position des Handflansches in Bezug zu zwei Achsenlinien, die einen rechten Winkel bilden. Eine Achsenlinie verläuft waagrecht zur Grundfläche und geht durch den Drehmittelpunkt [O] für die Handgelenkneigung. Die andere Achsenlinie verläuft senkrecht zur Grundfläche und geht durch den Drehmittelpunkt des Schultergelenks.



**Abb. A-5:** Kennungen für die Stellung am 6-achsigen Roboter: kippen/nicht kippen (F/N)

# INDEX

## !

( ' )-Befehl ..... 5-143

## A

ACCEL-Befehl ..... 8-84  
 ACL-Befehl ..... 8-4  
 ACT-Befehl ..... 8-6  
 ADD-Befehl ..... 5-11  
 ALIGN-Befehl ..... 8-8  
 alphanumerische Anzeige ..... 2-6  
 AN-Befehl ..... 5-12  
 Anweisung ..... 7-1  
   angehängte ..... 7-1  
 Arbeitsgegenstände  
   ergreifen ..... 5-145  
   palettieren ..... 5-149  
   plazieren ..... 5-145  
 Ausgabe ..... 5-95  
 Ausgabegeräte  
   anschließen ..... 5-155  
 Ausgänge  
   schalten ..... 5-93  
 Austauschbefehle ..... A-8

## B

BASE-Befehl ..... 8-9  
 BASIC-Befehle ..... 8-1  
 Batterie  
   anzeigen ..... 2-48  
 Befehle  
   direkt ausführen ..... 5-3  
   verarbeiten ..... 5-4  
 Betriebsgeschwindigkeit  
   einstellen ..... 5-128  
 Bewegung  
   für Teaching-Playback ..... 5-74  
 Bitstatus  
   direkt überprüfen ..... 5-135  
   überprüfen ..... 5-134  
 Bremsen ..... 2-44

## C

CALLP-Befehl ..... 8-11  
 CF-Befehl ..... 5-13  
 CHANGE-Befehl ..... 8-85  
 CL-Befehl ..... 5-16  
 CLOSE-Befehl ..... 8-12  
 CNT-Befehl ..... 8-13  
 COM OFF-Befehl ..... 8-16  
 COM ON-Befehl ..... 8-17  
 COM STOP-Befehl ..... 8-18  
 CP-Befehl ..... 5-18  
 CR-Befehl ..... 5-20

## D

DA-Befehl ..... 5-22  
 DACL-Befehl ..... 8-19  
 Daten  
   lesen ..... 5-28  
   übertragen ..... 5-109  
 Datentypen ..... 7-5  
 Datenwertvergleich: < ..... 5-126  
 Datenwertvergleich: = ..... 5-37  
 Datenwertvergleich: > ..... 5-57  
 Datenwertvergleich: ungleich ..... 5-88  
 Datum  
   einstellen ..... 2-49  
 Dauermodus ..... 2-28  
 DC-Befehl ..... 5-23  
 DEF ACT-Befehl ..... 8-21  
 DEF FN-Befehl ..... 8-22  
 DEF PLT-Befehl ..... 8-24  
 DEFINT-Befehl ..... 8-86  
 DEFIO-Befehl ..... 8-87  
 DEFJNT-Befehl ..... 8-88  
 DEFPOS-Befehl ..... 8-89  
 DELAY-Befehl ..... 8-90  
 DIM-Befehl ..... 8-26  
 DIV-Befehl ..... 5-24  
 DJ-Befehl ..... 5-25  
 DL-Befehl ..... 5-26  
 DLY-Befehl ..... 8-27  
 DP-Befehl ..... 5-27  
 DR-Befehl ..... 5-28  
 DRIVE-Befehl ..... 8-91  
 DS-Befehl ..... 5-30  
 DW-Befehl ..... 5-32

**E**

EA-Befehl .....	5-34
ED-Befehl .....	5-36
Ein-/Ausgabebefehle .....	A-7, A-14, A-17
Einführung .....	1-1
Eingabegeräte anschließen .....	5-155
Eingänge lesen .....	5-52
Einschaltzeit anzeigen .....	2-48
Einzel-Gelenkbewegung .....	5-25
Encoder zurücksetzen .....	2-46
END-Befehl .....	8-28
EQ-Befehl .....	5-37
ER-Befehl .....	5-39
Exklusiv-ODER-Verknüpfung .....	5-142
externe Geräte anschließen .....	5-155

