EPSON

Referenzhandbuch

SPEL III Version 6.1

Rev. 1

Garantie

Das Robotersystem sowie alle Optionen werden vor Versand an den Kunden sehr stengen Qualitätskontrollen, Tests und Untersuchungen unterzogen, um sicherzustellen, daß das System in einwandfreiem Zustand ist und unseren hohen Leistungsanforderungen genügt.

Alle Schäden bzw. Fehlfunktionen am Robotersystem, die trotz normaler Betriebsbedingungen und Handhabung entstanden sind, werden innerhalb der ersten 12 Monate nach Auslieferung kostenlos repariert. Ausgenommen davon sind jedoch die folgenden Fälle:

- Schäden oder Fehlfunktionen, die durch nachlässige Bedienung oder Bedienvorgänge verursacht wurden, die nicht oder anders in diesem Handbuch beschrieben sind.
- Unerlaubte Modifikationen oder Demontage.
- Schäden oder Fehlfunktionen, die durch unerlaubte Einstellungen oder Reparaturen verursacht wurden.
- Schäden oder Fehlfunktionen, die durch externe, roboterunabhängige Ursachen entstanden, wie z.B. Brand- oder Wasserschaden.

Service-Center

Wenn Reparaturen, Wartungsmaßnahmen oder Neueinstellungen notwendig werden, wenden Sie sich bitte an Ihr EPSON Service-Center. Halten Sie dabei die Informationen zur Produktbezeichnung, Seriennummer (M.CODE), Software-Version sowie eine kurze Problembeschreibung bereit.

EPSON Deutschland GmbH
Abteilung Roboter
Zülpicher Straße 6
40549 Düsseldorf
Tel.: (0211) 55603-0

Diese Garantiebestimmungen gelten jedoch nicht in allen Ländern im vollen Umfang. Genauere Informationen zum Haftungsumfang erhalten Sie bei Ihrem Seiko Epson Fachhändler, bei dem Sie das Robotersystem erworben haben.

Hersteller

SEIKO EPSON CORPORATION
Robots & FA System Department

Okaya Plant No. 2
1-16-15, Daiei-cho
Okaya-shi, Nagano-ken, 394
Japan

Tel.: 81-266-23-0020 (Zentrale)
81-266-24-2004 (Direktwahl)
Fax: 81-266-24-2017

Einführung

In diesem Referenzhandbuch finden Sie alle Informationen, die zur korrekten Anwendung der Roboter-Programmiersprache SPEL III notwendig sind. Daher sollten Sie vor Inbetriebnahme des Robotersystems sowohl dieses Referenzhandbuch als auch das mitgelieferte Bedienungshandbuch durchlesen, um so Ihre Arbeit mit dem Montageroboter problemlos zu optimieren.

Das vorliegende Referenzhandbuch umfaßt die folgenden Abschnitte:

- ❏ Wichtiges (lesen Sie diesen Abschnitt zuerst)
- ❏ Aufbau des Handbuchs
- ❏ Zusammenfassung aller Befehle, Anweisungen und Funktionen
- ❏ Erläuterungen zu den Befehlen, Anweisungen und Funktionen (in alphabetischer Reihenfolge)
- ❏ Fehlermeldungen

Dieses Referenzhandbuch beschreibt die Version 6.1 von SPEL III. Die Software-Version entspricht jeweils dem verwendeten Modell der Robotersteuerung:

SRC-300	Version 4.2
SRC-310	Version 5.2
SRC-310A, SRC-320	Version 6.1

Wichtiges

Im Vergleich zur Version 3 von SPEL III wurden in der erweiterten Version 6.1 wichtige Änderungen vorgenommen, die sowohl die Programmierung als auch die Bedienung betreffen. Bevor Sie also mit dem Programmieren beginnen, sollten Sie sich die folgenden Informationen zu den Änderungen sorgfältig durchlesen:

Einschalten der Motoren

Motoren werden nicht automatisch eingeschaltet, wenn die Steuerung eingeschaltet oder zurückgesetzt wird. Zum Einschalten der Motoren haben Sie folgende Möglichkeiten:

- Führen Sie über die Programmiereinheit den Befehl MOTOR ON aus
- Drücken Sie an der Bedieneinheit den Schalter MOTOR ON
- Senden Sie an den Anschluß REMOTE3 das Signal zum Einschalten der Motoren
- Fügen Sie den Befehl MOTOR ON in das Ablaufprogramm ein

Kalibrierung

Die Version 6 von SPEL unterstützt sowohl sogenannte INC-Roboter, die mit einem inkrementellen Encoder ausgerüstet sind sowie ABS-Roboter, die mit einem absoluten Encoder ausgerüstet sind.

Bei einem INC-Roboter muß nach dem Einschalten des Roboters eine Kalibrierung mit Hilfe des Befehls MCAL durchgeführt werden. Bei einem ABS-Roboter ist eine Kalibrierung nicht erforderlich.

Befehl HOME

Die Funktion HOME in der Version 6.1 dient dazu, den Manipulator in eine HOME-Position (oder Standby-Position) zu bringen, die mit Hilfe des Befehls HOMESET festgelegt wird. Als HOME-Position können Sie je nach Bedarf einen beliebigen Punkt wählen.

Bitte beachten Sie, daß bei Auslieferung des Systems keine HOME-Position (Standby-Position) definiert ist. Wenn Sie also den Befehl HOME ausführen lassen, ohne zuvor eine HOME-Position zu definieren, wird eine Fehlermeldung angezeigt.

Bewegungsgeschwindigkeit im TEACH-Modus

Unabhängig von den Einstellungen, die Sie über den Befehl SPEED gewählt haben, verfährt der Manipulatorarm im TEACH-Modus nur mit langsamer Geschwindigkeit. Diese Maßnahme dient der Erhöhung Ihrer persönlichen Sicherheit. Nähere Informationen dazu finden Sie in der Beschreibung der Befehle POWER bzw. LP und SPEED.

Aufbau des Handbuchs

In diesem Handbuch wird jeder Befehl des Programms SPEL III nach folgendem Schema beschrieben:

Aufbau der Befehlserläuterungen

BEFEHL, ANWEISUNG oder FUNKTION	SYMBOLE
FUNKTION	
FORMAT	
BESCHREIBUNG	
VERWANDTE BEFEHLE	
BEISPIEL	

FORMAT

Das Format aller Befehle wird in diesem Handbuch wie folgt erläutert:

Zeichen, die nicht in eckigen Klammern [ ] stehen	Diese Zeichen sollten genau wie dargestellt eingegeben werden, z.B. MOTOR ON, #endif, AOPEN "[Dateiname]" AS #[Dateinummer]
[ ]	Datentyp. Datenelemente sind in eckige Klammern [ ] gesetzt, z.B. EIN [Ausgang]
| |	Jeder der so gekennzeichneten Mehrfachwerte kann ausgewählt werden, z.B. LOAD { | PRG | } | PNT |
{ }	Angaben in geschweiften Klammern { } können ausgelassen werden. Genauere Hinweise zu den Auslassungsmöglichkeiten finden Sie in der Beschreibung des jeweiligen Befehls. Beispiel: LOAD { | PRG | } | PNT | In diesem Fall kann jede der folgenden drei Möglichkeiten verwendet werden: LOAD LOAD PRG LOAD PNT
{ }n	Wenn Sie für die Variable n eine ganze Zahl angeben, werden die durch { } gekennzeichneten Daten n-mal geschrieben.Wird n nicht durch eine ganze Zahl ersetzt, können die gekennzeichneten Daten unendlich oft geschrieben werden.
~	Zeigt die Fortsetzung zur nächsten oder von der vorhergehenden Zeile an.

Symbole

In diesem Handbuch werden die folgenden Symbole verwendet.

Kann als Befehl verwendet werden

Kann als Anweisung verwendet werden

Funktion

Bezieht sich auf INC-Robober, also Roboter die mit einem inkrementellen Encoder ausgerüstet sind. Wird ein so gekennzeichneter Befehl bei einem ABS-Roboter, also einem Roboter mit absolutem Encoder ausgeführt, erfolgt Fehlermeldung 123.

Zusammenfassung der Befehle, Anweisungen und Funktionen

Die folgende Übersicht listet alle Befehle, Anweisungen und Funktionen in SPEL III auf. Dabei sollten Sie besonders die Befehle, Anweisungen bzw. Funktionen beachten, die wie folgt gekennzeichnet sind:

★ Falls Sie zum ersten Mal mit einem EPSON-Roboter arbeiten, sollten Sie die Beschreibung unbedingt lesen
☆ Befehle, Anweisungen oder Funktionen in der Version 6.1, die neu oder geändert sind

Befehle, Anweisungen und Funktionen zur Systemverwaltung

☆ ★	POWER, LP	Aktivierung/Deaktivierung des Low-Power-Modus im TEACH-Modus
	CONSOLE, CNSOL	Definiert die im Auto-Modus zu verwendende Eingabeeinheit
★	RESET	Setzt die Robotersteuerung zurück
	SETENV	Definiert, löscht bzw. zeigt die Umgebungsvariable an
	FREE	Zeigt die noch verfügbare Speicherkapazität an
	RGSIZE	Definiert bzw. zeigt die Größe des Programmbereichs an
	PNTSIZE, PSIZE	Definiert bzw. zeigt die erlaubte Anzahl Positionsdaten an
	LIBSIZE, SIZE	Definiert bzw. zeigt die Anzahl verwendbarer Backup-Variablen und die dafür verfügbare Speicherkapazität an
	SYSINIT	Initialisiert den Hauptspeicher
	TON	Zeigt die Zeilennummern eines Programms an
	TOFF	Unterdrückt die Anzeige der Zeilennummern eines Programms
	TSTAT	Zeigt den Status einer Task an
☆	STAT ( )	Gibt den Status der Steuerung oder eines anderen an die RS-232C-Schnittstelle angeschlossenen Kontrollers aus
★	VER	Zeigt die Systemdaten an
	VERINIT	Initialisiert die Systemdaten
	MKVER	Erstellt ein Backup bestimmter Einstellungen im Dateienspeicher bzw. auf der Festplatte
	SETVER	Lädt die Daten, die mit dem Befehl MKVER in einer Datei gespeichert wurden, zurück in den entsprechenden Speicherbereich
	DATE	Setzt bzw. zeigt das aktuelle Datum an
	TIME	Setzt bzw. zeigt die aktuelle Uhrzeit an
	HOUR	Zeigt die Betriebsstunden der Steuerung an
	TIME ( )	Gibt die Betriebsstunden der Steuerung aus
	ERRHIST	Zeigt ein Fehlerprotokoll an
	LINEHIST	Zeigt ein Protokoll der Zzeilennummern an

	CALIB	Ersetzt den aktuellen Pulse-Wert für die Armposition durch die aktuellen CALPLS-Werte
	CALPLS	Definiert bzw. zeigt Positions- und Encoder-Pulse für die Kalibrierung an
	HOFS	Definiert den Offset-Pulse für die Softwarekorrektur des Nullpunktes oder zeigt diesen an
	!	Führt einen MS-DOS-Befehl aus

Befehle, Anweisungen und Funktionen zur Armsteuerung

	MCAL	Führt eine Maschninenkalibrierung aus
	MCORDR	Definiert die Reihenfolge der Achsenbewegung bei der Maschinenkalibrierung bzw. zeigt die aktuellen Werte an
	MCORG	Berechnet die Parameter für die Maschinenkalibrierung
	MCOFS	Definiert die Parameter für die Maschinenkalibrierung bzw. zeigt die aktuellen Parameter an
	HTEST	Zeigt die HTEST-Werte an
★	MOTOR	Schaltet die Stromzufuhr zu den Motoren ein bzw. aus
★	SFREE	Schaltet die Motoren frei
★	SLOCK	Schaltet eine Achse wieder ein (aus dem "Servo-free"-Zustand)
★	JUMP	Steuert eine PTP-Bewegung kombiniert mit einer Gate-Bewegung
	ARCH	Definiert die Bogenparameter für den JUMP-Befehl bzw. zeigt sie an
★	LIMZ	Definiert die Z-Achsenhöhe für einen JUMP-Befehl bzw. zeigt den aktuellen Wert an
	SENSE	Definiert und zeigt die Bedingungen an, die zu einem Stop einer mit SENSE gekennzeichneten JUMP-Bewegung (JUMP SENSE) über dem Zielpunkt führt
	JS(0)	Gibt an, ob eine SENSE-Bedingung nach Ausführung einer JUMP-SENSE-Bewegung erfüllt oder nicht erfüllt wurde
★	GO	Führt eine gleichzeitige PTP-Bewegung aller vier Achsen aus
★	PASS	Führt eine PTP-Bewegung aus und bewegt den Manipulatorarm dabei an festgelegten Punkte vorbei (Verschleifen von Hilfspunkten, ohne diese exakt zu überfahren)
	PULSE	Führt eine gleichzeitige PTP-Bewegung aller vier Achsen aus
	TGO	Führt eine PTP-Bewegung im gewählten Werkzeug-Koordinatensystem aus
	TMOVE	Führt im gewählten Werkzeug-Koordinatensystem eine linearinterpolierte Bewegung aus
	TILL	Definiert bzw. zeigt die Eingabebedingung an, durch die - wenn erfüllt - die aktuell ausgeführte Bewegung (gesteuert durch die Befehle JUMP, GO oder MOVE) beendet wird, indem der Manipulator abgebremst wird und in einer Zwischenposition stoppt
★	!...!	Verarbeitet Ein-/Ausgangsanweisungen parallel zur Ausführung von Bewegungsbefehlen
☆ ★	SPEED	Definiert bzw. zeigt die Geschwindigkeit für eine PTP-Bewegung an
★	TSPEED	Definiert bzw. zeigt die maximale Geschwindigkeit einer PTP-Bewegung im TEACH-Modus an

☆ ★	ACCEL	Definiert bzw. zeigt die Beschleunigung einer PTP-Bewegung an
★	WEIGHT	Definiert das Gewicht der Nutzlast (incl. Greifergewicht) und die Länge des 2. Arms bzw. zeigt die gewählten Parameter an
	ARC	Führt eine kreisinterpolierte Bewegung in horizontaler Ebene aus
	CARC	Führt eine kreisinterpolierte Bewegung in horizontaler Ebene, ohne Geschwindigkeitsverzögerung am Endpunkt aus
	MOVE	Führt eine linearinterpolierte Bewegung aller vier Achsen mit Geschwindigkeitsverzögerung aus und stoppt am Endpunkt
	CMOVE	Führt eine linearinterpolierte Bewegung aller vier Achsen ohne Geschwindigkeitsverzögerung am Endpunkt aus
	CURVE	Erstellt eine Datei zur freien CP-Steuerung einer Kurvenbewegung
	CVMOVE	Führt die mit CURVE definierten Daten als Bahnbewegung aus
☆	SPEEDS	Definiert bzw. zeigt die Geschwindigkeit für die CP-Bewegung an
☆	ACCELS	Definiert bzw. zeigt die Beschleunigung für die CP-Bewegung an
	TSPEEDS	Definiert bzw. zeigt die maximale Geschwindigkeit für eine CP-Bewegung im TEACH-Modus an
	HOME	Führt eine Bewegung zur Home-Position (Standby-Position) aus
	HOMESET	Definiert bzw. zeigt die Home-Position an
	HORDR	Definiert bzw. zeigt die Reihenfolge der Achsenbewegung bei Ausführung des HOME-Befehls an
★	PALET	Definiert bzw. zeigt definierte Paletten an
★	PALETn( )	Gibt die Position auf der angegebenen Palette entsprechend der festgelegten Nestnummer aus
★	SEL	Wählt und zeigt Einstellungen für Teach-In-Bewegung an
	SET	Definiert Werte für die Teach-In-Bewegungen
	FINE	Definiert den Bereich am Bewegungsendpunkt, in dem eine Bewegung als beendet erkannt wird bzw. zeigt diesen an
★	QP	Schaltet Modus für Schnellpause ein bzw. aus oder zeigt den aktuell eingeschalteten Modus an
	JRANGE	Definiert den zulässigen Verfahrbereich für die angegebene Achse in Pulsen
	RANGE	Definiert den zulässigen Verfahrbereich jeder Achse in Pulsen bzw. zeigt die derzeit zulässigen Bereiche an
	XYLIM	Definiert den zulässigen Verfahrbereich in X- und Y-Richtung im Koordinatensystem bzw. zeigt die Werte an
	CX(P) CY(P) CZ(P) CU(P)	Gibt den Koordinatenwert der X-, Y-, Z (= 3.)- bzw. U-Achse eines festgelegten Punktes an
	PLS( )	Gibt den Pulse-Wert der angegebenen Achse aus
	AGL( )	Gibt den Winkel für die ausgewählte Rotationsachse bzw. Position für die ausgewählte lineare Achse an

SELRB	Wählt ein Positionierungsgerät aus

Befehle und Anweisungen zum Editieren

★ NEW	Löscht das Quellprogramm im Quellprogrammbereich
★ CLEAR	Löscht Positionsdaten
★ LIST	Zeigt das Quellprogramm an
★ DELETE, DELET	Löscht Programmzeile(n)
★ RENUM	Führt eine Neunumerierung der Programmzeilen durch
★ PLIST, PLI	Zeigt Positionsdaten an
★ PDEL	Löscht die angegebenen Positionsdaten
★ Pn=Positionsfestlegung	Definiert einen Punkt
EDIT	Schaltet in den Editiermodus
FIND	Sucht nach einer Zeichenkette

Befehle, Anweisungen und Funktionen zur Verarbeitung der Ein- und Ausgänge

★ ON	Schaltet den angegebenen Ausgang ein
★ OFF	Schaltet eine Ausgang aus
OPORT( )	Gibt den Status eines Ausgangs aus
★ SW( )	Gibt den Status eines Eingangs aus
IN( )	Gibt den Status eines Eingangsports (8 Eingänge) als Binärwert aus
INBCD( )	Gibt den Status eines Eingangsports (8 Eingänge) als BCD-Code aus (Binär-Dezimal-Code)
OUT	Schaltet Ausgänge eines Ausgangsports (8 Ausgänge) entsprechend eines Binärwertes
OPBCD	Schaltet Ausgänge eines Ausgangsports (8 Ausgänge) entsprechend des BCD-Wertes
ON $	Schaltet einen Merker ein
OFF $	Schaltet einen Merker aus
SW($)	Gibt den Status eines Merkers aus
IN ($)	Gibt den Status eines Merker-Ports (8 Merker) aus
OUT $	Schaltet einen Merker-Port (8 Merker) entsprechend eines Binärwertes
ZEROFLG(0)	Gibt den Status des Merkers (0) vor der letzten Statusänderung aus
★ WAIT	Legt Timer-Intervalle fest, unterbricht die Programmausführung bis die angegebene Bedingung erfüllt ist
TMOUT	Definiert das Intervall für die Unterbrechung im WAIT-Befehl
TW(0)	Gibt den Status einer WAIT-Bedingung (Zeitüberschreitung) sowie das WAIT-Intervall aus
INPUT	Anforderung zur Dateneingabe über die Tastatur der Bedieneinheit

★	PRINT	Zeigt Daten auf dem Anzeigegerät der Bedieneinheit an
	LINE INPUT	Speichert eine Eingabezeile von der Bedieneinheit als Zeichenkettenvariable (String-Variable)
	CONFIG, CNFG	Definiert die Konfigurationsparameter für die RS-232C-Schnittstelle
	INPUT #	Dateneingabe von Kommunikationsschnittstelle RS-232C
	PRINT #	Datenausgabe an Kommunikationsschnittstelle RS-232C
	LINE INPUT #	Speichert eine Eingabezeile vom Kommunikationsanschluß als Zeichenkettenvariable
	LOF( )	Gibt die Anzahl der im RS-232C-Puffer gespeicherten Datenzeilen aus
	OPU PRINT	Gibt Zeichen an der Bedieneinheit aus
	DSW( )	Gibt den Eingabestatus der remoten Bedieneinheit als Dezimalwert aus
	PEEK( )	Liest Daten vom Ein-/Ausgabekanal
	POKE	Schreibt Daten an den Ein-/Ausgabekanal

Befehle und Anweisungen zur Änderung von Koordinaten und Koordinatensystemen

	ARM	Wählt oder zeigt eine Armnummer an
	ARMSET	Definiert oder zeigt einen zusätzlichen Manipulatorarm an
	TOOL	Wählt oder zeigt ein Werkzeug-Koordinatensystem an
	TLSET	Definiert oder zeigt ein Werkzeug-Koordinatensystem an
	LOCAL	Definiert oder zeigt ein lokales Koordinatensystem an
	LOCAL0	Definiert das Basis-Koordinatensystem des Arms Local0
	BASE	Definiert oder zeigt ein lokales Koordinatensystem an
	BASE 0	Definiert das Basis-Koordinatensystem des Arms Local0

Anweisungen und Funktionen zur Programmsteuerung

★	FUNCTION...FEND	Definiert Anfang und Ende einer Task
★	END	Beendet die Programmausführung
★	FOR...NEXT	Führt eine Reihe von Anweisungen mit der angegebenen Zahl von Wiederholungen in Form einer Schleife aus
★	GOSUB...RETURN	Verzweigt in ein Unterprogramm, führt es aus und verläßt es danach wieder
★	GOTO	Geht zur angegebenen Zeilennummer oder zu einem Label
	CALL	Ruft eine Task als Unterprogramm auf
★	IF..THEN..ELSE..ENDIF	Führt eine Anweisung mit Bedingungen aus
	SELECT...SEND	Definiert die Verzweigungsformel sowie die entsprechenden Anweisungssequenzen zum Verzweigen
	WHILE...WEND	Führt eine angegebene Anweisung aus, solange eine festgelegte Bedingung erfüllt ist
☆	TRAP	Definiert einen Interrupt-Vorgang
	ONERR...RETURN	Legt die Fehlerroutine bei Auftreten eines Fehlers fest
	ERA(0)	Gibt die Nummer der Achse aus, bei der ein Fehler aufgetreten ist

ERL(0)	Gibt die Zeilennummer aus, in der ein Fehler aufgetreten ist
ECLR	Löscht einen Fehlerstatus
ERR(0)	Gibt eine Fehlernummer aus
ERT(0)	Gibt die Nummer der Task aus,bei der ein Fehler aufgetreten ist
ERRMSG$( )	Gibt die zur angegebenen Fehlernummer gehörende Fehlermeldung aus

Befehle und Anweisungen zur Programmausführung

★	COMPILE, COM	Übersetzt ein Quellprogramm in ein ausführbares Programm
★	XQT	Führt ein Programm aus
★	PAUSE	Unterbricht kurzfristig die Programmausführung
	RUN	Kompiliert ein Quellprogramm und führt es aus
☆	HTASK	Definiert Programmtasks, die durch den Befehl PAUSE oder die Eingabe PAUSE kurzfristig unterbrochen werden
	HALT	Unterbricht kurzfristig die Ausführung einer laufenden Programmtask
	QUIT	Beendet laufende Tasks bzw. kurzfristig unterbrochene Tasks
	RESUME	Setzt die Ausführung einer durch HALT kurzfristig unterbrochenen Task fort
	MYTASK(0)	Gibt die Nummer der laufenden Task aus (mytask)
	CHAIN	Lädt Objektprogramme und Dateien mit Positionsdaten in den Hauptspeicher
	PRGNO	Legt fest, ob die Auswahl der Programmnummer an REMOTE 2 zyklisch (auf- bzw. absteigend) oder durch Eingabe eines Binärcodes erfolgt

Pseudo-Anweisungen

#define	Definiert eine ID-Zeichenkette, die durch eine angegebene Ersatzzeichenkette ersetzt wird
#ifdef...#endif	Bedingte Kompilierung
#ifndef...#endif	Bedingte Kompilierung
#include	Bezieht die angegebene Datei zur Kompilierung mit ein

Befehle und Anweisungen zur Dateiverwaltung

	FORMAT, FRMT	Formatiert den Dateienspeicher bzw. eine Diskette
	FILES	Zeigt Dateinamen und -größe der Dateien im Dateienspeicher bzw. auf der Diskette an
	WIDTH	Definiert die Anzahl Zeichen pro Zeile bzw. Zeilen pro Bildschirmseite am Monitor der Programmiereinheit
★	DIR	Zeigt den Inhalt des aktuellen Verzeichnisses an
	CHDIR, CD	Wechselt in ein Verzeichnis und zeigt das aktuelle Verzeichnis auf dem angegebenen Laufwerk an
	MKDIR, MD	Erstellt ein Unterverzeichnis
	RMDIR, RD	Entfernt (löscht) ein Unterverzeichnis

	RENDIR	Benennt ein Unterverzeichnis um
	PATH	Definiert, löscht bzw. zeigt den Pfad zur Ausführung einer Stapeldatei an
★	DLOAD, DLO	Lädt die angegebenen Dateien in den Hauptspeicher
★	DSAVE, DSA	Sichert Quellprogramm bzw. Positionsdaten im Dateienspeicher oder auf einer Diskette in den Hauptspeicher
	MERGE	Überträgt die Daten aus der Programmiereinheit in den Hauptspeicher und mischt sie mit den dortigen Daten
	DMERGE	Lädt die angegebene(n) Datei(en) in den Hauptspeicher und mischt sie mit dem Quellprogramm bzw. den Positionsdaten
	TYPE	Zeigt den Inhalt einer Datei an
★	KILL	Löscht Dateien
★	DEL, ERASE	Löscht Dateien
	COPY	Kopiert Dateien
	RENAME, REN, NAME	Benennt eine Datei um
	LINK	Verbindet zwei oder mehr Objektdateien
	AOPEN...CLOSE	Öffnet eine Datei zum Anfügen von Daten
	ROPEN...CLOSE	Öffnet eine Datei zum Einlesen von Daten
	WOPEN...CLOSE	Öffnet eine Datei zum Schreiben von Daten
	VLOAD	Lädt Variablendaten
	VSAVE	Sichert Variablendaten

Befehle und Anweisungen für Variablen

	SYS	Definiert Backup-Variablen
	LIBRARY	Zeigt die Backup-Variablen im Speicher an
	CLRLIB	Löscht Backup-Variablen
	ENTRY	Definiert globale Variablen
	EXTERN	Definiert die Referenz zu Variablen
	VARIABLE	Zeigt Variablen an

Funktionen zu numerischen Werten

	OPORT( )	Gibt den Status eines Ausgangs aus
★	SW( )	Gibt den Status eines Eingangs aus
	IN( )	Gibt den Status eines Eingangsports (8 Eingänge) als Binärwert aus
	INBCD( )	Gibt den Status eines Eingangsports (8 Eingänge) als BCD-Code aus (Binär-Dezimal-Code)
	SW($)	Gibt den Status eines Merkers aus
	IN ($)	Gibt den Status eines Merker-Ports (8 Merker) aus
	ZEROFLG(0)	Gibt den Status des Merkers (0) vor der letzten Statusänderung aus
	CTR( )	Gibt den Zählerstand eines Zählers aus

	CTRESET	Setzt einen Zähler zurück
	TMR( )	Timer-Funktion
	TMRESET	Setzt einen Timer zurück
	TIME( )	Gibt die Betriebsstunden der Robotersteuerung aus
	JS(0)	Gibt an, ob eine SENSE-Bedingung nach Ausführung einer JUMP-SENSE-Bewegung erfüllt oder nicht erfüllt wurde
★	PALETn( )	Gibt die Position auf der angegebenen Palette entsprechend der festgelegten Nestnummer aus
	CX(P) CY(P) CZ(P) CU(P)	Gibt den Koordinatenwert der X-, Y-, Z (= 3.)- bzw. U-Achse eines festgelegten Punktes an
	PLS( )	Gibt den Pulse-Wert der angegebenen Achse aus
	AGL( )	Gibt den Winkel für die ausgewählte Rotationsachse bzw. Position für die ausgewählte lineare Achse an
	ERA(0)	Gibt die Nummer der Achse aus, bei der ein Fehler aufgetreten ist
	ERL(0)	Gibt die Zeilennummer aus, in der ein Fehler aufgetreten ist
	ERR(0)	Gibt eine Fehlernummer aus
	ERT(0)	Gibt die Nummer der Task aus,bei der ein Fehler aufgetreten ist
	TW(0)	Gibt den Status einer WAIT-Bedingung sowie das WAIT-Zeitintervall aus.
	MYTASK(0)	Gibt die Nummer der laufenden Task aus (mytask)
	LOF( )	Gibt Anzahl der im RS-232C-Puffer gespeicherten Datenzeilen aus
	DSW( )	Gibt den Eingabestatus der remoten Bedieneinheit als Dezimalwert aus
	STAT ( )	Gibt den Status der Steuerung oder eines anderen an die RS-232C-Schnittstelle angeschlossenen Kontrollers aus
	SIN( )	Gibt den Sinuswert des angegebenen Winkels aus
	COS( )	Gibt den Kosinuswert des angegebenen Winkels aus
	TAN( )	Gibt den Tangenswert des angegebenen Winkels aus
	ATAN( )	Gibt den Arkustangens des angegebenen Wertes aus
	ATAN2( )	Gibt den Arkustangens von x/y aus
	SQR( )	Gibt die Quadratwurzel aus
	ABS( )	Gibt den Absolutwert aus
	SGN( )	Gibt das Vorzeichen des angegebenen numerischenWertes aus
	INT( )	Gibt die größte ganze Zahl aus, die kleiner oder gleich dem angegebenen Wert ist
	NOT( )	Gibt den invertierten Wert des ganzzahligen Bits aus
	LSHIFT( )	Verschiebt die numerischen Daten nach links
	RSHIFT( )	Verschiebt die numerischen Daten nach rechts

Anweisungen und Funktionen zu den Zeichenketten

STRING	Definiert eine Zeichenkettenvariable
ASC( )	Gibt den ASCII-Code (numerischer Wert) des ersten Zeichens der angegebenen Zeichenkette aus
CHR$( )	Gibt das Zeichen aus, das dem angegebenen ASCII-Code entspricht
LEFT$( )	Gibt die äußerst links gelegenen Zeichen der angegebenen Zeichenkette aus
MID$( )	Gibt die angegebene Anzahl von Zeichen ab der festgelegten Position in der angegebenen Zeichenkette aus
RIGHT$( )	Gibt die äußerst rechts gelegenen Zeichen der angegebenen Zeichenkette aus
LEN( )	Gibt die Anzahl Zeichen in der angegebenen Zeichenkette aus
SPACE$( )	Gibt eine Zeichenkette mit der angegebenen Anzahl von Leerzeichen aus
STR$( )	Gibt den angegebenen numerischen Wert als numerische Zeichenkette aus
VAL( )	Gibt den numerischen Wert der angegebenen numerischen Zeichenkette aus
ERRMSG$( )	Gibt die zur angegebenen Fehlernummer gehörende Fehlermeldung aus

Befehle zur Bedieneinheit

OPU PRINT	Gibt Zeichen an der Bedieneinheit aus
CHARSIZE	Definiert die Größe der an der Bedieneinheit angezeigten Zeichen
CLS	Löscht Zeichen
NORMAL	Definiert den invertierten Anzeigemodus an der Bedieneinheit
REVERSE	Schaltet den Modus für die invertierte Textanzeige für den angegebenen Bereich ein
CURSOR	Schaltet die Anzeige des Cursors (an der Bedieneinheit) ein bzw. aus
OPUNIT	Wählt den Betriebsmodus der Bedieneinheit aus
DSW( )	Gibt den Eingabestatus der remoten Bedieneinheit als Dezimalwert aus

Neue bzw. in dieser Version geänderte Befehle

Wenn Sie zuvor die Steuerungseinheit SRC-300 oder SRC-310 verwendet haben, sollten Sie diese Seite lesen.

1. Neue Befehle

POWER	Hat dieselbe Funktion wie der Befehl LP ON/OFF. Die Befehlsbezeichung wurde in POWER/LOW/HIGH geändert, um die Einstellung des Motorgeschwindigkeitsmodus leicht verständlich zu machen. Der Befehl LP ON/OFF ist ebenfalls verfügbar.

2. Ergänzte Funktionen/geänderte Befehle

ACCEL ACCELS SPEED SPEEDS	Die Anzeige des Grenzwertes bei Ausführung eines Bewegungsbefehls im Low-Power-Status wurde zum besseren Verständnis geändert.
STAT( )	Der Inhalt der Bits 19, 22 und 23 der Adresse 0 sowie von Bit 5 und 6 der Adresse 1 wurden ergänzt.
TRAP	Interrupt-Vorgang bei Schutzabschrankung geöffnet/schließen wurde hinzugefügt.

!

FUNKTION Führt einen MS-DOS-Befehl aus

FORMAT `![MS-DOS-Befehl]`

BESCHREIBUNG Führt einen MS-DOS-Befehl aus, während der SPEL-Editor aktiviert ist.

Nach Ausführung des MS-DOS-Befehls wird folgendes angezeigt:

Push any key

Nachdem Sie eine beliebige Taste gedrückt haben, wird der !-Befehl durch Ausgabe eines Prompts (einer Eingabeaufforderung) beendet.

Falls der COMMAND.COM-Befehlsprozessor zur Ausführung des angegebenen MS-DOS-Befehls erforderlich ist, sich jedoch nicht im aktuellen Verzeichnis des Computers befindet, tritt ein Fehler auf.

! ... !

> S

Parallelbearbeitung

FUNKTION Verarbeitet Ein-/Ausgangsanweisungen parallel zur Ausführung von Bewegungsbefehlen.

FORMAT `[Bewegungsbefehl] ![Parallel zu verarbeitende Anweisung]!`

Als Bewegungsbefehl gilt jeder Bewegungsbefehl mit Ausnahme von PASS oder CURVE.

BESCHREIBUNG Beginnt mit der Ausführung der in ! ...! stehenden Anweisungen gleichzeitig mit dem Beginn eines über einen Bewegungsbefehl festgelegten Verfahrwegs.

Die folgenden Anweisungen können als Parallelanweisungen verwendet werden und zwar entweder als einzelne Anweisung oder als Mehrfachanweisungen.

Dn	Legt das Timing für die Ausführung der Parallelbearbeitung fest. n ist eine reele Zahl zwischen 0 und 100. Die Ausführung einer Anweisung, die nach Dn folgt, beginnt, wenn n% des festgelegtenVerfahrwegs zurückgelegt wurden. Beim JUMP-Befehl ist die Bewegung der Z-Achse (vertikal) nicht Bestandteil des %-Verfahrwegs. In einer ! ... !-Anweisung kann Dn maximal 6mal verwendet werden.
ON/OFF n	Schaltet Ausgang Nr. n ein bzw. aus
ON/OFF $n	Schaltet Merker Nr. n ein bzw. aus
OUT p, d	Gibt Ausgabedaten d an Ausgangsport p aus
OUT $p, d	Gibt Ausgabedaten d an Merker-Port d aus
WAIT t	Verzögert die Ausführung der nächsten Anweisung um t Sekunden
WAIT SW(n)=i	Verzögert die Ausführung der nächsten Anweisung, bis der Eingang n im Zustand i ist (1=ein, 0=aus)
WAIT SW($n)=i	Verzögert die Ausführung der nächsten Anweisung, bis der Merker n im Zustand i ist (1=ein, 0=aus)
PRINT/INPUT	Gibt Daten an die Bedieneinheit aus bzw. von dort ein
PRINT #/ INPUT #	Gibt Daten an die Kommunikationsschnittstelle aus bzw. von dort ein

Bei Anweisungen zur Parallelbearbeitung ist die Definition von Zeitintervallen für ON (Ein) bzw. OFF (Aus) nicht erlaubt. Das heißt, der Versuch `JUMP !P1 ON 5,3!` zu kompilieren, würde Fehlercode 14 verursachen.

Die unten aufgeführten Befehle können mit einem RAIOC (Remote Access I/O Controller) verwendet werden. Näheres dazu lesen Sie im RAIOC-Handbuch.

ON #/OFF #, OUT #, ON #$/OFF #$, OUT #$

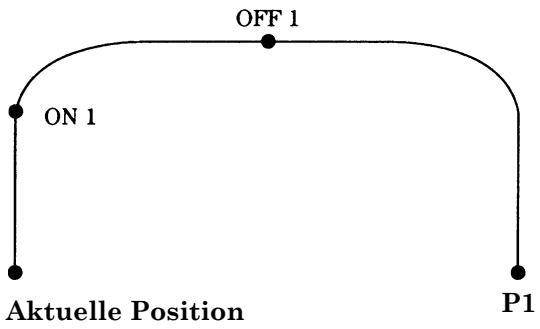
Falls die Ausführung der Parallelanweisungen nach Beendigung der Bewegungsbefehle und dem damit verbundenen Verfahrweg noch nicht abgeschlossen ist, wird die nachfolgende Programmausführung solange verzögert, bis alle Parallelanweisungen abgearbeitet und beendet sind.
Ähnliches gilt, wenn mit Hilfe von TILL an einer Zwischenposition des Verfahrwegs angehalten wurde. Auch hier wird die weitere Programmausführung solange verzögert, bis die Ausführung aller Parallelanweisungen abgeschlossen ist.
Auch beim JUMP-Befehl wird die Anweisung zu Beginn des ! ... !-Befehls (mit Ausnahme von Dn) gleichzeitig mit dem Beginn der (Hub-) Bewegung ausgeführt.
Sollen Anweisungen ausgeführt werden, nachdem die Hubbewegung der Z-Achse (=3. Achse) abgeschlossen ist, beginnen Sie die Anweisung mit D0 (Null).

VERWANDTE BEFEHLE JUMP, GO, MOVE, CMOVE, ARC, CARC, PULSE

BEISPIEL

```
JUMP P1 C0 LIMZ-45 SENSE !DO;ON 1;D50;OFF 1!
```

Parallel zur JUMP-Ausführung wird der Ausgang 1 gleichzeitig mit dem Beginn des JUMP-Verfahrwegs der ersten, zweiten und vierten Achse auf ON gesetzt. Ausgang 1 wird auf OFF geschaltet, wenn 50% des JUMP-Verfahrwegs beendet sind.




```
MOVE P1 !D10;ON 5;WAIT 0.5;OFF 5!
```

Parallel zur MOVE-Ausführung wird Ausgang 5 auf ON gesetzt, wenn 10% des MOVE-Verfahrwegs beendet sind. 0,5 Sekunden später wird Ausgang 5 wieder auf OFF gesetzt.

#define

S

FUNKTION

Definiert eine ID-Zeichenkette, die durch eine festgelegte Ersatzzeichenkette ausgetauscht werden soll.

FORMAT

```
#define [ID-Zeichenkette{([Parameter])}] {[Ersatzzeichenkette]}
```

BESCHREIBUNG

Durchsucht ein Programm nach der angegebenen ID-Zeichenkette, ersetzt diese an jeder gefundenen Stelle durch die Ersatzzeichenkette und kompiliert. Zwischen ID-Zeichenkette und der Ersatzzeichenkette muß mindestens ein Leerzeichen stehen.
Die festgelegte ID-Zeichenkette kann zum bedingten Kompilieren mit #ifdef- oder #ifndef-Befehlen verwendet werden. Parameter müssen durch runde Klammern ( ) eingeschlossen werden. Jeder Parameter muß durch ",", getrennt werden. Es dürfen maximal 8 Parameter angegeben werden. Wenn Parameter eingesetzt werden, kann dieser Befehl als Makro verwendet werden.

Hinweise zur ID-Zeichenkette

- Das erste Zeichen muß ein Alphazeichen, die übrigen könnenalphanumerisch oder Unterstriche (_) sein.
- Es sind maximal 8 Zeichen erlaubt. Bei Angabe von Parametern wird die erste "("-Zeichenkombination als ein Zeichen gezählt. Die weiteren Parameter sowie die letzte "("-Zeichenkombination werden nicht mitgezählt.
- Leerzeichen oder Tabulatoren sind nicht erlaubt.
- Wahlweise können Groß- und Kleinbuchstaben verwendet werden; sie werden entsprechend als Groß- bzw. Kleinbuchstaben erkannt.

Das bedeutet, die folgenden Zeichenketten werden als unterschiedliche ID-Zeichenketten erkannt und unterschieden:

XYZ, Xyz, xYZ

ID-Zeichenketten werden außerdem nach der Anzahl Zeichen erkannt und unterschieden. Beispielsweise wird "ABC" 5 Zeichen) als unterschiedlich zu ABC (3 Zeichen) erkannt und nicht ersetzt.

Hinweise zur Ersatzzeichenkette

- Leerzeichen und Tabulatoren sind nicht erlaubt.
- Argumente sind nicht erlaubt.
- ID-Zeichenketten, die mit anderen #define-Anweisungen verwendet werden, können nicht als Ersatzzeichenketten fungieren.
- Wird ein Kommentarzeichen ( ) eingebracht, werden die Zeichen nach dem ( )als Kommentar interpretiert und nicht in die Ersatzzeichenkette einbezogen.
- Die Ersatzzeichenkette ist nicht unbedingt erforderlich. In diesem Fall wird die angegebene ID-Zeichenkette durch nichts ersetzt, d.h., de facto wird die ID-Zeichenkette aus dem Programm gelöscht.

VERWANDTE BEFEHLE

#ifdef...#endif, #ifndef...#endif

BEISPIELE

Beispiel 1

Ohne Pseudo-Befehl #define	Mit Pseudo-Befehl #define
>10 FUNCTION MAIN	>10 FUNCTION MAIN
>20 INTEGER O_CYLF;O_CYLF=0	>20 #define O_CYLF 0
>30 INTEGER O_CYLR;O_CYLR=1	>30 #define O_CYLR 1
>40 INTEGER I_CYLF;I_CYLF=0	>40 #define I_CYLF 0
>50 INTEGER I_CYLR;I_CYLR=1	>50 #define I_CYLR 1
>60 INTEGER RDY; RDY=0	>60 #define RDY 0
.	.
.	.
.	.

Beispiel 2

```
100 #define EOF -1
110 PRINT EOF       -1 wird angezeigt
120 PRINT "EOF"     Die Zeichenkette EOF wrid angezeigt
130 PRINT EOF1      EOF ist nicht identisch mit EOF. In diesem Fall
                    wird der der Wert der Variablen EOF1 angezeigt
```

Beispiel 3

Die maximale Anzahl Zeichen pro Programmzeile beträgt 79. Jedoch kann bei Verwendung von #define zum Austausch von ID-Zeichenketten gegen Ersatzzeichenketten die Anzahl virtueller Zeichen pro Programmzeile auf 255 erhöht werden.

```
100 #define TEST 12345678901234567890123456789
110 PRINT TEST, TEST, TEST, TEST
```

Nach erfolgtem Austausch übersteigt die tatsächliche Anzahl dargestellter Zeichen die Summe von 80.

Beispiel 4

Verwendung des Befehls um die Zeichen an der Bedieneinheit OPU-300 als Makro anzeigen zu lassen.

```
100 #define CLS PRINT #24, CHR$ (&H1B)+"E?'"
110 #define LOCATE (a,b) CHR$(&H1B)+"Y"+CHR$(31+a)+CHR$(31+b)
120 #define BLKCLS (a,b) CHR$(&H1B)+"E"+CHR$(31+a)+CHR$(31+b)
130 CLS                                  Bildschirm löschen
140 PRINT #24, LOCATE(1,3)+BLKCLS(32,2)  2 Zeilen ab der 3. Zeile löschen
```

Der Kompiler von SPEL III überprüft ein Kundenprogramm zweimal. Pseudo-Befehle, Variablen und Label werden im ersten Durchgang verarbeitet, jede einzelne Zeile wird dann im zweiten Durchgang kompiliert. Das bedeutet, die Verwendung ist auf denselben Durchgang beschränkt. In den folgenden Fällen kann die Verwendung von #define zu einer Fehlermeldung führen:

- Registrierung einer Variablen und Label 1
- Verwendung einer ID-Zeichenkette, die durch den Befehl #define als Dateiname für den Befehl #include registriert ist.

LIST 1 bis LIST 4 geben Beispiele für solche Fehler.

LIST 1
```
10 FUNCTION MAIN
20 #define CHAR STRING
30 CHAR A$
40 PRINT A$
50 FEND
>COM
#ERROR 2 at 40
>
```

LIST 2
```
10 FUNCTION MAIN
20 #define LBL LABEL
30 GOTO LBL
40 LBL:
50 FEND
>COM
#ERROR 8 at 30
>
```

LIST 3
```
10 FUNCTION MAIN
20 #define LMAX 10
30 REAL LBUF (LMAX)
40 FEND
>COM
#ERROR 14 at 30
>
```

LIST 4
```
10 FUNCTION MAIN
20 #define ABC "TEST.PRG"
 30 include ABC
40 FEND
>COM
#ERROR 2 at 30
>
```

Wird die bereits registrierte ID-Zeichenkette jedoch in einer Ersatzzeichenkette verwendet, kann diese problemlos verarbeitet werden. Dies nennt man Verschachtelung von #define.

LIST 5
```
10 FUNCTION MAIN
20 #define ABC 10
30 #define DEF ABC
40 PRINT DEF
50 FEND
>RUN
 10
>
```

#ifdef...#endif
#ifndef...#endif

S

if define (falls definiert)
if not define (falls nicht definiert)

FUNKTION Bedingte Kompilierung

FORMAT

```
#ifdef [ID-Zeichenkette]
   .
   .
   .
#endif

#ifndef [ID-Zeichenkette]
   .
   .
   .
#endif
```

BESCHREIBUNG Werden diese Befehle zusammen mit Pseudo-Befehlen verwendet, führen #endif, #ifdef und #ifndef eine bedingte Kompilierung durch.

#ifdef überprüft zuerst, ob die angegebene ID-Zeichenkette derzeit durch #define definiert ist. Ist dies der Fall, werden die Anweisungen zwischen #ifdef und #endif kompiliert. Falls keine Definition durch #define gefunden wird, werden die Anweisungen zwischen #ifdef und #endif übersprungen und nicht kompiliert.