**F**

Fehler lesen .....	5-39
Fehlerbedingung zurücksetzen .....	5-119
Fehlercodes Übersicht .....	A-23
Fehlermeldung quittieren .....	2-8, 2-51
Fehlermeldungen .....	2-40
Feldvariablen .....	7-16
FINE-Befehl .....	8-29
FOR-NEXT-Befehl .....	8-31
FPRM-Befehl .....	8-33
Funktionen .....	7-29

**G**

GC-Befehl .....	5-41
Gelenk-Jog-Betrieb .....	2-14
Gelenkbewegung .....	5-25, 5-65
Gelenkbremsen lösen .....	2-44
Gerätekonfiguration .....	2-2
Geschwindigkeit definieren .....	5-123
Geschwindigkeits-Übersteuerung .....	5-99
GF-Befehl .....	5-43
GO-Befehl .....	5-44
GOHOME-Befehl .....	8-92
GOSUB-Befehl .....	8-34

GOTO-Befehl .....	8-35
GRASP-Befehl .....	8-93
Grundposition .....	5-50
GS-Befehl .....	5-45
GT-Befehl .....	5-47

**H**

HALT-Befehl .....	8-94
HAND-Befehl .....	8-95
Handgreifer öffnen .....	5-44
öffnen/schließen .....	2-22
schließen .....	5-41
Handgreiferzustand festlegen .....	5-43
Handsteuerbefehle .....	A-6, A-14, A-17
HE-Befehl .....	5-48
HLT-Befehl .....	5-49, 8-36
HND-Befehl .....	8-37
HO-Befehl .....	5-50
HOLD-Befehl .....	8-96
HRE-Befehl .....	8-39

**I**

IC-Befehl .....	5-51
ID-Befehl .....	5-52
IF ... THEN ... ELSE-Befehl .....	8-40
IN-Befehl .....	8-41, 8-97
INP-Befehl .....	5-53
INPUT#-Befehl .....	8-42
INPUT-Befehl .....	8-98
Interrupt-Eingang festlegen .....	5-34
Interrupt-Freigabestatus .....	5-22
Interrupt-Möglichkeit sperrern .....	5-22
IOBLOCK-Befehl .....	8-99
IP-Befehl .....	5-55

**J**

Jog-Betrieb .....	2-14
Jog-Geschwindigkeit einstellen .....	2-20
JOVRD-Befehl .....	8-43
JRC-Befehl .....	5-56
JSPEED-Befehl .....	8-100

**K**

Kommentarbefehl .....	5-143
Kommunikationsbefehle .....	A-9
Kommunikationskanäle	
öffnen .....	5-97
Komponentendaten .....	7-24
Konstanten .....	7-6
Koordinatenaddition .....	5-61
Koordinatenposition	
anfahren .....	5-69
Kreis-Interpolation .....	5-76, 5-78

**L**

LABEL-Befehl .....	8-44
LG-Befehl .....	5-57
Logische Werte .....	7-23
LR-Befehl .....	5-59

**M**

MA-Befehl .....	5-61
Marken .....	7-2
MC-Befehl .....	5-63
Mehrfach-Gelenkbewegung .....	5-65
MELFA-BASIC III-Befehle	
Übersicht .....	8-2
Menüpunkt	
auswählen .....	2-25
MJ-Befehl .....	5-65
ML-Befehl .....	5-67
MO-Befehl .....	5-68
Monitor-Funktion	
Ausgangssignale .....	2-38
Eingangssignale .....	2-37
Variablen .....	2-39
MOV-Befehl .....	8-45
MOVE-Befehl .....	8-101
MOVEMASTER-Befehle	
Programm-Beispiele .....	5-144
Übersicht .....	5-8
Movement Position-Befehl .....	8-46
MP-Befehl .....	5-69
MPB-Befehl .....	5-71
MPC-Befehl .....	5-74
MR-Befehl .....	5-76
MRA-Befehl .....	5-78
MS-Befehl .....	5-80
MT-Befehl .....	5-82
MTS-Befehl .....	5-84
MUL-Befehl .....	5-86
MVC-Befehl .....	8-47
MVR-Befehl .....	8-49

MVR2-Befehl .....	8-51
MVS-Befehl .....	8-53

**N**

N-Befehl .....	5-87
NE-Befehl .....	5-88
NT-Befehl .....	5-90
Nullpunkt .....	5-50
anfahren .....	5-90, 5-96
NW-Befehl .....	5-91
NX-Befehl .....	5-92