#ifndef überprüft die gegenteilige Bedingung zu #ifdef.
#ifndef überprüft zuerst, ob die angegebene ID-Zeichenkette derzeit durch #define definiert ist. Ist keine Definition durch #define vorhanden, werden die Anweisungen zwischen #ifndef und #endif kompiliert. Wird eine Definition durch #define gefunden, werden die Anweisungen zwischen #ifndef und #endif übersprungen und nicht kompiliert.

VERWANDTE BEFEHLE #define, COMPILE

BEISPIEL

```
   .
   .
   .
>100  INPUT #20,A$
>110  #ifdef DEBUG
>120    PRINT A$
>130  #endif
   .
   .
   .
```

Im obigen Beispiel erfolgt ein Ausdruck bzw. kein Ausdruck, abhängig davon, ob eine Definition #define DEBUG vorliegt oder nicht.
Das Beispiel in der Erläuterung zum Befehl #include kann durch die Verwendung von #ifdef und #ifndef folgendermaßen verändert werden:

VAL.PRG

```
>300 #ifdef VALUE
>310    ENTRY REAL WORK(100)
>320    ENTRY DOUBLE A_SIN, A_COS
>330 #endif
>340 #ifndef VALUE
>350    EXTERN REAL WORK(100)
>360    EXTERN DOUBLE A_SIN, A_COS
>370 #endif
```

FILE1

```
>10 FUNCTION MAIN
>20 #define VALUE
>30 #include "VAL.PRG"
>40 EXTERN FUNCTION INIT
>50 XQT !2, INIT
>60 FEND
```

Beachten Sie, daß FILE 2 und FILE 3, in denen die #include-Pseudo-Anweisung verwendet wird, identisch sind mit denen in den Erläuterungen zum #include-Befehl.

In FILE 1 wird VALUE durch die #define-Anweisung definiert. Dadurch daß in FILE 2 und FILE 3 VALUE nicht definiert ist, kann ausgewählt werden, welche Anweisung - #ifdef VALUE oder #ifndef VALUE - in jeder Datei ausgeführt wird. Der Vorteil dieser Methode besteht darin, daß beim Hinzufügen oder Löschen von Variablen die Änderung nur an einer Stelle vorgenommen werden muß, und zwar im Programm VAL.PRG.

Wird VALUE mit Hilfe der #define-Anweisung definiert, wird nur dieses Programmsegment zu FILE 1 hinzugefügt, da #ifdef...#endif kompiliert wird. Falls VALUE nicht definiert ist, wird #ifndef...#endif kompiliert und dieses Segment FILE 2 und FILE 3 hinzugefügt.

#include

S

(Einbeziehen)

FUNKTION Bezieht die angegebene Datei mit in die Kompilierung ein

FORMAT `#include "{[Laufwerk]}{[Pfad]}[Dateiname].PRG"`

Es sind bis zu 5 Verschachtelungen erlaubt.

BESCHREIBUNG Bezieht den Inhalt der angegebenen Quellprogrammdatei mit ein. #include erlaubt die Einbeziehung deklarierter globaler Variablen und #define-Definitionen aus anderen Dateien.

Die angegebenen Dateien sollten auf dem Laufwerk existieren.
Wird die Laufwerksbezeichnug ausgelassen, sucht #include auf dem aktuellen Laufwerk nach Dateien. Wird die Pfadangabe ausgelassen, sucht #include die Dateien im aktuellen Verzeichnis.

Eine Include-Datei kann eine sekundäre Include-Datei beinhalten. Beispielsweise kann FILE 2 in FILE 1 enthalten sein und FILE 3 in FILE 2. Dieser Vorgang wird als Verschachtelung bezeichnet. Es sind maximal 5 Schleifen (einschließlich der Originaldatei) erlaubt.

Es ist zwar erlaubt, daß die Zeilennummern der einzubeziehenden Datei mit denen des Originalprogramms identisch sind, sollte aber vermieden werden, da es sonst nicht mehr möglich ist, den aktuellen Stand der Programmausführung zu ermitteln.

BEISPIEL

Ohne Include-Datei	Mit Include-Datei
	INCLUDE FILE (VAL.PRG)
	>300 EXTERN REAL WORK(100)
	>310 EXTERN DOUBLE A_SIN,A_COS
FILE1 (MAIN.PRG)	**FILE1 (MAIN.PRG)**
>10 FUNCTION MAIN	>10 FUNCTION MAIN
>20 EXTERN FUNCTION INIT	>20 EXTERN FUNCTION INIT
>30 ENTRY REAL WORK(100)	>30 ENTRY REAL WORK(100)
>40 ENTRY DOUBLE A_SIN,A_COS	>40 ENTRY DOUBLE A_SIN, A_COS
>50 XQT !2 INIT	>50 XQT !2 INIT
>60 FEND	>60 FEND
FILE2 (INIT.PRG)	**FILE2 (INIT.PRG)**
>100 FUNCTION INIT	>100 FUNCTION INIT
>110 EXTERN REAL WORK(100)	>110 EXTERN REAL WORK(100)
>120 EXTERN DOUBLE A_SIN, A_COS	
>130 EXTERN FUNCTION DISP	>130 EXTERN FUNCTION DISP
.	.
.	.
.	.
>180 XQT !3 DISP	
>190 FEND	
FILE3 (PRI.PRG)	**FILE3 (PRI.PRG)**
>200 FUNCTION DISP	>200 FUNCTION DISP
>210 EXTERN REAL WORK(100)	>210 #include "VAL.PRG"
>220 EXTERN DOUBLE A_SIN, A_COS	.
.	.
.	.
.	
>290 FEND	

Wird keine Include-Datei verwendet, müssen für jede Datei, in der Variablen verwendet werden, die entsprechenden Variablen deklariert bzw.Variablennamen geändert und neue hinzugefügt werden.

So kann beispielsweise in dem vorherigen Beispiel durch Verwendung einer Include-Datei die Änderung einer Variablendefinition durch Änderung der Variablen an nur zwei Stellen (FILE1.PRG und VAL.PRG) durchgeführt werden. Dies ist besonders nützlich zur Steigerung der Programmeffizienz, besonders dann, wenn viele Tasks verwendet werden.

ABS( )

F

Absolut

FUNKTION Gibt den absoluten Wert aus.

FORMAT `ABS([numerischer Wert])`

BESCHREIBUNG Gibt den absoluten Wert des angegebenen numerischen Wertes aus.

VERWANDTE BEFEHLE SGN ( ), INT ( ), SQR ( )

BEISPIEL

```
>PRINT ABS(-3.54)
3.54
>
```

ACCEL

> S

Acceleration (Beschleunigung)

FUNKTION Definiert die Beschleunigung und die Verzögerung (in %) für eine PTP-Bewegung oder zeigt sie an.

FORMAT

(1) `ACCEL, A,B{,C,D,E,F}`

A: Beschleunigungswert
B: Wert für Verzögerung
C: Beschleunigungswert (Z-Achse = 3. Achse aufwärts)
D: Verzögerungswert (Z-Achse = 3. Achse aufwärts)
E: Beschleunigungswert (Z-Achse = 3. Achse abwärts)
F: Verzögerungswert (Z-Achse = 3. Achse abwärts)

Jeder der Werte A bis F muß eine ganze Zahl zwischen 1 und 100 sein (Prozentwert der maximalen Beschleunigung/Verzögerung).

Informationen zu den Standardwerten erhalten Sie im Handbuch zum Manipulatorarm (technische Daten).

(2) `ACCEL`

BESCHREIBUNG

(1) Definiert die Werte für die Beschleunigung bzw. Verzögerung bei einer PTP-Bewegung (Befehle GO, JUMP, PULSE und verwandte).

Jeder Wert muß eine ganze Zahl zwischen 1 und 100 sein.

Die Werte C, D, E und F der Z-Achse (= 3. Achse) gelten nur für den JUMP-Befehl. Werden die Werte für die Z-Achse ausgelassen, werden für C, D, E und F folgende Standardwerte eingesetzt:

C und E entspricht A (Beschleunigungswert)
D und F entspricht B (Verzögerung

(2) Zeigt die derzeit gültigen ACCEL-Werte wie folgt an:

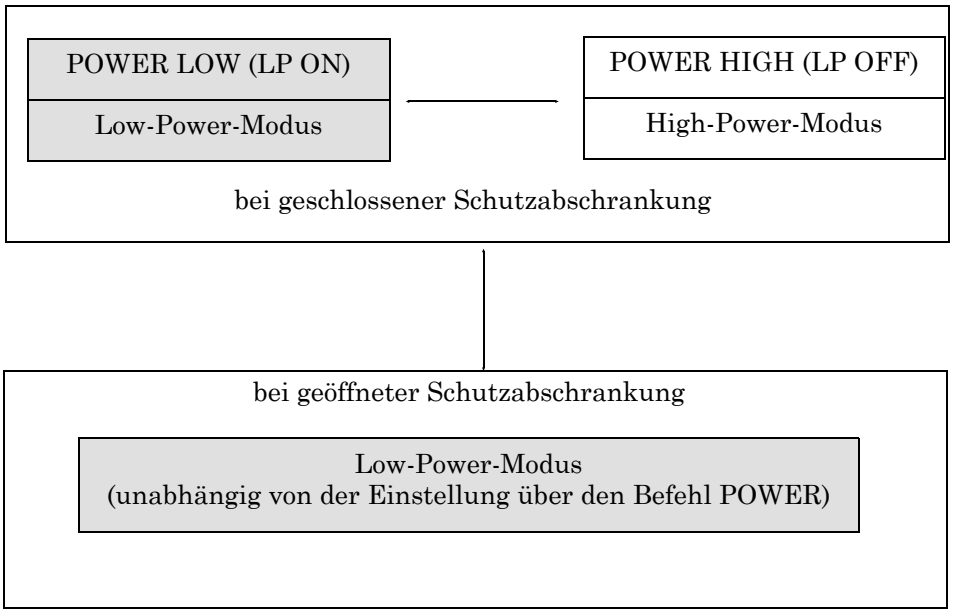
Beschl.-Wert	Verzög.-Wert
Beschl.-Wert (Z-Achse aufwärts)	Verzög.-Wert (Z-Achse aufwärts)
Beschl.-Wert (Z-Achse abwärts)	Verzög.-Wert (Z-Achse abwärts)

Der ACCEL-Wert wird auf die Standardwerte zurückgesetzt, wenn eine der folgenden Aktionen ausgeführt wird:

Einschalten der Stromzufuhr
Moduswechsel (TEACH/AUTO)
Software-Reset
Befehl MOTOR ON
Befehl SFREE, SLOCK
Befehl VERINIT
Drücken der Taste STOP
Drücken der Tastenkombination Strg + C

Im Teach-Modus ist die tatsächliche Beschleunigung im Low-Power-Modus anders als im High-Power-Modus.

Motor-Power-Status	Tatsächliche Beschleunigung
Low-Power-Status	Standardwert von ACCEL Wert von ACCEL } der niedrigere Wert
High-Power-Status	Wert von ACCEL



Soll mit einer höheren Geschwindigkeit gearbeitet werden, schalten Sie den High-Power-Modus über den Befehl POWER HIGH bzw. LP OFF ein und schließen Sie die Schutzabschrankung. Ist die Schutzabschrankung geöffnet, wird der Wert für die Beschleunigung auf den Standardwert gesetzt.

Wird der Befehl ACCEL ausgeführt, wenn sich der Roboter im Low-Power-Modus befindet, wird die folgende Meldung angezeigt. Bei der in der Meldung angezeigten Zahl handelt es sich um den Standardwert für den Befehl ACCEL; dieser kann je nach verwendetem Modell unterschiedlich sein. Obwohl bei dem folgenden Beispiel ein Beschleunigungswert von 100 eingestellt ist, wird sich der Roboter mit der Standardbeschleunigung von (10) bewegen, da er sich im Low-Power-Modus befindet.

```
>ACCEL 100,100
Low Power State : ACCEL ist begrenzt auf 10
>
>ACCEL
Low Power State : ACCEL ist begrenzt auf 10
     100        100
     100        100
     100        100
>
```

VERWANDTE BEFEHLE POWER (LP), SPEED, GO, JUMP, PASS, PULSE

BEISPIEL

```
>ACCEL 100,100,100,100,100,50        'Nur die Verzögerung für die
                                      Abwärtsbewegung der Z-Achse
                                      ist auf 50 gesetzt
>ACCEL
   100    100
   100    100
   100     50

>_
```

ACCELS

> S

Acceleration Straight (Tatsächliche Beschleunigung)

FUNKTION Definiert die Beschleunigung für eine CP-Bewegung bzw. zeigt den aktuellen Wert an.

FORMAT

(1) ACCELS [Beschleunigungswert]

Der Beschleunigungswert muß eine ganze Zahl zwischen 1 und 5000 (mm/s^2) sein.

Informationen zu den Standardwerten erhalten Sie im Handbuch zum Manipulatorarm (technische Daten).

(2) ACCELS

BESCHREIBUNG

(1) Definiert die Beschleunigung der Hand für eine CP-Bewegung in mm/s^2 (ARC, MOVE, CMOVE and verwandte Befehle).

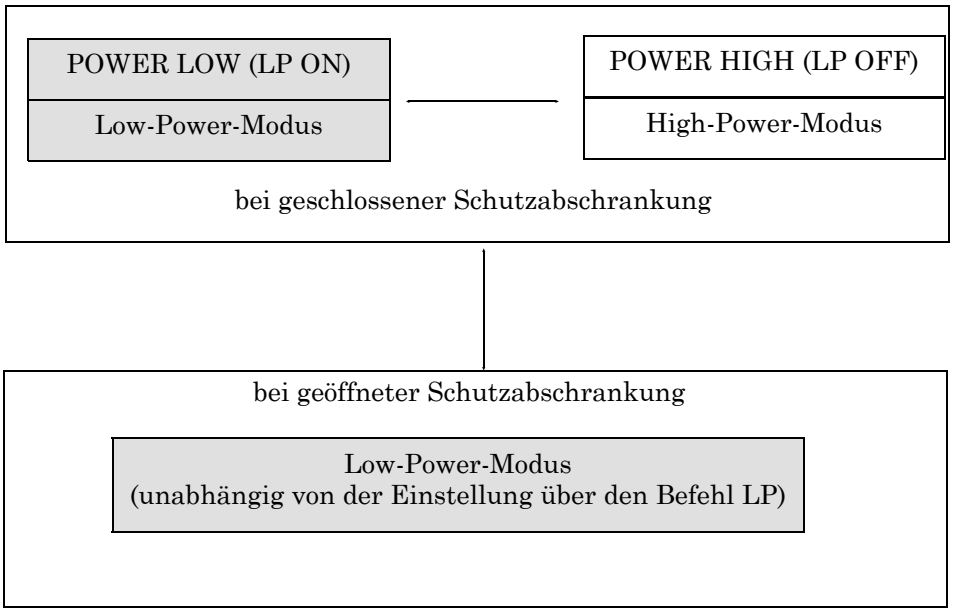
(2) Zeigt den derzeit gültigen ACCELS-Wert an.

Der ACCELS-Wert wird auf den Standardwert zurückgesetzt, wenn eine der folgenden Aktionen ausgeführt wird:

Einschalten der Stromzufuhr
Moduswechsel (TEACH/AUTO)
Software-Reset
Befehl MOTOR ON
Befehl SFREE, SLOCK
Befehl VERINIT
Drücken der Taste STOP
Drücken der Tastenkombination Strg + C

Im Teach-Modus ist die tatsächliche Beschleunigung im Low-Power-Modus anders als im High-Power-Modus.

Motor-Power-Status	Tatsächliche Beschleunigung
Low-Power-Status	Standardwert von ACCELS Wert von ACCELS } der niedrigere Wert
High-Power-Status	Wert von ACCELS



Soll mit einer höheren Geschwindigkeit gearbeitet werden, schalten Sie den High-Power-Modus über den Befehl POWER HIGH bzw. LP OFF ein und schließen Sie die Schutzabschrankung. Ist die Schutzabschrankung geöffnet, wird der Wert für die Beschleunigung auf den Standardwert gesetzt.

Wird der Befehl ACCELS ausgeführt, wenn sich der Roboter im Low-Power-Modus befindet, wird die folgende Meldung angezeigt. Bei der in der Meldung angezeigten Zahl handelt es sich um den Standardwert für den Befehl ACCELS; dieser kann je nach verwendetem Modell unterschiedlich sein. Obwohl bei dem folgenden Beispiel ein Beschleunigungswert von 1000 eingestellt ist, wird sich der Roboter mit der Standardbeschleunigung von (200) bewegen, da er sich im Low-Power-Modus befindet.

```
>ACCELS 1000
Low Power State : ACCELS ist begrenzt auf 200
>
>ACCEL
Low Power State : ACCELS ist begrenzt auf 200

    1000

>
```

VERWANDTE BEFEHLE POWER (LP), LP, SPEEDS, MOVE, CMOVE, ARC, CARC, CMOVE

BEISPIEL

```
>ACCELS 1000
>ACCELS
      1000

>
```

AGL ( )

F

Angle (Winkel)

FUNKTION Gibt den Winkel für die ausgewählte Rotationsachse bzw. die Position für die ausgewählte lineare Achse an.

FORMAT `AGL([Achsennummer])`

Die Achsennummer muß eine ganze Zahl zwischen 1 und 4 sein.

BESCHREIBUNG Ist die ausgewählte Achse eine Rotationsachse, wird der aktuelle Winkel, gemessen von der 0-Pulse-Position der ausgewählten Achse, in der Einheit Grad ausgegeben. Der ausgegebene Wert ist eine reele Zahl.

Ist die ausgewählte Achse eine lineare Achse, wird die aktuelle Position in Relation zur 0-Pulse-Position der ausgewählten Achse, in der Einheit mm ausgegeben. Der ausgegebene Wert ist eine reele Zahl.

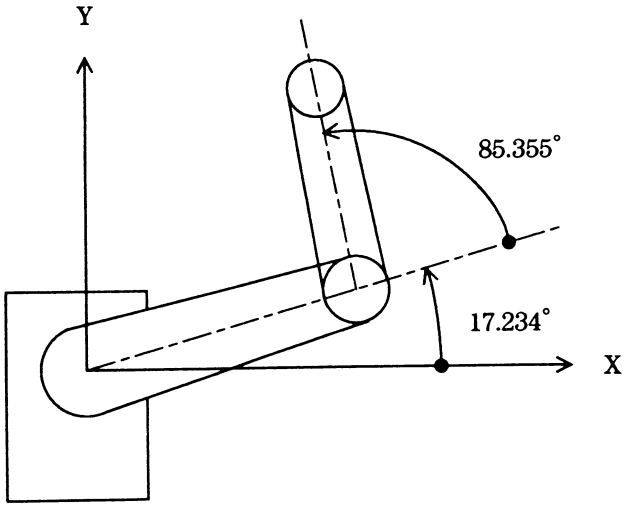
Das Vorzeichen des ausgegebenen Wertes (Plus oder Minus) entspricht dem Vorzeichen des entsprechenden Pulse-Wertes des Manipulators. Hinweise zur 0-Pulse-Position des Standardarms finden Sie im Manipulatorhandbuch.

Wurde mit ARM ein zusätzlicher Manipulator gewählt, gibt AGL den Winkel (bzw. die Position) von der 0-Pulse-Position des Standardarms zum gewählten Arm aus.

VERWANDTE BEFEHLE PLS( )

BEISPIEL

```
>PRINT AGL(1),AGL(2)
    17.234 ’   85.355
>_
```



AOPEN...CLOSE

S

Append Open (Öffnen zum Anfügen)

FUNKTION Öffnet eine Datei um Daten anzufügen.

FORMAT

```
AOPEN "[Dateiname]" AS #[Dateinummer]
  .
  .
  .
CLOSE #[Dateinummer]
```

Der Dateiname muß mit der Dateinamenerweiterung angegeben werden, die Dateinummer muß eine ganze Zahl zwischen 30 und 35 sein.

BESCHREIBUNG Öffnet die angegebene Datei und kennzeichnet sie mit der festgelegten Dateinummer. Diese Anweisung dient dazu, Daten an die angegebene Datei anzufügen. CLOSE schließt die Datei wieder und gibt die Dateinummer wieder frei.

Die angegebene Datei muß auf dem Datenträger existieren. Die Dateinummer kennzeichnet die Datei solange sie geöffnet ist, und wird von der Ausgabeanweisung zum Anfügen (PRINT #) und der Anweisung zum Schließen der Datei (CLOSE #) benötigt. Dementsprechend kann die Dateinummer, solange die aktuelle Datei nicht wieder geschlossen ist, keiner anderen Datei zugeordnet werden.

Es können maximal 6 Dateien gleichzeitig geöffnet sein. Solange dies jedoch der Fall ist, können die Befehle DLOAD und DMERGE nicht ausgeführt werden.

VERWANDTE BEFEHLE PRINT #, ROPEN, WOPEN

BEISPIEL

```
100 REAL DATA(200)
110 WOPEN "TEST.VAL" AS #30
120 FOR I=0 TO 100
130 PRINT #30,DATA(I)
140 NEXT
150 CLOSE #30
  .
  .
  .
200 AOPEN "TEST.VAL" AS #30
210 FOR I=101 TO 200
220 PRINT #30,DATA(I)
230 NEXT
240 CLOSE #30
250 '
```

ARC

> S

(Bogen)

FUNKTION Führt eine interpolare Bewegung in der XY-Ebene aus.

FORMAT `ARC P[Punkt-Nr. 1],P[Punkt-Nr. 2] {![Parallelanweisung]!}`

BESCHREIBUNG Führt eine interpolare Bogenbewegung von der aktuellen Position, durch Punkt 1 zu Punkt 2 durch. Der mit dem ARC-Befehl zurückgelegte Weg ist nur in der horizontalen Ebene ein echter Bogen.

Die angegebenen Armattribute (rechts oder links) müssen für die aktuelle Position, Punkt 1 und Punkt 2 identisch sein. Ist dies nicht der Fall, erfolgt eine Fehlermeldung.

Der ARC-Befehl kann den Trajektoriebereich des Verfahrwegs nicht im Vorfeld überprüfen. Das bedeutet, daß der Manipulator in der Bewegung möglicherweise versucht, den erlaubten Bereich zu verlassen, selbst wenn sich die Zielpositionen innerhalb des erlaubten Bereichs befinden. In diesem Fall hält der Manipulator abrupt an und kann dadurch erheblich beschädigt werden.

Der ARC-Befehl verwendet den mit SPEEDS festgelegten Wert für die Geschwindigkeit sowie den mit ACCELS festgelegten Beschleunigungswert.

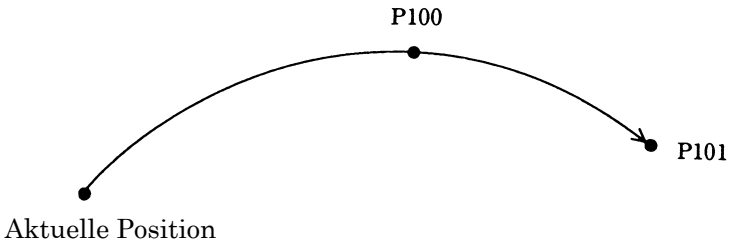
(1) Beim ARC-Befehl wird ein Weg von der aktuellen Position zu Punkt 2 interpoliert (errechnet), wobei die Z- und U-Koordinaten von Punkt 1 ignoriert werden. Daher sind ARC-Bögen nur für horizontale Arbeitsflächen geeignet. Für andere nicht horizontale Arbeitsflächen benutzen Sie besser CURVE und/oder CVMOVE.

(2) Da die ARC-Bewegung von der aktuellen Position aus beginnt, ist es u.U. erforderlich, den Manipulator mit Hilfe der Befehle GO (bzw. JUMP und verwandte) an die gewünschte Position zu bringen, und dann erst den ARC-Befehl auszuführen.

VERWANDTE BEFEHLE SPEEDS, ACCELS, !...!, CARC, MOVE, CMOVE, CVMOVE

BEISPIEL

`>ARC P100,P101` Beginnt an der aktuellen Position, führt eine interpolare Bogenbewegung durch Punkt P100 aus und hält bei Punkt P101.



ARCH

> S

("Torbogen")

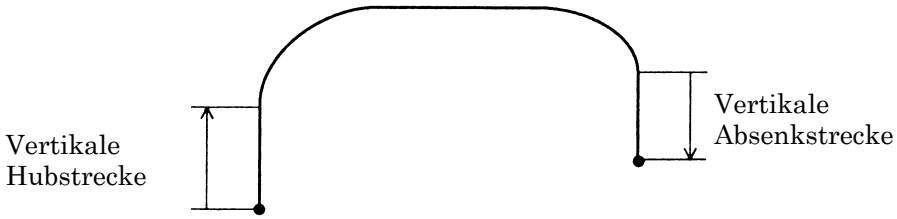
FUNKTION Definiert die Bogenparameter für den JUMP-Befehl bzw. zeigt sie an.

FORMAT (1) `ARCH [Bogen-Nr.],[vert. Hubstrecke],[vert. Absenkstrecke]`

Die Bogennummer muß eine ganze Zahl zwischen 0 und 6 sein. Die Angabe der vertikalen und Hub- und Absenkstrecke erfolgt in der Einheit mm.

(2) `ARCH`

BESCHREIBUNG (1) Definiert die vertikalen Streckenparameter für die JUMP-Bogenbewegung.
Mit ARCH können Sie sieben (0-6) vertikale Streckenparameter für Bogenbewegungen festlegen.
Mit ARCH können Sie jedoch keine Parameter für die Bogennummer 7 angeben, da diese für JUMP-Bewegungen ohne Verschleifen der Eckpunkte (Gate-Bewegung) reserviert ist.
Als vertikale Hubstrecke wird die vertikale Strecke über der Startposition bezeichnet.
Die vertikale Absenkstrecke ist die vertikale Strecke zur Zielposition.



ARCH-Werte werden über das Ausschalten hinaus gespeichert.
Durch den VERINIT-Befehl werden die ARCH-Werte auf die folgenden Standardwerte zurückgesetzt:

Bogennummer	Vertikale Hubstrecke	Vertikale Absenkstrecke
0	30	30
1	40	40
2	50	50
3	60	60
4	70	70
5	80	80
6	90	90

Falls die festgelegten Werte für die vertikale Hub- bzw. Absenkstrecke die tatsächliche vertikale Bewegungsstrecke übersteigen, wird eine Gate-Bewegung ohne Verschleifen der Eckpunkte ausgeführt.

Bei der Bogenbewegung werden die Parameter verwendet, die für die mit der JUMP-C-Bedingung ausgewählten Bogennummer gelten.

(2) Zeigt die derzeit gültigen ARCH-Werte an.

BEISPIEL

```
>ARCH 0,10,50          Definiert Bogennummer 0 mit 10 bzw. 50 mm
>JUMP P1 C0            Führt Sprungbewegung mit Verschleifen aus
>ARCH                  Zeigt definierte Bogenparameter an
arch0=10 50
arch1=40 40
arch2=50 50
arch3=60 60
arch4=70 70
arch5=80 80
arch6=90 90
>
```

ARM

> S

FUNKTION Wählt eine Armnummer bzw. zeigt sie an.

FORMAT (1) `ARM [Armnummer]`

Die Armnummer muß eine ganze Zahl zwischen 0 und 3 sein.

(2) `ARM`

BESCHREIBUNG (1) Wählt eine Armnummer.

Der ARM-Befehl ermöglicht es jedem zusätzlichen Arm, die allgemeinen Positionsdaten zu verwenden. Ist kein zusätzlicher Arm installiert, arbeitet der Standardarm (Arm Nr 0). Da bei Auslieferung des Roboters die Armnummer 0 definiert ist, ist es nicht notwendig, mit Hilfe des ARM-Befehls eine Armnummer auszuwählen.

ARM-Werte werden über das Ausschalten hinaus gespeichert.

Der VERINIT-Befehl setzt die Armnummer auf 0 zurück.

Werden zusätzliche Armnummern gewählt, die nicht über den ARM-SET-Befehl definiert wurden, erfolgt eine Fehlermeldung.

(2) Zeigt die aktuelle Armnummer an.

VERWANDTE BEFEHLE ARMSET

BEISPIEL

```
>ARM 3          Wählt den Zusatzarm 3. Wurde Arm 3 nicht über
                ARMSET definiert, erfolgt eine Fehlermeldung.
>ARM 0          Wählt den Standardarm

10 ARM 0
20 JUMP P1      Standardarm verfährt nach P1
30 ARM 1
40 JUMP P1      Zusatzarm verfährt nach P1
```

ARMSET

> S

FUNKTION Definiert einen Zusatzarm bzw. zeigt ihn an.

FORMAT (1) `ARMSET [Arm-Nr.],[1. Parameter],[2. Parameter],~`

`[3. Parameter]{,[4. Parameter]{,[5. Parameter]}}`

Die Armnummer kann eine ganze Zahl zwischen 1 und 3 sein.
Für die einzelnen Parameter muß eine reelle Zahl eingesetzt werden (siehe Übersicht auf der nächsten Seite).

(2) `ARMSET`

BESCHREIBUNG (1) Definiert die Parameter für einen Zusatzarm. Dies wird notwendig, wenn außer dem Standardarm ein Zusatzarm (oder eine Zusatzhand) installiert ist. Wenn Sie mit dem Zusatzarm arbeiten wollen, müssen Sie diesen mit dem ARM-Befehl auswählen.

Werden der dritte und vierte Parameter ausgelassen, gelten als Standardwerte die Werte des Standardarms.

Bei Ausschalten des Roboters werden die ARMSET-Werte nicht geändert. Beim VERINIT-Befehl werden die ARMSET-Werte auf "nicht definiert" zurückgesetzt.

(2) Zeigt die aktuellen ARMSET-Werte für alle definierten Arme an, auch den Wert für Arm 0 (Standardarm).

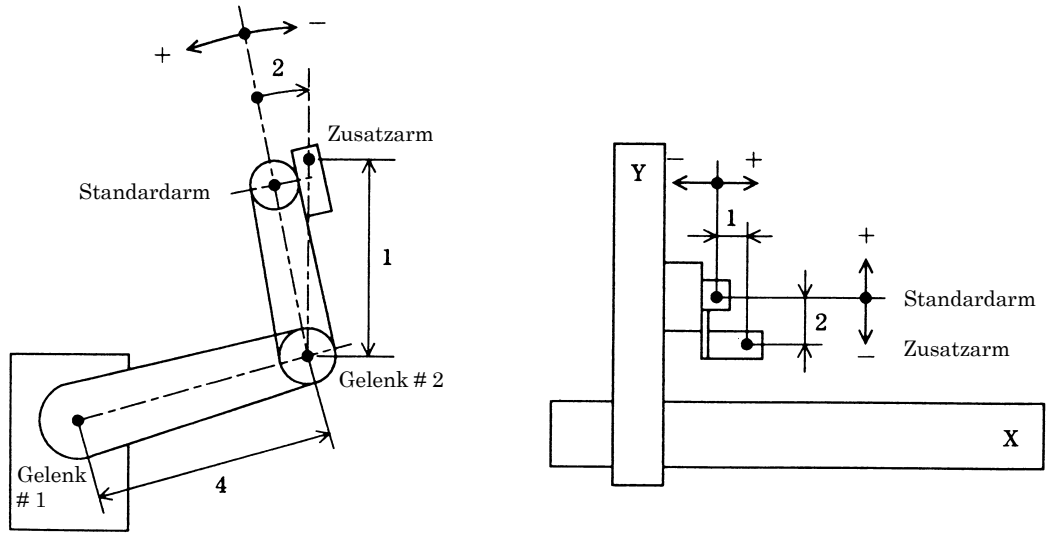
Wird ein Zusatzarm verwendet, ohne zuvor seine ARMSET-Parameter zu definieren, kann es vorkommen, daß der Roboter seine Zielposition nicht erreichen kann. Daher ist es wichtig, die ARMSET-Parameter vor Verwendung des Zusatzarms zu definieren, besonders in den folgenden Fällen:

- ein einzelner Datenpunkt soll von zwei oder mehr Armen angefahren werden
- bei Verwendung des Befehls PALET
- bei Angabe einer CP-Bewegung
- bei Definition einer relativen Position
- bei Verwendung lokaler Koordinaten

Beim Betrieb eines SCARA-Roboters mit drehbaren Gelenken oder Robotern mit Zylinderkoordinaten in einem Kartesischen Koordinatensystem werden die Berechnungen zu den Gelenkwinkeln auf der Basis dieser Parameter ausgeführt. Es ist daher sehr wichtig, sie mit Hilfe des ARMSET-Befehls zu definieren.

	Parameter-Nr.				
	1	2	3	4	5
Typ SCARA	Horizontale Entfernung des Mittelpunkts von Gelenk #2 zum Zentrum des Mittelpunkts der Zusatzachse (mm)	Versatz Gelenk #2 (Grad)	Höhenversatz (mm)	Horizontale Entfernung des Mittelpunkts von Gelenk #1 zum Mittelpunkt von Gelenk #2 (mm)	Winkelversatz der Zusatzachse (Grad)
Kartesischer Typ	Positionsversatz in X-Achsen-Richtung (mm)	Positionsversatz in Y-Achsen-Richtung (mm)	Höhenversatz (mm)	Dummy-Parameter (definiert mit 0)	Winkelversatz der Zusatzachse (Grad)

Versatz bedeutet der Versatz zum Standardarm.



Bei Ausführung des Befehls VERINIT werden die ARMSET-Werte gelöscht.

VERWANDTE BEFEHLE ARM

BEISPIEL

```
>ARMSET 1,300,-12,-30,300,0
>ARMSET
arm0 250 0 0 300 0
arm1 300 -12 -30 300 0
>
```

ASC ( )

F

ASCII

FUNKTION Gibt den ASCII-Code (numerischer Wert) des ersten Zeichens einer angegebenen Zeichenkette aus.

FORMAT

```
ASC(|[Name einer Zeichenkette]|)
    |"     [Zeichenkette]    "|
```

BESCHREIBUNG Gibt den ASCII-Code (numerischer Wert) des ersten Zeichens einer angegebenen Zeichenkette aus. Bei der angegebenen Zeichenkette kann es sich sowohl um eine konstante als auch um eine variable Zeichenkette handeln.

Falls die angegebene variable Zeichenkette jedoch keine Zeichen enthält (Null-Zeichenkette), tritt ein Fehler auf.

BEISPIEL

```
10 FUNCTION MAIN
20 STRING LETTERS$
30 LETTERS$="ABC"
40 PRINT ASC (LETTERS$)
50 FEND
>RUN

COMPILE END
65
>PRINT ASC ("ABC")
65
>
```

Ausgabe des ASCII-Codes für das erste Zeichen des aktuellen Wertes der variablen Zeichenkette LETTERS$

Kompilieren und Starten des Programms

ATAN( )

F

Arkustangens

FUNKTION Gibt den Arkustangens des angegebenen Wertes aus.

FORMAT `ATAN([numerischer Wert])`

BESCHREIBUNG Gibt den Arkustangens des angegebenen Wertes als Radiantenwert aus.

Der ausgegebene Wert liegt im Bereich von $-\pi/2$ bis $\pi/2$

Um den Radiantenwert in eine Gradzahl umzuwandeln, benutzen Sie folgende Gleichung:

Grad = Radiant*180/π
(π = 3,141593)

VERWANDTE BEFEHLE TAN( ), SIN( ), COS( ), ATAN2( )

BEISPIEL

```
>PRINT ATAN(-0.55)
```

Gibt Arkustangens von -0,55 aus

```
-.5028432
```

-0,5028432 rad

```
>PRINT ATAN(0.5773503)*180/3.141593
30
```

Der Arkustangens von 0,5773503 beträgt 30 Grad

```
PRINT TAN(30*3.141593/180)
.5773503
>
```

ATAN2( )

F

Arkustangens 2

FUNKTION Gibt den Arkustangens von x/y aus.

FORMAT `ATAN2([Wert der X-Koordinate],[Wert der Y-Koordinate])`

BESCHREIBUNG Gibt den Arkustangens von x/y als Radiantenwert aus.

Der ausgegebene Wert liegt im Bereich von $-\pi/2$ bis $\pi/2$.

Um den Radiantenwert in eine Gradzahl umzuwandeln, benutzen Sie folgende Gleichung:

Grad = Radiant*180/π
(π = 3,141593)

VERWANDTE BEFEHLE TAN( ), SIN( ), COS( ), ATAN( )

BEISPIEL

```
>PRINT ATAN2(100,100)
.7853982                    Dieser Winkel beträgt 0,7853982 rad.
>A=.7853982*180/3.141593        Wandelt 0,7853982 rad. in Grad um
>PRINT A
45
>
PRINT TAN(30*3.141693/180)
.5773503
>
```

BASE

> S

Basis für Koordinatensystem

FUNKTION Definiert ein lokales Koordinatensystem bzw. zeigt es an.

FORMAT (1) `BASE [Nr. des lokalen Koordinatensystems],[Positionsspezifikation]`

Die Nummer des lokalen Koordinatensystems muß eine ganze Zahl zwischen 1 und 15 sein.

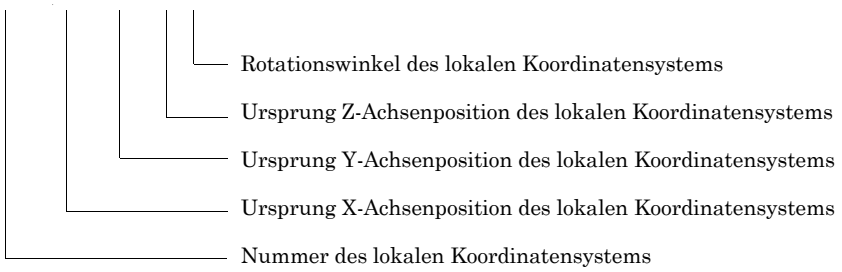
(2) BASE

BESCHREIBUNG (1) Definiert das lokale Koordinatensystem, indem dessen Ursprung und der Rotationswinkel in Verhältnis zum Roboter-Koordinatensystem angegeben werden.

Ist ein LOCAL 0-Koordinatensystem über die Befehle LOCAL0 oder durch BASE 0 definiert, gilt diese Beschreibung entsprechend dafür. Die Hinweise auf das "Roboter-Koordinatensystem" gelten in diesem Fall dann für das LOCAL 0-Koordinatensystem.

Beispiel

```
BASE 1,100,200,0,45
```



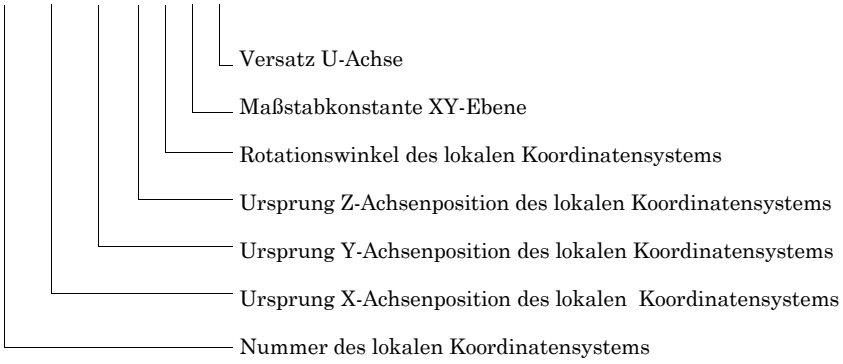

```
BASE 2,P10+X100
```

In diesem Fall werden nur die Werte für die X-U-Achsen berücksichtigt, die Armstellung und die Nummer des lokalen Koordinatensystems für P10 wurden ignoriert.

(2) Alle lokalen Koordinatensysteme, einschließlich der durch LOCAL definierten, werden angezeigt.

Beispiel

```
base1=100 200 0 45 1 45
```



Die Maßstabkonstante der XY-Ebene ist normalerweise (1). Sie wird mit dem durch den Befehl SLOCAL definierten Koordinatensystem angezeigt.

Der *"Versatz der U-Achse*" sowie der Wert für den "*Rotationswinkel des lokalen Koordinatensystems*" sind normalerweise identisch. Die Werte sind dann unterschiedlich, wenn Bit 2 des Softwareschalters auf On gesetzt wurde. Nähere Informationen dazu erhalten Sie im Handbuch zum SPEL Editor.

Die Nummern des lokalen Koordinatensystems werden auch von den Befehlen LOCAL und BASE verwendet.

Die Definitionen der lokalen Koordinatensysteme #1 bis #15 werden beim Abschalten der Stromzufuhr gelöscht, ebenso bei Ausführung der Befehle LOCAL0, BASE 0 bzw. VERINIT.

Durch Anfügen des Parameters &n kann eine Umkehrkonvertierung des lokalen Koordinatensystems durchgeführt werden. Beachten Sie, daß im TEACH-Modus festgelegte Punkte als Punktdaten in einem lokalen Koordinatensystem definiert sein können.

`P1=P*&2` Definiert den aktuellen Punkt als Koordinate im lokalen Koordinatensystem 2.

VERWANDTE BEFEHLE

BASE 0, LOCAL0, LOCAL

BEISPIEL

```
>BASE 1,100,0,0,45
```
Definiert das unten dargestellte Koordinatensystem als Local 1

```
>BASE
```
Zeigt alle lokalen Koordinatensysteme an

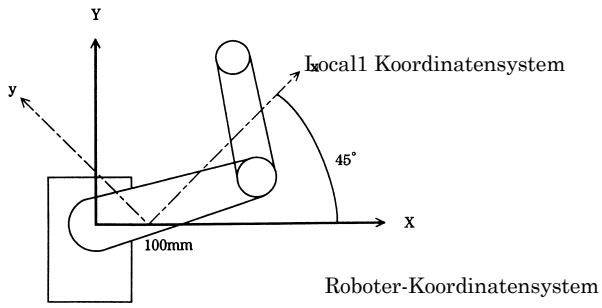
```
base0=0 0 0 0 1 0
base1=100 0 0 45 1 45
base12=100.54 11.22 0 1.554 1 1.554
>
>P5=100,200,50,0
>BASE 5,P5
```
Definiert über die Positionsdaten das lokale Koordinatensystem

```
>
>LOCAL
```
Zeigt über LOCAL das lokale Koordinatensystem an

```
local 0(p***:p***),(p***:p***)
```
Lokales Koordinatensystem definiert über BASE 0 (in diesem Fall wird P*** angezeigt)

```
local 1(p***:p***),(p***:p***)
```
Lokales Koordinatensystem definiert über BASE 1 (in diesem Fall wird P*** angezeigt)

```
rlocal12(p1 :p111),(p2 :p112)
```
'Lokales Koordinatensystem definiert über RLOCAL



Roboter-Koordinatensystem

BASE 0

> S

FUNKTION Definiert das Basis-Koordinatensystem des Roboters Local0.

FORMAT `BASE 0,[Positionsspezifikation]`

BESCHREIBUNG Jeder Roboter verfügt über ein absolutes Koordinatensystem, genannt das "*Roboter-Koordinatensystem*", dessen Position nicht verändert werden kann. Dieses Koordinatensystem dient als Referenz zur Definition eines lokalen Koordinatensystems, genannt "*Local0-Koordinatensystem*", dessen Position allerdings verändert werden kann. Das Local0-Koordinatensystem kann über die Befehle BASE 0 und LOCAL0 definiert werden.

Mit dem BASE0-Befehl wird das Local0- Koordinatensystem genauso definiert, wie lokale Koordinatensysteme mit dem Befehl BASE, mit Ausnahme der (0). Näheres dazu finden Sie in den Erläuterungen zum Befehl BASE.

In den meisten Fällen sind das Roboter-Koordinatensystem und das Local0-Koordinatensystem identisch, und es ist nicht nötig, zwischen den beiden zu differenzieren. Der Befehl BASE 0 wird verwendet, wenn lokales Koordinatensystem und Roboter-Koordinatensystem unterschiedlich sein sollen, wie z.B. bei der Durchführung von Wartungsarbeiten.

Durch Ausschalten der Steuerung werden die BASE 0-Parameter nicht verändert.

Bei Ausführung einer der folgenden Befehle werden die Werte des lokalen Koordinatensystems mit denen des Roboter-Koordinatensystems gleichgesetzt:

```
BASE 0,0,0,0,0
LOCAL0 (P1:P1),(P1:P1)
VERINIT
```

Da der Befehl BASE 0 die Position des Basis-Koordinatensystems verändert, sollte er nur wenn unbedingt notwendig verwendet werden.

Bei Ausführung des Befehls BASE 0 werden alle lokalen Koordinatensysteme einschließlich der unter LOCAL definierten gelöscht. Das bedeutet, sie müssen neu definiert werden.

Bei VERINIT wird das lokale Koordinatensystem auf (base 0 0 0 0 1 0), d.h. auf die gleichen Werte des Roboter-Koordinatensystems zurückgesetzt.

VERWANDTE BEFEHLE BASE, LOCAL0, LOCAL

CALIB

> S

Calibration (Eichung, Kalibrierung)

FUNKTION Ersetzt den aktuellen Pulse-Wert für die Armposition durch die aktuellen CALPLS-Werte.

FORMAT `CALIB {[Achsennummer] {,[Achsennummer]}3`

Die Achsennummer muß eine ganze Zahl zwischen 1 und 4 sein.

BESCHREIBUNG Berechnet und definiert automatisch den Offset-Wert (HOFS). Dieser Versatz ist notwendig, um den Ursprung aller Motoren an den entsprechenden mechanischen Ursprung des Manipulatorarms anzupassen.

Verwenden Sie den CALIB-Befehl, wenn sich der Pulse-Wert des Motors geändert hat, wie z.B. nach einem Austausch des Motors oder Getriebes.

Normalerweise sind die mit dem PULSE-Befehl gesetzten Werte für die Kalibrierposition identisch mit den CALPLS-Werten. Nach Wartungsarbeiten, wie z.B. Motoraustausch, sind diese Werte jedoch unterschiedlich, so daß eine Kalibrierung notwendig wird.