**O**

OADL-Befehl .....	8-55
OB-Befehl .....	5-93
OC-Befehl .....	5-94
OD-Befehl .....	5-95
ODER-Verknüpfung .....	5-98
OG-Befehl .....	5-96
ON ... GOTO-Befehl .....	8-59
ON COM GOSUB-Befehl .....	8-57
ON-GOSUB-Befehl .....	8-58
OPEN-Befehl .....	8-60
Operationsbefehle .....	A-8
OPN-Befehl .....	5-97
OR-Befehl .....	5-98
ORG-Befehl .....	8-62
OUT-Befehl .....	8-63, 8-104
OVR-Befehl .....	5-99
OVRD-Befehl .....	8-65

**P**

PA-Befehl .....	5-100
Palettengitterpunkte	
definieren .....	5-100
Palettenkoordinaten	
berechnen .....	5-111
Parameter	
anzeigen/einstellen .....	2-41
für Teaching-Playback .....	5-71
Übersicht .....	A-19
Parameterwerte	
lesen .....	5-105
schreiben .....	5-106
PC-Befehl .....	5-101
PD-Befehl .....	5-102
PL-Befehl .....	5-104
PLT-Befehl .....	8-67
PMR-Befehl .....	5-105

PMW-Befehl .....	5-106
Position	
anfahren .....	5-68, 6-11
austauschen .....	5-114
definieren .....	5-102
kopieren .....	5-104
löschen .....	5-101
speichern .....	5-48
Positionierbefehle .....	A-1, A-11, A-16
Positionsdaten	
ändern .....	4-8, 6-9, 6-13
anfahren .....	4-8, 6-9
angleichen .....	4-8
editieren .....	4-8
eingeben .....	6-9
ersetzen .....	6-9
lesen .....	5-53, 5-107
löschen .....	4-8, 6-9, 6-14
Positionskoordinaten	
addieren .....	5-125
lesen .....	5-139
Positionsnummer .....	5-2
dekrementieren .....	5-27
inkrementieren .....	5-55
Positionsspeicher	
löschen .....	5-91
PR-Befehl .....	5-107
PRINT#-Befehl .....	8-68
PRINT-Befehl .....	8-105
PRN-Befehl .....	5-109
Programm	
auswählen .....	2-7, 5-87
editieren .....	4-4, 6-5
erstellen .....	4-2, 5-144, 6-2
kopieren .....	2-33
löschen .....	2-36
neustarten .....	2-7
Rücksprung zum Hauptprogram .....	5-120
schützen .....	2-32
starten .....	2-7, 2-27, 5-117
stoppen .....	2-7, 2-30, 5-49
testen .....	2-13, 5-7
unterbrechen .....	5-34
zurücksetzen .....	2-8, 5-119
Programme	
alle löschen .....	2-43
Programmebenen .....	7-34
Programmende .....	5-36
Programmiermethoden .....	1-5
Auswahl .....	3-3
Einteilung .....	3-1
Programmierung	
mit der Teaching Box .....	4-1
Schutzmaßnahmen .....	5-6
Programminformationen	
lesen .....	5-115
Programmname	
ändern .....	2-35

Programmschleife .....	5-116
beenden .....	5-92
Programmschritt	
lesen .....	5-131
Programmspeicher	
löschen .....	5-91
Programmsteuerbefehle .....	A-4, A-13, A-16
Programmverzeichnis	
anzeigen .....	2-31
Programmzeile .....	5-2
direkt aufrufen .....	4-9
lesen .....	5-59
löschen .....	5-26
Programmzeilensprung .....	5-47
PT-Befehl .....	5-111
PW-Befehl .....	5-113
PX-Befehl .....	5-114

## Q

QN-Befehl .....	5-115
-----------------	-------

## R

RC-Befehl .....	5-116
Register .....	5-3, 5-4
RELEASE-Befehl .....	8-106
REM-Befehl .....	8-70
Reservierte Wörter .....	7-35
RESET-Befehl .....	8-107
RETURN-Befehl .....	8-71
RN-Befehl .....	5-117
Roboter	
stoppen .....	5-49
Roboterbewegung	
kontinuierliche .....	5-63
relative geradlinige .....	5-30
stoppen .....	2-7
unterbrechen .....	5-147
Roboterbewegung	
relative .....	5-32
Roboter gelenke	
Bewegungsrichtungen .....	2-14
Roboterstellung .....	5-13
RS-Befehl .....	5-119
RS232C-Schnittstelle .....	2-4
RS422-Schnittstelle .....	2-4
RT-Befehl .....	5-120