Die Kalibrierung kann dadurch erfolgen, daß der Manipulator an die gewünschte Kalibrierposition bewegt und anschließend der Befehl CALIB ausgeführt wird. Dies setzt den Pulse-Wert für die Kalibrierposition auf den CALPLS-Wert.

Um die Kalibrierung vornehmen zu können, müssen zuvor die HOFS-Werte ermittelt werden. Die HOFS-Werte werden automatisch errechnet, wenn Sie den Manipulatorarm an die gewünschte Kalibrierposition bewegen und CALIB ausführen. Anhand der Pulse-Werte für die Kalibrierposition und der CALPLS-Pulse-Werte errechnet die Steuereinheit automatisch die HOFS-Werte.

Wird keine Achsennummer angegeben, erfolgt ein Fehler.

CALIB sollte nur zu Wartungszwecken verwendet werden, d.h. nur wenn unbedingt nötig.

Durch den Befehl CALIB werden die HOFS-Werte ersetzt. Da eine unbeabsichtigte Änderung der HOFS-Werte unvorhersehbare Armbewegungen verursachen kann, sollten Sie den Befehl CALIB nur wenn unbedingt erforderlich verwenden.

VERWANDTE BEFEHLE CALPLS, HOFS

BEISPIEL

```
>CALPLS
```

Zeigt die aktuellen CALPLS-Werte an

BEISPIEL

```
>CALPLS
   65523 43320
   -1550 21351
>PULSE
   65526 49358
   -1542 21299
>CALIB 2
>PULSE
   65526 43320
   -1542 21299
>_
```

Zeigt die aktuellen CALPLS-Werte an

Zeigt die aktuellen PULSE-Positionswerte an

Führt die Kalibrierung nur für Achse 2 durch
Zeigt die (geänderten) PULSE-Werte an

CALL

S

Aufruf

FUNKTION Ruft eine Funktion als Unterprogramm auf.

FORMAT `CALL [Funktionsname]`

Es sind bis zu 10 Verschachtelungsebenen möglich.

BESCHREIBUNG CALL ruft eine Funktion (definiert mit FUNCTION ... FEND) als Unterprogramm auf (d.h. überträgt die Programmsteuerung an diese Funktion). Bei Erreichen des Befehls END oder FEND wird die Programmsteuerung an die Task zurückgegeben, aus der die CALL-Funktion aufgerufen wurde.

Da eine Funktion in einer CALL-Anweisung als Unterprogramm behandelt wird, können verschiedene Tasks dasselbe Unterprogramm aufrufen. Falls jedoch verschiedene Tasks dieselbe Funktion **gleichzeitig** aufrufen, kann es zu Konflikten bei der Wertzuweisung für Variablen in der Funktion kommen. Um dies zu verhindern, sollten Sie die Programme so anlegen, daß andere Tasks dieselbe Funktion erst dann aufrufen, wenn die Funktion durch eine vorherige Task abgeschlossen wurde. Dies nennt man ausschließliche Steuerung.

Um ein Unterprogramm innerhalb einer Task aufzurufen, verwenden Sie die Anweisung GOSUB...RETURN (siehe Beschreibung im entsprechenden Kapitel).

BEISPIEL

```
100 FUNCTION MAIN
105 OFF $0          Merker für ausschließliche Steuerung
110 XQT !2 SUB
 .
 .
 .
200 CALL ERROR
 .
 .
 .
300 FEND
310 FUNCTION SUB
 .
 .
 .
350 CALL ERROR
 .
 .
 .
400 FEND
410 FUNCTION ERROR
420 ON $0;IF ZEROFLG(0)=1 THEN WAIT SW($0)=0;GOTO 420
 .
 .
 .
490 OFF $0
500 FEND
```

C

CALPLS

> S

Kalibrierpulse

FUNKTION Definiert bzw. zeigt Positions- und Encoder-Pulse für die Kalibrierung an.

FORMAT (1) `CALPLS [Pulsewert 1. Achse],[Pulsewert 2. Achse],~`

`[Pulsewert 3. Achse],[Pulsewert 4.Achse]`

Der Pulsewert muß eine ganze Zahl sein.

(2) CALPLS

BESCHREIBUNG (1) Definiert und speichert die korrekten Positions-Pulse für die Kalibrierung.

Der Befehl CALPLS wird bei Wartungsarbeiten, z.B. Austausch eines Motors, verwendet, um die Nullposition des Motors an die mechanische Nullposition des entsprechenden Arms anzupassen. Diese Anpassung der Nullpositionen wird Kalibrierung genannt.

Normalerweise sind die PULSE-Werte für die Kalibrierposition identisch mit den CALPLS-Werten. Nach Wartungsarbeiten, wie z.B. Motoraustausch, sind diese Werte jedoch unterschiedlich, so daß eine Kalibrierung notwendig wird.

Die Kalibrierung kann dadurch erfolgen, daß der Manipulator an die gewünschte Kalibrierposition bewegt und anschließend der Befehl CALIB ausgeführt wird. Dies setzt den Pulsewert für die Kalibrierposition auf den CALPLS-Wert.

Um die Kalibrierung vornehmen zu können, müssen zuvor die HOFS-Werte ermittelt werden. Die HOFS-Werte werden automatisch errechnet, wenn Sie den Manipulator an die gewünschte Kalibrierposition bewegen und CALIB ausführen. Anhand der Pulsewerte für die Kalibrierposition und der CALPLS-Werte errechnet die Steuerung automatisch die HOFS-Werte.

Die CALPLS-Werte bleiben auch nach dem Ausschalten oder nach Ausführen der Befehle VERINIT oder SYSINIT unverändert.

(2) Zeigt die aktuellen CALPLS-Werte wie folgt an:

[Pulsewert 1.Achse]	[Pulsewert 2.Achse]
[Pulsewert 3.Achse]	[Pulsewert 4.Achse]

VERWANDTE BEFEHLE CALIB, HOFS

BEISPIEL

```
>CALPLS                 Zeigt die aktuellen CALPLS-Werte an
   65523 43320
   -1550 21351
>PULSE                  Zeigt die aktuellen PULSE-Positionswerte an
   65526 49358
   -1542 21299
>CALIB 4                Kalibrierung nur für Achse 4
>PULSE                  Zeigt (geänderte) PULSE-Werte an
   65526 43358
   -1542 21351
>_
```

CARC

> S

Continuous Arc (Bogenförmige Bewegung ohne Verzögerung)

FUNKTION

Führt eine kreisinterpolierte Bewegung in horizontaler Ebene, ohne Geschwindigkeitsverzögerung, bis zu einer festgelegten Endposition aus.

FORMAT

```
CARC P[Punkt Nr. 1],P[Punkt Nr. 2]{![Parallelanweisung]!}
```

BESCHREIBUNG

Generiert und veranlaßt eine interpolierte Bewegung als Kreisausschnitt, ausgehend von der aktuellen Position, durch Punkt Nr. 1 und Punkt Nr. 2, wobei keine Geschwindigkeitsverzögerung beim Passieren von Punkt Nr. 2 erfolgt. Detaillierte Informationen zu interpolierten Bogenbewegungen finden Sie in der Beschreibung des Befehls ARC.

Folgt unmittelbar auf den Befehl CARC ein weiterer CARC-Befehl oder der Befehl CMOVE, wird die Geschwindigkeit ohne Verzögerung konstant gehalten. Stellen Sie daher sicher, daß eine Serie von Befehlen zur kontinuierlichen Bewegung entweder mit dem Befehl MOVE oder ARC abgeschlossen und damit abgebremst und angehalten wird.

Falls dem CARC-Befehl einer der Befehle CARC, CMOVE, ARC oder MOVE unmittelbar folgt, achten Sie darauf, daß die aufeinanderfolgenden Wege nahtlos und ohne Knickstellen ineinander übergehen. Ansonsten kann es passieren, daß sich der Arm heftig und ruckartig bewegt, oder daß Fehler 152 erfolgt, und die Stromversorgung zu den Motoren unterbrochen und damit der Manipulator gestoppt wird.
Um dies zu verhindern, setzen Sie Bit 6 des Softwareschalters SS5 auf 1 (EIN). Folgt bei dieser Einstellung auf eine CARC-Bewegung von P1 nach P2 eine MOVE-, CMOVE-, ARC- oder CARC-Bewegung von P2 nach P3, beschreibt bzw. benutzt der Arm bei Erreichen von P2 einen nahtlosen Weg zu P3, indem er nah an P2 vorbeifährt aber nicht direkt hindurch.
Folgt bei dieser Einstellung nach der CARC-Bewegung von P1 nach P2 kein Bewegungsbefehl, verlangsamt der Arm die Bewegung und hält bei P2.

Wenn die CARC-Bewegung durch einen MOVE- oder ARC-Befehl gebremst und angehalten werden soll und die Strecke zum Abbremsen nicht ausreicht, erfolgt Fehler 153 und die Stromversorgung zu den Motoren wird unterbrochen, d.h. der Manipulator stoppt.

Folgt unmittelbar auf den CARC-Befehl ein Befehl bzw. eine Funktion oder Anweisung, durch die keine Bewegung ausgelöst wird, wird dieser Befehl bzw. diese Funktion oder Anweisung ausgeführt noch bevor der Arm die durch den CARC-Befehl vorgesehene Position erreicht hat. Daher sollten unmittelbar auf CARC prinzipiell nur die Befehle SPEEDS, ACCELS, CARC, CMOVE, ARC oder MOVE folgen.

Zur Verarbeitung von Anweisungen parallel zu CARC-Bewegungen sollten Sie die Parallelverarbeitung mit Hilfe des Befehls !...! verwenden (siehe Beschreibung im entsprechenden Kapitel).

VERWANDTE BEFEHLE

P=, ! ... !, ARC, SPEEDS, ACCELS, MOVE, CMOVE

BEISPIEL

```
20 JUMP P1
30 CARC P2,P3 !D50;ON 0;D100;OFF 0!
40 ARC P4,P5
```

CHAIN

S

Anhängen

FUNKTION Lädt Objektprogramm- und Positionsdatendateien in den Hauptspeicher und führt diese aus.

FORMAT `CHAIN "{[Laufwerk]}{[Pfadangabe]}[Dateiname]"`

Dateinamenerweiterungen sind nicht erlaubt.

BESCHREIBUNG Während der Ausführung eines Programms wird über diesen Befehl ein sekundäres Objektprogramm sowie die entsprechenden Positionsdaten in den Hauptspeicher geladen.

Das aktuell laufende Objektprogramm sowie die Positionsdaten werden durch das angegebene Objektprogramm und die entsprechenden Positionsdaten ersetzt. Das angegebene Programm wird von der ersten Zeile an ausgeführt. Wenn keine Laufwerksangabe erfolgt, sucht die CHAIN-Anweisung im aktuellen Verzeichnis nach Dateien.

Eine Dateinamenerweiterung für die angegebene Datei ist nicht erlaubt. Durch die Angabe des Dateinamen allein werden automatisch die folgenden 3 Dateiarten geladen:

"dateiname.OBJ"
"dateiname.SYM"
"dateiname.PNT

Falls eine dieser Dateien nicht im Dateispeicher existiert, erfolgt ein Fehler. Eine Fehlerbehebung mit Hilfe des Befehls ONERR ist nicht möglich.

Bei Ausführung von CHAIN werden alle Variablen mit Ausnahme der Backup-Variablen gelöscht.

Da mit dem CHAIN-Befehl kein Software-Reset erfolgt, bleiben die Bedingungen für die Ein- und Ausgänge sowie die Merker erhalten.

BEISPIEL

```
10 FUNCTION JOB1
20 IF IN(0)=1 THEN CHAIN "JOB2"    Führt JOB2 aus, wenn Eingangsport 0 auf 1 steht
30 IF IN(0)=2 THEN CHAIN "JOB3"    Führt JOB3 aus, wenn Eingangsport 0 auf 2 steht
40 IF IN(0)=3 THEN CHAIN "B:JOB4" Führt JOB4 von Diskette aus, wenn Eingangsport 0 auf 3 steht
50 FEND
```

CHARSIZE

> F

Zeichengröße

FUNKTION Definiert die Größe der an der Bedieneinheit angezeigten Zeichen.

FORMAT CHARSIZE [Größenwert]

Einstellbar ist eine Zeichengröße von 2, 4, 9 oder 16. Der Standardwert beträgt 2.

BESCHREIBUNG Definiert anhand des Größenwertes die Größe der Zeichen, die über den Befehl OPU PRINT an der Bedieneinheit ausgegeben werden. Die Größenwerte entsprechen den folgenden Zeichengrößen. Andere Größenwerte werden ignoriert.

Größenwert	Zeichengröße
2	Standard
4	x4
9	x9
16	x16

Bei Einschalten des Roboters wird automatisch der Größenwert 2 (Standardwert) eingestellt.

VERWANDTE BEFEHLE CLS CURSOR NORMAL RESERVE OPUNIT OPU PRINT

BEISPIEL

```
>CHARSIZE 9
>OPU PRINT 3,1,"EPSON"
>CHARSIZE 2
>OPU PRINT 5,5"CORP."
```

CHDIR, CD

Change Directory (Verzeichnis wechseln)

FUNKTION

Wechselt ins angegebene Verzeichnis und zeigt das Verzeichnis mit Pfadangabe an.

FORMAT

(1) `CHDIR {{[Laufwerk]}{[Pfadname]}[Verzeichnisname]}`

(2) `CHDIR`

BESCHREIBUNG

(1) Wechselt vom aktuellen Verzeichnis im angegebenen Laufwerk in das angegebene Verzeichnis.

Wenn kein Pfad oder Verzeichnis angegeben ist, wird das aktuelle Verzeichnis im angegebene Laufwerk angezeigt. Damit können Sie überprüfen, in welchem Verzeichnis Sie derzeit arbeiten.

Wird kein Laufwerk angegeben, nimmt CHDIR das aktuelle Laufwerk an.

(2) Zeigt das aktuelle Verzeichnis an.

Beim Einschalten wird das Stammverzeichnis das aktuelle Verzeichnis.

VERWANDTE BEFEHLE

DIR, SELDSK

BEISPIEL

`>CHDIR \`	Wechselt aus dem aktuellen Verzeichnis im aktuellen Laufwerk ins Stammverzeichnis
`>CHDIR..`	Wechselt vom aktuellen Verzeichnis ins Elternverzeichnis
`>CD \TEST\H55`	Wechselt in das Unterverzeichnis H55 im Verzeichnis \TEST
`>CD B:\BAK`	Wechselt vom aktuellen Verzeichnis auf Laufwerk B ins Verzeichnis \BAK
`>CD` `A:\TEST\H55\`	Zeigt das aktuelle Verzeichnis im aktuellen Laufwerk an
`>CD B:` `B:\BAK\`	Zeigt das aktuelle Verzeichnis in Laufwerk B an

CHR$( )

F

Character (Zeichen)

FUNKTION Gibt das Zeichen, das einem angegebenen ASCII-Code entspricht, aus.

FORMAT `CHR$([Zeichencode])`

Der Zeichencode muß eine ganze Zahl zwischen 0 und 255 sein.

BESCHREIBUNG Gibt das Zeichen, das dem angegebenen ASCII-Code entspricht aus.

BEISPIEL

```
>PRINT CHR$(&H41)+CHR$(&H42)+CHR$(&H43)
ABC
>
```

CLEAR

Löschen

FUNKTION Löscht Positionsdaten.

FORMAT CLEAR

BESCHREIBUNG Löscht Positionsdaten. (Initialisiert den Positionsdatenbereich.)

Wird CLEAR im Online-Modus ausgeführt, werden die Daten im Hauptspeicher der Steuereinheit gelöscht; bei Ausführung im Offline-Modus werden die Positionsdaten in der Programmiereinheit gelöscht.

VERWANDTE BEFEHLE PDEL, NEW

BEISPIEL

```
>P1=100,200,-20,0/R
>P2=0,300,0,20/L
>PLIST
P1=100,200,-20,0/R
P2=0,300,0,20/L
>CLEAR
>PLIST
>_
```

Da alle Positionsdaten gelöscht wurden, werden keine angezeigt

C

CLRLIB

Clear Library (Bibliothek löschen)

FUNKTION Löscht Backup-Variablen

FORMAT CLRLIB

BESCHREIBUNG Löscht die Backup-Variablen im Speicher.

VERWANDTE BEFEHLE LIBRARY, SYS

BEISPIEL

```
>CLRLIB
>
```

CLS

> S

Clear Screen (Bildschirm löschen)

FUNKTION Löscht Zeichen (an der Bedieneinheit).

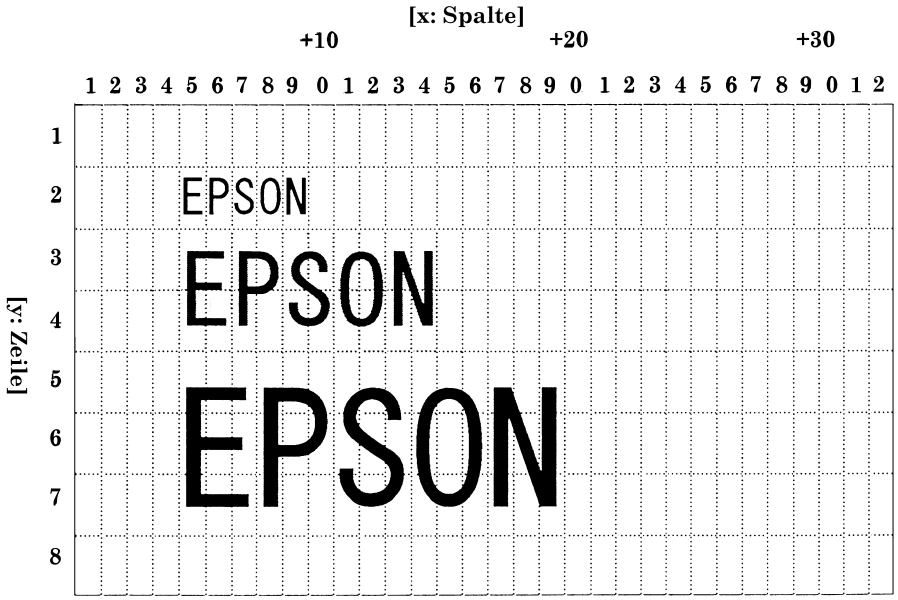
FORMAT `CLS {[X-Koordinate],[Y-Koordinate],[Spaltenanzahl],[Zeilenanzahl]`

BESCHREIBUNG Löscht die Zeichen in dem über die Werte (x, y) und (x+Spaltenanzahl, y+Zeilenanzahl) angegebenen Bereich.
Nachdem alle Zeichen gelöscht wurden, wird die Cursor auf die Position (x,y) gesetzt.
Wird der Befehl CLS ohne weitere Parameter eingegeben, werden alle Zeichen am Bildschirm der Bedieneinheit gelöscht. In diesem Fall wird der Cursor nach dem Löschen der Zeichen auf die Home-Position (1,1) gesetzt.

VERWANDTE BEFEHLE CHARSIZE, CURSOR, NORMAL, REVERSE, OPUNIT, OPU PRINT

BEISPIEL

`>CLS 5,3,10,2` Löscht alle in der folgenden Grafik grau schraffierten Zeichen



CMOVE

> S

Continuous Move (Kontinuierliche Bewegung)

FUNKTION

Führt eine interpolierte Linearbewegung aller vier Achsen gleichzeitig durch eine angegebene Position und ohne Geschwindigkeitsverzögerung durch.

FORMAT

```
CMOVE [Positionsspezifikation] {![Parallelanweisung]!}
```

BESCHREIBUNG

Führt eine Linearbewegung aller vier Achsen gleichzeitig von der aktuellen durch eine angegebene Position durch, wobei die Geschwindigkeit beim Passieren der angegebenen Position nicht verringert wird. Nähere Informationen zu interpolierten Linearbewegungen finden Sie in der Beschreibung des Befehls MOVE.

Folgt unmittelbar auf den Befehl CMOVE ein CARC-Befehl oder ein weiterer CMOVE-Befehl, wird die Geschwindigkeit ohne Verzögerung konstant gehalten. Stellen Sie daher sicher, daß eine Serie von Befehlen zur kontinuierlichen Bewegung entweder mit dem Befehl MOVE oder ARC abgeschlossen und damit abgebremst und angehalten wird.

Falls dem CMOVE-Befehl einer der Befehle CARC, CMOVE, ARC oder MOVE unmittelbar folgt, achten Sie darauf, daß die aufeinanderfolgenden Wege nahtlos und ohne Knickstellen ineinander übergehen. Ansonsten kann es passieren, daß sich der Arm heftig und ruckartig bewegt, oder daß Fehler 152 erfolgt, und die Stromversorgung zu den Motoren unterbrochen und damit der Manipulator gestoppt wird.

Um dies zu verhindern, setzen Sie Bit 6 des Softwareschalters SS5 auf 1 (EIN). Folgt bei dieser Einstellung auf eine CMOVE-Bewegung von P1 nach P2 eine MOVE-, CMOVE-, ARC- oder CARC-Bewegung von P2 nach P3, beschreibt bzw. benutzt der Arm bei Erreichen von P2 einen nahtlosen Weg zu P3, indem er nah an P2 vorbeifährt aber nicht direkt hindurch.

Folgt bei dieser Einstellung nach der CMOVE-Bewegung von P1 nach P2 kein Bewegungsbefehl, verlangsamt der Manipulator die Bewegung und hält bei P2.
Wenn die CMOVE-Bewegung durch einen MOVE- oder ARC-Befehl gebremst und angehalten werden soll und die Strecke zum Abbremsen nicht ausreicht, erfolgt Fehler 153 und die Stromversorgung zu den Motoren wird unterbrochen, d.h. der Manipulator stoppt.

Folgt unmittelbar auf den CMOVE-Befehl ein Befehl bzw. eine Funktion oder Anweisung, durch die keine Bewegung ausgelöst wird, wird dieser Befehl bzw. diese Funktion oder Anweisung ausgeführt noch bevor der Arm die durch den CMOVE-Befehl vorgesehene Position erreicht hat. Daher sollten unmittelbar auf CMOVE prinzipiell nur die Befehle SPEEDS, ACCELS, CARC, CMOVE, ARC oder MOVE folgen.
Zur Verarbeitung von Anweisungen parallel zu CMOVE-Bewegungen können Sie die Parallelverarbeitung mit Hilfe des Befehls !...! verwenden (siehe Beschreibung im entsprechenden Kapitel).

VERWANDTE BEFEHLE

P=, ! ... !, MOVE, SPEEDS, ACCELS, ARC, CARC

BEISPIEL

```
20 JUMP P1
30 CMOVE P2 !D50;ON 0;D100;OFF 0!
40 CARC P3,P4;MOVE P5
```

COMPILE, COM

(Zusammenstellen, Übersetzen)

FUNKTION Kompiliert eine Quellprogrammdatei in eine ausführbare Datei.

FORMAT `COMPILE{-V}{-L}{"{[Laufwerk]}{[Pfadangabe]}[Dateiname]"}`

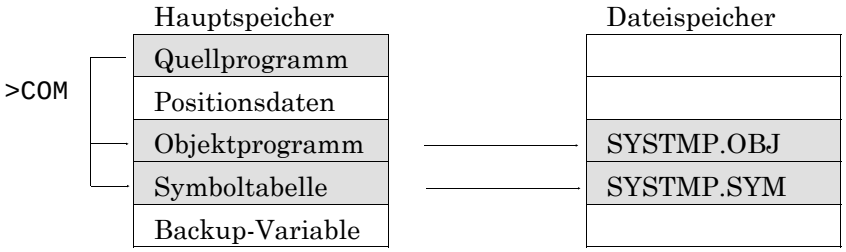
Dateinamenerweiterungen sind nicht erlaubt.

BESCHREIBUNG Um ein Programm ausführen zu lassen, muß das Quellprogramm in ein maschinenlesbares Format umgewandelt werden. Dieser Vorgang wird Kompilierung genannt. Mit Hilfe von COMPILE wird ein Quellprogramm kompiliert.

Bei Ausführung des Befehls COMPILE erzeugt SPEL III eine ausführbare Datei im Zwischencode, die als Objektdatei bezeichnet wird sowie eine Symboltabellendatei, in der die Variablen- und Funktionsnamen im Quellcode mit denen im Zwischencode verbunden werden.

Wird kein Dateiname angegeben, kompiliert SPEL III das im Hauptspeicher befindliche Quellprogramm und legt auch im Hauptspeicher eine Objekt- und eine Symboltabellendatei an. Um die im Hauptspeicher erzeugte Objektdatei ausführen zu lassen, verwenden Sie den Befehl XQT.

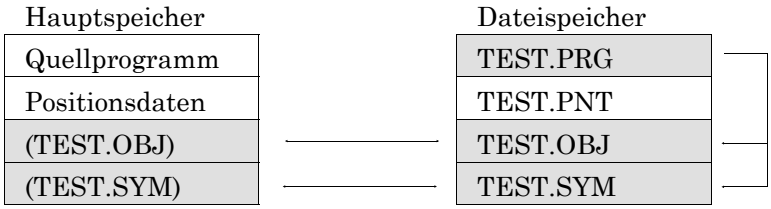
Nach dem Kompilieren erzeugt SPEL III im aktuellen Verzeichnis des aktuellen Laufwerks eine Objektdatei mit dem Namen SYSTMP.OBJ sowie eine Symboldatei mit dem Namen SYSTMP.SYM.



Bei Angabe eines Dateinamen kompiliert SPEL III die angegebene Quellprogrammdatei (Dateiname.PRG), die sich auf dem angegebenen Pfad (Pfadname) im angegebenen Laufwerk (Laufwerk) befindet und erzeugt im aktuellen Verzeichnis des aktuellen Pfades eine Objektdatei mit der Bezeichnung Dateiname.OBJ sowie eine Symboldatei mit der Bezeichnung Dateiname.SYM. Beachten Sie, daß "Dateiname" durch den tatsächlichen, von Ihnen vorgegebenen Dateinamen ersetzt wird. Zur Ausführung des Programms geben Sie XQT gefolgt vom Dateinamen ein. Nähere Informationen finden Sie unter dem Befehl XQT.

SPEL III reserviert im Hauptspeicher zwei Bereiche für die Kompilierung: den Objektbereich und den Symbolbereich. Bei Ausführung des COMPILE-Befehls mit Angabe eines Dateinamen erzeugt SPEL III sowohl im Objekt- bzw. Symbolbereich die Dateien Dateiname.OBJ bzw. Dateiname.SYM als auch im angegebenen Pfad auf dem angegebenen Laufwerk.

SPEL III akzeptiert den Befehl COMPILE mit Angabe des Dateinamen, der Pfad- und der Laufwerksangabe nur, wenn sich die Quellprogrammdatei auch tatsächlich im angegebenen Pfad bzw. Laufwerk befindet. Das bedeutet, die Quellprogrammdatei muß vor dem Kompilieren auf einer Diskette gesichert werden.



Wenn Sie den Parameter -V angeben, überprüft SPEL III, ob alle Variablentypen deklariert sind. Findet das Programm dabei eine Variable, deren Typ noch nicht deklariert wurde, erzeugt dies Fehler 2. Damit wird sichergestellt, daß für alle Variablen eine Typendeklaration festgelegt wurde.

Wird -V weggelassen, betrachtet SPEL III alle Variablen ohne Typendeklaration als reele Zahl.

Wenn Sie den Parameter -L angeben, wird die angegebene Zeile, in die über einen der Befehle GOTO/GOSUB/ONERR verzweigt werden soll, als eine ganzen Zahl von 4 Byte kompiliert.
Wird der Parameter -L weggelassen, wird die Zeile als eine ganzen Zahl von 2 Byte kompiliert.
Wird ein Programm ohne Angabe des Parameters -L kompiliert, kann es passieren, daß die folgende Meldung angezeigt wird:

```
short branch     [Zeilennummer]
```

Wenn ein Programm ohne Angabe des Parameters -L kompiliert wird, wird die angegebene Zeile, in die über einen der Befehle GOTO/GOSUB/ONERR verzweigt werden soll, als eine ganzen Zahl von 2 Byte kompiliert. Dies bedeutet, die Zeilen, in die verzweigt werden soll, dürfen ± 32 kByte (im Objektbereich) benötigen.
Die oben erwähnte Meldung bedeutet, daß die Zeilen, in die verzweigt werden soll, mehr als ± 32 kByte benötigen. In diesem Fall sollten Sie zur Kompilierung des Programms den Parameter -L angeben.

Wird bei der Kompilierung der Parameter -L angeben, ist die Objektgröße nach der Kompilierung etwas höher, als bei einer Kompilierung ohne den Parameter -L.

Bei Ausführung des COMPILE-Befehls mit Angabe eines Dateinamen erzeugt SPEL III sowohl im Objekt- bzw. Symbolbereich die Dateien Dateiname.OBJ bzw. Dateiname.SYM als auch im angegebenen Pfad auf dem angegebenen Laufwerk. Dadurch stimmt das gerade im Hauptspeicher befindliche Pogramm nicht mehr mit den Positionsdaten im Hauptspeicher überein. Dies sollten Sie bei Ausführung des kompilierten Programms bzw. bei der Arbeit mit den Dateien beachten.

VERWANDTE BEFEHLE RUN, XQT, QUIT, RESUME, VARIABLE

BEISPIEL

```
>COM
COMPILE END
>
```
Kompiliert die Quellprogrammdatei im Hauptspeicher

```
>XQT
>
```
Führt die Objektprogrammdatei im Hauptspeicher aus, das Programm wird gestartet

```
>COM"B:TEST"
COMPILE END
```
Kompiliert das Programm TEST.PRG auf Laufwerk B (sofern Laufwerk B installiert ist)

```
>XQT"B:TEST"
>
```
Führt die kompilierte Datei TEST.OBJ aus

```
>COM"TEST"
COMPILE END
>DLOAD "TEST"
```
Lädt das Quellprogramm und die Positionsdaten, die der Objektdatei und den Symboltabellen entsprechen, in den Hauptspeicher

```
>XQT
>
```
Führt das Programm, das sich im Hauptspeicher befindet aus (in diesem Fall das Programm "Test")

```
>COM-V
##ERROR 2 at 50
```
Überprüft bei der Kompilierung die Variablentypdeklaration

Zeile 50 enthält eine Variable ohne Typendeklaration

```
##ERROR 2 at 120
>
```
Zeile 120 enthält eine Variable ohne Typendeklaration

CONFIG, CNFG

Configuration (Konfiguration)

FUNKTION Setzt die Kommunikationsparameter für die RS-232C-Schnittstelle.

FORMAT

```
CONFIG {#[Port-Nr.],[Modus-Nr.],[Protokoll-Nr.],~
                    [Zeitsperre],[Baudraten-Nr.]}
```

Die Port-Nr. muß entweder 20 oder 21 sein (bei einer zusätzlichen RS-232C-Schnittstelle eine ganze Zahl von 20 bis 23).
Die Modus-Nr. muß eine ganze Zahl von 0 bis 47 sein.
Die Protokoll-Nr. muß eine ganze Zahl von 0 bis 19 sein.
Die Baudraten-Nr. muß eine ganze Zahl von 0 bis 7 sein.

BESCHREIBUNG CONFIG setzt die Kommunikationsparameter für eine RS-232C-Schnittstelle. Bei einer Standardinstallation muß die Port-Nr. mit 20 bzw. 21 angegeben werden. Ist eine zusätzliche RS-232C-Schnittstelle installiert, kann eine Port-Nr. von 20 bis 23 angegeben werden.

Näheres zu Modus-, Protokoll- und Baudratennummer finden Sie in den Tabellen auf den folgenden Seiten. Ist das TTY-Protokoll ausgewählt, bleibt eine Zeitsperre ohne Wirkung. Für die zusätzliche Port-Nr. 22 und 23 hat eine Modus-Nr. von 24 oder höher keine Wirkung.

Nachdem Sie mit Hilfe des Befehls CONFIG die Konfigurationsparameter der RS-232C-Schnittstelle geändert haben, müssen Sie das System zurücksetzen, damit die neuen Einstellungen wirksam werden. Dies wird nach jeder Änderung der Konfiguration notwendig, da das System die Konfigurationsparameter nur nach dem Systemstart liest.

Sie können das System auf folgende Weise zurücksetzen:

- Stromversorgung aus- und wieder einschalten
- Modus wechseln
- Im TEACH-Modus den Befehl RESET ausführen
- Falls eine Bedieneinheit oder ähnliches verwendet wird, den RESET-Taster im Auto-Modus drücken
- Über einen RS-232C-Port die Reset-Zeichen angeben

Wenn Sie den Befehl CONFIG ohne weitere Parameterangabe eingeben, können Sie die aktuellen Parametereinstellungen anzeigen lassen.

Bei Ausführung des Befehls VERINIT werden alle Schnittstellenkonfigurationsparameter (Modus, Protokoll, Zeitsperre und Baudrate) auf ihre Standardwerte zurückgesetzt: 2, 1, 3 und 0. Dieser Ziffern haben die folgende Bedeutung:

2:	7 Datenbits, gerade Parität, 1 Stop-Bit
1:	Basisprotokoll (Sekundärstation)
3:	Zeitsperre von 3 Sekunden
0:	Baudrate von 9600 bps

VERWANDTE BEFEHLE PRINT #, INPUT #, CON, VER, VERINIT

BEISPIEL

```
>CONFIG #20,0,4,0,6          'Setzt die Konfigurationsparameter für
                              Port-Nr. 20 auf:
                              7 Datenbits, gerade Parität, 2 Stop-Bits
                              TTY-Protokoll, XON/XOFF-Steuerung,
                               CR-LF-Terminator, Baudrate 19200 bps
```

ARGUMENTE UND EINSTELLUNGEN

Modus-Nr.

Modus Nr.	Daten-bits	Parität	Stop-Bits		Modus Nr.	Daten-bits	Parität	Stop-Bits
0	7	Gerade	2		24	6	Gerade	2
1	7	Ungerade	2		25	6	Gerade	1.5
2	7	Gerade	1		26	6	Gerade	1
3	7	Ungerade	1		27	-	-	-
4	8	Keine	2		28	6	Ungerade	2
5	8	Keine	1		29	6	Ungerade	1.5
6	8	Gerade	1		30	-	-	-
7	8	Ungerade	1		31	-	-	-
8	7	Gerade	1.5		32	6	Keine	2
9	-	-	-		33	6	Keine	1.5
10	7	Ungerade	1.5		34	6	Keine	1
11	-	-	-		35	-	-	-
12	7	Keine	2		36	5	Gerade	2
13	7	Keine	1		37	5	Gerade	1.5
14	7	Keine	1.5		38	5	Gerade	1
15	-	-	-		39	-	-	-
16	8	Gerade	2		40	5	Ungerade	2
17	8	Gerade	1.5		41	5	Ungerade	1.5
18	-	-	-		42	5	Ungerade	1
19	8	Ungerade	2		43	-	-	-
20	8	Ungerade	1.5		44	5	Keine	2
21	-	-	-		45	5	Keine	1.5
22	8	Keine	1.5		46	5	Keine	1
23	-	-	-		47	-	-	-

Protokollnummer

Protokoll-Nr.	Protokoll	Station	"Buffer busy"-Überprüfung	Terminator
0	TTY	-	Nein	CR
1	BASIC	2	-	-
2	BASIC	1	-	-
3	TTY	-	XON/XOFF	CR
4	TTY	-	XON/XOFF	CR-LF
5	TTY	-	XON/XOFF	LF
6	TTY	-	Nein	CR-LF
7	TTY	-	Nein	LF
8	BASIC2	2	-	-
9	BASIC2	1	-	-
10	TTY	-	RS/CS	CR
11	BASIC	-	RS/CS	-
12	BASIC	-	RS/CS	-
13	TTY	-	RS/CS & XON/XOFF	CR-LF
14	TTY	-	RS/CS & XON/XOFF	CR-LF
15	TTY	-	RS/CS & XON/XOFF	LF
16	TTY	-	RS/CS	CR-LF
17	TTY	-	RS/CS	LF
18	BASIC2	2	RS/CS	-
19	BASIC2	1	RS/CS	-

Hinweise zur Verwendung von RS/CS

- Die Übertragung bei RS wird nicht überprüft, die RS-Ausgabe (Anfrage zur Übertragung) ist immer L (Low).
  Die Steuerung verfügt über 20 Frames und einen Buffer von 120 Byte, so daß die Gefahr einer fehlerhaften Datenübertragung nur relativ gering ist, selbst wenn die Protokoll-Nr. 10, 16 oder 17 (TTY, RS/CS) verwendet wird. Werden jedoch große Datenmengen an die Steuerung übertragen, kann es vorkommen, daß die Daten nicht korrekt empfangen werden, da diese Protokolle ständig zur Datenübertragung bereit sind. Arbeiten Sie in diesem Fall mit den Protokollnummern 13, 14 oder 15 (TTY, RS/CS & XON/XOFF).
- Bei den Protokollnummern 18 bzw. 19 (BASIC, RS/CS), ist die Zeitsperre nicht wirksam. solange die Steuerung darauf wartet, Daten zu senden.

Baudratennummer (Übertragungsgeschwindigkeit)

Baudraten-Nr.	Baudrate (bps)
0	9600
1	4800
2	2400
3	1200
4	600
5	300
6	19200
7	38400

CONSOLE, CNSOL

Bedienpult

FUNKTION

Definiert die im Auto-Modus zu verwendende Eingabeeinheit.

FORMAT

```
CONSOLE {|OP          |}
        |#[Port-Nr.]|
        |BUS         |
```

Die Port-Nr. muß entweder 20 oder 21 sein. Der Standardwert ist OP.

BESCHREIBUNG

Ist der Parameter OP angegeben, ist das Gerät, das am Haupt-Remote-Anschluß angeschlossen ist, die Eingabeeinheit. Der Roboter kann über Schalter gesteuert werden. Der Haupt-Remote-Anschluß wird über Bit 1 des Software-Schalters SS1 ausgewählt. Bei Auslieferung des Roboters ist REMOTE2 (Bedieneinheit) als Haupt-Remote-Anschluß ausgewählt.

Wird eine Portnummer angegeben, ist der angegebene Port der RS-232C-Schnittstelle die Eingabeeinheit.

Über eine als Eingabeeinheit definierte RS-232C-Schnittstelle kann der Manipulator durch Verwendung von SPEL-III-Anweisungen im Auto-Modus wie im Teach-Modus gesteuert werden.

Ist der Parameter BUS angegeben, ist der serielle Bus BUS1 (Option) die Bedieneinheit. Der Roboter kann dann über einen Host-Computer im seriellen bus-Netzwerk gesteuert werden.

Wenn Sie nur den Befehl CONSOLE ohne Paramter eingeben, wird die aktuell ausgewählte Bedieneinheit angezeigt.

Mit dem Befehl VER können Sie den Wert den Wert der aktuellen Port-Nr. anzeigen lassen.

Durch VERINIT wird der CONSOLE-Wert auf den Standardwert OP zurückgesetzt.

VERWANDTE BEFEHLE

CONFIG, VER, VERINIT

BEISPIEL

```
>CONSOLE #20    Definiert RS-232C-Anschluß Nr. 20 als Verbindung
                zum PC
>CONSOLE #21    Definiert RS-232C-Anschluß Nr. 21 als Verbindung
                zum PC
```

COPY

Kopieren

FUNKTION Kopiert eine Datei an eine andere Stelle.

FORMAT

```
COPY {[Laufwerk 1]}{[Pfadangabe 1]}{[Dateiname 1]},~
     {[Laufwerk 2]}{[Pfadangabe 2]}{[Dateiname 2]}
```

Der Dateiname muß mit der Dateinamenerweiterung angegeben werden.

BESCHREIBUNG Kopiert die angegebene Datei - Dateiname 1 (Quelle) - als angegebene Datei - Dateiname 2 (Ziel).

Platzhalter (*,?) sind bei der Angabe der Dateinamen erlaubt.

Der angegebene Pfad für Quell- und Zieldatei (Laufwerk\Pfadname\Dateiname) darf nicht identisch sein.

Spezifikation der Quelldatei (Dateiname 1)
COPY [Laufwerk 1]
Kopiert alle Dateien im aktuellen Verzeichnis des angegebenen Laufwerks (Laufwerk 1).
COPY [Laufwerk 1] [Pfadangabe 1]
Kopiert alle Dateien im angegebenen Pfad (Pfadangabe 1) des angegebenen Laufwerks (Laufwerk 1).
COPY [Laufwerk 1] [Pfadangabe 1] [Dateiname 1]
Kopiert die angegebene Datei (Dateiname 1). Wird kein Laufwerk angegeben, wird das aktuelle Laufwerk angenommen; bei Auslassung des Pfades (Pfadangabe 1) wird das aktuelle Verzeichnis angenommen.

Spezifikation der Zieldatei (Dateiname 2)
[Laufwerk 2]
Kopiert die Datei ins aktuelle Verzeichnis im angegebenen Laufwerk (Laufwerksangabe 2). Die kopierte Datei erhält denselben Namen wie die Originaldatei. Wird die Laufwerksangabe 2 ausgelassen, wird das aktuelle Verzeichnis im aktuellen Laufwerk angenommen.
[Laufwerk 2] [Pfadangabe 2]
Kopiert die Datei ins angegebene Verzeichnis (Pfadangabe 2). Die kopierte Datei erhält denselben Namen wie die Originaldatei.
[Laufwerk 2] [Pfadangabe 2] [Dateiname 2]
Kopiert die Datei mit dem angegebenen Namen (Dateiname 2) in das angegebene Verzeichnis (Pfadangabe 2) im angegebenen Laufwerk (Laufwerksangabe 2).

Wenn Sie eine Datei innerhalb eines Verzeichnis kopieren wollen, ist es sinnvoll, dieses Verzeichnis mit dem Befehl CHDIR(CD) zum aktuellen Verzeichnis zu machen. Dann brauchen Sie nach dem COPY-Befehl nur noch die Dateinamen ohne die Pfadnamen für Quelle und Ziel anzugeben.

VERWANDTE BEFEHLE

DIR, CHDIR, MKDIR

BEISPIEL

`COPY \BAK\*.PRG \USR\*.PRG`	Kopiert alle Dateien mit der Erweiterung *.PRG aus dem Verzeichnis \BAK des aktuellen Laufwerks ins Verzeichnis \USER des aktuellen Laufwerks, wobei die ursprünglichen Dateinamen beibehalten werden.
`COPY B: A:`	Kopiert alle Dateien im aktuellen Verzeichnis auf Laufwerk B ins aktuelle Verzeichnis von Laufwerk A, wobei die ursprünglichen Dateinamen beibehalten werden.

COS( )

F

Cosine (Kosinus)

FUNKTION Gibt den Kosinuswert des angegebenen Winkels aus.

FORMAT `COS([Radiant])`

BESCHREIBUNG Gibt den Kosinuswert des angegebenen Winkels als Radiantwert aus.

Winkelangaben in Grad müssen in Radiantwerte mit Hilfe der folgenden Gleichung umgewandelt werden:

Radiant = Grad*π/180 (π = 3,141593)

VERWANDTE BEFEHLE SIN( ), TAN( ), ATAN( ), ATAN2( )

BEISPIEL

```
>PRINT COS(0.55)                Zeigt den Kosinuswert von 0,55 rad. an
.8525245
>PRINT COS(30*3.141593/180)     Zeigt den Kosinuswert von 30 Grad an
.8660254
```

Berechnung des Kosinus von 30 Grad in einem Programm

```
10 FUNCTION TEST
20 REAL A
30 A=30*3.141593/180
40 PRINT COS(A)
50 FEND
```

CTR( )

F

Counter (Zähler)

FUNKTION Gibt den Zählerstand eines Zählers aus.

FORMAT CTR([Nr. des Eingangs])

Die Nummer des Eingangs ist der Eingang, dessen ON/OFF-Zyklen gezählt werden sollen
Der ausgegebene Wert ist eine ganze Zahl von 0 bis 32767.

BESCHREIBUNG Gibt den Zählerstand des durch den Parameter "Nr. des Eingangs" definierten Zählers aus.

VERWANDTE BEFEHLE CTRESET

BEISPIEL

```
100 CTRESET 3                          Definiert und setzt Zähler 3 zurück
110 ON 0                               Ausgang 0 einschalten
120 TURN:
130 IF CTR(3)<5 THEN GOTO TURN
140 OFF 0                              'Wenn der Zählerstand 5 ist,
                                       Ausgang 0  ausschalten
```

C

CTRESET

> S

Counter Reset (Zähler zurücksetzen)

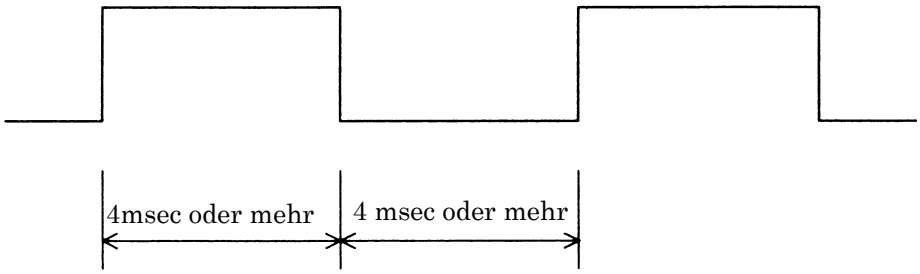
FUNKTION Setzt einen Zähler zurück

FORMAT CTRESET[Nr. des Eingangs]

Die Nummer des Eingangs muß eine ganze Zahl von 0 bis 127 sein.
Die Anzahl der Zähler kann maximal 16 betragen.

BESCHREIBUNG Definiert den angegebenen Eingang als Zähler und startet den Zähler. Ist der angegebene Eingang bereits als Zähler definiert, wird er zurückgesetzt und neu gestartet.

Wird der Eingang von Off auf On geschaltet, wird der Zähler um 1 hochgesetzt. Die folgende Abbildung zeigt die Mindest-Impulsdauer.



Bei Ausschalten des Roboters werden alle Zähler freigegeben.

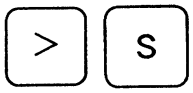
Mit Hilfe des Befehls CTR lassen Sie den Zählerstand anzeigen.

VERWANDTE BEFEHLE CTR( )

BEISPIEL

```
100 CTRESET 3                          Definiert und setzt Zähler 3 zurück
110 ON 0                               Ausgang 0 einschalten
120 TURN:
130 IF CTR(3)<5 THEN GOTO TURN
140 OFF 0                              Wenn der Zählerstand 5 ist,
                                       Ausgang 0 ausschalten
```

CURSOR



FUNKTION Schaltet die Anzeige des Cursors (an der Bedieneinheit) ein bzw. aus.

FORMAT

```
CURSOR |ON  |
       |OFF|
```

BESCHREIBUNG Bei der Einstellung ON wird der Cursor angezeigt.; bei der Einstellung OFF wird der Cursor nicht angezeigt.

Beim Einschalten des Roboters ist die ursprüngliche Einstellung OFF.

VERWANDTE BEFEHLE CHARSIZE, CLS, NORMAL, REVERSE, OPUNIT, OPU PRINT

CURVE

> S

FUNKTION

Erstellt eine Datei zur freien CP-Steuerung einer Kurvenbewegung (CP=Continuous Path=kontinuierlicher Weg).

FORMAT

```
CURVE {[Laufwerk]}{[Pfadangabe]}[Dateiname],|O|,[Modus]~
                                            |C|
                                  [Achse],[Kurvenpunktspezifikation]
```

Dateinamenerweiterungen sind nicht erlaubt.
Wählen Sie zur Moduseinstellung eine ganze Zahl zwischen 0 und 3.
Wählen Sie als Achse eine ganze Zahl von 2 bis 4.

BESCHREIBUNG

Erstellt aus einer Serie von festgelegten, aufeinanderfolgenden Punkten eine Datei zur freien CP-Bewegung. Mit Hilfe dieser Datei werden CVMOVE-Kurvenbewegungen ausgeführt.
Es ist nicht notwendig, vor dem CURVE-Befehl Einstellungen zur Geschwindigkeit bzw. Beschleunigung vorzunehmen. Geschwindigkeit und Beschleunigung können jederzeit vor Ausführung des CVMOVE-Befehls geändert werden.

Über die Befehle BASE oder LOCAL definierte Punkte können in die Serie eingesetzt werden, um die Kurve an die gewünschte Position zu verschieben.

Werden Laufwerk und Pfad angegeben, wird die Datei im angegebenen Verzeichnis auf dem angegebenen Laufwerk erstellt. Werden diese Angaben ausgelassen, wird die Datei im aktuellen Verzeichnis auf dem aktuellen Laufwerk angelegt.

Der Dateiname darf aus maximal 8 alphanumerischen Zeichen und/ oder Unterstrichen bestehen. Beim Erstellen der Datei hängt CURVE automatisch die Erweiterung .CRV an.

|O| bzw. |C| definieren offene bzw. geschlossene Kurven.
Bei einer offenen Kurve hält die Bewegung beim angegebenen Endpunkt.
Bei einer geschlossenen Kurve durchläuft die Bewegung den angegebenen Endpunkt und stoppt erst nach der Rückkehr zum Startpunkt.

Folgende Moduseinstellungen (Stop am Endpunkt und Tangentialkorrekturen) sind möglich:

Moduseinstellung	Stop am Endpunkt	Tangentialkorrektur
0	Ja	Nein
1	Ja	Nein
2	Ja	Ja
3	Nein	Ja

Bei der Tangentialkorrektur wird eine kontinuierliche Werkzeugausrichtung der Tangente an die Kurve in der XY-Ebene vorgenommen. Diese Option wird angegeben, wenn Werkzeuge installiert sind, die eine ständige Tangentialausrichtung erfordern wie z.B. Abschneidevorrichtungen.
Da Tangentialkorrekturen bei geschlossenen Kurven eine komplette 360-Grad-Drehung des Werkzeugs erfordern, muß vor Ausführung des Befehls CVMOVE zuerst der Bereich mit Hilfe des Befehls RANGE vergrößert werden.

Die Anzahl der Achsen muß eine ganze Zahl zwischen 2 und 4 sein.
Bei Auswahl von 2 wird eine Kurve nur in der XY-Ebene ohne Z-Achsenbewegung oder U-Achsenrotation erzeugt, bei Auswahl von 3 liegt die erzeugte Kurve im XYZ-Raum ohne U-Achsenrotation und bei 4 wird eine Kurve im XYZ-Raum mit einer U-Achsenrotation erzeugt.

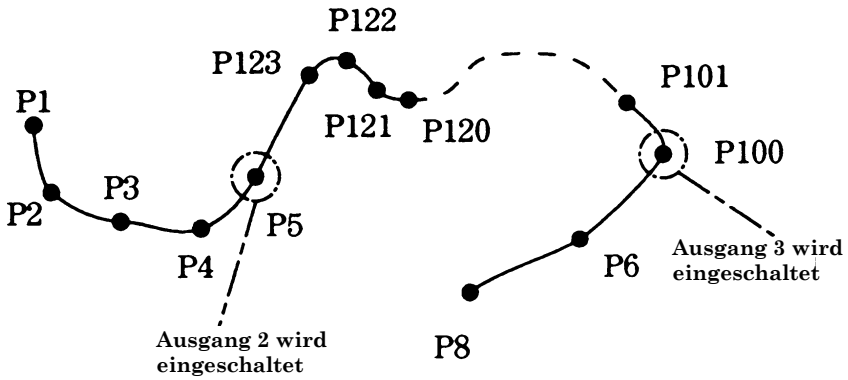
Spezifikationen zu den Kurvenpunkten (individuelle Punkte und/oder Serien von kontinuierlich ansteigenden bzw. absteigenden Punkten) werden durch Kommas getrennt. Eine kontinuierlich ansteigende bzw. absteigende Punktserie kann durch einen Bindestrich (-) zwischen dem ersten und dem letzten Punkt der Serie gekennzeichnet werden. So ist z.B. (P1-P5) identisch mit (P1, P2, P3, P4, P5).
Einschließlich der Zwischenpunkte bei kontinuierlichen Serien können für geschlossene Kurven zwischen 3 und 50 Punkte angegeben werden, bei offenen Kurven zwischen 4 und 200 Punkte.

Der CURVE-Befehl darf maximal fünf Ein-/ bzw. Ausgangsanweisungen zur Ausführung während der Kurvenbewegung, jeweils durch Kommas getrennt, enthalten. Die Ein-/ bzw. Ausgangsanweisung wird jeweils dann ausgeführt, wenn der Arm den Punkt (oder den Endpunkt einer Serie) durchläuft, der der E/A-Anweisung vorangeht (siehe Beispiel).

VERWANDTE BEFEHLE

CVMOVE, FILES

```
BEISPIEL
>CURVE BEISPIEL, 0, 0, 4, P1-P5, ON 2, P123-P100, ON 3, P6, P8      Definiert die
                                                                    u.a. Kurve
>FILES
  .
  .
  .
BEISPIEL CRV  6
  .
  .
  .
space   65
>JUMP P1
>CVMOVE BEISPIEL
>_
```



CVMOVE

> S

Curve Move (Kurvenbewegung)

FUNKTION

Führt eine mit dem Befehl CURVE erzeugte Datei zur freien CP-Bewegung aus (CP=Continuous Path=Kontinuierlicher Weg).

FORMAT

```
CVMOVE {[Laufwerk]}{[Pfadangabe]}[Dateiname]
```

BESCHREIBUNG

Führt freie CP-Bewegungen anhand von CURVE-Dateien aus.

Werden Laufwerk und Pfad angegeben, sucht CVMOVE im angegebenen Verzeichnis auf dem angegebenen Laufwerk nach Dateien. Werden diese Angaben ausgelassen, sucht CVMOVE die Datei im aktuellen Verzeichnis auf dem aktuellen Laufwerk.

Dateinamen werden ohne die Erweiterung (.CRV) angegeben.

Bei Dateien mit freien Kurven, die mit Punktdaten erstellt wurden, denen bereits lokale Attribute zugeordnet wurden, können die Bewegungspositionen mit Hilfe der Befehle LOCAL bzw. BASE geändert werden.

Der Befehl CVMOVE kann vorab keine Überprüfung des Trajektoriebereichs durchführen. Das bedeutet, auch wenn die Zielpositionen innerhalb des erlaubten Bereichs liegen, kann es vorkommen, daß der Roboter in der Bewegung versucht, den erlaubten Bereich zu verlassen und dadurch mit einem heftigen Ruck anhält, so daß der Manipulator beschädigt wird. Um dies zu verhindern, sollten Sie zuvor den Bereich mit langsamer Geschwindigkeit überprüfen.

Der Befehl CVMOVE verwendet die unter SPEEDS bzw. ACCELS festgelegten Werte für Geschwindigkeit bzw. Beschleunigung.

VERWANDTE BEFEHLE

SPEEDS, ACCELS, CURVE

BEISPIEL

```
>CURVE BEISPIEL,0,0,4,P1-P5,ON 2,P123-P100,ON 3,P6,P8      Definiert die
                                                           Kurvenparameter
>FILES
  .
  .
  .
BEISPIEL CRV        6
  .
  .
  .
space           65
>JUMP P1                                                   Sprungbewegung
                                                           zum Startpunkt
>CVMOVE BEISPIEL                                           Ausführung der
                                                           Bahnbewegung
>_
```

CX(P) CY(P), CZ(P), CU(P)

F

Coordinate of X-axis (Koordinate der X-Achse)

FUNKTION Gibt die Koordinaten der X-, Y-, Z- bzw. U-Achse eines angegebenen Punktes aus.

FORMAT

(1)
```
C|X|(P[Punktnummer])
  |Y|
  |Z|
  |U|
```

(2)
```
C|X|(P*)
  |Y|
  |Z|
  |U|
```

BESCHREIBUNG

(1) Gibt den Koordinatenwert der X-, Y-, Z- bzw. U-Achse eines angegebenen Punktes aus.

(2) Gibt alle Koordinatenwerte der aktuellen Position aus.

VERWANDTE BEFEHLE PLS( )

BEISPIEL

```
>P1=0, 0, 0, 0
>P2=100, 200, 300, -50, 180
>PRINT CX (P1)
0
>PRINT CY (P2)
200
>PRINT CZ (P2)
-50
>PRINT CU (P*)
-56.234
>
```

C

DATE

(Datum)

FUNKTION Definiert das aktuelle Datum bzw. zeigt es an.

FORMAT

(1) `DATE [Monat]-[Tag]-[Jahr]`

(2) `DATE`

BESCHREIBUNG

(1) Definiert das aktuelle Jahr, den Monat und den Tag. DATE stellt automatisch den aktuellen Wochentag ein.

Dieses Datum wird für die Dateienstatistik verwendet.

(2) Zeigt das aktuelle Datum an.

VERWANDTE BEFEHLE TIME

BEISPIEL

```
>DATE
Current date is Sat 8-20-1994

>DATE 5-8-1995
>DATE
Current date is Mon 5-08-1995
>
```

DEL, ERASE

Delete (Löschen)

FUNKTION Löscht eine oder mehrere Dateien.

FORMAT

```
DEL {[Laufwerk]}{[Pfadangabe]}{[Dateiname]}
```

Mindestens einer der Parameter [Daten] muß angegeben werden.
Jeder Dateiname muß mit der Dateinamenerweiterung angegeben werden.

BESCHREIBUNG Löscht die angegebene(n) Datei(en).

Dateiname und Erweiterungen können Platzhalter (*, ?) enthalten. Wird anstelle eines Dateinamen *.* eingegeben, erscheint die folgende Meldung:

Are you sure (Y/N)?

Wollen Sie tatsächlich alle Dateien im aktuellen Verzeichnis löschen, geben Sie Y bzw. y ein.
Wollen Sie keine Datei löschen, geben Sie N bzw. n ein.

Wird kein Laufwerk angegeben, werden mit DEL die Dateien auf dem aktuellen Laufwerk gelöscht.

Wird kein Pfad angegeben, löscht DEL die Dateien im aktuellen Verzeichnis.

Wird kein Dateiname angegeben, löscht DEL alle Dateien im aktuellen Verzeichnis. Dies entspricht der Eingabe von *.* anstelle eines Dateinamen.

VERWANDTE BEFEHLE KILL

BEISPIEL

```
>DEL B:\BAK
>DEL *.PNT
```

DELETE, DELET

(Löschen)

FUNKTION Löscht eine oder mehrere Programmzeilen.

FORMAT

```
DELETE  |[Zeilennummer]|                              |
        |[Erste Zeilennummer] -                       |
        |[Erste Zeilennummer]-[letzte Zeilennummer]|
        |                    -[letzte Zeilennummer]|
```

BESCHREIBUNG Löscht die angegebenen Programmzeilen wie folgt:

[Zeilennummer]
Löscht die angegebene Zeilennummer.
[Erste Zeilennummer]-
Löscht alle Zeilennummern ab der angegebenen Zeilennummer bis zum Programmende.
[Erste Zeilennummer]-[letzte Zeilennummer]
Löscht alle Zeilen ab der angegebenen ersten Zeilennummer bis und einschließlich der angegebenen letzten Zeilennummer.
-[letzte Zeilennummer]
Löscht alle Zeilen bis zur und einschließlich der angegebenen letzten Zeilennummer.

VERWANDTE BEFEHLE NEW, PDEL

BEISPIEL

```
>DELETE 30-40
```
Löscht ab Zeile 30 bis einschließlich Zeile 40.

```
>LIST
 10 FUNCTION MAIN
 20 JUMP P1
 50 END
```

```
10 FUNCTION MAIN
20 JUMP P1
30 WAIT 3
40 JUMP P5
50 END
```

```
>DELETE 100
```
Löscht Zeile 100

```
>DELETE 200-
```
Löscht alle Zeilen ab Zeile 200 aufwärts

DIR

Directory (Verzeichnis)

FUNKTION Zeigt den Inhalt eines Verzeichnisses an.

FORMAT `DIR {[Laufwerk]}{[Pfadangabe]}{[Dateiname]}{/P}{/W}`

BESCHREIBUNG Zeigt in dem unten dargestellten Format die folgenden Informationen zu den angegebenen Verzeichnissen und Dateien an: Dateiname bzw. Verzeichnisname, Dateigröße sowie Datum und Uhrzeit der letzten Bearbeitung.

AUTO	BAT	12345	2-18-94	1:00
↑	↑	↑	↑	↑
Dateiname	Erweiterung	Dateigröße (in Byte)	Datum der letzten Bearbeitung	Uhrzeit der letzten Bearbeitung

Bei Unterverzeichnissen wird anstelle der Dateigröße <DIR> angezeigt.

Wird keine Laufwerksangabe gemacht, arbeitet DIR auf dem aktuellen Laufwerk. Bei Auslassung der Pfadangabe arbeitet DIR im aktuellen Verzeichnis.

Dateiname und Erweiterung können Platzhalter (*, ?) enthalten. Werden Dateiname und Erweiterung ausgelassen, entspricht dies der Eingabe von *.*. In diesem Fall werden die Informationen zu allen Dateien im angegebenen Laufwerk und Verzeichnis angezeigt.

Sie können auch entweder den Dateinamen oder die Erweiterung weglassen. Dies entspricht der Verwendung des Platzhalters * anstelle des Dateinamen bzw. der Erweiterung. Beachten Sie jedoch, daß bei Angabe eines Dateinamen ohne Erweiterung, der identisch ist mit einem Verzeichnisnamen im selben Pfad, DIR in das entsprechende Verzeichnis wechselt. Das heißt, es werden alle Informationen zu den Unterverzeichnissen und Dateien in diesem Verzeichnis angezeigt.

Die folgenden Befehle sind gleichwertig:

Befehl	Gleichwertiger Befehl
DIR	DIR *.*
DIR .PRG	DIR *.PRG

/P aktiviert die seitenweise Anzeige. Es wird jeweils eine Bildschirmseite mit Informationen angezeigt. Zur Anzeige der nächsten Bildschirmseite drücken Sie die Leertaste.

/W aktiviert die weitformatige Anzeige. Es werden nur die Dateinamen, jeweils 5 in einer Zeile, angezeigt.

/P/W aktiviert die seitenweise weitformatige Anzeige. Es wird eine Bildschirmseite mit Informationen angezeigt. Zur Anzeige der nächsten Bildschirmseite drücken Sie die Leertaste. Es werden nur die Dateinamen angezeigt, jeweils 5 in einer Zeile.

VERWANDTE BEFEHLE FILES

BEISPIEL

```
>DIR
>DIR B:*.PRG
>DIR TEST.OBJ
```

DLOAD, DLO

> S

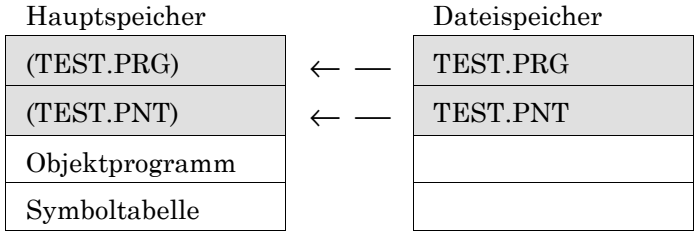
Disk Load (Diskette laden)

FUNKTION Lädt die angegebenen Dateien in den Hauptspeicher.

FORMAT

```
DLOAD "{[Laufwerk]}{[Pfad]}[Dateiname]{|.PRG|}]"
                                       |.PNT|
```

BESCHREIBUNG Lädt die angegebene Datei entweder aus dem Dateispeicher oder von der Diskette in den Hauptspeicher.



Wird die Dateinamenerweiterung angegeben, wird nur die entsprechende Datei geladen. Wird die Erweiterung .PRG angegeben, wird nur das Quellprogramm [Dateiname.PRG] geladen. Bei Angabe der Erweiterung .PNT werden nur die Positionsdaten [Dateiname.PNT] geladen.

Wird keine Dateinamenerweiterung angegeben, werden die Dateien [Dateiname.PRG] und [Dateiname.PNT] nacheinander geladen. Falls eine dieser Dateien nicht existiert, erfolgt ein Fehler. Existiert zwar die Datei [Dateiname.PRG] jedoch nicht die Datei [Dateiname.PNT], erfolgt der Fehler nach dem Laden der Datei [Dateiname.PRG].

Die Anweisung DLOAD "[Dateiname.PNT]" kann in ein Programm eingefügt werden, um die Positionsdatendateien während der Programmausführung laden zu lassen. Die Anweisung DLOAD "[Dateiname.PRG]" ist jedoch nicht erlaubt.

Bei Auslassung der Laufwerksangabe lädt DLOAD die Datei(en) aus dem aktuellen Laufwerk.
Wird kein Pfad angegeben, lädt DLOAD die Datei(en) aus dem aktuellen Verzeichnis.

DLOAD überschreibt das aktuelle Quellprogramm sowie die Positionsdaten im Hauptspeicher. Daher sollten Sie das aktuelle Quellprogramm und die Positionsdaten falls notwendig in den Dateispeicher oder auf eine Diskette kopieren.

VERWANDTE BEFEHLE FILES, DIR, DSAVE, DMERGE

BEISPIEL

```
>DLOAD "TEST"
>DLOAD "TEST.PRG"
>DLOAD "B:TEST.PNT"
>
```

DMERGE

> S

Disk Merge (Diskette mischen)

FUNKTION Lädt die angegebene(n) Datei(en) in den Hauptspeicher und mischt sie mit dem Quellprogramm bzw. den Positionsdaten.

FORMAT

```
DMERGE "{[Laufwerk]}{[pfadangabe]}[Dateiname{|.PRG|}]"
                                          |.PNT|
```

BESCHREIBUNG Lädt die angegebene(n) Datei(en) entweder aus dem Dateispeicher oder von der Diskette in den Hauptspeicher und mischt sie mit dem Quellprogramm bzw. den Positionsdaten.

Werden Dateiname und Erweiterung angegeben, wird nur diese Datei geladen und gemischt.
Gültig sind nur die Dateinamenerweiterungen .PRG und .PNT.
Wird nur der Dateiname ohne Erweiterung angegeben, werden die Dateien [Dateiname.PRG] und [Dateiname.PNT] nacheinander geladen und gemischt. Falls eine dieser Dateien nicht existiert, erfolgt ein Fehler. Existiert zwar die Datei [Dateiname.PRG] jedoch nicht die Datei [Dateiname.PNT] erfolgt der Fehler nach dem Laden und Mischen der Datei [Dateiname.PRG].

Falls die geladene Datei eine Zeilennummer enthält, die einer Zeilennummer des Quellprogramms im Hauptspeicher entspricht, wird der Inhalt der Zeile des Quellprogramms durch den Inhalt der geladenen Zeilennummer ersetzt. Dies gilt in ähnlicher Weise für die Positionsdaten. Entspricht eine geladene Punktnummer einer Punktnummer im Hauptspeicher, werden die Positionsdaten im Hauptspeicher durch die geladenen Positionsdaten ersetzt.

Die Anweisung DMERGE "[Dateiname.PNT]" kann in ein Programm eingefügt werden, um die Positionsdatendateien während der Programmausführung zu mischen. Die Anweisung DMERGE "[Dateiname.PRG] ist nicht erlaubt.

Wird die Angabe des Laufwerks ausgelassen, lädt DMERGE die angegebene(n) Datei(en) aus dem aktuellen Verzeichnis und mischt sie.

Bei Auslassung der Pfadangabe lädt DMERGE die angegebene(n) Datei(en) aus dem aktuellen Verzeichnis und mischt sie.

VERWANDTE BEFEHLE FILES, DIR, DSAVE

BEISPIEL

```
>DMERGE "TEST.PRG"
>LIST
10 FUNCTION TEST
20 JUMP P2     ⇐
30 WAIT 1
35 ON 2
40 FEND
45 FEND
```

Hauptspeicher

```
10 FUNCTION MAIN
20 JUMP P1
30 WAIT 1
40 FEND
```

\+

Dateispeicher

```
10 FUNCTION TEST
20 JUMP P2
35 ON 2
45 FEND
```

DSAVE, DSA

Save to Disk (auf die Festplatte bzw. Diskette speichern)

FUNKTION

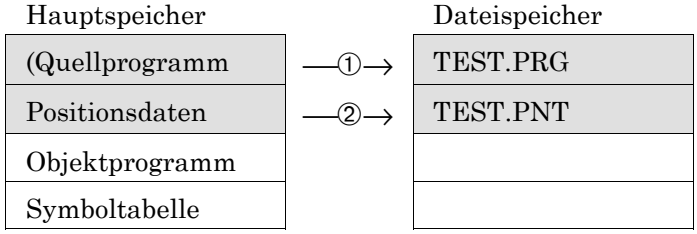
Speichert das Quellprogramm sowie die Positionsdatendateien im Hauptspeicher in den Dateispeicher bzw. auf die Diskette.

FORMAT

```
DSAVE "{[Laufwerk]}{[Pfadangabe]}[Dateiname{|.PRG|}]"
                                          |.PNT|
```

BESCHREIBUNG

Speichert das im Hauptspeicher befindliche Quellprogramm und die Positionsdatendateien, benennt die Dateien wie angegeben und speichert sie im Dateispeicher bzw. auf eine Diskette.



Die folgenden Konventionen gelten für die Vergabe der Dateinamen:

- maximal acht Zeichen,
- erlaubte Zeichen sind:
  alphanumerische Zeichen
  Symbole wie z.B. ! # $ % & ( ) { } - _ @ ^
- es können sowohl Klein- als auch Großbuchstaben verwendet werden; Kleinbuchstaben werden automatisch in Großbuchstaben umgewandelt.

Als Dateinamenerweiterung sind nur .PRG und .PNT zugelassen.

Wird der Parameter [Dateiname.PRG] angegeben, wird nur das Quellprogramm gespeichert. Bei Angabe von [Dateiname.PNT] entsprechend nur die Positionsdatendatei. Wird keiner der Parameter angegeben, werden sowohl die .PRG- als auch die .PNT-Dateien gespeichert.

Wird der DSAVE-Befehl ausgeführt, obwohl der Hauptspeicher weder ein Quellprogramm noch eine Positionsdatendatei enthält, enthalten die gespeicherten Dateien keinerlei Daten.

Die Anweisung DSAVE "[Dateiname.PNT]" kann in ein Programm eingefügt werden, um die Positionsdaten während der Programmausführung zu sichern.

In diesem Fall können auch andere Dateinamenerweiterungen als .PNT verwendet werden. Sie sollten jedoch die Erweiterung .PNT für Positionsdatendateien verwenden.

Bei Auslassung der Laufwerksangabe speichert DSAVE die Dateien auf dem aktuellen Laufwerk; wird kein Pfad angegeben, werden die Dateien im aktuellen Verzeichnis gespeichert.

Existiert im angegebenen Verzeichnis auf dem angegebenen Laufwerk bereits eine Datei mit demselben Dateinamen, erfolgt ein Fehler. Löschen Sie in diesem Fall die nicht benötigten Dateien mit dem Befehl DEL.

VERWANDTE BEFEHLE DIR, FILES, DEL, KILL, DLOAD

BEISPIEL

```
>DSAVE "TEST.PRG"
>DSAVE "TEST.PNT"
>KILL "TEST"
>DSAVE "TEST"
>
```

DSW( )

F

Dip-Switch (DIP-Schalter)

FUNKTION Gibt den Status der Tastschalter der Bedieneinheit (OPU) als Dezimalwert aus.

FORMAT `DSW([Portnummer])`

Die Portnummer kann eine ganze Zahl von 2 bis 6 sein.

BESCHREIBUNG Gibt den Status von REMOTE1 - 3 als Dezimalwert ausgegeben

Bei dieser Version von SPEL III haben die Portnummern 0 und 1 keine Bedeutung. (Wenn Sie DSW( ) eingeben und Portnummer 0 oder 1 wählen, wird als Wert 0 ausgegeben.)

DSW(2)	Bit 0	RESET	REMOTE2 (OPU-300)
	1	PAUSE	
	2	START	
	3	nicht definiert	
	4	nicht definiert	
	5	nicht definiert	
	6	nicht definiert	
	7	MONITOR	
DSW(3)	Bit 0	↑	
	1	↓	
	2	←	
	3	→	
	4	F1	
	5	F2	
	6	F3	
	7	F4	
DSW(4)	Bit 0	nicht definiert	
	1	Auto-Modus	
	2	TEACH-Modus	
	3	nicht definiert	
	4	nicht definiert	
	5	nicht definiert	
	6	Schutzabschrankung	REMOTE1
	7	Notaus	REMOTE1, REMOTE2, TEACH

DSW(5)	Bit 0	RESET	REMOTE3
	1	PAUSE	
	2	START	
	3	HOME	
	4	Programm-Nr. 2^0	
	5	Programm-Nr. 2^1	
	6	Programm-Nr. 2^2	
	7	Programm-Nr. 2^3	
DSW(6)	Bit 0	MCAL	
	1	Motor ein	
	2	Motor aus	
	3	nicht definiert	
	4	nicht definiert	
	5	nicht definiert	
	6	nicht definiert	
	7	nicht definiert	

DSW(5) und DSW(6) von REMOTE3 funktionieren nur für die Portnummern, die als REMOTE3 konfiguriert wurden.Wenn Sie DSW( ) für eine Portnummer eingeben, die nicht als REMOTE3 konfiguriert wurde, wird als Wert 0 ausgegeben.

VERWANDTE BEFEHLE

PRGNO, OPUNIT

BEISPIEL

Wenn Sie die folgenden Anweisungen in Ihr Programm einfügen, können Sie prüfen lassen, ob die Taste F1 an der OPU-300 gedrückt wurde.

```
10 FUNCTION CHECKSW
20 IF (DSW (3) AND &H10)=&H10 THEN ...
30 IF DSW (3) = 16 THEN PRINT "F1 ist betätigt"
```

ECLR

S

Error Clear (Fehler löschen)

FUNKTION Löscht einen Fehlerstatus.

FORMAT ECRL

BESCHREIBUNG Löscht einen Fehlerstatus (Fehlernummer).

ECLR wird zusammen mit dem Befehl ONERR verwendet.

Wenn Sie ONERR in ein Programm einfügen, ist es möglich, im Falle eines Fehlers bei der Programmabarbeitung in ein Unterprogramm zur Fehlerbehebung zu verzweigen.

Wollen Sie nach Ablauf des Unterprogramms wieder in das Hauptprogramm zurückkehren, muß der Fehlerstatus mit Hilfe des Befehls ECLR gelöscht werden. Wird ECLR nicht verwendet, bleibt der Fehlerstatus bestehen und ruft dadurch das Unterprogramm zur Fehlerbehebung erneut auf.

VERWANDTE BEFEHLE ONERR

BEISPIEL

```
10 ONERR ERR_SUB          Erfolgt bei der Programmausführung ein
                          Fehler, zu Zeile 70 verzweigen.
20 FOR I=0 TO 199
30 JUMP PI
40 NEXT I
50 END
60 '
70 ERR_SUB:
80 ECLR                   Löscht den Fehlerstatus
.
.
.
220 RETURN
```

EDIT

(Editieren)

FUNKTION Schaltet in den Editiermodus.

FORMAT `EDIT {[Laufwerk]}{[Pfadangabe]}[Dateiname] {.[Erweiterung]}`

BESCHREIBUNG Schaltet in den Editiermodus um Textdateien wie z.B. Stapeldateien zu bearbeiten. Es gibt zwei Befehle, mit denen Sie den Editiermodus verlassen können, und zwar QUIT und END. Bei beiden Befehlen wird in den Programmiermodus zurückgeschaltet.

Im Programmiermodus können zwar Programm- und Positionsdatendateien editiert werden, jedoch keine Textdateien (z.B. Stapeldateien). Diese Art Dateien können nur im Editiermodus bearbeitet werden.

Wenn Sie eine Datei bearbeiten wollen, die keine Dateinamenerweiterung hat, geben Sie einfach den Namen ohne Erweiterung an.

```
End of input file
```

Existiert der angegebene Name jedoch nicht, erscheint die folgende Meldung als Hinweis, daß eine neue Datei erstellt wird:

```
New file
```

Zum Verlassen des Editiermodus verwenden Sie einen der modusspezifischen Befehle: bei Eingabe von QUIT verlassen Sie den Editiermodus ohne die Datei zu speichern, bei END wird die Datei zuvor gesichert, dann verläßt SPEL III den Editiermodus. Nachdem Sie den Editiermodus verlassen haben, erscheint je nach verwendetem Befehl eine der folgenden Meldungen:

```
>QUIT
Edit End ...
>
>END
Edit End ... file save
>
```

Wird eine Datei über den Befehl EDIT geladen, wird jeder Zeile eine Zeilennummer vorangestellt, beginnend mit 10 und um jeweils 10 ansteigend. Wird die Datei jedoch mit END gesichert, werden die Zeilennummern wieder herausgenommen.

Im Editiermodus können nur die folgenden Befehle verwendet werden:

Edit-Befehle (Editieren) LIST DELETE RENUM NEW FREE
Exit-Befehle (Modus verlassen) QUIT (Zurück zum Programmiermodus ohne Datensicherung) END (Zurück zum Programmiermodus nach Datensicherung)
Dateiverwaltung COPY DIR DEL (ERASE) RENAME CD (CHDIR) MD (MKDIR) RD (RMDIR) RENDIR

SPEL III hält für den Editier- und den Programmiermodus jeweils separate Speicherbereiche frei. Dadurch können Sie über den EDIT-Befehl im Editiermodus arbeiten, ohne daß die Daten im Quellprogrammbereich des Hauptspeichers verlorengehen.

+ Die Anzahl Zeichen pro Zeile darf maximal 79 betragen. Das bedeutet, wenn die Datei geladen wird, darf eine Zeile nicht mehr als 74 Zeichen (ohne Zeilennummer) enthalten. Bei Überschreiten dieser Anzahl werden alle weiteren Zeichen in der entsprechenden Zeile ignoriert.

Im Editiermodus sind nur die zuvor genannten Editierbefehle zulässig. Wenn Sie andere Befehle verwenden wollen (z.B. XQT), kehren Sie mit Hilfe des Befehls QUIT oder END in den Programmiermodus zurück. Solange der Editiermodus aktiv ist, kann das Programm nicht ausgeführt werden (selbst bei eingeschaltetem AUTO-Modus).

BEISPIEL

```
EDIT CNFG.SYS
New file
>10 TASK=8
>20 ERRBUF=20
>30 LINBUF=256
>END
Edit End ... file save
>
```

END

> S

FUNKTION Beendet die Programmausführung.

FORMAT END

BESCHREIBUNG Kennzeichnet das Ende eines Programms; das Programm wird nur bis zu dieser Zeilennummer ausgeführt.

Enthält ein Programm Unterprogramme, wird das Hauptprogramm durch den Befehl END am Ende vom nachfolgenden Unterprogramm getrennt. Wird END jedoch ausgelassen, wird das Unterprogramm als Teil des Hauptprogramms ausgeführt.

BEISPIEL

```
10 FUNCTION TEST
20 JUMP P5
30 IF SW(4)=0 THEN GOSUB BUZZER
40 JUMP P6 ;ON 0 ;WAIT 0.5
50 JUMP P7 ;OFF 0 ;WAIT 0.5
60 GOTO 20
70 END                                  Programmende
80 '
90 BUZZER:
100 ON 4
110 WAIT SW (4)=1 OR SW(5)=1
120 OFF 4
130 WAIT SW(4)=1
140 RETURN
150 '
160 FEND
```

Im obigen Beispiel könnte END auch entfallen. Werden Zeile 60 und Zeile 70 wie folgt verändert, so ist END unbedingt erforderlich:

```
60 IF SW(5) = 1 THEN GOTO 20
70 END
```

ENTRY

S

FUNKTION Definiert globale Variablen.

FORMAT `ENTRY [Variablentyp] [Variablenname] {,[Variablenname]}n`

BESCHREIBUNG Definiert die angegebene(n) Variable(n) als globale Variable(n) und speichert die entsprechenden Werte in einer speziell zugeordneten Symboldatei.

Mit dem Befehl ENTRY unterstützt SPEL III zusätzlich zu den lokalen Variablen (Variablen, die nur innerhalb **einer** Datei verwendet werden können) auch globale Variablen, d.h. Variablen, die für alle Dateien gelten und hierfür ein bestimmtes Programm enthalten.

Ist eine globale Variable einmal definiert, kann diese in einer anderen Datei verwendet werden. Dazu fügen Sie dort den Befehl EXTERN ein.

Die Verwendung globaler Variablen vereinfacht das Erstellen neuer Programme aus verschiedenen Dateien durch modulare Kompilierung und Verknüpfungen (linking).

VERWANDTE BEFEHLE EXTERN, LINK

BEISPIEL

FILE1 (MAIN.PRG)

```
10 FUNCTION MAIN
20 INTEGER I                        I ist eine lokale Variable
30 ENTRY REAL WORK (100)            Globale Variable definieren
40 EXTERN FUNCTION SUB_TASK
50 WORK(5)=10.3
 .
 .
 .
1000 FEND
```

FILE2 (INIT.PRG)

```
2000 FUNCTION SUB_TASK
2010 J=1
2020 EXTERN REAL WORK(100)          Bezug auf globlale Variable
                                    definieren
2030 A=J+WORK(5)                    Wert von WORK(5) aus FILE1
                                    verwenden
 .
 .
 .
3000 FEND
```

ERA(0)

F

Error Axis (Fehlerachse)

FUNKTION Gibt die Nummer der Achse aus, bei der ein Fehler aufgetreten ist.

FORMAT ERA(0)

BESCHREIBUNG Gibt die Nummer der Achse an, bei der ein Fehler aufgetreten ist. Als Achsennummer wird eine Zahl von 0 bis 4 ausgegeben. Bei Ausgabe von 0 (Null) verursacht keine der Achsen einen Fehler.

VERWANDTE BEFEHLE ONERR, TRAP, ERR(0), ERL(0), ERT(0), ERD(0)

BEISPIEL

```
100 TRAP ERROR CALL ER_PRINT
 .
 .
 .
3000 FUNCTION ER_PRINT
3010 IF END (0)=0 OR ERD(0)=1 THEN
3020 PRINT #20, "Fehler Roboter aufgetreten."
3030 PRINT #20, "Tasknummer, in der Fehler auftrat ",ERT(0),"."
3040 IF ERA(0) THEN PRINT #20, "Achse, die Fehler verursachte
",ERA(0),"."
3050 ELSE
3060 PRINT #20, "Fehler trat auf bei RAIOC #",ERD(0),"."
3070 ENDIF
3080 PRINT #20, "Fehler-Nummer ist ",ERR(0),"."
3090 END
3100 FEND
```

ERL(0)

F

Error line number (Fehler-Zeilennummer)

FUNKTION Ausgabe der Zeilennummer, in der ein Fehler auftrat.

FORMAT ERL(0)

BESCHREIBUNG Gibt die Zeilennummer aus, in der ein Fehler aufgetreten ist.

ERL(0) wird gemeinsam mit dem Befehl ONERR verwendet. Bei Auftreten eines Fehlers in einem Programm mit der Befehlsschleife ONERR....RETURN kann es u.U. schwierig sein, die Fehlerquelle auszumachen. Mit dem Befehl ERR(0) wird die Zeilennummer ausgegeben, in der der Fehler auftrat.

Da der ECLR-Befehl innerhalb des Fehlerbearbeitungsprogramms ONERR die Fehler-Zeilennummer auf 0 zurücksetzt, sollten Sie darauf achten, daß Sie den Befehl ERL(0) vor ECLR ausführen.

VERWANDTE BEFEHLE ONERR, ECLR, ERR(0)

E

BEISPIEL

```
  .
  .
  .
.120 ONERR 1000
  .
  .
  .
1000 'Fehlerbearbeitungsroutine
1010 ELINE=ERL(0)
1020 ECODE=ERR(0)
1030 ECLR
1040 PRINT "Fehler"
1050 PRINT "   Zeile  =..", ELINE
1060 PRINT "   Nummer =  ", ECODE
1070 RETURN
```

ERR(0)

F

Error (Fehler)

FUNKTION Gibt eine Fehlernummer aus.

FORMAT `ERR(0)` (bei der Angabe in Klammern handelt es sich um die Ziffer 0).

BESCHREIBUNG ERR(0) wird gemeinsam mit dem Befehl ONERR verwendet. Bei Auftreten eines Fehlers in einem Programm mit der Befehlsschleife ONERR...RETURN ist es u.U. schwierig, die Fehlernummer auszumachen. Mit dem Befehl ERR(0) wird die Fehlernummer ausgegeben.

Da der ECLR-Befehl innerhalb des Fehlerbearbeitungsprogramms ONERR die Fehler-Zeilennummer auf 0 zurücksetzt, sollten Sie darauf achten, daß Sie den Befehl ERL(0) vor ECLR ausführen.

VERWANDTE BEFEHLE ONERR, ECLR, ERL(0)

BEISPIEL

```
10 ONERR 60
20 FOR I=0 TO 199
30 JUMP PI
40 NEXT I
50 END
60 '
70 'ERR SUB
80 A=ERR(0)
90 PRINT A
100 ECLR
110 RETURN
```

ERRHIST

Error History (Fehlerprotokoll)

FUNKTION Zeigt ein Fehlerprotokoll an.

FORMAT `ERRHIST {[Anzahl der anzuzeigenden Fehler]}`

Die Anzahl der anzuzeigenden Fehler kann eine ganze Zahl von 1 bis 50 sein. Der Standardwert beträgt 20.

BESCHREIBUNG Zeigt die zuletzt erfolgten Fehler an (maximal 50).

Die folgenden Informationen werden angezeigt:

Fehlernummer
Achse, auf die sich der Fehler bezieht
Tasknummer
RAIOC-Gerätenummer (Adresse), falls anwendbar
Datum und Uhrzeit des Fehlers

Ist die Anzahl der anzuzeigenden Fehler definiert, wird diese Anzahl Fehler angezeigt. Ist die Anzahl nicht definiert, werden die letzten 20 erfolgten Fehler angezeigt.

Wenn Sie in der Datei CONFIG.SYS ERRBUF=30 angeben, werden die letzten 30 Fehler angezeigt.

Wird der Fehler 2 (Syntax-Fehler) während der Programmausführung ausgegeben, werden nur diese Fehler aufgezeichnet.

Wird der Befehl ONERR verwendet, werden diese Fehler nicht aufgezeichnet.

Der Inhalt des ERRHIST-Puffers sowie die angezeigten Informationen bleiben auch nach dem Ausschalten erhalten.

Das Ausschalten wird mit Fehler 48 registriert.

BEISPIEL

```
>ERRHIST 7
Error (axis)   Line      Task      Dev     Date    Time
    5                                      10-21   20:04:48
    11                                     10-21   20:05:07
    2                                      10-21   20:11:09
    125 ( 2)                               10-21   20:11:18
    48                                     10-21   20:12:06
   125 (2)                                 10-22   20:12:26
   125 (2)     160       1                 10-22   13:01:15
>
```

ERRMSG$

F

FUNKTION Gibt die zur angegebenen Fehlernummer gehörenden Fehlermeldung aus.

FORMAT `ERRMSG$([Fehlernummer])`

BESCHREIBUNG Gibt die jeweilige Fehlermeldung aus (eine Liste der Fehlermeldungen finden Sie in der Fehlercode-Tabelle). Die Fehlermeldungen werden in der über Software-Schalter SS1 definierten Sprache ausgegeben.

VERWANDTE BEFEHLE ERR(0), ERL(0), ONERR, TRAP

BEISPIEL

```
>PRINT ERRMSG$ (124)
Numerischer Wert außerhalb des zulässigen Bereichs.
```

ERT(0)

> S

Error Task (Fehler-Task)

FUNKTION Gibt die Nummer der Task aus, bei der ein Fehler aufgetreten ist.

FORMAT ERT(0)

Die Zahl in Klammern ist die Ziffer 0 (Null).

BESCHREIBUNG Gibt die Nummer der Task aus, bei der ein Fehler aufgetreten ist.

Die ausgegebene Nummer ist eine Zahl zwischen 1 und 16.

VERWANDTE BEFEHLE TRAP, ERR(0), ERD(0), ERA(0), ERL(0)

BEISPIEL

```
100 TRAP ERROR ER_PRINT
 .
 .
 .
3000 FUNCTION ER_PRINT
3010 IF ERD(0)=0 OR ERD(0)=1 THEN
3020  PRINT #20, "Fehler Roboter aufgetreten."
3030  PRINT #20, "Tasknummer, in der Fehler auftrat ",ERT(0),"."
3040  IF ERA(0) THEN PRINT #20,"Achse, die Fehler verursachte
",ERA(0),"."
3050 ELSE
3060  PRINT #20, "Fehler trat auf bei RAIOC #",ERD(0),"."
3070 ENDIF
3080 PRINT #20, "Fehler-Nummer ist ",ERR(0),"."
3090 END
3100 FEND
```

E

EXTERN

S

External (Extern)

FUNKTION Definiert die Referenz auf globale Variablen.

FORMAT `EXTERN [Variablentyp] [Variablenname]{,[Variablenname]}n`

BESCHREIBUNG Setzt den Bezug auf eine oder mehrere globale Variablen, die mit der Anweisung ENTRY definiert wurden. Wenn Sie den EXTERN-Befehl in eine Datei einfügen, kann diese Datei auf den Wert einer globalen Variablen aus einer anderen Datei zugreifen.

VERWANDTE BEFEHLE ENTRY

BEISPIEL

FILE1 (MAIN.PRG)

```
10 FUNCTION MAIN
20 INTEGER I                          I ist eine lokale Variable
30 ENTRY REAL WORK (100)              Globale Variable definieren
40 EXTERN FUNCTION INIT
50 WORK(5)=10.3
.
.
.
1000 FEND
```

FILE2 (INIT.PRG)

```
2000 FUNCTION INIT
2010 J=1
2020 EXTERN REAL WORK(100)            Bezug auf globlale Variable
                                      definieren
2030 A=J+WORK(5)                      Wert von WORK(5) aus FILE1
                                      verwenden
.
.
.
3000 FEND
```

FILES

Dateien

FUNKTION Zeigt Dateiname und -größe der Dateien im Dateienspeicher bzw. auf einer Diskette an.

FORMAT `FILES {[Laufwerk]}{[Pfadname]}{[Dateiname]}`

BESCHREIBUNG Zeigt den Namen sowie die Größe der Dateien im Dateienspeicher bzw. auf einer Diskette an. Gleichzeitig wird die freie Speicherkapazität im Dateienspeicher angezeigt. Die Dateigröße wird in Kbyte angegeben.

Wird die Laufwerksangabe weggelassen, nimmt FILES das aktuelle Laufwerk an; wird kein Pfad angegeben, nimmt FILES das aktuelle Verzeichnis an.

Sowohl der Dateiname als auch die Dateinamenerweiterungen können sog. Platzhalter (*, ?) enthalten. Werden weder Dateiname noch Erweiterung angegeben, entspricht dies der Eingabe *.* statt des Dateinamens. FILES zeigt dann Dateiname und -größe aller Dateien und Verzeichnisse im angegebenen Laufwerk und Pfad an.

Wenn sich SPEL-80M im Offline-Modus befindet, entsprechen die DFILS-Funktionen dem Befehl FILES.