**S**

SC-Befehl .....	5-121
SD-Befehl .....	5-123
Servospannung	
ein-/auschalten .....	2-26
SET-Befehl .....	8-108
SF-Befehl .....	5-125
Sicherheitshinweise	
Grundlegende .....	1-1
SKIP-Befehl .....	8-72
SLIM-Befehle	
Übersicht .....	8-83
SM-Befehl .....	5-126
Software-Version	
lesen .....	5-138
SP-Befehl .....	5-128
SPD-Befehl .....	8-73
SPEED-Befehl .....	8-109
Stellungsdaten	
ändern .....	5-13
Stellungsmerker .....	A-30
Steuergerät	
alphanumerische Anzeige .....	2-6
Gerätebeschreibung .....	2-3
Gerätfunktionen .....	2-7
STOP-Befehl .....	8-74
Störungssuche .....	A-28
STR-Befehl .....	5-131
SUB-Befehl .....	5-133
SUBSTITUTE-Befehl .....	8-82
Substitutionsbefehle .....	A-8
SV-Befehl .....	8-75

**T**

TB-Befehl .....	5-134
TBD-Befehl .....	5-135
Teaching Box	
Bedienung .....	2-14
Gerätebeschreibung .....	2-9
Menübaum .....	2-24
TI-Befehl .....	5-136
TL-Befehl .....	5-137
TOOL-Befehl .....	8-77

**U**

Übersteuerung .....	5-99
Uhrzeit/Datum	
einstellen .....	2-49
UND-Verknüpfung .....	5-12
Unterprogrammsprung .....	5-45

**V**

Variablen .....	7-10
Verfahrenbefehle .....	A-1, A-11, A-16
VR-Befehl .....	5-138

**W**

WAIT-Befehl .....	8-110
Warteimpulse .....	5-113
Werkzeug-Jog-Betrieb .....	2-18
Werkzeug-Koordinatensystem .....	2-18
Werkzeugbewegung	
geradlinige .....	5-84
mit Gelenk-Interpolation .....	5-82
Werkzeuglänge	
einstellen .....	5-137
lesen .....	5-141
WH-Befehl .....	5-139
WHILE ~ WEND-Befehl .....	8-78
WT-Befehl .....	5-141
WTH-Befehl .....	8-80
WTHIF-Befehl .....	8-81

**X**

XO-Befehl .....	5-142
XYZ-Jog-Betrieb .....	2-16
XYZ-Koordinatensystem .....	2-16

**Z**

Zähler .....	5-3
laden .....	5-16
Zählernummer .....	5-3
Zählerwert	
ausgeben .....	5-94
dekrementieren .....	5-23
einstellen .....	5-121
inkrementieren .....	5-51
lesen .....	5-20, 5-53
vergleichen .....	5-18
Zeichenkettennummer .....	5-3
Zeichenkettenregister .....	5-3
Zeichentypen .....	7-3
Zeilen .....	7-2
Zeilennummern .....	7-2
Zeitglied .....	5-136
Zusatzfunktionen .....	A-10, A-15, A-18