VERWANDTE BEFEHLE DIR, WIDTH, DLOAD, DSAVE, DMERGE, KILL

BEISPIEL

```
>FILES
01         PRG     2
01         PNT     1
01         OBJ     2
01         SYM     1
ABC        CRV     1
02         PRG     2
02         PNT     1
EXAMPLE    PRG     5
CNFG       SYS     1
AUTO       BAY     1
EXAMPLE    PNT     1
EXAMPLE2   PNT     1
   space   181
>
```

FIND

>

Suchen

FUNKTION Sucht nach einer Zeichenkette.

FORMAT `FIND {"}[Zeichenkette]{"} {{[Laufwerk]}{[Pfad]}[Dateiname]}`

Die Angabe der Dateinamenerweiterung ist unbedingt erforderlich.

BESCHREIBUNG Sucht nach einer Zeichenkette und zeigt die Zeile bzw. Zeilen in der angegebenen Datei an, in der die Zeichenkette enthalten ist.

Wenn Sie mit dem Befehl FIND eine Zeichenkette mit Kleinbuchstaben und Leerzeichen suchen wollen, muß die zu suchende Zeichenkette durch Anführungszeichen eingeschlossen sein. Werden keine Anführungszeichen gesetzt, sucht FIND trotz Angabe von Kleinbuchstaben nach Großbuchstaben!

Die zu suchende Zeichenkette kann auch Platzhalter (*, ?) enthalten. Wollen Sie mit dem FIND-Befehl nach Zeichenketten mit Platzhaltern (*, ?) oder dem umgekehrten Schrägstrich (\) suchen, muß jedem Zeichen (*, ?, \) ein umgekehrter Schrägstrich vorangehen und in der Zeichenkette mit *, \? bzw. \\ angegeben werden.

Der Dateiname bzw. die Erweiterung können Platzhalter enthalten (*, ?).

Wird kein Dateiname angegeben, sucht FIND im Hauptspeicherprogramm nach der angegebenen Zeichenkette und zeigt die entsprechende(n) Zeile(n) an.

Bei Auslassung der Laufwerksangabe sucht FIND auf dem aktuellen Laufwerk.

Bei Auslassung der Pfadangabe sucht FIND im aktuellen Verzeichnis.

BEISPIEL Als Beispiel-Dateiname ONOFF.PRG

```
>FIND L?D ONOFF.PRG
ONOFF.PRG
20 INTEGER LED
40 FOR LED=0 TO 31
50      ON LED
70 FOR LED=0 TO 31
80      OFF LED
>
FIND ST*SW ONOFF.PRG
ONOFF.PRG
30 INTEGER STOPSW
100 IF SW(STOPSSW)=0 THEN GOTO 40
>
>FIND "ON LED" ONOFF.PRG
50      ON LED
>
```

```
10 FUNCTION ONOFF
20 INTEGER LED
30 INTEGER STOPSW
40 FOR LED=0 TO 31
50      ON LED
60 NEXT
70 FOR LED=0 TO 31
80      OFF LED
90 NEXT
100 IF SW(STOPSW)=0 THEN  GOTO 40
110 FEND
```

FINE

> S

FUNKTION

Definiert den Bereich am Bewegungsendpunkt, in dem eine Bewegung als beendet erkannt wird.

FORMAT

```
FINE {[1. Achse],[2. Achse],[3. Achse],[4. Achse]}
```

Der Wert für die Achseneinstellung kann eine ganze Zahl zwischen 0 und 32.767 sein.

BESCHREIBUNG

Setzt für jede Achse den zulässigen Bereich in Pulsen. Die Positionen werden bestätigt, wenn sich alle Achsen innerhalb des angegebenen Bereiches befinden.

Die Überprüfung der Positionen beginnt, sobald die Sub-CPU alle Pulse für die Zielposition an das Servosystem gesandt hat. Durch die Servo-Verzögerung hat der Manipulator die Zielposition dann noch nicht erreicht. Die Überprüfung wird nach wenigen Millisekunden fortgesetzt, und zwar solange, bis sich jede Achse innerhalb des angegebenen Bereichs befindet. Nach Beendigung der Positionsüberprüfung wird die Steuerung an den nächsten Befehl übergeben.

Bei relativ großen Bereichen wird das Erreichen der Position recht früh bestätigt, was dazu führen kann, daß die Positionierung am Zielpunkt relativ ungenau ist.

Da die Servo-Verzögerung normalerweise weniger als ein paar tausend Pulse beträgt, hat eine Bereichseinstellung von wenigen tausend Pulsen oder mehr negative Auswirkungen auf die Positionierung.

Je nach herrschenden Bedingungen führen die FINE-Einstellungen, selbst bei der Einstellung 0, nicht unbedingt zu einer genaueren Positionierung.

Wurde für keine der vier Achsen eine Einstellung angegeben, zeigt FINE die aktuellen Einstellungen wie folgt an:

[1. Achseneinstellung]	[2. Achseneinstellung]
[3. Achseneinstellung]	[4. Achseneinstellung]

Bei Ausführung der Befehle MOTOR ON, SLOCK, SFREE oder VERINIT werden die FINE-Werte auf die Standardwerte zurückgesetzt. Diese Werte sind abhängig vom verwendeten Manipulatormodell.

VERWANDTE BEFEHLE

GO, JUMP, MOVE, ARC, CVMOVE, PULSE

BEISPIEL

```
>FINE  50,50,50,50

>
>FINE
       50    50
       50    50
```

FOR...NEXT

S

FUNKTION

Führt eine Reihe von Anweisungen mit der angegebenen Anzahl von Wiederholungen aus.

FORMAT

```
FOR [vn]=[Anfangswert] TO [Endwert] {STEP [Steigerungswert]}
   .
   .
   .
NEXT {[vn]}
```

(vn wird durch den Variablennamen ersetzt)

Für den Anfangs-, End- und Steigerungswert sind Eingaben von ± 0,000001 bis± 9999999 erlaubt. Es sind maximal 20 Verschachtelungen möglich.

BESCHREIBUNG

Führt eine Reihe von Anweisungen mit der angegebenen Anzahl von Wiederholungen aus.

Die Anzahl der wiederholten Schleifen (Angabe, wie oft die Anweisungen zwischen FOR und NEXT wiederholt werden) wird durch die Angaben für den Anfangs-, End- und Steigerungswert bestimmt.

Bei Weglassung des Steigerungswerts wird der Standardwert 1 angenommen. Der Steigerungswert kann auch negativ sein, allerdings muß dann der Anfangswert größer als der Endwert sein.

Verschachtelungen sind erlaubt; es dürfen maximal 9 FOR...NEXT-Schleifen in der ersten FOR-NEXT-Schleife enthalten sein. (Erläuterungen zu Verschachtelungen finden Sie im Abschnitt zum #include- Befehl).

Der Variablenname nach NEXT kann weggelassen werden. Enthält ein Programm jedoch verschachtelte FOR...NEXT-Schleifen, sollte auf NEXT der Variablenname folgen, um die Lesbarkeit des Programmlistings zu erhöhen.

BEISPIEL

```
10 FOR SP=10 TO 100 STEP
```

Ändert die Geschwindigkeit von 10 auf 100 in 10er-Intervallen, springt bei jedem Geschwindigkeitswechsel von P1 nach P5

```
20   SPEED SP
30     FOR I=1 TO 5
40       JUMP PI
50    NEXT I
60 NEXT SP
```

FORMAT

FUNKTION Formatiert den Dateienspeicher bzw. eine Diskette.

FORMAT

```
FORMAT {[Laufwerk]} {|/9|}
                    |/A|
```

BESCHREIBUNG Formatiert den Dateienspeicher bzw. eine Diskette im angegebenen Laufwerk. Wird die Laufwerksangabe ausgelassen, wird das aktuelle Laufwerk formatiert.

Wenn Sie alle FORMAT-Parameter angegeben haben, zeigt FORMAT eine ähnliche Meldung wie die folgenden an und wartet auf eine Eingabe:

Für Laufwerk A

```
RAM disk Format OK (Y/N)?
?_
```

Für Laufwerk B

```
Floppy disk Format OK (Y/N)?
?_
```

Zum Starten des Formatiervorgangs drücken Sie die Taste Y (=Yes), ansonsten die Taste N (=No).

Der Parameter /9 gibt an, daß die Diskette im 2DD-Format formatiert werden soll; wird dieser Parameter weggelassen, wird die Diskette im Format 2HD formatiert.

Wenn der Dateienspeicher mit Hilfe des optionalen 1-Mbyte-RAM-Moduls erweitert wurde, bewirkt der Parameter /A die Formatierung des Dateienspeichers ohne Löschen des aktuellen Inhalts. Bei Eingabe von /A erscheint die folgende Meldung:

```
RAM disk size adjust OK (Y/N) ?
?_
```

Wenn Sie hier Y eingeben, wird der Dateienspeicher formatiert und die Größe des Inhalts erhöht. Näheres hierzu finden Sie entweder im Benutzerhandbuch (User s Guide) oder in der Begleitdokumentation des RAM-Moduls.

Bei der Ausführung des Befehls FORMAT kann es vorkommen, daß die folgende Meldung angezeigt wird.

```
Memory error !! ddress $xxxxxxxx
```

Ist die hinter dem Zeichen $ angezeigte Adreßnummer gleich 1 oder kleiner, bedeutet dies, daß der DIP-Schalter SW1 auf der MPU-Platine auf einen zusätzlich installierten RAM eingestellt ist, obwohl de facto kein RAM-Modul installliert ist. Schalten Sie in diesem Fall das entsprechende Bit des Software-Schalters aus (OFF).

$00080003	Bit 1 von SD2
$00099001	Bit 1 und 2 von SD2

Wird eine andere Adreßnummer angezeigt, wenden Sie sich bitte an Ihren EPSON-Service.

Die Ausführung des FORMAT-Befehls bewirkt (außer bei Angabe des Parameters /A), daß alle Daten aus dem Dateienspeicher bzw. von der Diskette gelöscht werden. Daher sollten Sie die Daten, falls erforderlich, vor dem Formatieren sichern.

BEISPIEL

```
>FORMAT
RAM disk Format OK (Y/N)?
?Y
>_
```

FREE

FUNKTION Zeigt die verfügbare Speicherkapazität an.

FORMAT FREE

BESCHREIBUNG Zeigt die freie Speicherkapazität sowie alle nicht verwendeten Variablen an.

Im Online-Modus zeigt dieser Befehl den verfügbaren Hauptspeicher in Byte wie folgt an:

PRG = Verfügbarer Speicher für das Quellprogramm
VAR = Nicht benutzte Backup-Variablen, verfügbarer Speicher (Gesamtzahl der Variablen, Gesamtspeicher) ← über LIBSIZE definierte Werte
OBJ = Verfügbarer Speicher für das Objektprogramm

Im Offline-Modus zeigt der Befehl die verfügbare Speicherkapazität der Programmiereinheit in Byte wie folgt an:

PRG = Verfügbarer Speicher für das Quellprogramm
PNT = Verfügbarer Speicher für Positionsdaten

BEISPIEL

```
>FREE
PRG = 65536     Verfügbarer Speicher für Quellprogramm
VAR = 10,462    Nicht verwendete Backup-Variablen, verfügbarer Speicher
      (10,512)  (Gesamtzahl der Variablen, Gesamtspeicher) ← über
                                                  LIBSIZE definierte Werte
OBJ = 99660     Verfügbarer Speicher für Objektprogramm
>
```

F

FUNCTION...FEND

S

Function...Function End

FUNKTION Definiert eine Funktion.

FORMAT

```
FUNCTION [Funktionsname]
   .
   .
   .
FEND
```

BESCHREIBUNG Die erste Zeile muß den Befehl FUNCTION sowie den Funktionsnamen enthalten, die letzte Zeile den Befehl FEND. Das Programm zwischen FUNCTION und FEND wird Funktion (TASK) genannt. Zur Ausführung vieler Tasks oder bei Verwendung von CALL-Anweisungen können mehrere Funktionen der Reihe nach definiert werden (siehe Beispiel unten).

Ein Funktionsname wird benötigt, um jede der Funktionen aufzurufen. Für den Namen gelten dabei die folgenden Einschränkungen:

- es sind maximal 8 Zeichen erlaubt
- es sind nur alphanumerische Zeichen oder Unterstriche (_) erlaubt
- die Eingabe von Groß- und Kleinbuchstaben ist erlaubt; Kleinbuchstaben werden in Großbuchstaben umgewandelt
- das erste Zeichen muß ein Buchstabe, darf aber nicht das [P] sein
- Schlüsselwörter (wie z.B. GO oder FOR) sind nicht erlaubt. Folgt ein Unterstrich oder eine Zahl auf ein Schlüsselwort wird dies als Schlüsselwort erkannt.

Die erste Funktion, die in einem Programm definiert wird, ist die sog. "Hauptfunktion" (Main function). Die Hauptfunktion wird in Task 1 ausgeführt.

Der Eintrag XQT in der Hauptfunktion gibt die Tasknummern und die entsprechenden Funktionsnamen an und steuert die Ausführung aller nachfolgenden Funktionen.

VERWANDTE BEFEHLE XQT, RUN, QUIT, HALT, CALL

BEISPIEL

```
10 FUNCTION MAIN
20 '
30 XQT !2 TASK2          Führt Programm TASK2 in Task 2 aus
40 XQT !3 TASK3          Führt Programm TASK3 in Task 3 aus
 .
 .
 .
```

```
100 FEND
110 '
120 FUNCTION TASK 2       Programm TASK2
130 ST2:
140 WAIT SW($1)=1
 .
 .
 .
200 GOTO ST2
210 FEND
220 '
230 FUNCTION TASK3        Programm TASK3
240 ST3:
250 WAIT SW(1)=1
 .
 .
 .
300 GOTO ST3
310 FEND
```

GO

> S

FUNKTION

Führt eine gleichzeitige PTP-Bewegung aller vier Achsen aus. Verlangsamt die Geschwindigkeit und stoppt am angegebenen Punkt.

FORMAT

(1) `GO [ps] {TILL} {![Parallelanweisung]!}`

(2) `GO [ps] {TILL SW(Eingang])} {= |0|} {![Parallelanweisung]!}`
`|1|`

[ps] = Positionsspezifikation

BESCHREIBUNG

Führt eine gleichzeitige PTP-Bewegung aller vier Achsen aus. Verlangsamt die Geschwindigkeit und stoppt am angegebenen Punkt.

Der GO-Befehl verwendet den mit SPEED gesetzten Geschwindigkeitswert sowie die mit ACCEL festgelegten Beschleunigungs- bzw. Verzögerungswerte.

Die TILL-Bedingung wird verwendet, um den Manipulator an einer Zwischenposition abzubremsen und anzuhalten, wenn die aktuelle TILL-Bedingung erfüllt wird. Wird die TILL-Bedingung nicht erfüllt, verfährt der Manipulator bis zur Zielposition.

(1) GO mit TILL-Bedingung
Überprüft, ob die aktuelle TILL-Bedingung erfüllt wird. In diesem Fall wird der Befehl beendet, indem der Manipulator abgebremst und gestoppt wird.

(2) GO mit TILL-Bedingung, SW (Eingang) Bedingung und (0 oder 1) Eingangsbedingung:
Überprüft, ob dieselbe Zeileneingangsbedingung erfüllt wird. In diesem Fall wird der Befehl beendet, indem der Manipulator an einer Zwischenposition abgebremst und angehalten wird.

GO mit TILL-Bedingung, SW(Eingang) Bedingung, jedoch ohne Eingangsbedingung:
Die Eingangsbedingung erhält den Standardwert 1. Geht der angegebene Eingang auf ON, wird der Manipulator an einer Zwischenposition abgebremst und angehalten.

Der GO-Befehl unterscheidet sich folgendermaßen vom JUMP-Befehl:

Beim GO-Befehl bewegt sich die Z-Achse in einer geraden Linie von der Ausgangs- zur Zielposition, während der Manipulator beim JUMP-Befehl vor dem horizontalen Verfahrweg angehoben und anschließend wieder abgesenkt wird. Aus diesem Grund kann es beim GO-Befehl leichter vorkommen, daß die Hand gegen ein Werkstück schlägt. Daher ist es in den meisten Fällen sinnvoller, den JUMP- statt des GO-Befehls zu verwenden.

VERWANDTE BEFEHLE P=, ! ... !, SPEED, ACCEL, TILL, SW ( ), PASS

BEISPIEL

`100 TILL SW(1)=0 AND SW(2)=1`	Definiert die TILL-Bedingungen (Eingang 1 ist OFF, Eingang 2 ist ON)
`110 GO P1 TILL`	Stoppt, wenn die aktuelle TILL-Bedingung (Zeile 100) erfüllt wird
`120 GO P2 TILL SW(2)=1`	Stoppt, wenn Eingang 2 auf ON schaltet
`130 GO P3 TILL`	Stoppt, wenn die aktuelle TILL-Bedingung (Zeile 100) erfüllt ist.

GOSUB...RETURN

S

Goto Subroutine (Zu einem Unterprogramm verzweigen) ...
Return (Zurück zum übergeordneten Programm)

FUNKTION

Verzweigt in ein Unterprogramm, führt es aus und verläßt es danach wieder.

FORMAT

```
GOSUB |Zeilennummer|
      |    Label    |
  .
  .
  .
|Zeilennummer|
|      Label  |
  .
  .
  .
RETURN
```

Es sind maximal 10 Verschachtelungen möglich.

BESCHREIBUNG

Verzweigt zur angegebenen Zeilennummer bzw. zum angegebenen Label, führt das danach folgende Unterprogramm aus und kehrt anschließend zum Hauptprogramm zurück.
Achten Sie darauf, daß jedes Unterprogramm mit RETURN beendet wird. Dadurch kehrt die Programmausführung zu der Stelle im Programm zurück, die auf GOSUB folgt.
Verschachtelungen sind erlaubt; es können bis zu 9 GOSUB...RETURN-Schleifen in die erste GOSUB...RETURN-Schleife eingebettet werden. (Hinweise zu Verschachtelungen finden Sie in den Erläuterungen zum #include-Befehl.)

Insgesamt können die Befehle GOSUB und GOTO maximal 447mal miteinander kombiniert werden. Beim Versuch diese Anzahl zu überschreiten, erfolgt ein Fehler.

VERWANDTE BEFEHLE

GOTO

BEISPIELE

Beispiel 1

```
.
.
210 GOSUB 300

220 WAIT SW(1)=1
.
.
290 END
300 I=0
.
.
.
400 RETURN
.
.
.
```

Beispiel 2

```
.
.
210 GOSUB UP_1    Verzweigung zu
                  Unterprogramm UP_1
220 WAIT SW(1)=1
.
.
290 END
300 UP_1:
310 I=0
.
.
400 RETURN
.
.
.
```

GOTO

S

Go to (Gehe nach)

FUNKTION Verzweigt zur angegebenen Zeilennummer.

FORMAT

```
GOTO |Zeilennummer|
     |   Label     |
```

BESCHREIBUNG Verzweigt ohne Bedingung zur angegebenen Zeile bzw. zum angegebenen Label.

Existiert die angegebene Zeile bzw. das Label nicht, erfolgt Fehler 8.

Insgesamt können die Befehle GOSUB und GOTO maximal 447mal miteinander kombiniert werden. Beim Versuch diese Anzahl zu überschreiten, erfolgt eine Fehlermeldung.

BEISPIEL

```
10 SPEED 100
20 JUMP P50
30 WAIT 0.5
40 JUMP P80
50 WAIT 1.2
60 GOTO 20          Verzweigt zu Zeile 20, setzt die Ausführung fort
```

BEISPIEL 2

```
100 GOTO TIMER      Verzweigt zur mit TIMER gekennzeichneten Zeile,
                    setzt die Ausführung fort
  .
  .
  .
300 TIMER:
```

HALT

> S

(Unterbrechen)

FUNKTION Unterbricht kurzfristig die Ausführung einer laufenden Task.

FORMAT `HALT ![Tasknummer]`

Die Tasknummer muß eine ganze Zahl zwischen 1 und 16 sein.

BESCHREIBUNG Unterbricht eine Task d.h. hält vorläufig die Ausführung der laufenden Task an.
Zur Wiederaufnahme der unterbrochenen Task wird der Befehl RESUME, zur Beendigung der Befehl QUIT verwendet.

VERWANDTE BEFEHLE RUN, XQT, QUIT, RESUME

BEISPIEL

```
10 FUNCTION MAIN
20 XQT !2 BLINK        Führt Funktion BLINK als Task 2 aus
30 LOOP:
40 WAIT 3              Setzt Timer auf 3 Sekunden (Task 2 läuft)
50 HALT !2             Unterbricht Task 2 (nach Zeile 40, 3 Sekunden sind
                       vergangen)
60 WAIT 3              Setzt Timer auf 3 Sekunden (Task 2 ist unterbrochen)
70 RESUME !2           Nimmt Task 2 wieder auf (nach Zeile 60, 3 Sekunden
                       sind vergangen)
80 GOTO LOOP
90 FEND
100 '
110 FUNCTION BLINK     Schaltet einen Ausgang in 0,2-Sekunden-Interval-
                       len ein und aus
120 LOOP1:
130 ON 1
140 WAIT 0.2
150 OFF 1
160 WAIT 0.2
170 GOTO LOOP1
180 FEND
```

HOFS

> S

H Offset

FUNKTION

Definiert den Offset-Pulse für die Softwarekorrektur des Nullpunktes oder zeigt diesen an.

FORMAT

(1) `HOFS [Offset-Wert 1. Achse],[Offset-Wert 2. Achse]~`
`[Offset-Wert 3. Achse],[Offset-Wert 4. Achse]`

(2) HOFS

BESCHREIBUNG

(1) Definiert den Versatz (Offset) vom Encoder-Nullpunkt zum mechanischen Nullpunkt; dieser Offset wird für die softwareseitige Korrektur des Nullpunktes benötigt.
Obwohl die Steuerung der Manipulatorbewegungen auf dem Nullpunkt des Encoders basiert, der an jedem Achsenmotor befestigt ist, stimmen der Nullpunkt des Encoders und der mechanische Nullpunkt des Manipulators nicht unbedingt überein. Die Korrekturpulse für den HOFS-Offset-Pulse werden benötigt, um den mechanischen Nullpunkt anhand des Encoder-Nullpunktes per Software zu korrigieren.

Die HOFS-Werte bleiben über das Ausschalten hinaus gespeichert. Auch bei Ausführung des VERINIT-Befehls bleiben die Werte unverändert.

HOFS-Werte können direkt definiert werden.

Wenn Sie die HOFS-Werte automatisch berechnen lassen wollen, bewegen Sie den Manipulator an eine bestimmte Kalibrierposition und führen den Befehl CALIB aus. Die Steuerung berechnet die HOFS-Werte automatisch anhand der CALPLS-Pulsewerte und der Pulsewerte für die Kalibrierposition.

(2) Zeigt die aktuellen HOFS-Werte wie folgt an:

[Wert für die 1. Achse] [Wert für die 2. Achse]
[Wert für die 3. Achse] [Wert für die 4. Achse]

Vor Auslieferung des Roboters wird der HOFS-Wert korrekt definiert. Bei einer unnötigen Änderung des HOFS-Wertes besteht die Gefahr unerwünschter Manipulatorbewegungen bzw. von Fehlern bei der Positionierung. Wir empfehlen daher **dringend**, die HOFS-Werte nur dann zu ändern, wenn dies unbedingt notwendig ist.

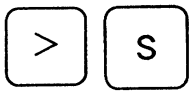
VERWANDTE BEFEHLE

CALIB, CALPLS

BEISPIEL

```
 >HOFS
     100       120
      50         0
>HOFS 100, 120, 50, 0
>HOFS
     100       120
      50         0
```

HOME



FUNKTION Führt eine Bewegung zur Home-Position (Standby-Position) aus.

FORMAT HOME

BESCHREIBUNG Führt eine langsame PTP-Bewegung zu der mit HOMESET definierten Home-Position (Standby-Position) in der Reihenfolge der HORDR-Werte aus.

Hat der Roboter die Home-Position erreicht, wird der HOME-Ausgang des REMOTE2-Anschlusses an der Steuereinheit aktiv.

VERWANDTE BEFEHLE HOMESET, HORDR

BEISPIEL

```
>HOME
>
```

HOMESET

> S

HOME SET (ing)

FUNKTION Definiert die Home-Position (Standby-Position) oder zeigt sie an.

FORMAT

(1) `HOMESET [Pulsewert 1. Achse],[Pulsewert 2. Achse],~`
`[Pulsewert 3. Achse],[Pulsewert 4. Achse]`

Der Pulsewert muß eine ganze Zahl sein.

(2) `HOMESET`

BESCHREIBUNG

(1) Definiert die Home-Position in Pulsewerten.

HORDR definiert die Reihenfolge der Achsenbewegung bei Ausführung des HOME-Befehls bzw. zeigt sie an.

(2) Zeigt die aktuellen HOMESET-Werte an.

☞ Bei Ausführung des VERINIT-Befehls werden die HOMESET-Werte gelöscht.

VERWANDTE BEFEHLE HOME, HORDR

BEISPIEL

```
>HOME
!!Error 143
```
Fehler beim Versuch HOME auszuführen, obwohl keine HOMESET-Werte angegeben sind

```
>HOMESET
!!Error 143
```
Fehler beim Versuch, die Pulsewerte der Home-Position anzeigen zu lassen, obwohl keine HOMESET-Werte definiert sind

```
>HOMESET 0,0,0,0
>HOMESET
     0         0
     0         0
>HOME
>_
```

```
>HOMESET PLS(1),PLS(2),PLS (3),PLS (4)
>
```
Durch Verwendung der PLS-Funktion wird die aktuelle Position als Home-Position definiert

HORDR

> S

Home Order (Home-Reihenfolge)

FUNKTION

Definiert die Reihenfolge der Achsenbewegung bei Ausführung des HOME-Befehls oder zeigt sie an.

FORMAT

(1) `HORDR [Wert 1],[Wert 2],[Wert 3],[Wert 4]`

(2) HORDR

BESCHREIBUNG

(1) Definiert die Reihenfolge der Achsenbewegung bei Ausführung des HOME-Befehls.

Die Achse bzw. Gruppe von Achsen, die mit Wert 1 definiert wurde, führt die Bewegung zuerst aus. Nach Beendigung der ersten Bewegung folgt die mit Wert 2 definierte(n) Achse(n) und so weiter, bis die Home-Position erreicht ist.

Jeder Achse wird ein Bit von 0 bis 3 wie folgt zugeordnet:

Achse	1. Achse	2. Achse	3. Achse	4. Achse
Bit-Nr.	Bit3	Bit2	Bit1	Bit0
Binärcode	&B1000	&B0100	&B0010	&B0001

Durch den VERINIT-Befehl werden die HORDR-Werte auf die Standardwerte zurückgesetzt. Nähere Informationen zu den jeweiligen Standardwerten des verwendeten Manipulatorarms erhalten Sie in der Dokumentation zum Manipulatorarm.

(2) Zeigt die aktuellen HORDR-Werte in hexadezimaler Notation wie folgt an:
[Wert 1] [Wert 2]
[Wert 3] [Wert 4]

VERWANDTE BEFEHLE

HOME, HOMESET

BEISPIEL

```
>HORDR &B0010,&B1000,&B0100,&B0001
```
Definiert die HORDR-Reihenfolge (3. Achse, 1. Achse, 2. Achse, 4. Achse)

```
>HORDR
     02    08
     04    01
```
Hexadezimal-Anzeige

H

HOUR

FUNKTION Zeigt die Betriebsstunden der Steuerung an.

FORMAT HOUR

BESCHREIBUNG Zeigt als ganze Zahl die Betriebszeit der Steuerung in Stunden an.

VERWANDTE BEFEHLE TIME ( )

BEISPIEL

```
>HOUR
  2550
>
```

HTASK

> S

Halt Task (Task unterbrechen)

FUNKTION Definiert die Tasks, die durch den PAUSE-Befehl oder die Eingabe PAUSE kurzfristig unterbrochen werden.

FORMAT

```
HTASK {[0,]}[Tasknummer]{,[Tasknummer]}n
```

Die Tasknummer muß eine ganze Zahl zwischen 1 und 16 sein, im obigen Format ist die Ziffer Null (0) gemeint.

BESCHREIBUNG Definiert die Tasks, die durch den PAUSE-Befehl oder die Eingabe von PAUSE kurzfristig unterbrochen werden können. Die Eingabe von PAUSE erfolgt über die folgenden 3 Ports
PAUSE-Schalter an der Bedieneinheit
Remote1-Anschluß Schutzabschrankung-Eingabe
Remote3-Anschluß PAUSE-Eingabeterminal

Tasks, die nicht über den Befehl HTASK definiert wurden, können weder durch den PAUSE-Befehl noch durch die Eingabe von PAUSE unterbrochen werden. Dies gilt jedoch nicht bei der Schutzabschrankung-Eingabe an Remote1-Anschluß; hierbei wird die Robotersteuerungstask auch dann unterbrochen, wenn die Task nicht durch HTASK definiert wurde.

Nach der Unterbrechung wird der Vorgang durch START fortgesetzt. Durch Eingabe von RESET werden alle unterbrochenen Tasks komplett beendet.

Der Parameter "0" bestimmt den Status der START LED (An/Aus), wenn die durch HTASK definierte Task unterborchen ist, während andere Tasks abgearbeitet werden. Der Status wird folgednermaßen festgelegt:

	START LED	PAUSE LED
ohne Parameter "0"	Aus	An
mit Parameter "0"	An	An

Nach dem Einschalten werden alle Tasks bei Ausführung/Eingabe des Befehls PAUSE durch den HTASK-Initialwert kurzfristig unterbrochen; der Parameter "0" ist nicht gesetzt. Das bedeutet, wenn nach dem Einschalten keine Werte über den Befehl HTASK definiert werden, werden alle Tasks durch PAUSE kurzfristig unterbrochen.

Die HTASK-Werte bleiben erhalten, bis sie entweder neu definiert werden oder die Stromzufuhr abgeschaltet wird. Andere Vorgänge, wie z.B. die Eingabe von RESET oder das Drücken des RESET-Schalters ändern die HTASK-Werte nicht.

Die Task, über die die Roboterbewegung gesteuert wird, muß durch HTASK definiert werden.
Verwenden Sie HTASK nur einmal bei Task 1; definieren Sie sie nicht merhmals in einem Programm.
Wurde ein HTASK-Wert einmal definiert, gilt er für alle auszuführenden Programme, und zwar solange bis die Stromzufuhr abgeschaltet wird. Bei Verwendung des Befehls HTASK sollten Sie die HTASK-Werte in allen Programmen, einschließlich der Dateien im Dateispeicher, definieren, um alle Tasks bei Eingabe von PAUSE zu unterbrechen.

BEISPIEL

```
20 HTASK 1,3,4
```

Bei Eingabe von PAUSE werden die Tasks 1, 3 und 4 kurzfristig unterbrochen

H

HTEST

>

FUNKTION Zeigt die HTEST-Werte an.

FORMAT HTEST

BESCHREIBUNG Zeigt die HTEST-Werte aller Achsen wie folgt an:

[HTEST-Wert der 1. Achse] [HTEST-Wert der 2. Achse]
[HTEST-Wert der 3. Achse] [HTEST-Wert der 4. Achse]

Als HTEST-Wert wird die Differenz zwischen dem tatsächlichen Pulse-Wert und dem logischen Pulse-Wert bei der Kalibrierung mit MCAL bezeichnet, bei der der Home-Sensor dort eingestellt wird, wo das Encoder-Z-Phasensignal entdeckt wird.

VERWANDTE BEFEHLE MCAL, HOFS

BEISPIEL

```
>HTEST
176  -75
218   59
>
```

IF...THEN...ELSE...ENDIF

S

FUNKTION Ausführung einer Anweisung mit Bedingung.

FORMAT

```
(1) IF [Bedingung] THEN
      .
      .
      .
    {ELSE}
      .
      .
      .
    ENDIF
```

Es sind maximal 20 Verschachtelungen erlaubt.

```
(2) IF [Bedingung] THEN [Anweisung]{;[Anweisung]}n~
                    {ELSE [Anweisung]{;[Anweisung]}n}
```

BESCHREIBUNG

(1) Definiert eine IF-Bedingung.
Ist die IF-Bedingung erfüllt, werden die Anweisungen zwischen IF und ELSE ausgeführt, bei Nichterfüllen der IF-Bedingung werden die Anweisungen zwischen ELSE und ENDIF ausgeführt.

Die Zeilen von ELSE bis ENDIF können weggelassen werden. In diesem Fall wird die Steuerung bei Nichterfüllen der IF-Bedingung an die Zeile übergeben, die unmittelbar auf die Zeile ENDIF folgt.

Sie können mehrere IF...ENDIF-Anweisungen verschachteln.

Die IF-Bedingung kann jeden durch SPEL III unterstützten Operator enthalten.

(2) Wird die IF-Bedingung der angegebenen Zeile erfüllt, werden die Anweisungen zwischen THEN und ELSE ausgeführt. Bei Nichterfüllen der Bedingung werden alle auf ELSE folgenden Anweisungen ausgeführt.

Die Eingaben ab ELSE sind optional und können weggelassen werden. In diesem Fall wird die Steuerung bei Nichterfüllen der IF-Bedingung an die nächste Zeile übergeben.

BEISPIEL

```
100 IF SW(0)=1 THEN PRINT "Ausgang0 EIN" ELSE PRINT
      "Ausgang0 AUS"
110 '
120 IF SW(1)=1 THEN
130   IF SW(2)=1 THEN
140     PRINT "Ausgang1 EIN Ausgang2 EIN"
150   ELSE
160     PRINT "Ausgang1 EIN Ausgang2 AUS"
170   ENDIF
180 ELSE
190   IF SW(2)=1 THEN
200     PRINT "Ausgang1 AUS Ausgang2 EIN"
210   ELSE
220     PRINT "Ausgang1 AUS Ausgang2 AUS"
230   ENDIF
240 ENDIF
```

I

IN( )

F

Input (Eingang)

FUNKTION Gibt den Eingangsstatus eines Eingangsports (8 Eingänge) aus.

FORMAT `IN([Portnummer])`

Die Portnummer muß eine ganze Zahl von 0 bis 15 sein.

BESCHREIBUNG Gibt den Eingangsstatus des mit der Portnummer angegebenen Eingangsports als Dezimalwert aus.

Jeder Port besteht aus 8 Eingängen, die Standard-16E/16A-Schnittstelle besteht aus 2 Ports. Portnummer, LSB/MSB und Eingänge sind wie folgt miteinander verbunden:

Port-Nummer	MSB 7	6	5	4	3	2	1	LSB 0
0	7	6	5	4	3	2	1	0
1	15	14	13	12	11	10	9	8

LSB: least significant bit (niedrigste Bitstelle)
MSB: most significant bit (höchste Bitstelle)

VERWANDTE BEFEHLE OUT, INBCD( ), OPBCD

BEISPIEL

```
A=IN(1)
```

Liest den Eingangsstatus von Eingangsport 1 (Eingänge 8-15) als Dezimalwert in Variable A.
Sind z.B. Eingang 9 und 10 EIN und die übrigen Eingänge AUS, wird der Wert 6 an die Variable A ausgegeben.

IN($)

F

Input $ (Eingang $)

FUNKTION Gibt den Eingangsstatus eines Merker-Ports (8 Merker) aus.

FORMAT `IN($[Portnummer])`

Die Portnummer muß eine ganze Zahl zwischen 0 und 63 sein.

BESCHREIBUNG Gibt den Eingangsstatus des mit der Portnummer angegebenen Merker-Ports als Dezimalwert aus.

Jeder Port besteht aus 8 Bit. Da es sich um insgesamt 512 Bit handelt, besteht ein Merker aus 64 Ports.

Portnummer, LSB/MSB und Merker sind folgendermaßen miteinander verbunden:

Port-Nummer	MSB 7	6	5	4	3	2	1	LSB 0
0	7	6	5	4	3	2	1	0
1	15	14	13	12	11	10	9	8
2	23	22	21	20	19	18	17	16
.								
62	503	502	501	500	499	498	497	496
63	511	510	509	508	507	506	505	504

LSB: least significant bit (niedrigste Bitstelle)
MSB: most significant bit (höchste Bitstelle)

VERWANDTE BEFEHLE OUT $

BEISPIEL

```
A=IN ($62)      'Liest den Eingangsstatus von Port 62 (Bit 496-503)
                als Dezimalwert in Variable A.
                Sind z.B. Bit 500-502 EIN und die übrigen Bits AUS,
                wird der Wert 112 an die Variable A ausgegeben.
```

I

INBCD( )

F

Input by Binary Coded Decimal (Eingang als Binär-Dezimal-Code)

FUNKTION

Gibt den Eingangsstatus eines Eingangsports (8-Bit-Eingang) als BCD-Code aus.

FORMAT

```
INBCD([Portnummer])
```

Die Portnummer muß eine ganze Zahl von 0 bis 15 sein.

BESCHREIBUNG

Gibt als ("oberer Rang, unterer Rang") Binär-Dezimal-Code (0-9) den Status des über die Portnummer festgelegten Eingangsports aus.

Jeder Port besteht aus 8 Eingängen, die Standard-16E/16A-Schnittstelle besteht aus 2 Ports. Portnummer, LSB/MSB und Eingänge sind wie folgt miteinander verbunden:

Port-Nummer	MSB 7	6	5	4	3	2	1	LSB 0
0	7	6	5	4	3	2	1	0
1	15	14	13	12	11	10	9	8

LSB: least significant bit (niedrigste Bitstelle)
MSB: most significant bit (höchste Bitstelle)

VERWANDTE BEFEHLE

OPBCD, IN( ), OUT

BEISPIEL

```
>B=INBCD(1)          Speichert in Variable B den Eingangsstatus von
                     Eingangsport 1, ausgegeben als BCD-Code
>PRINT INBCD(1)      'Zeigt den Status von Eingang 8-15 als BCD-Code an.
                     (In diesem Beispiel ist Port 1: 01010011)
53
>_
```

INPUT

> S

FUNKTION Aufforderung zur Dateneingabe über die Tastatur der Bedieneinheit.

FORMAT `INPUT [Variablenname 1]{,[Variablenname n]}n`

BESCHREIBUNG Erlaubt die Eingabe von numerischen Werten und Zeichenketten über die Bedieneinheit und speichert sie in Variablen.

Mit einem einzigen INPUT-Befehl können mehrere Variablennamen definiert werden. Die Variablennamen müssen durch Kommas (,) getrennt werden.

Sie können Namen sowohl für numerische Variablen als auch für Zeichenkettenvariablen angeben; die Art der Eingabedaten muß jedoch dem definierten Variablentyp entsprechen.

Bei Ausführung von INPUT erscheint die Eingabeaufforderung (?) an der Bedieneinheit. Drücken Sie nach Eingabe der Daten die Return-Taste.

Bei mehreren Variablen muß die Anzahl der Dateneinheiten der Anzahl der mit INPUT definierten Variablennamen entsprechen. Außerdem müssen die Daten durch Kommas (,), den sogenannten Begrenzungszeichen getrennt werden.

Wenn Sie numerische Werte eingeben, sollten Sie folgendes beachten:

Andere Zeichen als numerische werden ignoriert. Entspricht die Anzahl der Dateneinheiten nicht der Anzahl der über INPUT definierten Namen für die numerischen Variablen, erfolgt ein Fehler.

Werden numerische Zeichen und Alphazeichen gemischt, wird nur das erste bzw. die ersten von Alphazeichen eingeschlossenen numerischen Zeichen in der Variablen gespeichert.

Beispiel
`A123BC45` (123) wird gespeichert

Bei Eingabe von Zeichenketten können numerische und alphabetische Zeichen gemischt verwendet werden.

VERWANDTE BEFEHLE STRING $, PRINT

BEISPIEL

```
>INPUT A              Definiert die numerische Variable A
?52                   Speichert 52 in der numerischen Variablen A
>INPUT A$, B$         Definiert die Zeichenkettenvariablen A$ und B$
?SEIKO, EPSON         Sichert SEIKO in der Zeichenkettenvariablen A$,
                      EPSON in der Zeichenkettenvariablen B$
>
```

INPUT

> S

FUNKTION Dateneingabe von der Kommunikationsschnittstelle (RS-232C).

FORMAT `INPUT # [Portnummer],[Variablenname 1]{,[Variablenname n]}n`

Die Portnummermuß eine ganze Zahl zwischen 20 und 23 sein.

BESCHREIBUNG Erlaubt die Eingabe numerischer Werte und Zeichenketten von der über die Portnummer angegebenen Kommunikationsschnittstelle und speichert die Eingaben in Variablen.

Mit INPUT # können mehrere Variablennamen definiert werden. Die Variablennamen müssen durch Kommas (,) getrennt werden.

Bei mehreren Variablen muß die Anzahl der Dateneinheiten mit der Anzahl der über INPUT # definierten Variablennamen identisch sein.

Es sind sowohl numerische Variablennamen als auch Zeichenkettennamen erlaubt; die Art der Eingabedaten muß dem Variablentyp entsprechen.

Wenn Sie numerische Werte eingeben, sollten Sie folgendes beachten:

Andere Zeichen als numerische werden ignoriert. Entspricht die Anzahl der Dateneinheiten nicht der Anzahl der über INPUT # definierten Namen für die numerischen Variablen, erfolgt ein Fehler.

Werden numerische Zeichen und Alphazeichen gemischt, wird nur das erste bzw. die ersten von Alphazeichen eingeschlossenen numerischen Zeichen als Variable gespeichert.

Beispiel
`A123BC45` (123) wird gespeichert

Bei Eingabe von Zeichenketten können numerische und alphabetische Zeichen gemischt verwendet werden.

VERWANDTE BEFEHLE LOF( ), PRINT #

BEISPIEL

```
>INPUT #20,A      Erlaubt die Eingabe einer numerischen Dateneinheit
                  an Port 20, speichert sie in der numerischen Variablen A
>INPUT #21,B,C    Erlaubt die Eingabe von zwei numerischen Daten-
                  einheiten an Port 21, speichert eine in der numerischen
                  Variablen B, die andere in der numerischen Variablen C
>INPUT #20,A$     Erlaubt die Eingabe einer Zeichenkette an Port 20,
                  speichert sie in der Zeichenkettenvariablen A$
```

INT( )

F

Integer (ganze Zahl)

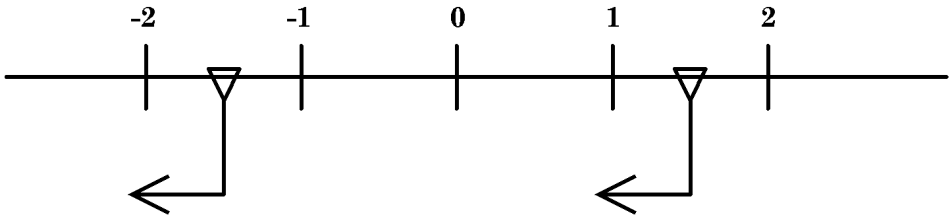
FUNKTION Gibt die größte ganze Zahl aus, die kleiner oder gleich dem angegebenen Wert ist.

FORMAT `INT([numerischer Wert])`

BESCHREIBUNG Gibt die größte ganze Zahl aus, die kleiner oder gleich dem angegebenen Wert ist.

Beachten Sie, daß bei Angabe von negativen Werten der absolute Wert der ausgegebenen Zahl größer ist als der des angegebenen Wertes. Dies erläutert das folgende Beispiel.

Beispiel
INT(1.55) → 1
INT(-1.55) → -2



VERWANDTE BEFEHLE ABS( ), SGN( ), SQR( )

BEISPIEL

```
>PRINT INT(111.55)
111
>PRINT SGN(-111.55)
-112
>
```

I

JRANGE

> S

Joint Range

FUNKTION Definiert den zulässigen Verfahrbereich für die angegebene Achse in Pulsen.

FORMAT

```
JRANGE [Achsennummer],[unterer Pulsewert],[oberer Pulsewert]
```

Die Achsennummer muß eine ganze Zahl zwischen 1 und 4 sein.

BESCHREIBUNG Definiert den zulässigen Verfahrbereich für die angegebene Achse anhand eines untere und oberen Grenzwertes in Pulsen. Beim RANGE-Befehl müssen alle Parameter für den unteren und oberen Grenzwert jeder Achse definiert werden. Obwohl beim Befehl JRANGE nur 3 Parameter angegeben werden müssen, ist es sinnvoll, den Verfahrweg für nur eine Achse zu definieren.

Der Wert für den unteren Bereich darf den des oberen Bereichs nicht übersteigen, da ansonsten ein Fehler erfolgt und eine Bewegung des Manipulators nicht ausgeführt werden kann.

Der definierte Verfahrbereich kann mit dem Befehl RANGE überprüft werden.

Die zulässigen Maximalwerte für den Verfahrbereich des Manipulators hängen vom jeweiligen Modell ab. Informationen dazu erhalten Sie in der Dokumentation zum Manipulator.

Die über den Befehl JRANGE definierten Werte bleiben auch über das Ausschalten des Roboters bzw. bei Ausführung des Befehls VERINIT erhalten.

VERWANDTE BEFEHLE RANGE, LIMPLS

BEISPIEL

```
>JRANGE 2, -60000, 70000
```

Definiert für die 2. Achse einen Verfahrbereich von -60000 bis 70000

JS (0)

F

Jump Sense

FUNKTION Gibt aus, ob eine SENSE-Bedingung nach Ausführung einer JUMP-Bewegung mit SENSE-Bedingung erfüllt ist oder nicht.

FORMAT `JS(0)`

BESCHREIBUNG Gibt aus, ob eine SENSE-Bedingung nach Ausführung einer JUMP-Bewegung mit SENSE-Bedingung erfüllt ist oder nicht.

1: SENSE-Bedingung ist erfüllt, Manipulator wurde über der Zielposition angehalten.

0: SENSE-Bedingung ist nicht erfüllt, JUMP-Bewegung wurde bis zur Zielposition ausgeführt.

VERWANDTE BEFEHLE SENSE, JUMP, STAT(1)

BEISPIEL

```
 .
 .
 .
100 SENSE SW(0)=1
 .
 .
 .
200 JUMP P1 C2 SENSE
210 IF JS(0)=1 THEN GOSUB ERRPRC;GOTO 200
220 ON 1
 .
 .
 .
1000 ERRPRC:
 .
 .
 .
```

In dem obigen Programm wird die Bewegung zu P1 gesteuert, ein Bauteil montiert und anschließend Ausgang 1 auf EIN gesetzt.

Wenn ein Peripheriegerät ein Signal "Fehler bei Materialversorgung" an Eingang 1 sendet, verzweigt das Programm in das Unterprogramm ERRPRC, um die Materialversorgung wiederherzustellen. Der Zustand des Eingangs wird überprüft, wenn die Abwärtsbewegung der dritten (Z-) Achse beginnt.

JUMP

> S

FUNKTION Steuert eine PTP-Bewegung kombiniert mit einer Gate-Bewegung (Bewegung ohne Verschleifen der Eckpunkte).

FORMAT

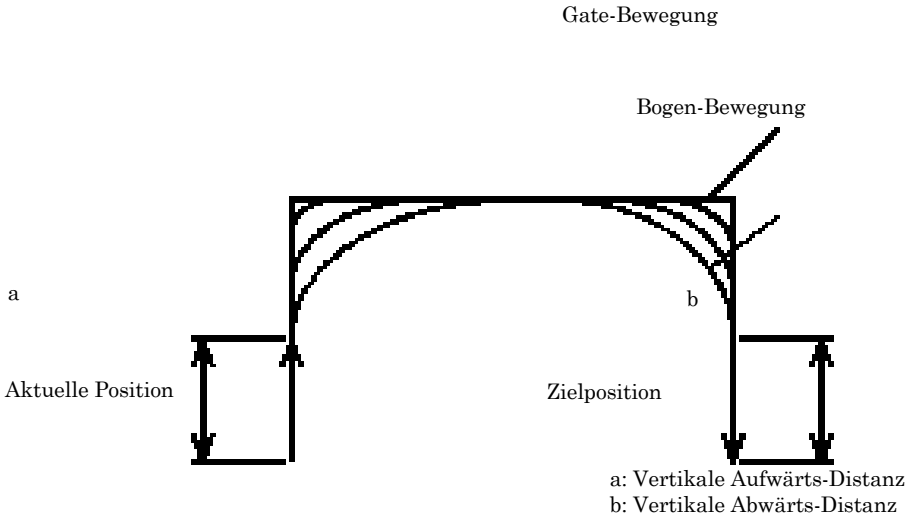
```
JUMP [ps] {C[Bogennummer]} {LIMZ[Wert der Z-Koordinate]}~
                                {|SENSE|} {![Parallelanweisgung]!}
                                 |TILL |
```

Die Bogennummer muß eine ganze Zahl zwischen 0 und 7 sein.
[PS] = Positionsspezifikation

BESCHREIBUNG Steuert eine Gate-Bewegung zum Zielpunkt.

Bei der Gate-Bewegung wird die dritte Achse aus der aktuellen Position bis zur höchsten Position der Z-Koordinate angehoben (z=0), verfährt horizontal bis zum Zielpunkt, wobei die 4. Achse bis zum definierten Winkel gedreht wird. Anschließend wird der Manipulator am Zielpunkt abgesenkt.

Der Parameter C[Bogennummer] bestimmt den Trajektoriebereich des Bogens während des Bewegungsvorgangs "Anheben aus der aktuellen Position - horizontaler Verfahrweg - Absenken an der Zielposition". Diese Bewegungsablauf nennt man Bogenbewegung.



Der Standardwert für jede Bogennummer wird in der folgenden Liste angegeben (mm-Angaben). Die Werte können über den Befehl ARCH geändert werden.

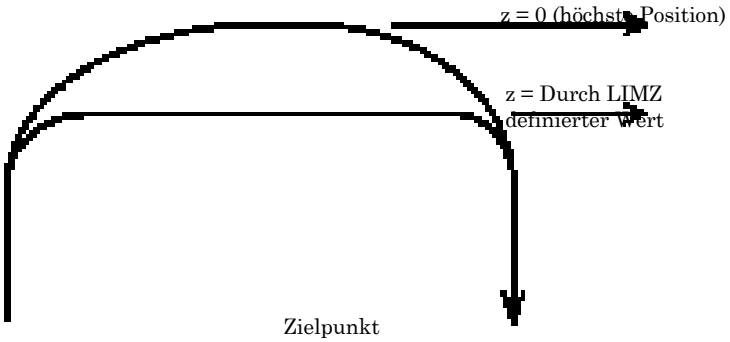
Bogennummer	0	1	2	3	4	5	6	7
a	30	40	50	60	70	80	90	Gate-Bewegung
b	30	40	50	60	70	80	90	

J

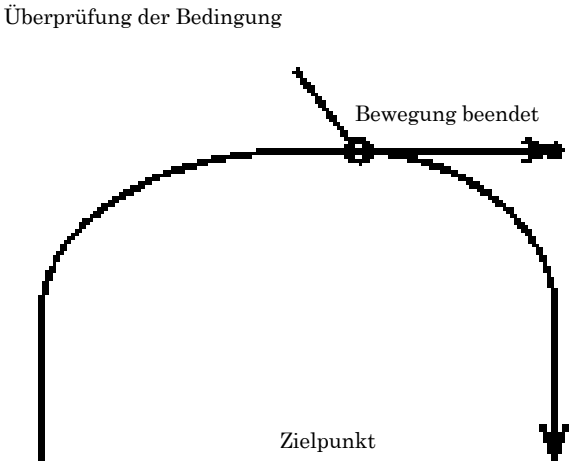
Bei Angabe des Parameters LIMZ [Wert der Z-Koordinate] wird die Z-Achse bis zur angegebenen maximalen Höhe angehoben und verfährt dann horizontal.

Wird die LIMZ-Bedingung weggelassen, wird der Manipulator bis zur Höhe der Z-Achse, (angegeben durch den aktuellen LIMZ-Wert) angehoben.

Die Angabe für die LIMZ-Z-Achsenhöhe bezieht sich auf den Z-Achsenwert des Roboter-Koordinatensystems und nicht auf den Z-Achsenwert für die ARM-, TOOL-, oder LOCAL0-Koordinaten. Beachten Sie dies, wenn Sie Werkzeuge oder Greifer mit unterschiedlicher Bedienhöhe verwenden.



JUMP-Befehl mit SENSE-Bedingung:
Das System überprüft, ob die SENSE-Bedingung vor Absenken der Z-Achse erfüllt ist. Ist sie erfüllt, wird der Befehl ausgeführt, indem der Manipulator über dem Zielpunkt stoppt. Mit Hilfe der Befehle JS(0) bzw. STAT(1) können Sie überprüfen, ob die SENSE-Bedingung erfüllt und der Befehl vollständig beendet ist oder ob die SENSE-Bedingung nicht erfüllt ist und der Manipulator am Zielpunkt gestoppt wurde.



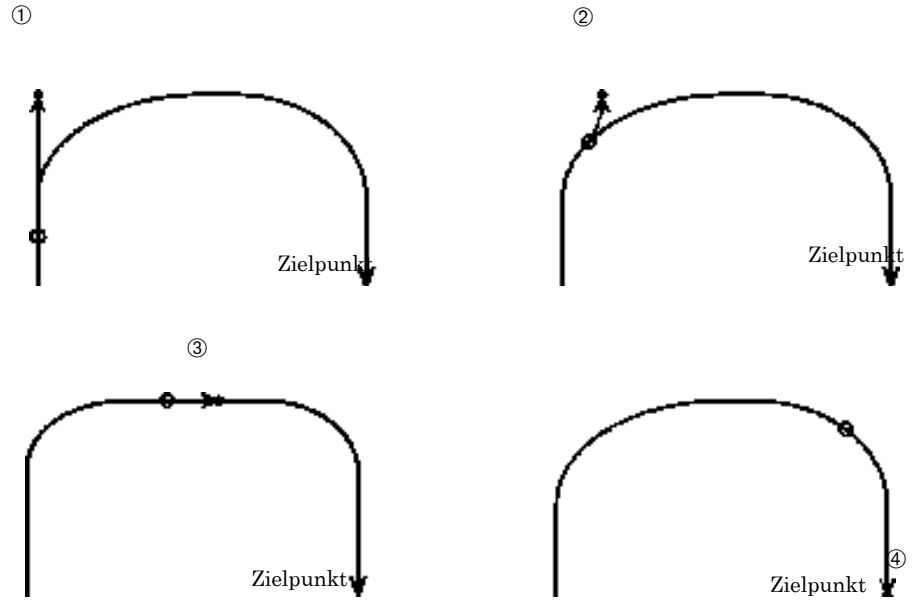
JUMP-Befehl mit TILL-Bedingung:
Das System überprüft, ob die aktuelle TILL-Bedingung während der Bewegung des Manipulators bis zum Absenken der Z-Achse erfüllt ist. Hat die Absenkbewegung der Z-Achse begonnen, wird die TILL-Bedingung nicht überprüft. Beachten Sie dies, wenn Sie eine Bogen-Bewegung verwenden wollen.

Ist die Bedingung erfüllt, wird der Befehl abgeschlossen, indem der Manipulator abgebremst und gestoppt wird. Der Manipulator stoppt in der über den Befehl LIMZ definierten Höhe. Wurde kein LIMZ-Wert definiert, bewegt sich der Manipulator bis zur Höhe z = 0. Mit Hilfe des Befehls STAT(1) können Sie überprüfen, ob die TILL-Bedingung erfüllt und der Befehl vollständig beendet ist, oder ob die TILL-Bedingung nicht erfüllt wurde und der Manipulator am Zielpunkt gestoppt wurde.

Die folgende Tabelle erläutert die Positionen, an denen die Bedingung erfüllt ist und die Art der Bewegungsbeendigung.

①	während der vertikalen Aufwärtsbewegung	wird angehoben und stoppt
②	während einer kombinierten Bewegung (horizontal und vertikal)	horizontale Bewegung - wird abgebremst und stoppt in der Bewegung der Z-Achse - wird angehoben und stoppt
③	während der horizontalen Bewegung	wird abgebremst und stoppt
④	nach Beginn der Abwärtsbewegung	stoppt am Zielpunkt

N = Position, an der die Bedingung erfüllt ist
• = Position, an der der Manipulator stoppt



Über eine Parallelanweisung kann die Ausführung bestimmter Befehle während des horizontalen Verfahrwegs gesteuert werden. Näheres dazu finden Sie im Kapitel über den Befehl !...!.

Zur Festlegung der Geschwindigkeit und der Beschleunigung verwenden Sie die Befehle SPEED und ACCEL. Es ist jedoch auch möglich, die Geschwindigkeits- und Beschleunigungswerte für die Aufwärtsbewegung der Z-Achse, den horizontalen Verfahrweg einschließlich der U-Achsendrehung und die Abwärtsbewegung der Z-Achse separat festzulegen.

VERWANDTE BEFEHLE P=, !...!, SPEED, ACCEL, LIMZ, SENSE, TILL, JS(0), STAT(1)

BEISPIEL

```
>JUMP P10+X50 C0 LIMZ-20 SENSE !D50;ON 0;D80;ON 1!
>
```

Die Anwendung des JUMP-PASS-Befehls

Der JUMP-PASS-Befehl bietet eine Kombination aus dem JUMP- und dem PASS-Befehl. Auf diese Weise können Hindernisse ohne Zwischenstops umfahren werden.

Die obere Verfahrhöhe ist Z=0 und kann durch LIMZ begrenzt werden. Die Anwendung von LIMZ ist identisch zu JUMP. Die Verfahrhöhe ist nicht von den PASS-Punkten abhängig. Die dort gespeicherten Z-Achsenhöhen werden ignoriert.

Parallelbearbeitung mit Hilfe des Befehls !...! ist möglich. Sie können die Befehle WAIT, ON, OFF; SW( ); D verwenden.

Der Parameter D wird wie folgt verwendet:

D0	bis	D 99	gilt zwischen Startpunkt und 1. Pass-Punkt
D100	bis	D199	gilt zwischen 1. und 2. Pass-Punkt
D200	bis	D299	gilt zwischen 2. und 3. Pass-Punkt
D300	bis	D399	gilt zwischen 3. und 4. Pass-Punkt
D400	bis	D499	gilt zwischen 4. und 5. Pass-Punkt

BEISPIEL

```
JUMP,P1,P2,P3,P4,P5 !D50; ON 1; D150 OFF1!
```

Bei der Anwendung von SENSE liegt der Haltepunkt über dem letzten Punkt im JUMP-PASS-Befehl.

Die maximale Anzahl von Punkten innerhalb einer JUMP-PASS-Bewegung ist auf 50 begrenzt.

Anwendungsbeispiele:

```
JUMP p1,p2,p31,p15,p8
JUMP p1-p5
JUMP p1,p3,p6,p4 limz-50
JUMP p1-p2 !D50;on1!
JUMP p5,p4,p6 sense
JUMP p12,p13,p31 c0 limz-35 !D1; on1;D170; off 3!
```

VERWANDTE BEFEHLE ARCH, TILL

KILL

FUNKTION Löscht eine oder mehrere Dateien.

FORMAT `KILL "{[Laufwerk]}{[Pfad]}[Dateiname]"`

BESCHREIBUNG Löscht die angegebene(n) Datei(en).

Die Dateinamenerweiterung muß nicht unbedingt angegeben werden. Wird die Erweiterung angegeben, wird nur die entsprechende Datei gelöscht.

Wird nur der Dateiname ohne die Erweiterung angegeben, werden die folgenden Dateien in dieser Reihenfolge gelöscht:

Dateiname.PRG (Quellprogrammdatei)
Dateiname.PNT (Positionsdatendatei)
Dateiname.OBJ (Objektdatei)
Dateiname.SYM (Symboltabellendatei)

Falls in diesem Fall entweder die Datei "Dateiname.PRG" oder die Datei "Dateiname PNT" nicht existiert, erfolgt ein Fehler. Existiert zwar die PRG-Datei jedoch nicht die PNT-Datei, erfolgt ein Fehler, nachdem die PRG-Datei gelöscht wurde.

Bei Auslassung der Laufwerksangabe löscht KILL die Dateien auf dem aktuellen Laufwerk.
Bei Auslassung der Pfadangabe löscht KILL die Dateien aus dem aktuellen Verzeichnis.

Dateiname und Erweiterung können Platzhalter (*,?) enthalten.

Im Offline-Modus von SPEL-80M funktioniert der Befehl DKILL in der gleichen Weise wie KILL.

VERWANDTE BEFEHLE DEL, FILES, DIR, DSAVE

BEISPIEL

```
>FILES
ABCD  PRG  2
ABCD  PNT  1
space     59
>KILL"ABCD.PRG"
>FILES
ABCD  PNT  1
Space     61
```

K

LEFT$( )

F

FUNKTION Gibt die äußerst links gelegenen Zeichen einer angegebenen Zeichenkette aus.

FORMAT
```
LEFT$(|[Variablenname d. Zeichenkette]|,[Anzahl Zeichen])
      |        "[Zeichenkette]"       |
```

BESCHREIBUNG Gibt die angegebene Anzahl der äußerst links gelegenen Zeichen einer angegebenen Zeichenkette aus. Die Zeichenkette kann durch eine Zeichenkettenvariablen definiert werden.

VERWANDTE BEFEHLE RIGHT$( ), MID$( )

BEISPIEL
```
10   FUNCTION MAIN
20   STRING LETTER$
30   LETTER$="ABCDE"
40   PRINT LEFT$ (LETTER$,2)
50   FEND
>RUN
COMPILE END
AB
>
```

L

LEN( )

F

Length (Länge)

FUNKTION Gibt die Anzahl von Zeichen in einer angegebenen Zeichenkette aus.

FORMAT

```
LEN(|[Variablenname d. Zeichenkette]|)
    |      "[Zeichenkette]"          |
```

BESCHREIBUNG Gibt die Anzahl der Zeichen in einer angegebenen Zeichenkette aus. Die Zeichenkette kann durch eine Zeichenkettenvariablen definiert werden.

BEISPIEL

```
>PRINT LEN("ABCDE")
5
>
```

LIBRARY

FUNKTION Zeigt die Namen der Backup-Variablen im Speicher an.

FORMAT LIBRARY

BESCHREIBUNG Zeigt die Namen der Backup-Variablen im Speicher sowie den Variablentyp an.

VERWANDTE BEFEHLE SYS, VARIABLE

BEISPIEL

```
>LIST
10  FUNCTION BACKUPV
20  SYS BYTE ER_RB(10),ER_RA(5)
30  SYS REAL V(20)
40  FEND
>RUN
COMPILE END
>LIBRARY
 BYTE  ER_RB (10)
 BYTE  ER_RA (5)
 REAL V(20)
>
```

LIBSIZE, SIZE

Library Size (Bibliotheksgröße)

FUNKTION Definiert die Anzahl verwendbarer Backup-Variablen und die verfügbare Speicherkapazität oder zeigt die entsprechenden Daten an.

FORMAT (1) `LIBSIZE [Anzahl Backup-Variablen],~ [verfügbarer Speicher]`

Die Anzahl der Backup-Variablen muß eine ganze Zahl von 0 bis 794 sein. Der verfügbare Speicher beträgt zwischen 14 und 32768 Bytes. Standardwerte sind 10 bzw. 512.

(2) `LIBZISE`

BESCHREIBUNG (1) Definiert die Anzahl verwendbarer Backup-Variablen und die verfügbare Speicherkapazität .
Matrixvariablen werden unabhängig von der Größe der Matrix als eine Variable gezählt.

Nach Eingabe des Befehls erscheint die folgende Meldung:

```
>Backup Variable, Object all clear OK?
```

Zum Ändern des LIBSIZE-Wertes drücken Sie **Y** oder **y**.
Soll der LIBSIZE-Wert nicht geändert werden, drücken Sie **N** oder **n**.
Durch die Änderung des aktuellen LIBSIZE-Wertes wird im Hauptspeicher sowohl der gesamte Bereich für Backup-Variablen als auch der Objektbereich gelöscht. Dies gilt jedoch nicht für den Quellprogramm- bzw. Positionsdatenbereich. Daher müssen, falls verwendet, die Backup-Variablen nach Ausführung des LIBSIZE-Befehls erneut gesichert werden.

Da der Hauptspeicher in der Robotersteuerung eine feste Größe hat, verringert sich bei Definition einer größeren Anzahl von Backup-Variablen oder des verfügbaren Speichers in entsprechender Weise der Objektbereich. Daher sollten Sie mit der Erhöhung der Anzahl von Backup-Variablen bzw. des verfügbaren Speichers vorsichtig sein.

Die Speicherkapazität, die eine Variable beansprucht, hängt vom Variablentyp ab. Zum Beispiel benötigt eine 2 Byte große ganze Zahl einschließlich ihrer Registriertabelle (Systemarbeitsbereich) ca. 10 Byte. Der benötigte Speicher für die Registriertabelle ist bei allen Variablen gleich, unabhängig vom Variablentyp oder ob es sich um eine Matrixvariable handelt oder nicht. Daher sollten Sie so weit wie möglich Matrixvariablen verwenden, um den Speicher effizient zu nutzen.

Die folgenden Variablentypen stehen zur Verfügung:

BYTE	1 Byte (-128 bis +127)		ganze Zahlen
INTEGER	2 Byte (-32768 bis +32767		
LONG	4 Byte (-2147483648 bis +2147483647)		
REAL	4 Byte	7 Ziffern	reele Zahlen
DOUBLE	8 Byte	14 Ziffern	

(2) Zeigt die derzeit verwendbaren Backup-Variablen sowie den verfügbaren Speicher an.

VERWANDTE BEFEHLE PRGSIZE, PNTSIZE, FREE, SYS

BEISPIEL

```
>LIBSIZE 500,10000
>LIBSIZE
 500,10000
>
```

LIMZ

> S

Limit Z Axis (Grenzbereich für Z-Achse)

FUNKTION

Definiert die Z-Achsenhöhe für einen JUMP-Befehl bzw. zeigt den aktuellen Wert an.

FORMAT

(1) `LIMZ [Definitionswert für Z-Achse]`

```
      .
      .
      .
   JUMP [Positionsdefinition]
```

Der Standardwert für LIMZ ist 0.

(2) `JUMP [Positionsdefinition] LIMZ [Wert für Z-Achse]`

(3) `LIMZ`

BESCHREIBUNG

LIMZ definiert die Höhe der Z-Achse für eine JUMP-Bewegung.
Durch den Befehl MOTOR ON, einen Software-Reset oder die Befehle SLOCK, SFREE und VERINIT wird der LIMZ-Wert standardmäßig auf 0 gesetzt.

(1) LIMZ-Befehl
Definiert die Höhe der Z-Achse.
Die Eingabe mehrerer LIMZ-Befehle ist erlaubt; der zuletzt eingegebene LIMZ-Wert wird der jeweils aktuelle bis zur Eingabe eines neuen Wertes.

JUMP-Befehl
Führt eine JUMP-Bewegung in der durch LIMZ festgelegten Höhe der Z-Achse durch.

(2) JUMP mit LIMZ-Bedingung:
Definiert die Höhe der Z-Achse nur für den gleichen JUMP-Befehl.

(3) Zeigt den aktuellen LIMZ-Wert an.

Die Definition des LIMZ-Wertes für die maximale Z-Achsenhöhe bezieht sich auf den Z-Achsenwert im Roboterkoordinatensystem und nicht auf den Z-Achsenwert für die ARM-, TOOL oder LOCAL 0-Koordinaten. Beachten Sie dies, wenn Sie Werkzeuge oder Greifer mit unterschiedlichen Bedienhöhen verwenden wollen. Wird die LIMZ-Bedingung weggelassen, wird der Manipulator bis zu der durch den aktuellen LIMZ-Wert definierten Z-Achsenhöhe angehoben.

VERWANDTE BEFEHLE

JUMP

BEISPIEL

```
10 LIMZ -10       Definiert den LIMZ-Wert (Z=-10)
20 JUMP P1        Führt eine JUMP-Bewegung mit dem aktuellen
                  LIMZ-Wert (Z=-10) aus
30 JUMP P2 LIMZ -30          Führt eine JUMP-Bewegung mit dem Wert
                  Z=-30 aus
40 JUMP P3        Führt eine JUMP-Bewegung mit dem aktuellen
                  LIMZ-Wert (Z=-10) aus
```

LINEHIST

>

Line History (Zeilenprotokoll)

FUNKTION Zeigt ein Protokoll der Zeilennummern an.

FORMAT `LINEHIST [Tasknummer]{,Tasknummer}7`

BESCHREIBUNG Zeigt der Reihenfolge nach die 10 zuletzt ausgeführten Zeilennummern an.

Sie können Zeilenprotokolle für maximal 8 Tasks definieren. Mehrfach-Tasks werden in der Reihenfolge ihrer Ausführung angezeigt, wobei für jede Task die 10 zuletzt ausgeführten Zeilennummern angezeigt werden.

Die Kapazität für die Zeilenprotokolle kann auf zwanzig Protokolle erhöht werden, indem Sie in der CNFG.SYS-Datei den Eintrag LINEBUF=20 einfügen.

Bei Ausschalten des Gerätes werden die Zeilenprotokolle gelöscht.

BEISPIEL

```
>LINEHIST 1,2
Task 1         Task 2       Time
  40                        13:26:37
  30                        13:26:37
  40                        13:26:37
  30                        13:26:37
                 140        13:26:37
                 110        13:26:37
  40                        13:26:38
  30                        13:26:38
  40                        13:26:38
  30                        13:26:38
  40                        13:26:38
  30                        13:26:38
                 120        13:26:38
                 130        13:26:38
                 140        13:26:38
                 110        13:26:38
                 120        13:26:38
                 130        13:26:38
                 140        13:26:38
                 110        13:26:38
```

Programm

```
10  FUNCTION MAIN
20  XQT !2,SUB
30  WAIT 0.1
40  GOTO 30
50  FEND
100  FUNCTION SUB
110  WAIT 0.2
120  '
130  '
140  GOTO 110
150  FEND
```

L

LINE INPUT

> S

FUNKTION Speichert eine an der Bedieneinheit eingegebene Datenzeile in eine Zeichenkettenvariable.

FORMAT `LINE INPUT [Name der Zeichenkettenvariablen]`

BESCHREIBUNG Speichert eine an der Bedieneinheit eingegebene Zeile mit Zeichenkettendaten in eine Zeichenkettenvariable.

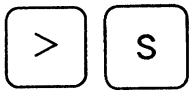
Bei Ausführung des Befehls LINE INPUT erscheint die Eingabeaufforderung "?" an der Bedieneinheit. Drücken Sie nach Eingabe der Datenzeile die Return-Taste.

VERWANDTE BEFEHLE INPUT

BEISPIEL

```
100 FUNCTION MAIN
110 STRING A$
120 LINE INPUT A$          Liest eine Datenzeile und speichert sie in
                           der Zeichenkettenvariablen A$
130 PRINT A$
140 FEND
>RUN
COMPILE END
?A,B,C
A,B,C                      Zeigt die gelesene Zeichenkette an
>
```

LINE INPUT



FUNKTION Speichert eine vom Kommunikationsport (RS-232C) gesandte Datenzeile in einer Zeichenkettenvariablen.

FORMAT `LINE INPUT #[Portnummer],[Name der Zeichenkettenvariablen]`

Die Portnummer muß eine ganze Zahl zwischen 20 und 23 sein.

BESCHREIBUNG Speichert eine vom Kommunikationsport gesandte Zeile mit Zeichenkettendaten in eine Zeichenkettenvariable.

BEISPIEL

```
LINE INPUT #20,A$
```

Empfängt eine Zeichenkette von Port 20 und speichert die Zeile in der Zeichenkettenvariablen A$

LINK

FUNKTION Verbindet zwei oder mehr Objektdateien.

FORMAT

```
LINK[Datei 1]+[Datei 2]{+[Datei n]}n{,[LINK-Datei]}
```

Die Angabe einer Dateinamenerweiterung ist nicht erlaubt.

BESCHREIBUNG Verbindet zwei oder mehr Objektdateien, die sich im Dateispeicher oder auf einer Diskette befinden, zu einer einzigen Objektdatei und sichert diese neue Datei unter dem angegebenen LINK-Dateinamen (mit der Erweiterung .OBJ). Gleichzeitig werden die Symboldateien zu einer einzigen Symboldatei verbunden und die neue Datei mit dem LINK-Dateinamen (Erweiterung .SYM) gesichert.

Auf diese Weise kann durch modulare Kompilierung aus verschiedenen Objekt- und Symboldateien eine einzige ausführbare Objekt- bzw. Symboldatei erstellt werden.

Für jede Datei kann eine Laufwerkbezeichnung angegeben werden. Bei Auslassung wird das aktuelle Laufwerk angenommen.

Der LINK-Dateiname kann weggelassen werden. In diesem Fall wird die Objektdatei als SYSTMP.OBJ und die Symboldatei als SYSTMP.SYM gesichert.

Bei einer verbundenen Objektdatei (d.h. eine Datei, die aus mehreren verbundenen Objektdateien besteht) wird der erste Funktionsablauf (FUNCTION...END) aus Datei 1 die Hauptfunktion und wird als Task 1 ausgeführt.

Um eine verbundene Objektdatei ausführen zu können, ist ein Positionsdatendatei mit demselben Dateinamen und der Dateinamenerweiterung .PNT erforderlich. So muß z.B. für eine verbundene Objektdatei mit dem Namen SYSTMP.OBJ eine Positionsdatendatei mit dem Namen SYSTMP.PNT angelegt werden.

VERWANDTE BEFEHLE COMPILE, FILES, DIR

BEISPIEL

```
>COM"MAIN"
COMPILE END
>COM"SUB"
COMPILE END
>LINK MAIN+SUB,B:TEST        Verbindet MAIN.OBJ und SUB.OBJ
                             zu TEST.OBJ
>
```

LIST

FUNKTION Zeigt das Quellprogramm an.

FORMAT

```
(1) LIST {|[Zeilennummer]                                |}
         |[erste Zeilennummer] -                         |
         |[erste Zeilennummer]-[letzte Zeilennummer]      |
         |                    -[letzte Zeilennummer]      |
         |*                                               |
         |*-                                              |

(2) LIST
```

BESCHREIBUNG

(1) Zeigt die durch ihre Zeilennummern definierten Programmzeilen an. Die Zeilennummern werden wie folgt definiert:

`[Zeilennummer]`
Zeigt die angegebene Zeile an.
`[erste Zeilennummer]-`
Zeigt alle Zeilen ab "erste Zeilennummer" bis zum Ende des Programms an.
`[erste Zeilennummer] - [letzte Zeilennummer]`
Zeigt alle Zeilen ab "erste Zeilennummer" bis einschließlich "letzte Zeilennummer" an. Die erste Zeilennummer muß kleiner sein als die letzte Zeilennummer, da ansonsten Fehler 2 erfolgt.
`- [letzte Zeilennummer]`
Zeigt alle Zeilen bis einschließlich "letzte Zeilennummer" an.
*
Zeigt die Zeile an, bei der die Anzeige durch Drücken der STOP- oder BREAK-Taste unterbrochen wurde.
*-
Zeigt bis zum Ende des Programms alle Zeilen ab der Zeile an, bei der die Anzeige durch Drücken der STOP- oder BREAK-Taste unterbrochen wurde.

(2) Zeigt alle Zeilen an.

BEISPIEL

`>LIST 50-`	Zeigt alle Zeilen ab Zeile 50 bis zum Programmende an
`>LIST -100`	Zeigt alle Zeilen bis einschließlich Zeile 100 an
`>LIST 20`	Zeigt Zeile 20 an
`>LIST 100-200`	Zeigt alle Zeilen ab Zeile 100 bis 200 einschließlich an
`>LIST`	Zeigt alle Zeilen an
`>LIST *`	Zeigt die Zeile an, bei der die Anzeige unterbrochen wurde
`>LIST *-`	Zeigt ab der Zeile, bei der die Anzeige unterbrochen wurde, bis zum Programmende alle Zeilen an.

LOCAL (RLOCAL, LLOCAL, SLOCAL)

> S

FUNKTION Definiert die lokalen Koordinatensysteme bzw. zeigt sie an.

FORMAT

```
(1) LOCAL[Nr des lokalen Koordinatensystems]~
       (P[Punkt-Nr. 1]:P[Punkt Nr. 2]),(P[Punkt Nr.3]:~
        P[Punkt Nr. 4])
```

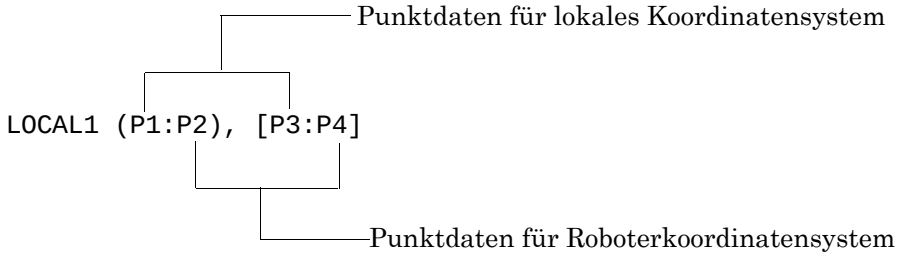
Die Nummer für das lokale Koordinatensystem muß ganze Zahl zwischen 1 und 15 sein.

```
(2) LOCAL
```

BESCHREIBUNG

(1) Definiert ein lokales Koordinatensystem durch Angabe zweier Punkte darin, und zwar P[Punkt Nr. 1] und P[Punkt Nr. 3]), die mit zwei Punkten im Roboterkoordinatensystem übereinstimmen, und zwar P[Punkt Nr. 2] und P[Punkt Nr. 4].

<Beispiel>



Eine übersichtliche Schreibweise ist: LOCAL1 (P1:P11), [P2:P12] (sprich Peins:Peins-eins bzw. Pzwei:Pzwei-zwei).

Stimmt die Distanz zwischen den zwei im lokalen Koordinatensystem definierten Punkten nicht mit der der zwei im Roboterkoordinatensystem definierten Punkte überein, wird das lokale Koordinatensystem mittig zwischen die Punkte des Roboterkoordinatensystems plaziert.

In ähnlicher Weise wird die Z-Achse des lokalen Koordinatensystems durch die Position bestimmt, an der die beiden Mittelpunkte übereinstimmen.

Die U-Achse des lokalen Koordinatensystems wird automatisch entsprechend der X- und Y-Koordinaten der definierten Punkte korrigiert. Das heißt, die zwei Punkte im Roboterkoordinatensystem können anfangs jeden beliebigen Wert für die U-Koordinaten haben.

Möglicherweise möchten Sie die U-Achse des lokalen Koordinatensystems lieber anhand der U-Koordinaten der zwei Punkte im Roboterkoordinatensystem korrigieren, anstatt eine automatische Korrektur vornehmen zu lassen (z.B. Korrektur der Rotationsachse durch manuelle Positionierung - Teaching).

Dazu müssen Sie Port 0/Bit 1 über den Befehl SWITCH (Softwareschalter) einschalten und dann den Befehl LOCAL verwenden. Informationen über Software-Schalter erhalten Sie in der Dokumentation zu Robotersteuerung bzw. im Handbuch zum SPEL Editor.

(2) Zeigt alle definierten lokalen Koordinatensysteme, einschließlich der über BASE bestimmten an.

<Beispiel>

Über LOCAL definierte lokale Koordinatensysteme:

`local 0 (p5 :p55 ), (p6 :p56 )`	Local0-Koordinatensystem
`local 1 (p1 :p11 ), (p2 :p12 )`	Über LOCAL definiertes lokales Koordinatensystem
`llocal 5 (p50 :p60 ), (p51 :p61 )`	Über LLOCAL definiertes lokales Koordinatensystem
`rlocal 10(p70 :p80 ), (p7 :p81 )`	Über RLOCAL definiertes lokales Koordinatensystem
`slocal 12(p90 :p100), (p91 :p101)`	Über SLOCAL definiertes lokales Koordinatensystem

Über BASE definierte lokale Koordinatensysteme

`local 0 (p***:p***), (p***:p***)`	Lokales Koordinatensystem 0
`local 2 (p***:p***), (p***:p***)`	

Zur Bewegung des Manipulators zum Punkt Z=0 im Roboterkoordinatensystem (dem höchsten Punkt der dritten Achse bei einem normalen Roboter) mit Hilfe der Punktdaten eines lokalen Koordinatensystems, dessen Z-Achse versetzt ist (offset), wird das Symbol @ verwendet:

```
JUMP P1:Z@
          └──── Punktdaten des lokalen Koordinatensystems
```

Die obigen Erläuterungen gelten auch für den Fall, daß das Local0-Koordinatensystem über LOCAL0 oder BASE 0 definiert wurde. Hier gelten die Aussagen zum Roboterkoordinatensystem entsprechend für das Local0 lokale Koordinatensystem .

Obwohl LOCAL wie zuvor beschrieben, prinzipiell mit Mittelpunkten zur Positionierung der Achsen eines lokalen Koordinatensystems arbeitet, bietet SPEL III drei verschiedene Varianten des LOCAL-Befehls, die jeweils andere Positionierungsmethoden verwenden und u.U. je nach Anwendungsfall besser geeignet sind. Zu diesen Varianten gehören die Befehle LLOCAL, RLOCAL und SLOCAL. (Dennoch korrigieren alle drei die U-Achse mit der gleichen Methode wie LOCAL).

LLOCAL (Left Local)

Der Befehl LLOCAL definiert ein lokales Koordinatensystem durch Festlegung von [Punkt Nr. 1] entsprechend [Punkt Nr. 2] im Roboterkoordinatensystem (einschließlich der Z-Achsenrichtung).
RLOCAL (Right Local)

Der Befehl RLOCAL definiert ein lokales Koordinatensystem durch Festlegung von [Punkt Nr. 3] entsprechend [Punkt Nr. 4] im Roboterkoordinatensystem (einschließlich der Z-Achsenrichtung).
SLOCAL (Scale Local)

Bei Definition der XY-Ebene im lokalen Koordinatensystem bestimmt SLOCAL [Punkt Nr. 1] und [Punkt Nr. 3] des lokalen Koordinatensystems entsprechend [Punkt Nr. 2] und [Punkt Nr. 4] im Roboterkoordinatensystem. Dies erlaubt die automatische Berechnung des Koordinaten-Skalierfaktors; der geometrische Ort kann maßstabsgetreu vergrößert oder verkleinert werden. Für die Z-Achse definiert SLOCAL [Punkt Nr. 1] im lokalen Koordinatensystem entsprechend [Punkt Nr. 2] im Roboterkoordinatensystem.

Für die Befehle BASE und LOCAL gelten dieselben Nummern der lokalen Koordinatensysteme.

Die Definition lokaler Koordinatensysteme von #1 bis #15 wird bei Ausschalten bzw. Ausführung einer der Befehle LOCAL0, BASE 0 oder VERINIT gelöscht.

Im Punktelisting wird das lokale Koordinatensystem "n" durch den Zusatz /n gekennzeichnet.

Durch Anfügen des Parameters &n kann eine Umkehrkonvertierung des lokalen Koordinatensystems durchgeführt werden. Beachten Sie, daß im TEACH-Modus festgelegte Punkte als Punktdaten in einem lokalen Koordinatensystem definiert sein können.

`P1=P*&2` Definiert den aktuellen Punkt als Koordinate im lokalen Koordinatensystem 2

VERWANDTE BEFEHLE BASE, LOCAL 0, BASE 0

BEISPIEL

```
>PLIST
P1=0,0,0,0/1                          Punkte im lokalen Koordinaten-
                                      system #1
P2=100,0,0,0/1
P11=0,100,0,0
P12=100,100,0,0                       'Punkte im Roboterkoordinaten-
                                      system
>LOCAL1 (P1:P11),(P2:P12)             Definiert das lokale Koordinaten-
                                      system #1
>
>LOCAL                                Zeigt alle lokalen Koordinaten-
                                      systeme an
local 0(p***:p***),(p***:p***)        Über BASE 0 definiertes Local0-
                                      Koordinatensystem
local 1(p1  :p11 ),(p2  :p12 )        Über LOCAL definiertes lokales
                                      Koordinatensystem #1
>
```

LOCAL0 (RLOCAL0, LLOCAL0)

> S

Local Zero

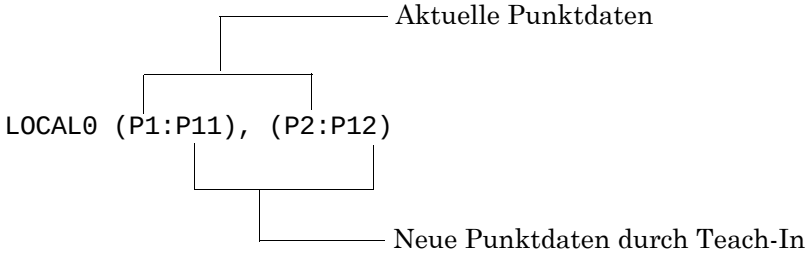
FUNKTION Definiert das Basiskoordinatensystem des Roboters, Local 0.

FORMAT
```
LOCAL0(P[Punkt Nr. 1]:P[Punkt Nr. 2])(P[Punkt Nr. 3]:~
                      P[Punkt Nr. 4])
```

BESCHREIBUNG Alle definierten lokalen Koordinatensysteme basieren auf einem einzigen lokalen Basiskoordinatensystem, dem sogenannten "Local0-Koordinatensystem". Normalerweise entspricht das Local0-Koordinatensystem dem Roboterkoordinatensystem, d.h. dem absoluten Koordinatensystem des Roboters.

Auch wenn die Position des Roboterkoordinatensystems nicht verändert werden kann, so kann doch die Position des Local0-Koordinatensystems mit Hilfe des Befehls LOCAL0 (oder BASE 0) verändert werden. Bedenken Sie aber, daß eine Neudefinierung des Local0-Koordinatensystems nicht erforderlich ist, solange es dem Roboterkoordinatensystem entspricht.

Wurde der Roboter zum Zwecke der Wartung oder aus anderen Gründen von seinem Maschinentisch ab- und wieder aufmontiert, entsprechen die Punktdaten nicht mehr den Punkten im System und machen dadurch die aktuellen Punktdaten ungültig. In diesem Fall können die aktuellen Punktdaten wieder gültig gemacht werden, indem man den Roboter mit zwei Punkten der aktuellen Punktdaten programmiert und dann über den Befehl LOCAL0 das Local0-Koordinatensystem definiert.



Bewegen Sie den Manipulator zu der jeweiligen Position, die P1 bzw. P2 der aktuellen Punktdaten entspricht, und programmieren Sie den Roboter so, daß er P1 als P11 bzw. P2 als P12 erkennt. Führen Sie dann den Befehl LOCAL0 wie zuvor beschrieben aus.

Der Befehl LOCAL0 und seine Varianten LLOCAL0 und RLOCAL0 definieren die XY-Ebene sowie die Z- und U-Achse in der gleichen Weise wie der Befehl LOCAL und seine Varianten. Lesen Sie dazu die Erläuterungen zum Befehl LOCAL.

Die Werte des Local0-Koordinatensystems bleiben über das Ausschalten hinaus gültig.

Wollen Sie sicherstellen, daß das Local0-Koordinatensystem dem Roboterkoordinatensystem entspricht, verwenden Sie einen der folgenden Befehle:

```
LOCAL0  (P1:P1),(P1:P1)
BASE  0,0,0,0,0
VERINIT
```

Da der Befehl LOCAL0 die Position des Basiskoordinatensystems verändert, sollte er nur wenn unbedingt erforderlich verwendet werden.

Bei Ausführung des Befehls LOCAL0 werden alle lokalen Koordinatensysteme gelöscht, auch die über BASE definierten; d.h., sie müssen erneut definiert werden.

Der Befehl VERINIT initialisiert das Local0-Koordinatensystem, d.h. es ist dann identisch mit dem Roboterkoordinatensystem.

VERWANDTE BEFEHLE LOCAL, BASE 0, BASE

LOF( )

F

Line of File (Zeile einer Datei)

FUNKTION Gibt die Anzahl der im RS-232C-Puffer gespeicherten Datenzeilen aus.

FORMAT LOF([Portnummer])

Die Portnummer muß eine ganze Zahl zwischen 20 und 23 sein.

BESCHREIBUNG Die an den RS-232C-Port gesandten Daten (z.B. von einem Bildverarbeitungssystem an die Robotersteuerung) werden in einem Puffer gespeichert und nacheinander über den Befehl INPUT # aus dem Puffer wieder ausgelesen. Der Befehl LOF gibt die Anzahl der im Puffer gespeicherten Datenzeilen aus bzw. zeigt 0 an, wenn keine Datenzeilen gespeichert sind.

VERWANDTE BEFEHLE INPUT #

BEISPIEL

```
>PRINT LOF (2)
5
>
```

Zeigt die Anzahl der im Puffer von Port 20 gespeicherten Datenzeilen an

L

LP, POWER

> S

Lower Power

FUNKTION Schaltet den Motor-Power-Modus ein oder aus bzw. zeigt den aktuellen Status des Modus an.

FORMAT

LP {|ON |} |OFF|	POWER {|LOW |} POWER |HIGH|
Standardwert = ON	Standardwert = LOW

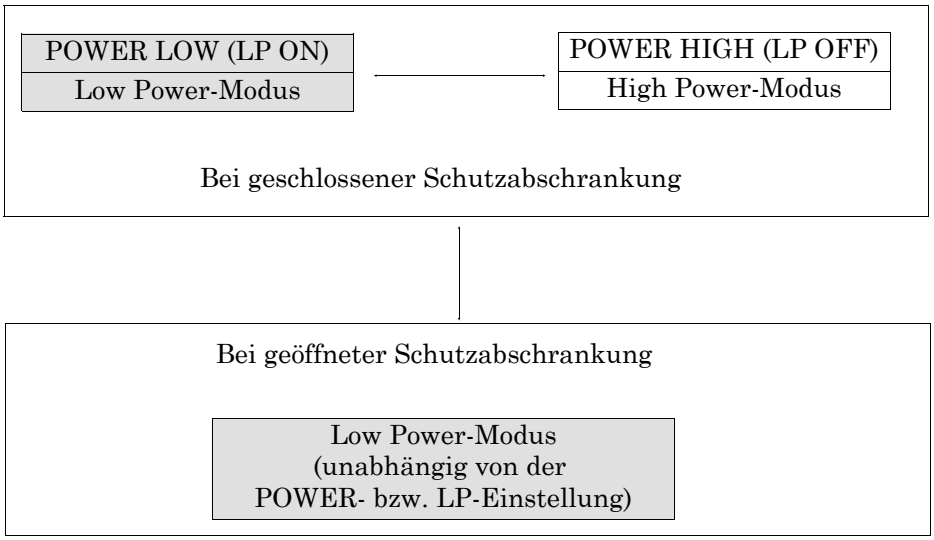
BESCHREIBUNG Schaltet den Motor-Power-Modus ein oder aus bzw. zeigt den aktuellen Status des Modus an.

LP Wird der Parameter ON gesetzt, wird der Low Power-Modus eingeschaltet.
Wird der Parameter OFF gesetzt, wird der High Power-Modus eingeschaltet.
Ohne Angabe einer der Parameter ON oder OFF zeigt der Befehl LP den aktuellen Status des Modus an.