Headquarters		Europäische Vertretungen		Verkaufsbüros Deutschland	
MITSUBISHI ELECTRIC EUROPE B.V. Gothaer Str. 8 <b>D-40880 Ratingen</b> <b>DEUTSCHLAND</b> Telefon: +49 (0) 2102/486-0 Fax: +49 (0) 2102/486-112	EUROPA	N.V. GETRONICS Belgium S.A. Pontbeeklaan 43 <b>B-1731 Asse-Zellik</b> Telefon +32 (0) 2 / 467 17 51 Fax: +32 (0) 2 / 467 17 45	BELGIEN	INEA d.o.o. Ljubljanska 80 <b>SI-61230 Domzale</b> Telefon +386 (0) 17 21 80 00 Fax: +386 (0) 17 24 16 72	SLOWENIEN
	ITALIEN	louis poulsen Geminivej 32 <b>DK-2670 Greve</b> Telefon +45 (0) 43 / 95 95 95 Fax: +45 (0) 43 / 95 95 91	DÄNEMARK	AutoCont Control Systems s.r.o. Nemocnicni 12 <b>CZ-702 00 Ostrava 2</b> Telefon +420 (0) 69 / 615 21 11 Fax: +420 (0) 69 / 615 21 12	TSCHECHIEN
	JAPAN	Beijer Electronics OY Elannontie 5 <b>FIN-01510 Vantaa</b> Telefon +358 (0) 9 / 615 20 11 Fax: +358 (0) 9 / 615 20 500	FINNLAND	GTS Darülaceze Cad. No. 43A KAT: 2 <b>TR-80270 Okmeydani-Istanbul</b> Telefon +90 (0) 212 / 320 1640 Fax: +90 (0) 212 / 320 1649	TÜRKEI
MITSUBISHI ELECTRIC CORPORATION Mitsubishi Denki Bldg., 2-2-3 Marunouchi <b>Tokyo 100-8310</b> <b>JAPAN</b> Telefon: +81 (0) 3/32 18 31 76 Fax: +81 (0) 3/32 18 24 22		IP Systèmes 8, Rue du Colonel Chambonnet <b>F-69672 Lyon Bron Cedex</b> Telefon +33 (0) 4 / 72 14 18 00 Fax: +33 (0) 4 / 72 14 18 01	FRANKREICH	MITSUBISHI ELECTRIC EUROPE B.V. Brunnenweg 7 <b>D-64331 Weiterstadt</b> Telefon: +49 (0) 6150 / 13 99 0 Fax: +49 (0) 6150 / 13 99 99	
	SPANIEN	MITSUBISHI ELECTRIC EUROPE B.V. Westgate Business Park, Ballymount <b>IRL-Dublin 24</b> Telefon +353 (0) 1 / 419 88 00 Fax: +353 (0) 1 / 419 88 90	IRLAND	MITSUBISHI ELECTRIC EUROPE B.V. Kurze Straße 40 <b>D-70794 Filderstadt</b> Telefon: +49 (0) 711 / 77 05 98 0 Fax: +49 (0) 711 / 77 05 98 79	
MITSUBISHI ELECTRIC EUROPE B.V. Pol. Ind. Can Magi-C. Joan Buscallá, 2-4 AC 420 <b>E-08190 Sant Cugat del Vallés (Barcelona)</b> <b>SPANIEN</b> Telefon: +34 9 3/565 31 60 Fax: +34 9 3/589 15 79		ILAN & GAVISH Automation Service 24 Shenkar St., Qiryat-Arie 49513 <b>IL-Petach-Tikva 49001</b> Telefon +972 (0) 3 / 922 18 24 Fax: +972 (0) 3 / 972 39 24 07 61	ISRAEL	MITSUBISHI ELECTRIC EUROPE B.V. Am Söldnermoos 8 <b>D-85399 Hallbergmoos</b> Telefon: +49 (0) 811 / 99 87 40 Fax: +49 (0) 811 / 998 74 10	
MITSUBISHI ELECTRIC EUROPE B.V. Travellers Lane <b>GB-Hatfield Herts. AL10 8 XB</b> <b>ENGLAND</b> Telefon: +44 (0) 1707/27 61 00 Fax: +44 (0) 1707/27 86 95	UK	Getronics Industrial Automation bv Donauweg 10 <b>NL-1043 AJ Amsterdam</b> Telefon +31 (0) 20 / 586 15 92 Fax: +31 (0) 20 / 586 19 27	NIEDERLANDE	MITSUBISHI ELECTRIC EUROPE B.V. Eibacher Schulstraße 37 <b>D-90451 Nürnberg</b> Telefon: +49 (0) 911 / 64 64 66 Fax: +49 (0) 911 / 64 94 80 0	
	USA	Beijer Electronics AS Teglværksveien 1 <b>N-3002 Drammen</b> Telefon +47 (0) 32 / 24 30 00 Fax: +47 (0) 32 / 84 85 77	NORWEGEN		
MITSUBISHI ELECTRIC AUTOMATION INC. 500, Corporate Woods Parkway <b>Vernon Hills, Illinois 60061</b> <b>USA</b> Telefon: +1 (0) 847/478 21 00 Fax: +1 (0) 847 / 478 22 83		GEVA Wiener Straße 89 <b>A-2500 Baden</b> Telefon +43 (0) 2252 / 85 55 20 Fax: +43 (0) 2252 / 488 60	ÖSTERREICH		
		MPL Technology Sp. z o.o. ul. Wroclawska 53 <b>PL-30011 Kraków</b> Telefon +48 (0) 12 / 632 28 85 Fax: +48 (0) 12 / 632 47 82	POLEN		
		Mitsubishi Electric Europe B.V. 12/1 Goncharnaya St, suite 3C <b>RUS-109240 Moscow</b> Telefon +7 (0) 95 / 915-8624/02 Fax: +7 (0) 95 / 915-8603	RUSSLAND		
		Beijer Electronics AB Box 325 <b>S-20123 Malmö</b> Telefon +46 (0) 40 / 35 86 00 Fax: +46 (0) 40 / 93 23 01	SCHWEDEN		
		ECONOTEC AG Postfach 282 <b>CH-8309 Nürensdorf</b> Telefon +41 (0) 1 / 838 48 11 Fax: +41 (0) 1 / 838 48 12	SCHWEIZ		