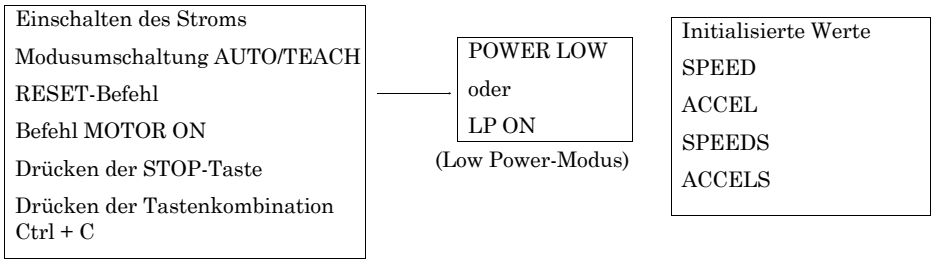
POWER Wird der Parameter LOW gesetzt, wird der Low Power-Modus eingeschaltet.
Wird der Parameter HIGH gesetzt, wird der High Power-Modus eingeschaltet.
Ohne Angabe einer der Parameter LOW oder HIGH zeigt der Befehl POWER den aktuellen Status des Modus an.

```
>POWER
Mode :LOW        ←   aktueller Modus
State:LOW        ←   tatsächlicher Status
```

Die Funktion der Befehle LP and POWER sind identisch.



Aus Sicherheitsgründen wird der Low Power-Modus im TEACH-Modus immer eingeschaltet, d.h., bei folgenden Operationen (Reset-Operationen) wird der Motor-Power-Modus standardmäßig auf LOW (niedrig) gesetzt. Dabei werden die Einstellungen für die Geschwindigkeit und die Beschleunigung auf die Standardwerte zurückgesetzt (initialisiert). Informationen zu diesen Standardwerten erhalten Sie in den Spezifikationen den Manipulatorarms, da diese modellabhängig sind.



Im Low Power-Modus wird die Motorleistung gedrosselt, so daß die tatsächliche Einstellung für die Geschwindigkeit des Motors niedriger ist als der Standardwert. Wird eine höhere Geschwindigkeit direkt oder in einem Programm angegeben, wird die Geschwindigkeit auf den Standardwert zurückgesezt.

Motor-Power-Status	Tatsächliche Bewegungsgeschwindigkeit
Low-Power-Status	Standardwert des Befehls zur Geschwindigkeitseinstellung Aktueller, über den Befehl zur Geschwindigkeitseinstellung definierter Wert } der niedrigere Wert
High-Power-Status	Aktueller, über den Befehl zur Geschwindigkeitseinstellung definierter Wert

Falls Sie im AUTO-Modus eine höhere Geschwindigkeit für Bewegungsbefehle in einem Programm einstellen wollen, fügen Sie den Befehl POWER HIGH oder LP OFF in das Programm ein.

Wenn sich das System im Low Power-Modus befindet und der Manipulatorarm von Hand oder durch ein Operation zur Abwärtsbewegung bewegt wird, kann u.U. Fehler 173 aufgrund der gedrosselten Motorkraft auftreten.

Bei Eingabe eines Befehls zur Geschwindigkeitssteuerung oder zur Programmausführung im Low Power-Modus wird folgende Meldung angezeigt. Sie besagt, daß sich das Robotersystem im Low Power-Status befindet und mit niedriger Geschwindigkeit arbeitet.

```
Low Power State : xxxxx ist begrenzt auf xxx
```

VERWANDTE BEFEHLE POWER, SPEED, ACCEL, SPEEEDS, ACCELS

BEISPIEL

```
>SPEED 50                          Definiert eine höhere Geschwindig-
                                   keit
 Low Power State : SPEED ist begrenzt auf 5
>ACCEL 100,100
 Low Power State : ACCEL ist begrenzt auf 10
>JUMP P1                           Bewegung bei gedrosselter
                                   Geschwindigkeit
>SPEED; ACCEL                      Zur Anzeige des aktuellen
                                   Geschwindigkeitsstatus
 Low Power State : SPEED is begrent auf 10
       50
       50          50

 Low Power State : ACCEL ist begrenzt auf 10
      100       100
      100       100
      100       100
>XQT
>LP OFF                            Stellt den High Power-Modus ein
                                   (=POWER HIGH)
>JUMP P2                           Bewegung mit hoher
                                   Geschwindigkeit
```

LSHIFT( )

F

Left Shift

FUNKTION Verschiebt die numerischen Daten nach links.

FORMAT `LSHIFT([numerische Daten],[Anzahl zu verschiebender Bits])`

BESCHREIBUNG Verschiebt die angegebenen numerischen Daten um die angegebene Anzahl von Bits nach links (zur höchsten Bitstelle). Für jede nach links verschobene Stelle wird auf der rechten Seite eine 0 in die freigewordene Stelle eingesetzt.

VERWANDTE BEFEHLE RSHIFT( ), NOT( )

BEISPIEL

```
>PRINT LSHIFT(1,2)
 4
>I=5
>PRINT LSHIFT(I,1)
 10
>
```

L

MCAL

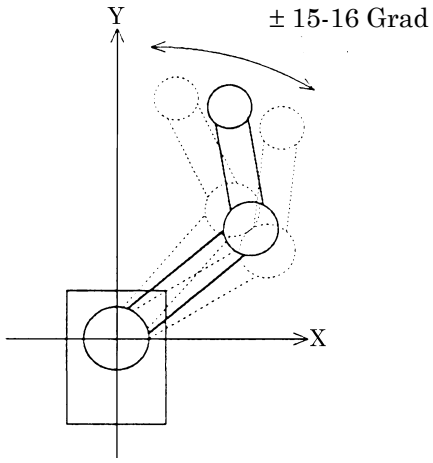
INC > S

Machine Calibration (Maschinenkalibrierung)

FUNKTION Führt eine Maschinenkalibrierung durch.

FORMAT MCAL

BESCHREIBUNG Eine Maschinenkalibrierung mit Hilfe dieses Befehls ist bei allen INC-Robotern (inkrementellen Robotern) erforderlich, bei denen ein inkrementeller Encoder installiert ist. Die Maschinenkalibrierung muß nach dem Einschalten des Stroms erfolgen. Wurde die Maschinenkalibrierung nicht durchgeführt und ein Befehl zur Bewegungssteuerung oder der Befehl PLIST* soll ausgeführt werden (diese Befehle benötigen die aktuellen Positionsdaten), erfolgt ein Fehler.
Die Maschinenkalibrierung wird in der über den Befehl MCORDR definierten Reihenfolge der Achsenbewegung durchgeführt. Der werkseitige Standardwert für den Befehl MCORDR ist modellabhängig. Nähere Informationen dazu erhalten Sie in der Dokumentation zum Manipulatorarm.
Führen Sie den MCAL-Befehl in der Mitte des Bewegungsbereichs aus. Bei einem Roboter mit einem Multi-Kalibrierungspunktsystem bewegen Sie die erste und die zweite Achse ± 15-16 Grad im Uhrzeigersinn bzw. gegen den Uhrzeigersinn. Wird der Befehl MCAL nahe am Limit des Bewegungsbereich ausgeführt überschreitet der Arm den Bereich. In diesem Fall erfolgt ein Fehler oder der Manipulator stößt gegen den mechanischen Stopper, so daß eine Kalibrierung nicht durchgeführt werden kann.
Die Bewegungsdistanz bei einer Maschinenkalibrierung ist modellabhängig. Nähere Informationen dazu erhalten Sie in der Dokumentation zum Manipulatorarm.



Bei einem BN-Roboter sollten Sie die vierte Achse bei ausgeschalteter Steuerung nicht um mehr als 180° drehen; ansonsten erfolgt Fehler 237, wenn Sie den Strom einschalten und den Befehl MCAL ausführen. Schalten Sie in diesem Fall den Strom ab, bewegen Sie die vierte Achse an die entsprechende vorherige Position zurück und schalten Sie den Strom erneut ein.

VERWANDTE BEFEHLE MCORDR, MCORG, MCOFS

BEISPIEL

```
>MOTOR ON
>MCAL
```

MCOFS

INC >

Machine Calibration Offset

FUNKTION

Definiert die Parameter für die Maschinenkalibrierung bzw. zeigt die aktuellen Parameter an.

FORMAT

```
MCOFS  {[Wert 1], [Wert 2],~
        [Wert 3], [Wert 4],~
        [Wert 5], [Wert 6],~
        [Wert 7]}
```

BESCHREIBUNG

Definiert die Parameter für die Maschinenkalibrierung bzw. zeigt die aktuellen Parameter an.
Dieser Befehl definiert die Daten, die zur Ausführung der Maschinenkalibrierung mit Hilfe des Befehls MCAL notwendig sind.
Die Bedeutung der einzelnen Werte veranschaulicht die folgende Tabelle:

Wert 1	Sensorlogik nach Ausführung des Befehls MCORG	hexadezimal
Werte 2 bis 5	Pulse-Werte zwischen dem Zeitpunkt, an dem der Sensor eingeschaltet ist, bis zum Zeitpunkt, an dem die Z-Phase an der festgelegten Position der ersten bis vierten Achse erkannt wird.	dezimal
Werte 6 und 7	Differenz zwischen dem Logikwert und der Breite der Sensorkante an der festgelegten Position der ersten und zweiten Achse	dezimal

Werden nach dem Befehl keine Werte angegeben, werden die aktuell eingestellten Parameter für die Kalibrierung wie folgt angezeigt.

```
[Wert 1]
[Wert 2]        [Wert 3]
[Wert 4]        [Wert 5]
[Wert 6]        [Wert 7]
```

Wenn Sie alle Werte angeben, werden sie als Parameter für die Maschinenkalibrierung eingestellt. Die Parameter zur Kalibrierung werden durch Bewegung des Manipulatorarms mit Hilfe des Befehls MCORG errechnet. Sollten die Werte jedoch bereits feststehen, können Sie diese Werte über den Befehl MCOFS direkt eingeben.

Die mit dem Befehl MCOFS definierten Werte bleiben auch über das Ausschalten hinaus bzw. nach Ausführung des VERINIT-Befehls gültig.

Bei Ausführung der Maschinenkalibrierung mit Hilfe des Befehls MCAL wird die aktuelle Position anhand dieser Daten erkannt, so daß die Angabe der Daten unbedingt notwendig ist. Werden hierbei die falschen Daten angegeben, sind die Roboterkoordinaten nicht korrekt, so daß auch die Roboterbewegungen nicht korrekt sind.

Dieser Befehl zur Angabe der Parameter für eine Kalibrierung sollte nur nach Wartungsarbeiten verwendet werden, z.B., wenn die MPU-Platine ausgetauscht wurde. Verwenden Sie diesen Befehl nicht für andere Aufgaben.

Die folgenden Erläuterungen sind lediglich ein kurzer Überblick über das Vorgehen beim Austausch einer MPU-Platine. Nähere Informationen dazu erhalten Sie in der entsprechenden Dokumentation zur Wartung.

1. Lassen Sie sich die aktuellen Parameter für die Maschinenkalibrierung mit Hilfe des Befehls MCOFS anzeigen, bevor Sie die MPU-Platine austauschen.
2. Notieren Sie sich alle der aufgeführten 7 Parameter.
3. Setzen Sie eine neue MPU-Platine in die Steuerung ein.
4. Geben Sie die zuvor notierten Parameter nach dem Befehl MCOFS in der richtigen Reihenfolge ein und führen Sie den Befehl aus.

Die Parameter für die Kalibrierung können auch mit Hilfe der Befehle MKVER und SETVER eingestellt werden bzw. mit Hilfe der Funktionstasten **V-BKUP** und **V-RSTR** im SPEL-Editor.

VERWANDTE BEFEHLE MCORG MCAL VER MKVER

BEISPIEL

```
>MCOFS
04
1364  50
4506  6304
-479  -469
>MCOFS &H04,1364,50,4506,6304,-479,-469
```

MCORDR

INC > S

Machine Calibration Order

FUNKTION

Definiert die Reihenfolge der Achsenbewegung bei der Maschinenkalibrierung bzw. zeigt die aktuellen Werte an.

FORMAT

(1) `MCORDR {[Wert 1], [Wert 2],~`
`[Wert 3], [Wert 4]}`

Die jeweiligen Standardwerte sind modellabhängig.

(2) `MCORDR`

BESCHREIBUNG

(1) Definiert die Reihenfolge der Achsenbewegung bei der Maschinenkalibrierung mit Hilfe des Befehls MCAL. Die mit dem Wert 1 angegebene Achse (oder mehrere Achsen) führt die Kalibrierung zuerst aus, anschließend die über den Wert 2 definierte Achse, dann die über den Wert 3 sowie die über den Wert 4 definierte Achse.

Jede Achse ist einem Bit von Bit0 bis Bit3 wie folgt zugeordnet:

Achse	1. Achse	2. Achse	3. Achse	4. Achse
Bit-Nr.	Bit3	Bit2	Bit1	Bit0
Binär-Code	&B1000	&B0100	&B0010	&B0001

Informationen zu den werkseitigen Standardwerten erhalten Sie in der Dokumentation zum Manipulatorarm.

Die über den Befehl MCORDR definierten Werte bleiben auch über das Ausschalten des Roboters hinaus gültig. Bei Ausführung des Befehls VERINIT werden sie auf ihre Standardwerte zurückgesetzt.

Bei einem Roboter mit einer Kugelumlaufspindel geben Sie die vorherige Reihenfolge für die dritte und vierte Achse an oder dieselbe Reihenfolge, da ansonsten bei Ausführung des Befehls MCAL ein Fehler erfolgt.

(2) Zeigt die aktuellen MCORDR-Werte in hexadezimaler Form wie folgt an:

[Wert 1] [Wert 2]
[Wert 3] [Wert 4]

VERWANDTE BEFEHLE

MCAL. MCOFS

BEISPIEL

```
>MCORDR &B0010,&B1000,&B0100,&B001   'Definiert die Reihenfolge
                                     wie folgt:
>MCORDR                              3., 1., 2., 4. Achse
02 08
04 01                                Die Werden werden in
                                     hexadezimaler Form angezeigt.
```

MCORG

INC >

Machine Calibration Origin

FUNKTION Berechnet die Parameter für die Maschinenkalibrierung.

FORMAT `MCORG [Achsennummer]{,[Achsennummer]}3`

Die Achsennummer muß eine ganze Zahl von 1 bis 4 sein.

BESCHREIBUNG Berechnet die Parameter, die zur Durchführung der Maschinenkalibrierung mit Hilfe des Befehls MCAL erforderlich sind.

Dieser Befehl sollten nur nach Wartungsarbeiten verwendet werden, z.B. nach Austausch eines Motors. Verwenden Sie diesen Befehl nur wenn unbedingt notwendig.

Wenn eine Achsennummer angegeben wird, wird der Parameter dieser Achse berechnet. Sollen bei einem Roboter mit einer Kugelumlaufspindel die Parameter für die dritte und vierte Achse berechnet werden, müssen Sie beide Achsen gleichzeitig angeben. Wird nur eine angegeben, erfolgt ein Fehler. Auch wenn keine Achsennummer angegeben wird, erfolgt ein Fehler.

Die zur Kalibrierung berechneten Parameter bleiben auch über das Ausschalten hinaus gespeichert und werden auch bei einer Initialisierung z.B. durch den Befehl VERINIT nicht verändert. Die in der Steuerung gespeicherten Parameter können mit Hilfe des Befehls MCOFS angezeigt werden.

Bei Ausführung des Befehls MCORG bewegen sich alle angegebenen Achsen. Stellen Sie also vor der Ausführung des Befehls sicher, daß sich keine Hindernisse um den Roboter herum befinden. Stellen Sie vor allem sicher, daß sich die dritte Achse in einer höheren Position befindet.

Bei einem Roboter mit einem Multi-Kalibrierungspunktsystem ist die Armposition, in der der Befehl MCORG ausgeführt werden darf, begrenzt.

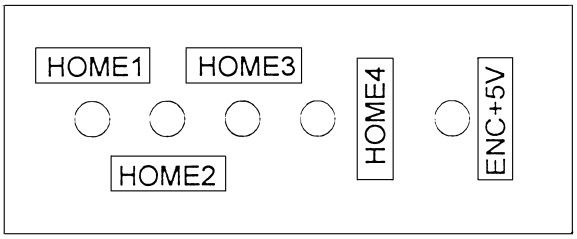
Führen Sie den Befehl MCORG in keiner anderen Position aus.

Bei anderen Robotern kann dieser Befehl in jeder Position ausgeführt werden.

Die Berechnung der Parameter für die Maschinenkalibrierung (über den Befehl MCORG) bei einem Roboter mit einem Multi-Kalibrierungspunktsystem wird im folgenden beschrieben.

Bewegen Sie die dritte und vierte Achse des Roboters innerhalb des Bewegungsbereichs.

Legen Sie die Armposition fest. Verwenden Sie dazu die LEDs am Sensormonitor an der Rückseite des Manipulatorsockels. Die LEDs werden nachfolgend dargestellt. Die Ziffern 1 bis 4 mit der Bezeichnung "HOME" entsprechen der Achsennummer des Manipulatorarms.



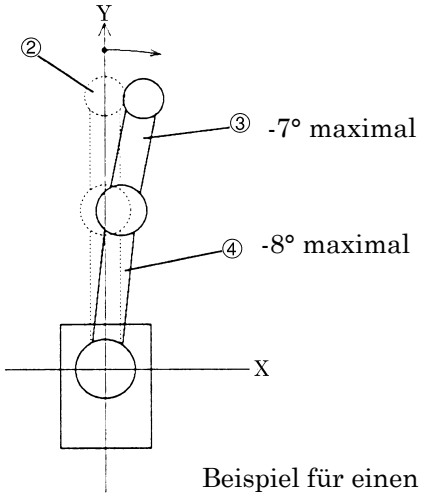
Wenn Sie die erste oder zweite Achse bewegen, geht die jeweilige LED entsprechend der Armbewegung an bzw. aus.

Bewegen Sie den Arm, wie durch die gestrichelte Linie in der folgenden Abbildung dargestellt, so daß der Arm parallel zur Y-Achse des Roboter-Koordinatensystems positioniert ist. In diesem Fall gehen die LEDs HOME1 und HOME2 an.

Beobachten Sie die LED HOME2 und bewegen Sie den zweiten Arm manuell im Uhrzeigersinn. Halten Sie dabei den ersten Arm fest, so daß dieser sich dabei nicht bewegt. Positionieren Sie den zweiten Arm ungefähr in der Mitte des Bereichs, bei dem die LED zum ersten Mal ausgeht.

Beobachten Sie die LED HOME1 und bewegen Sie den ersten Arm manuell im Uhrzeigersinn. Positionieren Sie den ersten Arm ungefähr in der Mitte des Bereichs, bei dem die LED zum ersten Mal ausgeht.

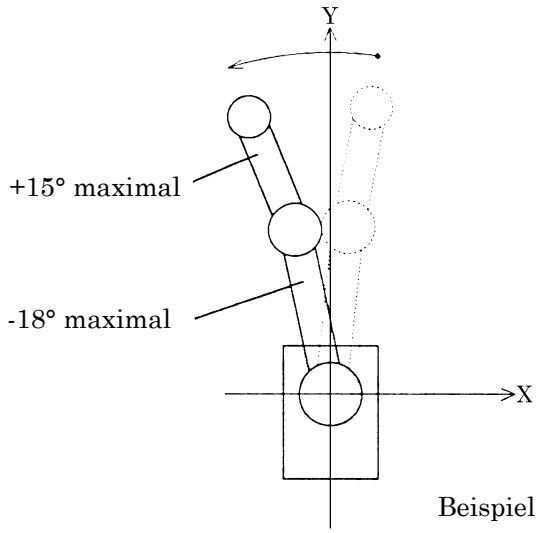
Für den ersten und zweiten Arm ist die ungefähre Position weniger als der in der folgenden Abbildung angegebene Grad. Der Grad ist modellabhängig. Nähere Einzelheiten dazu finden Sie in der Dokumentation zum Manipulatorarm.



Beispiel für einen BN-Roboter

Führen Sie den Befehl MOTOR ON aus und schalten Sie die Motoren ein.

Wenn Sie in dieser Position den Befehl MCORG ausführen, bewegen sich der erste und zweite Arm entgegen dem Uhrzeigersinn. Der maximale Bewegungsgrad entspricht den oben dargestellten Werten. Entfernen Sie alle Hindernisse in diesem Bereich, bevor Sie den Befehl MCORG ausführen.

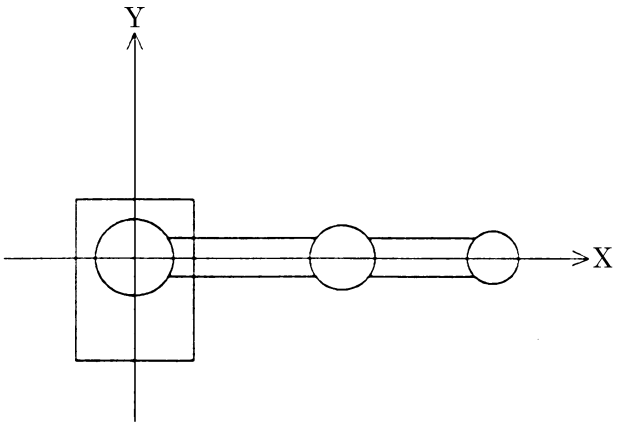


Beispiel für einen BN-Roboter

Führen Sie den Befehl MCAL aus, um die Maschinenkalibrierung durchzuführen.

Führen Sie in der 0-Pulse-Position des ersten und zweiten Arms den Befehl PULSE aus.
>PULSE 0,0,0,0

Stellen Sie sicher, daß der erste und zweite Arm gerade sind und sich auf der X-Achse des Roboter-Koordinatensystems befinden. Ist dies nicht der Fall, ist die Armposition bei Ausführung des Befehls MCORG nicht korrekt.



PULSE 0,0,0,0

Wird der Befehl MCORG bei einem Roboter mit einem Multi-Kalibrierungspunktsystem nicht in der korrekten Position ausgeführt, werden die falschen Parameter für die Kalibrierung berechnet. Dies bedeutet, daß auch die Bewegungen des Roboters nicht korrekt sind.
Sollten Sie den Befehl MCORG versehentlich ausführen, dürfen Sie keine Bewegungsbefehle ausführen lassen. Führen Sie in diesem Fall den Befehl MCORG an der korrekten Position erneut aus oder ermitteln Sie die werkseitig vorgegebenen Werte mit Hilfe des Befehls MCOFS und geben Sie diese Daten ein.

VERWANDTE BEFEHLE

MCAL, MCOFS

BEISPIEL

```
>MCORG 1,2        'Berechnet die Parameter für die Maschinenkalibrierung
                   der ersten und zweiten Achse.
```

MERGE

FUNKTION

Überträgt die Daten aus der Programmiereinheit in den Hauptspeicher der Steuerung und mischt sie mit den dort bereits gespeicherten Daten.

FORMAT

```
MERGE {|PRG|}
      |PNT|
```

BESCHREIBUNG

MERGE überträgt die in der Programmiereinheit gespeicherte Programm- und/oder Punktdatendatei in den Hauptspeicher der Steuerung und mischt diese Daten mit denen der dort gespeicherten Programm-/Punktdatendatei.

Wollen Sie nur die Programmdateien mischen, führen Sie den Befehl MERGE PRG aus.

Wollen Sie nur die Punktdatendateien mischen, führen Sie entsprechend den Befehl MERGE PNT aus.

Zum Mischen sowohl der Programm- als auch der Punktdatendateien führen Sie nur den Befehl MERGE ohne Parameter aus.

Enthalten die Programmdateien der Programmiereinheit und des Hauptspeichers dieselbe Zeilennummer, überschreibt MERGE den Zeileninhalt der Hauptspeicherdatei mit dem Zeileninhalt der Datei aus der Programmiereinheit.

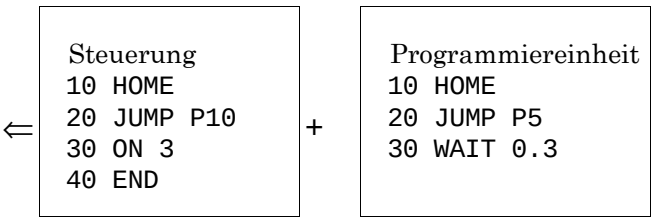
Dies gilt entsprechend für Punktdatendateien mit derselben Punktnummer, d.h., die Punktdaten der Hauptspeicherdatei werden mit denen der Datei aus der Programmiereinheit überschrieben.

VERWANDTE BEFEHLE

LIST, PLIST

BEISPIEL

```
>MERGE PRG
>LIST
10 HOME
20 JUMP P5
25 WAIT 0.3
30 ON 3
40 END
>
```




```
>MERGE      Mischt die Programm- und Punktdatendateien
```

MID$( )

F

Middle $

FUNKTION Gibt die angegebene Anzahl von Zeichen in einer angegebenen Zeichenkette ab der festgelegten Position aus.

FORMAT

```
MID$(|[Zeichenkettenvariablen]|,[Startposition],[Zeichenanzahl])
     |    "[Zeichenkette]"    |
```

BESCHREIBUNG Gibt die angegebene Anzahl von Zeichen in einer angegebenen Zeichenkette ab der festgelegten Position aus.

Die angegebene Zeichenkette kann als Zeichenkettenvariablen definiert werden.

Die Startposition wird vom äußerst links gelegenen Zeichen der Zeichenkette aus gerechnet.

VERWANDTE BEFEHLE LEFT$, RIGHT$

BEISPIEL

```
>PRINT MID$("ABCDE",3,2)
CD
>A$="1234567"
>PRINT MID$(A$,2,3)
234
>
```

MKDIR, MD

Make Directory (Verzeichnis erstellen)

FUNKTION Erstellt ein Unterverzeichnis.

FORMAT `MKDIR {[Laufwerk]}{[Pfad]}[Verzeichnis]`

BESCHREIBUNG Legt auf dem angegebenen Pfad ein Verzeichnis an.

Bei Auslassung des Pfadnamens wird das Unterverzeichnis im aktuellen Verzeichnis erstellt.

Bei Auslassung des Laufwerknamens wird das Unterverzeichnis auf dem aktuellen Laufwerk angelegt.

Es kann mit einem MKDIR-Befehl nur jeweils ein Unterverzeichnis erstellt werden.

VERWANDTE BEFEHLE DIR

BEISPIEL

```
>MD \USR
>MD \USR\PNT
```

MKVER

Make Ver

FUNKTION Erstellt ein Backup bestimmter Einstellungen im Dateispeicher bzw. auf der Festplatte.

FORMAT `MKVER {/A}`

BESCHREIBUNG Erstellt ein Backup bestimmter Einstellung sowie von allen Daten im Hauptspeicher im aktuellen Verzeichnis des aktuellen Laufwerks mit entsprechenden Dateinamen. Die Dateinamen werden automatisch vergeben.

Folgende Daten und Einstellungen, die mit den entsprechenden Befehlen definiert wurden, werden als Backup gesichert:

CALPLS	HOFS	MCORDR	MCOFS
SEL	SET	ARCH	
TSPEED	TSPEEDS	HOMESET	HORDR
ARM	ARMSET	TOOL	TLSET
RANGE	XYLIM	BASE 0	
CONFIG	CONSOLE	SELDISK	WIDTH
PRGNO	OPUNIT	WEIGHT	MAXDEV
PRGSIZE	PNTSIZE	LIBSIZE	
Merker-Einstellungen			Einstellung an Remot

Wird der Parameter /A nicht angegeben, werden die Daten und Backup-Variablen mit den folgenden Dateinamen gesichert.

Daten	Dateiname
Einstellungen	MK#0.BAT
Backup-Variable	LIB#0.BIN

Die Batch-Datei "MK#0.BAT" enthält Befehle zur Angabe der o.g. Daten und Einstellungen sowie einen Befehl zum Laden der Backup-Variablendatei "LIB#0.BIN" in den Hauptspeicher.

Wird der Parameter /A angegeben, werden die o.g. Einstellungen sowie alle Daten aus dem Haupspeicher mit den folgenden Dateinamen gesichert.

Daten	Dateiname
Einstellungen	MK#0.BAT
Quellprogramm	PRG#0.PNT
Positionsdaten	PNT#0.PNT
Objektprogramm	OBJ#0.BIN
Symboltabelle	SYM#0.BIN
Backup-Variable	LIB#0.BIN

Die Batch-Datei "MK#0.BAT" enthält Befehle zur Angabe der o.g. Daten und Einstellungen sowie einen Befehl zum Laden der zuvor genannten fünf Dateien in den Hauptspeicher.

Über den Befehl MKVER werden Backup-Variablen gesichert, jedoch nicht die anderen Variablen.

Um jede der Dateien, die mit dem Befehl MKVER gesichert wurden, in den ursprünglichen Speicherbereich zu laden, gibt es zwei Methoden:

Führen Sie den Befehl SETVER aus.

Führen Sie die Batch-Datei MK#0.BAT aus.

Um die Dateien, die über den Befehl MKVER im Dateispeicher erstellt wurden, auf die Festplatte der Programmiereinheit zu kopieren, verwenden Sie die Funktionstaste **BACKUP** im SPEL-Editor.

VERWANDTE BEFEHLE

SETVER

BEISPIEL

Zu den gespeicherten Einstellungen gehören wichtige Basisdaten wie z.B. die HOFS-Werte. Wenn der Roboter zum ersten Mal installiert wird, empfiehlt es sich, die Backup-Daten mit Hilfe des Befehls MKVER zu erstellen und diese Dateien aufzubewahren.

Dieser Befehl wird verwendet, wenn die MPU- oder SPU-Platine ausgetauscht wurde. Er ist besonders hilfreich, um die Einstellungen auf die neue Platine zu laden.

Sichern Sie die Daten auf der auszutauschenden (alten) Platine in einer Datei.

```
>MKVER                          Die folgenden Dateien werden
                                erstellt
 CPU Data Backup                MK#0.BAT
 Backup der Backup-Variablen    LIB#0.BIN
```

Erstellen Sie ein Backup aller Dateien einschließlich der zuvor genannten Dateien im Dateispeicher auf der Festplatte der Progrmmiereinheit. Verwenden Sie dazu die Funktionstaste **BACKUP**.

Installieren Sie eine neue Platine in die Steuereinheit.

Laden Sie alle Dateien auf der Festplatte der Programmiereinheit in den Dateispeicher der Steuereinheit. Verwenden Sie dazu die Funktionstaste **RESTOR**.

Führen Sie den Befehl SETVER aus.

```
>SETVER
```

MOTOR

> S

M

FUNKTION Schaltet die Stromzufuhr zu den Motoren ein bzw. aus.

FORMAT

```
MOTOR |ON  |
      |OFF |
```

BESCHREIBUNG Mit MOTOR ON werden die Motoren aller Achsen bestromt, die Lageregelung wird aktiviert und die Bremsen gelöst; beim Ausschalten wird die Stromzufuhr zu allen Motoren unterbrochen und die Bremsen angezogen.

Nach einem Abschalten mit Not-Aus oder nach Auftauchen eines Fehlers, der das Rücksetzen über den Befehl RESET erforderlich macht, führen Sie zuerst den Befehl RESET und anschließend MOTOR ON aus.

VERWANDTE BEFEHLE SFREE, SLOCK, RESET

BEISPIEL

```
>MOTOR OFF
>MOTOR ON
>
>SFREE 1,2,4        Schaltet die Achsen 1, 2, und 4 frei
>
```

MOVE

> S

FUNKTION Bewegt alle vier Achsen gleichzeitig durch lineare Interpolation.

FORMAT

(1) `MOVE [PS] {TILL}`

(2) `MOVE [PS] {TILL SW(Eingang){=|0|}}{![Parallelanweisung]!}`
`|1|`

Der Parameter [PS] steht für die Positionsangaben.

BESCHREIBUNG Bewegt durch eine lineare Interpolation alle vier Achsen gleichzeitig.

Der MOVE-Befehl verwendet den mit SPEEDS angegebenen Geschwindigkeitswert sowie die mit ACCELS angegebenen Beschleunigungswerte. Sollte der mit SPEEDS vorgegebene Geschwindigkeitswert die maximal mögliche Geschwindigkeit einer der vier Achsen überschreiten, wird die Stromzufuhr zu den Motoren abgeschaltet und die Robotertätigkeit unterbrochen.

Eine Bewegung nur der U-Achse ist mit dem MOVE-Befehl nicht möglich.

Eine vorherige Überprüfung des Trajektoriebereichs durch den MOVE-Befehl ist nicht möglich. Das bedeutet, selbst wenn die Zielpositionen innerhalb des erlaubten Bereichs liegen, kann es vorkommen, daß der Manipulator in der Bewegung versucht, diesen zulässigen Bereich zu verlassen. In diesem Fall wird die Bewegung durch einen heftigen Ruck unterbrochen, wobei der Arm beschädigt werden kann. Daher sollten Sie zuvor den gesamten Bereich mit langsamer Geschwindigkeit überprüfen.

Die TILL-Bedingung wird verwendet, um den Manipulator an einer Zwischenposition abzubremsen und anzuhalten, wenn die aktuelle TILL-Bedingung erfüllt ist. Ist die TILL-Bedingung nicht erfüllt, verfährt der Manipulator bis zur Zielposition.

(1) MOVE mit TILL-Bedingung
Überprüft, ob die aktuelle TILL-Bedingung erfüllt ist. In diesem Fall wird der Befehl beendet, indem der Manipulator abgebremst und gestoppt wird.

(2) MOVE mit TILL-Bedingung, SW(Eingang)-Bedingung und (0 oder 1) Eingangsbedingung:
Überprüft, ob dieselbe Zeileneingangsbedingung erfüllt wird. In diesem Fall wird der Befehl beendet, indem der Manipulator an einer Zwischenposition abgebremst und angehalten wird.

MOVE mit TILL-Bedingung, SW(Eingang)-Bedingung, jedoch ohne Eingangsbedingung:
Die Eingangsbedingung erhält den Standardwert 1. Geht der angegebene Eingang auf ON, wird der Manipulator an einer Zwischenposition abgebremst und angehalten.

VERWANDTE BEFEHLE P=, ! ... !, SPEEDS, ACCELS, TILL, SW( ), CMOVE, ARC, CARC, CVMOVE

BEISPIEL

```
100 TILL SW(1)=0 AND SW(2)=1    Definiert die Eingangsbedingung
                                (Eingang 1 ist Aus, Eingang 2 ist Ein)
110 MOVE P1 TILL                Stoppt, wenn die aktuelle TILL-
                                Bedingung (Zeile 100) erfüllt ist
120 MOVE P2 TILL SW(2)=1        'Stoppt, wenn Eingang 2 Ein ist
130 MOVE P3 TILL                Stoppt, wenn die aktuelle TILL-
                                Bedingung (Zeile 100) erfüllt ist
```

MYTASK(0)

F

FUNKTION Gibt die Nummer der laufenden Task aus (mytask).

FORMAT MYTASK(0)

Ziffer 0 in Klammern ( )

BESCHREIBUNG Gibt die Nummer der laufenden Task als numerischen Wert aus.

BEISPIEL In dem folgenden Beispielprogramm werden die Ausgänge 1 bis 8 ein- bzw. ausgeschaltet.

```
>LIST
10  FUNCTION MAIN
20  XQT !2, TASK        Führt Programm TASK in Task 2 aus
30  XQT !3, TASK        Führt Programm TASK in Task 3 aus
40  XQT !4, TASK        Führt Programm TASK in Task 4 aus
50  XQT !5, TASK        Führt Programm TASK in Task 5 aus
60  XQT !6, TASK        Führt Programm TASK in Task 6 aus
70  XQT !7, TASK        Führt Programm TASK in Task 7 aus
80  XQT !8, TASK        Führt Programm TASK in Task 8 aus
90  '
100 '
110 ON MYTASK(0)        'Ausgang der Mytask-Nummer (hier Ausgang 1)
                        einschalten
120 OFF MYTASK(0)'Ausgang der Mytask-Nummer (hier Ausgang 1)
                         ausschalten
130 GOTO 110
150 '
200 FUNCTION TASK
210 ON MYTASK(0)         Ausgang der Mytask-Nummer (hier Ausgang
                        2 ... 8) einschalten
220 OFF MYTASK(0) Ausgang der Mytask-Nummer
                        (hier Ausgang 2 ... 8) ausschalten
230 GOTO 210
240 FEND
```

NEW

FUNKTION Löscht das Quellprogramm aus dem Quellprogrammbereich.

FORMAT NEW

BESCHREIBUNG Löscht das Quellprogramm aus dem Quellprogrammbereich.

Wird der Befehl im Online-Modus ausgeführt, wird das Quellprogramm im Hauptspeicher der Steuerung gelöscht (Initialisierung des Quellprogrammbereichs).

Bei Ausführung des Befehls im Offline-Modus wird das Quellprogramm in der Programmiereinheit gelöscht.

NEW wird verwendet, um den Speicher vor Eingabe eines neuen Programmes zu löschen.

VERWANDTE BEFEHLE CLEAR

BEISPIEL

```
>LIST
10 'NEW PROGRAM
20 HOME
30 JUMP P1
40 JUMP P2
50 JUMP P3
60 END
>
>NEW
>LIST
>_
```

Da das zuvor gespeicherte Programm durch den NEW-Befehl gelöscht wurde, wird nichts angezeigt

NORMAL

> S

FUNKTION Deaktiviert den invertierten Anzeigemodus (dunkle Zeichen auf hellem Hintergund) an der Bedieneinheit.

FORMAT `NORMAL [x-Spalte],[Y-Zeile],[Zeichenanzahl]`

X muß eine ganze Zahl von 1 bis 32 sein.

Y muß eine ganze Zahl von 1 bis 8 sein.

Die Anzahl der Zeichen muß eine ganze Zahl von 1 bis 32 sein.

BESCHREIBUNG Deaktiviert den invertierten Anzeigemodus für die angegebene Anzahl von Zeichen ab der durch die X- und Y-Werte definierten Position.

Ist der angegebene Bereich länger als eine Zeile, verschiebt sich dieser zusätzliche Bereich nicht in die nächste Zeile, sondern kehrt an den Zeilenanfang zurück und zeigt die Zeichen in normaler Form an.

VERWANDTE BEFEHLE REVERSE, CHARSIZE, CLS, CURSOR, OPUNIT, OPU, PRINT

BEISPIEL

```
>NORMAL 10,2,3
```

NOT ( )

F

FUNKTION Gibt den invertierten Wert des ganzzahligen Bits aus.

FORMAT `NOT([ganze Zahl])`

BESCHREIBUNG Gibt den invertierten Wert des angegebenen ganzzahligen Bits aus.

VERWANDTE BEFEHLE LSHIFT ( ), RSHIFT ( )

OFF

> S

FUNKTION Schaltet einen Ausgang aus und ggf. nach einer bestimmten Zeit wieder ein.

FORMAT `OFF [Ausgang]{,[Intervall]{,[asynchrone Einstellung]}}`

Der Ausgang muß eine ganze Zahl zwischen 0 und 127 sein.
Das Intervall wird in Sekunden (Mindesteinheit ist 0,01) angegeben.
Für die asynchrone Einstellung kann 0 oder 1 (Standardwert ist 1) gewählt werden.

BESCHREIBUNG Wird nur der Ausgang angegeben, wird dieser Ausgang ausgeschaltet.

Wird auch ein Zeitintervall angegeben, wird der Ausgang ausgeschaltet (OFF) und nach Ablauf des festgelegten Intervalls wieder eingeschaltet. War der Ausgang bereits vor Ausführung des Befehls OFF ausgeschaltet, wird er nach Ablauf des Intervalls eingeschaltet.

Asynchrone Einstellungen sind verwendbar, wenn das Intervall folgendermaßen definiert wird:

1 Schaltet den Ausgang aus und nach Ablauf des Intervalls wieder ein, führt dann den nächsten Befehl aus.

0 Schaltet den Ausgang aus und führt gleichzeitig den nächsten Befehl aus.

Wird keine Einstellung gewählt, ist dies identisch mit Einstellung 1.

Wird ein Ausgang angegeben, der für den Anschluß Remote3 konfiguriert ist, erfolgt ein Fehler. Die Ausgänge von Remote3 werden automatisch entsprechend dem Status der Roboter-Steuerung ein- bzw. ausgeschaltet.

Der Befehl RESET schaltet alle Ausgänge aus (OFF).

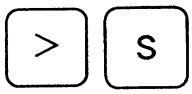
Bei einem Not-Aus werden alle Ausgänge ausgeschaltet. Um den aktuellen Status beizubehalten, müssen Sie Bit 6 von Softwareschalter SS1 einschalten. Nähere Informationen dazu erhalten Sie im Handbuch zum SPEL Editor.

VERWANDTE BEFEHLE ON, OUT, OPBCD, SW ( )

BEISPIEL

`>OFF 1`	Schaltet Ausgang 1 aus
`>OFF 1,3`	Schaltet Ausgang 1 aus und nach einem Intervall von 3 Sekunden wieder ein
`>OFF 1,3,0 ;GO P1`	Schaltet Ausgang 1 aus und beginnt gleichzeitig mit einer Bewegung nach P1. Nach 3 Sekunden wird Ausgang 1 wieder eingeschaltet.

OFF $



FUNKTION Schaltet einen Merker aus.

FORMAT `OFF $[Merker Nr.]`

Die Merker-Nr. muß eine ganze Zahl zwischen 0 und 511 sein.

BESCHREIBUNG Schaltet den angegebenen Merker aus.

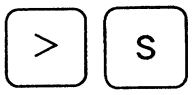
VERWANDTE BEFEHLE ON $, SW ($), IN ($)

BEISPIEL

```
>OFF $9
```

Schaltet Merker 9 aus

ON



FUNKTION

Schaltet den angegebenen Ausgang ein und ggf. nach einer bestimmten Zeit wieder aus.

FORMAT

```
ON [Ausgang]{,[Intervall]{[asynchrone Einstellung]}}
```

Der Ausgang muß eine ganze Zahl zwischen 0 und 127 sein.
Das Intervall wird in Sekunden (Mindesteinheit ist 0,01) angegeben.
Für die asynchrone Einstellung kann 0 oder 1 (Standardwert ist 1) gewählt werden.

BESCHREIBUNG

Wird nur der Ausgang angegeben, wird dieser Ausgang eingeschaltet.

Bei Angabe eines Intervalls wird der Ausgang nach Ablauf des Intervalls wieder ausgeschaltet, nachdem er zuvor eingeschaltet worden war.

Asynchrone Einstellungen sind verwendbar, wenn das Intervall folgendermaßen definiert wird:

1 Schaltet den Ausgang ein und führt nach Ablauf des Intervalls den nächsten Befehl aus.

0 Schaltet den Ausgang ein und führt gleichzeitig den nächsten Befehl aus.

Wird keine Einstellung gewählt, ist dies identisch mit Einstellung 1.

Wird ein Ausgang angegeben, der für den Anschluß Remote3 konfiguriert ist, erfolgt ein Fehler. Die Ausgänge von Remote3 werden automatisch entsprechend dem Status der Roboter-Steuerung ein- bzw. ausgeschaltet.

Der Befehl RESET schaltet alle Ausgänge aus (OFF).

Bei einem Not-Aus werden alle Ausgänge ausgeschaltet. Um den aktuellen Status beizubehalten, müssen Sie Bit 6 von Software-Schalter SS1 einschalten. Nähere Informationen dazu erhalten Sie im Handbuch zum SPEL Editor.

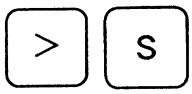
VERWANDTE BEFEHLE

OFF, OUT, OPBCD, SW ( )

BEISPIEL

`>ON 1`	Schaltet Ausgang 1 ein
`>ON 1,3`	Schaltet Ausgang 1 für 3 Sekunden ein und anschließend wieder aus
`>ON 1,3,0 ;GO P1`	Schaltet Ausgang 1 für 3 Sekunden ein und dann wieder aus. Beginnt beim Einschalten von Ausgang 1 mit der Bewegung nach P1

ON $



FUNKTION Schaltet einen Merker ein.

FORMAT `ON $[Merker Nr.]`

Die Merker-Nr. muß eine ganze Zahl zwischen 0 und 511 sein.

BESCHREIBUNG Schaltet den angegebenen Merker ein.

VERWANDTE BEFEHLE OFF $, SW ($), IN ($)

BEISPIEL `>ON $5` Schaltet Merker 5 ein

ONERR ... RETURN

S

On Error ... Return (Bei Fehler ... zurückkehren zu)

FUNKTION Legt die Fehlerroutine bei Auftreten eines Fehlers fest.

FORMAT

(1)
```
ONERR      |Zeilennummer|
           |Label       |
  .
  .
  .
|Zeilennummer|
|Label       |
  .
  .
  .
ECRL
  .
  .
  .
RETURN
```

(2)
```
ONERR 0
```

BESCHREIBUNG

(1) Nach Auftreten eines Fehlers, entweder bei der Zeilennummer oder beim Label, verzweigt das Programm in die Fehlerroutine. Mit der Anweisung RETURN wird die Programmsteuerung wieder an die Zeile übergeben, die auf den Fehler folgt.
Normalerweise wird bei Auftreten eines Fehlers die Fehlernummer angezeigt und die Ausführung des Programms abgebrochen. Wird jedoch der Befehl ONERR eingefügt, ist es möglich, bei einem Fehler während der Programmabarbeitung in eine Fehlerroutine zu verzweigen, und somit die Programmausführung fortzusetzen.
Die Fehlerroutine muß den Befehl ECLR zum Löschen des Fehlerstatus beinhalten. Verschachtelte Fehlerroutinen innerhalb einer Fehlerroutine sind nicht erlaubt. (Näheres zu Verschachtelungen finden Sie in der Beschreibung des #include-Befehls.)

ONERR ... RETURN kann in jeder Task, in der eine Fehlerbearbeitung erfolgen soll, eingesetzt werden, und zwar mehrmals.

(2) Löscht den Befehl ONERR.

VERWANDTE BEFEHLE ECLR, ERR(0), ERL(0)

BEISPIEL

```
10 ONERR 60
20 FOR I=0 TO 199
30 JUMP PI
40 NEXT I
50 END
60 '
70 'ERR SUB       'Unterroutine zur Fehlerbearbeitung
80 A=ERR(0)
90 PRINT A
100 ECLR          'Löscht den Fehlerstatus
110 RETURN
```

OPBCD

> S

Output by Binary Coded Decimal (Ausgabe entsprechend BCD-Wert)

FUNKTION

Schaltet die Ausgänge eines Ausgangsports (8 Ausgänge) entsprechend des BCD-Wertes.

FORMAT

```
OPCD [Portnummer],[Ausgangsdaten]
```

Die Portnummer muß eine ganze Zahl zwischen 0 und 15 sein.
Als Ausgangsdaten muß eine ganze Zahl von 0 bis 99 angegeben werden.

BESCHREIBUNG

Schaltet die Ausgänge eines Ausgangsports (definiert durch die Portnummer) entsprechend des BCD-Wertes.

Jeder Port besteht aus 8 Ausgängen; die standardmäßige 16E/16A-Schnittstelle besteht aus 2 Ports.

Portnummer, LSB/MSB und Ausgänge sind wie folgt miteinander verbunden:

Port-nummer	MSB 7	6	5	4	3	2	1	LSB 0
0	7	6	5	4	3	2	1	0
1	15	14	13	12	11	10	9	8

LSB bedeutet "Least significant bit" (niedrigste Bitstelle).
MSB bedeutet "Most significant bit" (höchste Bitstelle)

Die Ausgangsdaten werden durch den BCD-Wert der oberen und unteren 4 Bit dargestellt. Da der BCD-Wert 9 der größte durch einen 4-Bit-BCD-Wert darstellbare Wert ist, können Daten von BCD 00 bis 99 ausgegeben werden.

Wird ein Ausgang angegeben, der für den Anschluß Remote3 konfiguriert ist, erfolgt ein Fehler. Die Ausgänge von Remote3 werden automatisch entsprechend dem Status der Roboter-Steuerung ein- bzw. ausgeschaltet.

Der Befehl RESET schaltet alle Ausgänge aus (OFF).

Bei einem Not-Aus werden alle Ausgänge ausgeschaltet. Um den aktuellen Status beizubehalten, müssen Sie Bit 6 von Softwareschalter SS1 einschalten. Nähere Informationen dazu erhalten Sie im Handbuch zum SPEL Editor.

VERWANDTE BEFEHLE

INBCD, OUT

BEISPIEL

```
OPBCD 0,17
```

Schaltet Ausgang 0, 1, 2 und 4 ein,
und Ausgang 3, 5, 6, und 7 aus

7	6	5	4	3	2	1	0	Bit
0	0	0	1	0	1	1	1	→ &H17

OPORT ( )

F

Output Port (Ausgangsport)

FUNKTION Gibt den Status eines Ausgangs aus.

FORMAT `OPORT([Ausgangs-Nr.)`

Die Ausgangs-Nr. muß eine ganze Zahl von 0 bis 127 sein.

BESCHREIBUNG Gibt den Status des angegebenen Ausgangs als 0 oder 1 aus.

0: Aus (Off-Status)
1: Ein (On-Status)

Der Befehl OPORT ( ) funktioniert nicht bei Ausgängen,die für den Anschluß Remote3 konfiguriert sind und gibt in diesem Fall 0 aus.

BEISPIEL

```
>ON 5
>PRINT OPORT (5)
1
>
```

OPUNIT

> S

Operating Unit (Bedieneinheit)

FUNKTION Wählt den Betriebsmodus der Bedieneinheit aus.

FORMAT `OPUNIT {[Modusnummer]}`

Die Modusnummer muß eine ganze Zahl von 0 bis 3 sein.

BESCHREIBUNG Wählt den Betriebsmodus der Bedieneinheit aus. Wird keine Modusnummer angegeben, wird die aktuell eingestellte Modusnummer angezeigt.

Im Modus 0 (Systemmodus) werden verschiedene Funktionen einer entsprechenden Taste an der Bedieneinheit zugeordnet, um dadurch z.B. Monitorfunktionen oder Funktionen zur Auswahl von Dateien zu steuern. Außerdem werden die für diese Funktionen erforderlichen Meldungen angezeigt.

Im Modus 0 können Sie den Tasten zusätzlich eigene Funktionen zuordnen. Achten Sie jedoch darauf, daß Sie eine eigene Funktion nicht einer Taste zuordnen, der im Modus 0 ursprünglich bereits eine Funktion zugeordnet ist.

Im Modus 3 (Anwendermodus) sind die meisten der ursprünglich zugeordneten Systemfunktionen unwirksam, so daß Sie die Tasten nach eigenen Erfordernissen verwenden können. In diesem Modus wird der Bildschirm ausschließlich für den Anwender verwendet, das System benutzt ihn nicht. Die folgenden Tasten sind frei verfügbar:

F1 bis **F4**, ↑, ↓, ←, → und **MENU**.

Die Tasten **START**, **PAUSE** und **RESET** sind immer für Systemfunktionen reserviert und können nicht frei belegt werden.

In den Modi 1 und 2 ist auch die Monitorfunktion verfügbar. Im Modus 2 ist zusätzlich die Anzeige von Fehlermeldungen verfügbar, d.h., bei einem Fehler wird eine Fehlermeldung ausgegeben. In den Modi 1 und 2 sind die folgenden Tasten frei verfügbar:

F1 bis **F4**, ↑, ↓, ←, → (im weiteren **TASTEN** genannten).

Wird die Monitorfunktion nicht verwendet, sind die **TASTEN** frei verfügbar, wird die Monitorfunktion jedoch verwendet, ist die erforderliche Funktion je einer Taste zugeordnet. In diesem Fall sollten Sie darauf achten, daß eine durch den Anwender definierte Funktion nicht einer Taste zugeordnet wurde, die bereits mit einer Monitorfunktion belegt ist.

Wird in Modus 2 ein Fehler ausgegeben, wird der Bildschirm vollständig gelöscht, selbst wenn eine Anwendermeldung angezeigt wird, und eine Fehlermeldung wird angezeigt.

Modus Nr.	Modus-Name	Frei/reserviert	Freie Tasten
0	Systemmodus	Reserviert (bei allen Funktionen)	----------
1	Anwendermodus 1	frei mit Monitorfunktion	**TASTEN** (reserviert bei eingeschalteter Monitorfunktion)
2	Anwendermodus 2	frei mit Monitorfunktion Anzeige von Fehlermeldungen	**TASTEN** (reserviert bei eingeschalteter Monitorfunktion)
3	Anwendermodus 3	alle frei	**TASTEN**, **MENU**

Eingaben über die Tasten der Bedieneinheit werden durch die Funktion DSW( ) gelesen.

Bei Einschalten der Stromzufuhr ist standardmäßig der Modus 0, Systemmodus eingeschaltet.

VERWANDTE BEFEHLE

DSW( ), OPU PRINT, CHARSIZE, CLS, CURSOR, NORMAL, REVERSE

BEISPIEL

```
1000 OPUNIT 2                                          Modusauswahl
1010 OPU PRINT 1,3,"Select operation mode."            Meldungsanzeige
                                                       zur Auswahl des
                                                       Betriebsmodus
1020 OPU PRINT 1,4,"F1:Normal operation"
1030 OPU PRINT 1,5,"F2:Dummy operation"
1040 OPU PRINT 1,6,"F4:Operation End"
1050 WAIT (  DSW(3) AND &B10110000 ) 0                 Wartet auf
                                                       Funktionstasten-
                                                       eingabe
1060 IF DSW(3) AND &B10000000 THEN GOTO F4KEY
1070 IF DSW(3) AND &B00100000 THEN GOTO F2KEY
1080 IF DSW(3) AND &B00010000 THEN GOTO F1KEY
```

OPU PRINT

> S

Operating Unit Print

FUNKTION Gibt vorgegebene Zeichen an der Bedieneinheit aus.

FORMAT

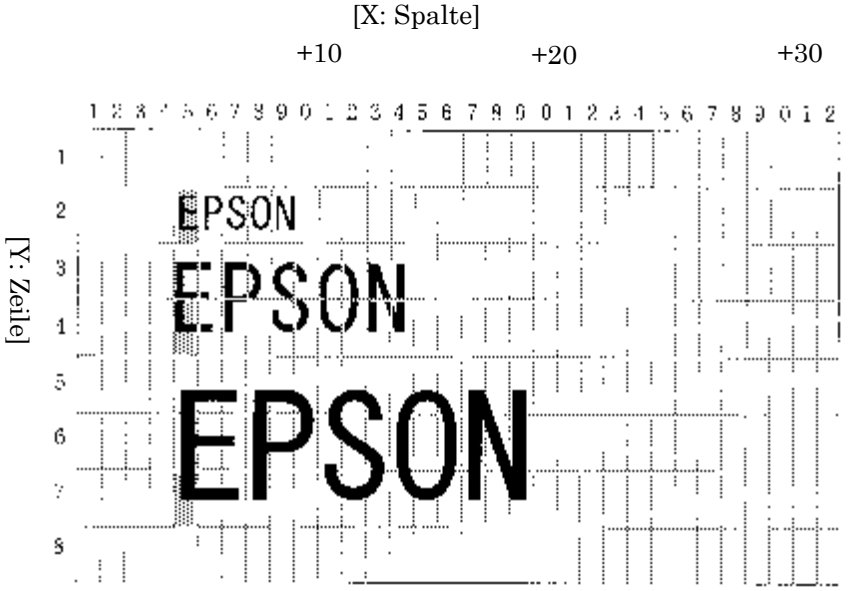
```
OPU PRINT [X-Spalte],[Y-Zeile],|"[Zeichenkette]"|{,|"[Zeichenkette]"|}n
                               | [num. Wert]    |  | [num. Wert]    |
                               | [Variablenname]|  | [Variablenname]|
                               | [Funktionsname]|  | [Funktionsname]|
```

BESCHREIBUNG Gibt die angegebene Zeichenkette an der Bedieneinheit aus und zeigt sie an der durch den X- und Y-Wert definierten Position an.

VERWANDTE BEFEHLE CHARSIZE, CLS, CURSOR, NORMAL, REVERSE, OPUNIT

BEISPIEL

```
>CHARSIZE 2
>OPU PRINT 5,2, "EPSON"
>CHARSIZE 4
>OPU PRINT 5,4, "EPSON"
>CHARSIZE 9
>OPU PRINT 5,7, "EPSON"
```



Durch den X- und Y-Wert definierte Position

OUT

> S

FUNKTION Schaltet die Ausgänge eines Ausgangsports (8 Ausgänge) entsprechend eines Binärwertes.

FORMAT `OUT [Portnummer],[Ausgangsdaten]`

Die Portnummer muß eine ganze Zahl von 0 bis 15 sein.
Als Ausgangsdaten muß eine ganze Zahl von 0 bis 255 angegebn werden.

BESCHREIBUNG Sendet Daten an den über die Portnummer angegebenen Ausgang.

Jeder Port besteht aus 8 Ein-/Ausgängen. Eine Standard-16E/16A-Schnittstelle hat 2 Ports.

Portnummer, LSB/MSB und Ein-/Ausgänge sind folgendermaßen miteinander verbunden:

Port-nummer	MSB 7	6	5	4	3	2	1	LSB 0
0	7	6	5	4	3	2	1	0
1	15	14	13	12	11	10	9	8

LSB bedeutet "Least significant bit" (niedrigste Bitstelle)
MSB bedeutet "Most significant bit" (höchste Bitstelle)

Die Ausgangsdaten können entweder als Dezimal- oder Hexadezimalwert (mit &H) angegeben werden.

Wird ein Ausgang angegeben, der für den Anschluß Remote3 konfiguriert ist, erfolgt ein Fehler. Die Ausgänge von Remote3 werden automatisch entsprechend dem Status der Roboter-Steuerung ein- bzw. ausgeschaltet.

Der Befehl RESET schaltet alle Ausgänge aus (OFF).

Bei einem Not-Aus werden alle Ausgänge ausgeschaltet. Um den aktuellen Status beizubehalten, müssen Sie Bit 6 von Softwareschalter SS1 einschalten.

VERWANDTE BEFEHLE IN, OPBCD, INBCD

BEISPIEL

`> OUT 0,28` Ausgang 2, 3 und 4 sind eingeschaltet, die übrigen sind ausgeschaltet (an Port 0)

7	6	5	4	3	2	1	0	Bit
0	0	0	1	1	1	0	0	→ 28

`> OUT 1,&H66` Ausgang 9, 10, 13 und 14 sind eingeschaltet, die übrigen sind ausgeschaltet (an Port 1)

7	6	5	4	3	2	1	0	Bit
0	1	1	0	0	1	1	0	→ &H66

OUT $

> S

FUNKTION Schaltet Merker-Port (8 Merker) entsprechend eines Binärwertes.

FORMAT `OUT $ [Portnummer],[Ausgangsdaten]`

Die Portnummer muß eine ganze Zahl zwischen 0 und 63 sein.
Als Ausgangsdaten muß eine ganze Zahl von 0 bis 255 angegeben werden.

BESCHREIBUNG Sendet Daten an einen über die Portnummer definierten Merker.

Jeder Port besteht aus 8 Bit. Da es sich um insgesamt 512 Bit handelt, besteht ein Merker aus 64 Ports.

Portnummer, LSB/MSB und Merker sind folgendermaßen miteinander verbunden:

Port-nummer	MSB 7	6	5	4	3	2	1	LSB 0
0	7	6	5	4	3	2	1	0
1	15	14	13	12	11	10	9	8
62	503	502	501	500	499	498	497	496
63	511	510	509	508	507	506	505	504

LSB bedeutet "Least significant bit" (niedrigste Bitstelle).
MSB bedeutet " Most significant bit" (höchste Bitstelle).

VERWANDTE BEFEHLE IN ($)

BEISPIEL

```
>OUT $3,5
```

PALET

> S

Pallet (Palette)

FUNKTION Definiert bzw. zeigt Paletten an.

FORMAT (1) `PALET[Paletten-Nr.] P[Punkt 1],P[Punkt 2],P[Punkt 3]{,P[Punkt 4]}~`

`[Anzahl Reihen],[Anzahl Spalten]`

Die Paletten-Nr. muß eine ganze Zahl zwischen 0 und 15 sein. [Anzahl Reihen] ist die Anzahl der Reihen von P[Punkt 1] bis P[Punkt 2]. [Anzahl Spalten] ist die Anzahl der Spalten von P[Punkt 1] bis P[Punkt 3]

Das Produkt aus [Anzahl Reihen] mal [Anzahl Spalten] darf das Ergebnis von 32.767 nicht überschreiten.

(2) PALET

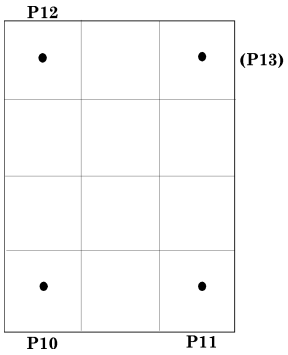
BESCHREIBUNG (1) Definiert eine Palette, indem die entsprechenden Punktdaten (für mindestens 3 Punkte 1, 2 und 3) im TEACH-Modus gelehrt werden. Dabei wir die Anzahl der Reihen und Spalten von Punkt 1 zu Punkt 2 und von Punkt 1 zu Punkt 3 angegeben.

Hat die Palette eine regelmäßige rechteckige Form, brauchen nur drei der vier Eckpunkte definiert zu werden. Bei einer Palette mit weniger regelmäßiger Form müssen dementsprechend alle vier Eckpunkte angegeben werden.

Zur Palettendefinition werden zuerst 3 oder 4 Eckpunkte im TEACH-Modus gelehrt. Definieren Sie die Palette anschließend wie folgt:

Die linke Seite der folgenden Abbildung zeigt eine Palette mit drei Punkten von Punkt 1 zu Punkt 2 bzw. mit vier Punkten von Punkt 1 zu Punkt 3. Dabei entspricht P10 Punkt 1, P11 Punkt 2 und P12 Punkt 3; d.h., es gibt drei Reihen von P10 zu P11 und vier Spalten von P10 zu P12. Also muß zur Definition dieser Palette folgendes angegeben werden:

```
PALET1 P10,P11,P12,3,4
```



10	11	12
7	8	9
4	5	6
1	2	3

Wie auf der rechten Seite der vorherigen Abbildung dargestellt, wird jedem Punkt der Palette automatisch eine Nestnummer zugeordnet, in unserem Beispiel beginnend mit P10. Diese Nestnummern sind außerdem für den Befehl PALET(n) erforderlich.

Beachten Sie, daß eine falsche Reihenfolge der Punkte oder der Nestnummern zwischen den einzelnen Punkten eine falsche Definition der Palettenform zur Folge hat.

Die Palettenebene wird durch die Z-Achsen-Koordinatenwerte der drei Eckpunkte P10, P11 und P12 definiert. Auf diese Weise können Sie auch senkrechte Paletten definieren.

(2) Zeigt alle definierten Paletten an.

VERWANDTE BEFEHLE

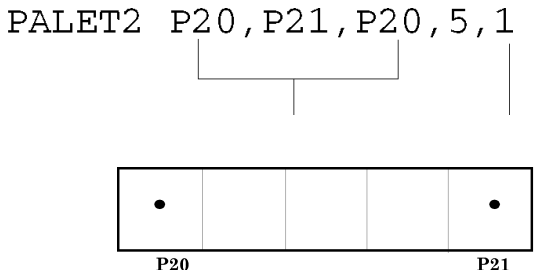
PALET(n)

BEISPIEL

```
100 PALET1 P1,P2,P3,P4,4,5     'Definiert eine Palette mit vier Punkten,
                                Punkt 1 bis 4
110 FOR I=1 TO 20
120   JUMP PALET1 (I)          Springt alle 20 Nester an
130 NEXT

>PALET
palet 1 P1 ,P2 ,P3 ,P4 , 4, 5
>
```

Eine einreihige Palette kann mit einem PALET-Befehl durch drei Punkte definiert werden. Dazu müssen Sie beide Endpunkte wie folgt definieren; außerdem müssen Sie 1 als die Anzahl der Nester zwischen demselben Punkt angeben.



PALET(n)

F

Palette

FUNKTION

Gibt die Position auf der festgelegten Palette entsprechend der Nestnummer an und wird immer mit einem Bewegungsbefehl kombiniert.

FORMAT

```
PALET[Paletten-Nr.]([Nest-Nr.])
```

Die Paletten-Nr. muß eine ganze Zahl von 0 bis 15, die Nestnummer eine ganze Zahl von 1 bis 32767 sein.

P

BESCHREIBUNG

Gibt die Position auf der festgelegten Palette (Paletten-Nr.) entsprechend der angegebenen Nestnummer (Nest-Nr.) an.

Bei der Positionierung mit dem Befehl PALET(n) sind die Operatoren +, - und : erlaubt.

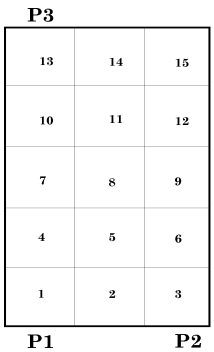
VERWANDTE BEFEHLE

PALET

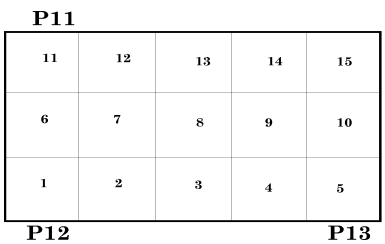
BEISPIEL

Dieses Beispielprogramm veranlaßt den Roboter Teile aus der Palette 1 in die Palette 2 zu transportieren.

```
10 PALET1 P1,P2,P3,3,5          Definiert Palette 1
20 PALET2 P12,P13,P11,5,3       Definiert Palette 2
30 FOR = I=1 TO 15
40 JUMP PALET1 (I)              Bewegung zu Palette 1, Nest-Nr. I
50 ON 1                         Teil aufnehmen
50 WAIT 0.5
70 JUMP PALET2 (I)              Bewegung zu Palette 2, Nest-Nr. I
80 OFF 1                        Teil absetzen
90 WAIT 0.5
100 NEXT I
110 END
```



Palette 1



Palette 2

PASS

> S

FUNKTION

Führt eine PTP-Bewegung aller vier Achsen aus und bewegt den Manipulatorarm dabei an festgelegten Punkten vorbei (Verschleifen von Hilfspunkten, ohne diese exakt zu überfahren).

FORMAT

```
PASS [Punktserie]{,|ON |[Ausgang],[Punktserie]}n
                     |OFF|
```

BESCHREIBUNG

Führt eine PTP-Bewegung aller vier Achsen aus und bewegt den Manipulatorarm dabei an festgelegten Punktfolge-Punkten vorbei (Verschleifen von Hilfspunkten, ohne diese exakt zu überfahren).

Zur Definition von Punktfolge-Punkten, verwenden Sie Punkte (P0, P1,...), die jeweils durch Kommas (,) getrennt werden. Zur Definition einer durchgehenden Folge, sowohl in aufsteigender als auch in absteigender Reihenfolge, verwenden Sie einen Bindestrich (-), wie z.B. Pi-Pj).

Wollen Sie einen Ausgang während der Bewegung ein- oder ausschalten, fügen Sie ON bzw. OFF, eingeschlossen in Kommas, zwischen die einzelnen Punkte ein. Die ON- bzw. OFF-Anweisung wird ausgeführt, bevor der Manipulatorarm den Punkt erreicht, der der ON- bzw. OFF-Anweisung unmittelbar vorangeht.

Folgt unmittelbar auf einen PASS-Befehl ein weiterer PASS-Befehl, erfolgt die Bewegung kontinuierlich, d.h., der Manipulatorarm stoppt nicht am letzten Punkt des vorangehenden PASS-Befehls.

Folgt auf einen PASS-Befehl ein anderer Befehl zur Bewegungssteuerung (nicht PASS), stoppt der Manipulatorarm am letzten Punkt des vorangehenden PASS-Befehls. Eine Beeinflussung über den FINE-Befehl wird jedoch nicht ausgeführt.

Folgt auf einen PASS-Befehl ein Befehl, eine Anweisung oder Funktion, der/die nicht zur Bewegungssteuerung dient, wird dieser Befehl bzw. diese Anweisung oder Funktion ausgeführt, bevor der Manipulatorarm den letzten Punkt des vorangehenden PASS-Befehls erreicht.

Soll an der Zielposition ein Positionierung über den FINE-Befehl ausgeführt werden, geben Sie nach dem PASS-Befehl eine GO-Anweisung ein, wobei Sie die Zielposition wie in den folgenden Beispielen definieren:

```
PASS P5;GO P5;ON 1;MOVE P10
```

Der Befehl PASS ist besonders nützlich zur Vermeidung von Kollisionen mit Hindernissen oder für Entwurfszeichnungen mit einem Kartesianischen Roboter.

VERWANDTE BEFEHLE

SPEED, ACCEL, GO

BEISPIEL

```
>PASS P0,P2-P5,ON 2,P6,P120-P110
>_
```

PATH (Pfad)

FUNKTION Definiert, löscht bzw. zeigt den Pfad zur Ausführung einer Stapeldatei an.

FORMAT

(1) `PATH={{[Laufwerk]}[Pfad]{;{[Laufwerk]}[Pfad]}n}`

(2) `PATH`

BESCHREIBUNG

(1) Definiert und löscht den Pfad zur Ausführung einer Stapeldatei; zeigt gleichzeitig den aktuellen Pfad an. Bei Eingabe des Dateinamen wird in dem angegebenen Pfad nach der angegebenen Datei gesucht; anschließend wird die Datei ausgeführt.

Ist kein Pfad angegeben, kann die Stapeldatei nicht ausgeführt werden; achten Sie also vor Ausführung einer Datei darauf, daß der Pfad mit Hilfe des PATH-Befehls korrekt angegeben ist.

Sie können auch mehrere Pfade angeben, indem Sie die einzelnen Pfadangaben durch ein Semikolon (;) voneinander trennen. Sind mehrere Pfade angegeben, wird jeder Pfad, beginnend mit dem ersten, durchsucht und die Datei ausgeführt, sobald das System sie gefunden hat.

Wird der Befehl "PATH=" ohne Angabe eines Pfadnamens eingegeben, wird die Definition des Pfades gelöscht. In diesem Fall ist die Ausführung einer Datei nicht möglich.

(2) Zeigt den aktuell definierten Pfad an.

Der PATH-Befehl ist nur zur Ausführung einer Stapeldatei gültig.

Beim Einschalten des Roboters ist kein Pfad definiert.

VERWANDTE BEFEHLE SETENV

BEISPIEL

```
>PATH=A:\;A:\BIN          Angabe von zwei Pfaden
>PATH
>PATH=A:\;A:\BIN
>PATH=\                   Das Hauptverzeichnis ist als Pfad definiert
>PATH
PATH=\
>
```

PAUSE

S

FUNKTION Unterbricht kurzfristig die Ausführung eines Programms.

FORMAT PAUSE

BESCHREIBUNG Unterbricht kurzfristig die Ausführung eines Programms.

Normalerweise unterbricht der Befehl PAUSE die Ausführung aller Tasks.

Wurden jedoch eine oder mehrere Tasks mit Hilfe des Befehls HTASK definiert, werden nur die so definierten Tasks kurzfristig unterbrochen, die übrigen Tasks werden weiter ausgeführt.

BEISPIEL

```
10 JUMP P1
20 ON 1
30 WAIT 0.5
40 PAUSE
50 JUMP P2
60 OFF 1
70 WAIT 0.5
80 GOTO 10
```

Kurzfristige Unterbrechung. Wird als nächste Aktion der Start-Taster gedrückt, wird das Programm ab Zeile 50 weiter ausgeführt; wird als nächste Aktion der Reset-Schalter gedrückt, wird ein Reset (Zurücksetzen des Systems) ausgeführt.

PDEL

Point DELete (Punkt löschen)

FUNKTION Löscht die angegebenen Positionsdaten.

FORMAT

```
PDEL|[Punktnummer]                              |
    |[erste Punktnummer]-                       |
    |[erste Punktnummer]-[letzte Punktnummer]|
    |                   -[letzte Punktnummer]|
```

Die Punktnummer muß als ganze Zahl angegeben werden, wobei ein Bereich erlaubt ist, der zwischen 0 und 1 niedriger ist, als der PNTSIZE-Wert.

Da der ursprüngliche PNTSIZE-Wert 200 beträgt, liegt der ursprünglich erlaubte Bereich zwischen 0 und 199.

BESCHREIBUNG Löscht die Informationen zu den angegebenen Positionsdaten.

Die Angabe der Punktnummern wird wie folgt definiert:

PDEL [Punktnummer]
Löscht die Positionsdaten der angegebenen Punktnummer.
PDEL [erste Punktnummer]-
Löscht alle Positionsdaten beginnend mit der "ersten Punktnummer" bis zur letzten vorhandenen Punktnummer.
PDEL [erste Punktnummer]-[letzte Punktnummer]
Löscht alle Punktdaten beginnend mit der "ersten Punktnummer" bis zur "letzten Punktnummer" einschließlich. Um Fehler 2 zu vermeiden, muß der Wert der ersten Punktnummer niedriger sein als der Wert der letzten Punktnummer.
PDEL -[letzte Punktnummer]
Löscht alle Positionsdaten bis zur "letzten Punktnummer" einschließlich.

VERWANDTE BEFEHLE PLIST, CLEAR, PNTSIZE

BEISPIEL

```
P1=10,300,-20,0/L
P2=0,300,-40,0
P10=-50,350,0,0
>PDEL 1-2          Punkt 1 und Punkt 2 löschen
>PLIST
 P10=-50,350,0,0
>PDEL 50           Punkt 50 löschen
>PDEL 100-         Von Punkt 100 bis zum letzten Punkt löschen
```

PEEK ( )

F

FUNKTION Liest die Daten vom Ein-/Ausgabekanal.

FORMAT `PEEK([Adresse])`

Der Wert für die Adresse muß eine ganze Zahl zwischen 0 und 4095sein.

BESCHREIBUNG Liest die Daten, die von der optionalen VME-I/O-Platine kommen. Die Adreßnummer ist der Offset-Wert der Basisadresse des Ein-/Ausgabekanals (FFF000H).

Beispiel

Adreßnummer	Adresse
0	FFF000H
1	FFF001H
2	FFF002H
3	FFF003H

Bei Ausführung des Befehls PEEK mit einer ungültigen Adresse erfolgt ein Systemfehler (Busfehler).

VERWANDTE BEFEHLE POKE

BEISPIEL

```
>PRINT PEEK(0)        Liest die Daten an Adresse FFF000H
5
>
```

PLIST

Punkte auflisten

FUNKTION Zeigt die Positionsdaten an.

FORMAT

```
PLIST {|[erste Punktnummer]-{[letzte Punktnummer]|}{/W}
       |-[letzte Punktnummer]                       |
       |*                                           |
       |[Punktnummer]                               |
```

BESCHREIBUNG

PLIST
Wird weder ein Wert für die "erste Punktnummer" noch ein Sternchen (*) angegeben, werden alle Positionsdaten im Speicher ausgegeben.
PLIST [Punktnummer]
Zur Ausgabe der Positionsdaten für nur einen Punkt, geben Sie nur diese Punktnummer an.
PLIST [erste Punktnummer]-
Wird kein Wert für die "letzte Punktnummer" angegeben, werden die Positionsdaten von der "ersten Punktnummer" bis zum letzten Punkt im Speicher ausgegeben.
PLIST [erste Punktnummer]-[letzte Punktnummer]
Zeigt die Positionsdaten von der "ersten Punktnummer" bis zur "letzten Punktnummer" an. In diesem Fall muß die Zahl für die "erste Punktnummer" kleiner sein als die Zahl für die "letzte Punktnummer". Ist dies nicht der Fall, erfolgt Fehler 2.
PLIST -[letzte Punktnummer]
Wird kein Wert für die "erste Punktnummer" angegeben, werden die Positionsdaten vom Anfangs- bis zum Endpunkt im Speicher ausgegeben.
PLIST *
Zur Anzeige der Positionsdaten für die aktuelle Position ein Sternchen * eingeben.

Bei Angabe des Parameters /W werden die Positionsdaten in einem festen Format ausgegeben (s. Beispiel).

VERWANDTE BEFEHLE PDEL, CLEAR, PNTSIZE

BEISPIEL

```
>PLIST
P0=450,400,0,0
P1=1.325,330.17,-58.2,-8/L
P11=-200,400,-100,150/1/R
P13=-450.456,200,0,0/2
P14=450.000,-200.000,0,0,0/2
>
```

```
>PLIST /W
P000=   450.000,   400.000,     0.000,     0.000
P001=     1.325,   330.170,   -58.200,    -8.000  /L
P011=  -200.000,   400.000,  -100.000,   150.000/1/R
P013=  -450.000,   200.000,     0.000,     0.000/2
P014=   450.000,  -200.000,     0.000,     0.000/2
>
>PLIST *
       254.3       365.256    [X-Koordinate]    [Y-Koordinate]
     -82.963           147    [Z-Koordinate]    [U-Koordinate]
PLIST 8
PLIST 100-/W
PLIST -150
PLIST 50-100
```

PLS ( )

F

Pulse

FUNKTION Gibt den Pulse-Wert der angegebenen Achse aus.

FORMAT `PLS([Achsennummer])`

Die Achsennummer muß eine ganze Zahl zwischen 1 und 4 sein.

BESCHREIBUNG Gibt den aktuellen Pulse-Wert (ganze Zahl von 4 Byte) der angegebenen Achse aus.

Der Befehl PLS( )dient zur Überwachung der Armausrichtung.

Soll der Pulse-Wert einer Achse während einer Roboterbewegung verwendet werden, muß der Befehl PLS( ) in einer Task (einem Prozeß) verwendet werden, bei der es sich nicht um die Task zur Robotersteuerung handelt.

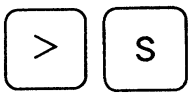
VERWANDTE BEFEHLE PUSLE, AGL( )

BEISPIEL

Anzeige der Pulse-Werte der Achsen 1 und 2 in 2 Sekundenintervallen

```
10 FUNCTION MAIN
20 LONG J1P,J2P
25 XQT !2,RB
30 J1P=PLS(1)
40 J2P=PLS(2)
50 PRINT J1P,J2P
60 WAIT 2
70 FEND
100 FUNCTION RB
110 SELRB 1
120 GO P1;WAIT 0.3
130 GO P2;WAIT 0.3
140 GOTO 120
150 FEND
```

Pn=Positionsdefinition



Point Number (Punktnummer)

FUNKTION Definiert einen Punkt.

FORMAT P[Punktnummer]=[Positionsdefinition)

BESCHREIBUNG Definiert einen Punkt und fügt ihn in die Punktdaten ein.

Die Positionsdefinition kann in einem der drei folgenden Formate angegeben werden, wobei alle drei Formate in jedem Befehl bzw. in jeder Anweisung für die Positionsdefinition verwendet werden können.

```
[X-Koordinate],[Y-Koordinate],[Z-Koordinate],[U-Koordinate]~

                              {/[Nr des lokalen Koordinatensystems]}{/|L|}
                                                                   |R|
```

```
P|[Punkt-Nr]|{/[Nr lokales Koordinatensystem]}{/|L|}{|+||X|[Koordinatenwert]}4
                                              |R|  |-||Y|
                                                   |:||Z|
                                                     |U|
```

```
PALET[Palettennummer](Nestnummer]){|+||X|[Koordinatenwert]}4
                                  |-||Y|
                                  |:||Z|
                                    |U|
```

Wird der Wert /[Nummer des lokalen Koordinatensystems] definiert, fügt SPEL III den definierten Punkt in die Punktdaten für das entsprechende lokale Koordinatensystem ein.

Die Parameter /L und /R sind nur bei SCARA-Robotern wirksam.

Mit Hilfe des Parameters /L wird der Punkt als linke Armstellung definiert; mit Hilfe des Parameters /R wird der Punkt als rechte Armstellung definiert. Wird keine Parameterangabe gemacht, wird der Punkt automatisch als rechte Armstellung interpretiert (mit Ausnahme einiger besonderer Konfigurationen).

P* bezeichnet den Punkt, an dem sich der Arm derzeit befindet.

Die X-, Y-, Z- und U-Koordinaten der Positionsdaten können geändert werden, indem einer Datenangabe wie z.B. P[Punktnummer] einer der Operatoren +, - oder : folgt.

X, Y, Z und U bezeichnen die Achsenkoordinaten.

Mit Hilfe des Operators + wird ein definierter Wert [Koordinatenwert] zum angegebenen Koordinatenwert der entsprechenden Achse hinzuaddiert.

Mit Hilfe des Operators - wird ein definierter Wert [Koordinatenwert] vom angegebenen Koordinatenwert der entsprechenden Achse subtrahiert.

Mit Hilfe des Operators : wird ein definierter Wert [Koordinatenwert] durch den angegebenen Koordinatenwert der entsprechenden Achse ersetzt.

VERWANDTE BEFEHLE

PLIST, PDEL, LOCAL, PALET

BEISPIEL

```
>P1=300,200,-50,100
>P2=-400,200,-80,100/l          Definiert die linke Armposition
>
>P3=P2+X20                      Addiert den Wert 20 mm zur
                                X-Koordinate von P2 und definiert den
                                resultierenden Punkt als P3

>PLIST 3
P3=-380,200,-80,100/L
>
>P4=P2-Y50:Z-30/R               Subtrahiert den Wert 50 mm von der
                                Y-Koordinate des Punkts P2, setzt den
                                Wert -30 mm für die Z-Koordinate ein und
                                definiert den resultierenden Punkt P4
                                als rechte Armposition
>PLIST 4
P4=-400,150,-30,100/R
>
>P5=P*                          'Definiert den aktuellen Punkt als P5
>
>P6=PALET3(5)+U90               Addiert den Wert 90 ° zur U-Koordinate
                                der Palette 3(5) und definiert den
                                resultierenden Punkt als P6
>
```

PNTSIZE, PSIZE

Point Size (Punktgröße)

FUNKTION Definiert oder zeigt die erlaubte Anzahl von Positionsdaten an.

FORMAT (1) PNTSIZE [Anzahl der Positionsdaten]

Die Anzahl der Positionsdaten muß eine ganze Zahl zwischen 1 und 1000 sein. Der Standardwert beträgt 200.

(2) PNTSIZE

BESCHREIBUNG (1) Definiert die erlaubte Anzahl der Positionsdaten im Hauptspeicher der Steuerung.

(2) Zeigt die aktuell erlaubte Anzahl der Positionsdaten an.

Bei einer Änderung der Anzahl von Positionsdaten wird zwar nicht der Programmteil gelöscht, jedoch der Bereich im Hauptspeicher für die Punktdaten sowie der Variablen- und der Objektbereich. Daher sollten Sie bei der Änderung der Anzahl von Positionsdaten folgendes berücksichtigen:

Erstellen Sie **vor der Änderung** eine Sicherungskopie der Positionsdaten im Hauptspeicher.

Speichern Sie **nach der Änderung** die Backup-Variablen zurück.

Nach der Eingabe von PNTSIZE [Anzahl der Positionsdaten] erscheint die folgende Eingabeaufforderung:

```
>Point, Backup Variable, Object all clear --> OK?
```

Wenn Sie eine neue Anzahl eingeben wollen, geben Sie Y (= Yes) bzw. y ein.

Wenn Sie die aktuelle Anzahl beibehalten wollen, geben Sie N (= No) bzw. n ein.

Da die Größe des Hauptspeichers der Robotersteuerung nicht veränderlich ist, wird bei einer Erhöhung der erlaubten Anzahl von Positionsdaten automatisch die Größe des Objektbereichs entsprechend reduziert. Aus diesem Grund sollte die Erhöhung der erlaubten Anzahl von Positionsdaten auf ein Minimales beschränkt werden.

Die erlaubte Anzahl der Punkte liegt zwischen 0 und dem Wert für die erlaubte Anzahl der Positionsdaten minus 1.

Beachten Sie, daß Positionsdaten, deren Zahl höher ist als die erlaubte Anzahl, nicht an den Hauptspeicher der Robotersteuerung gesendet werden können.

Beispiel: Die erlaubte Anzahl von Positionsdaten im Hauptspeicher ist auf den Standardwert 200 eingestellt. Wenn Sie in diesem Fall eine Positionsdatendatei mit 1000 Punkten an die Programmiereinheit senden, werden die Punktnummern 0 bis 199 gesendet, die Punktnummern 200 bis 999 werden nicht gesendet und eine Fehlermeldung wird angezeigt.

Der mit dem Befehl PNTSIZE definierte Wert bleibt auch über das Ausschalten hinaus und auch bei Ausführung des Befehls VERINIT gespeichert

VERWANDTE BEFEHLE PRGSIZE, LIBSIZE, FREE, SYS

BEISPIEL

```
>PNTSIZE 500
>PNTSIZE
         500
>
```

POKE

> S

FUNKTION Schreibt Daten an einen Ein-/Ausgabekanal.

FORMAT `POKE [Adreßnummer],[Daten]`

Die Adreßnummer muß eine ganze Zahl zwischen 0 und 4095 sein.

BESCHREIBUNG Schreibt Daten an ein optionales VME-Eingabe-/Ausgabe-Board. Bei der Adreßnummer handelt es sich um den Differenzwert zur Eingabe-/Ausgabe-Basisadresse (FFF000H).

Beispiel

Adreßnummer	Adresse
0	FFF000H
1	FFF001H
2	FFF002H
3	FFF003H

Wird mit dem Befehl POKE eine ungültige Adresse angegeben, erfolgt ein Systemfehler (Busfehler).

VERWANDTE BEFEHLE PEEK( )

BEISPIEL

```
>POKE 3,&H10          Schreibt 10H an Adresse FFF803H
>
```

POWER

> S

Lower Power

FUNKTION Schaltet den Motor-Power-Modus ein oder aus bzw. zeigt den aktuellen Status des Modus an.

FORMAT
```
POWER {|LOW |}
POWER  |HIGH|
```
Der Standardwert ist LOW.

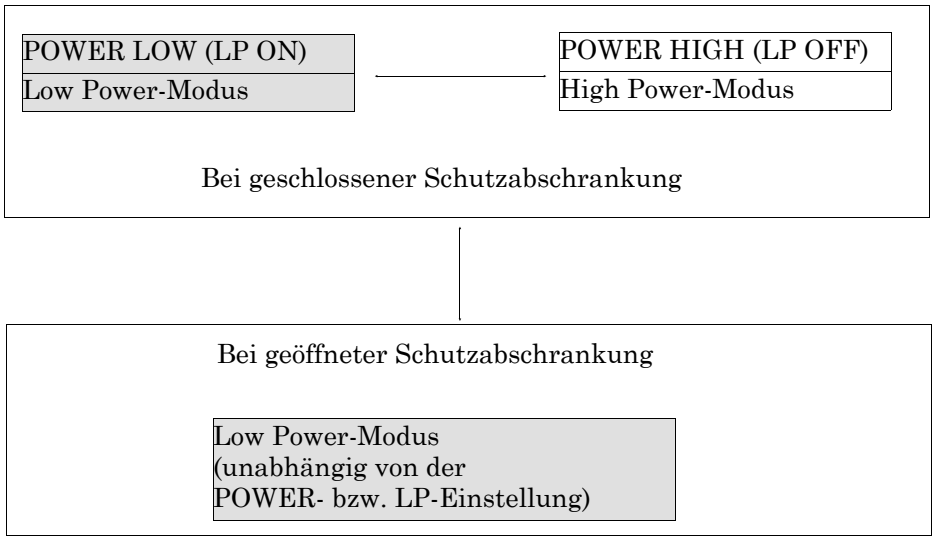
BESCHREIBUNG Schaltet den Motor-Power-Modus ein oder aus bzw. zeigt den aktuellen Status des Modus an.

Wird der Parameter LOW gesetzt, wird der Low Power-Modus eingeschaltet. Wird der Parameter HIGH gesetzt, wird der High Power-Modus eingeschaltet.

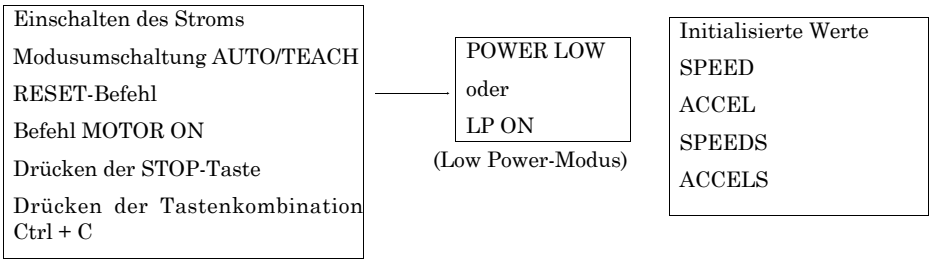
Ohne Angabe einer der Parameter LOW oder HIGH zeigt der Befehl POWER den aktuellen Status des Modus an.

```
>POWER
Mode :LOW      ←   aktueller Modus
State:LOW      ←   tatsächlicher Status
```

Die Funktion der Befehle LP and POWER sind identisch.



Aus Sicherheitsgründen wird der Low Power-Modus im TEACH-Modus immer eingeschaltet, d.h., bei folgenden Operationen (Reset-Operationen) wird der Motor-Power-Modus standardmäßig auf LOW (niedrig) gesetzt. Dabei werden die Einstellungen für die Geschwindigkeit und die Beschleunigung auf die Standardwerte zurückgesetzt (initialisiert). Informationen zu diesen Standardwerten erhalten Sie in den Spezifikationen den Manipulatorarms, da diese modellabhängig sind.



Im Low Power-Modus wird die Motorleistung gedrosselt, so daß die tatsächliche Einstellung für die Geschwindigkeit des Motors niedriger ist als der Standardwert. Wird eine höhere Geschwindigkeit direkt oder in einem Programm angegeben, wird die Geschwindigkeit auf den Standardwert zurückgesezt.

Motor-Power-Status	Tatsächliche Bewegungsgeschwindigkeit
Low-Power-Status	Standardwert des Befehls zur Geschwindigkeits-einstellung Aktueller, über den Befehl zur Geschwindigkeits-einstellung definierter Wert der niedrigere Wert
High-Power-Status	Aktueller, über den Befehl zur Geschwindigkeitseinstellung definierter Wert

Falls Sie im AUTO-Modus eine höhere Geschwindigkeit für Bewegungsbefehle in einem Programm einstellen wollen, fügen Sie den Befehl POWER HIGH oder LP OFF in das Programm ein.

Wenn sich das System im Low Power-Modus befindet und der Manipulatorarm von Hand oder durch ein Operation zur Abwärtsbewegung bewegt wird, kann u.U. Fehler 173 aufgrund der gedrosselten Motorkraft auftreten.

Bei Eingabe eines Befehls zur Geschwindigkeitssteuerung oder zur Programmausführung im Low Power-Modus wird folgende Meldung angezeigt. Sie besagt, daß sich das Robotersystem im Low Power-Status befindet und mit niedriger Geschwindigkeit arbeitet.

```
Low Power State : xxxxx ist begrenzt auf xxx
```

VERWANDTE BEFEHLE LP, SPEED, ACCEL, SPEEEDS, ACCELS

BEISPIEL

```
>SPEED 50                          Definiert ein höhere Geschwindig-
                                   keit
 Low Power State : SPEED ist begrenzt auf 5
>ACCEL 100,100
 Low Power State : ACCEL ist begrenzt auf 10
>JUMP P1                           Bewegung bei gedrosselter
                                   Geschwindigkeit
>SPEED; ACCEL                      Zur Anzeige des aktuellen
                                   Geschwindigkeitsstatus
 Low Power State : SPEED is begrent auf 10
       50
       50          50

 Low Power State : ACCEL ist begrenzt auf 10
      100        100
      100        100
      100        100
>XQT
>LP OFF                            Stellt den High Power-Modus ein
                                   (=POWER HIGH)
>JUMP P2                           Bewegung mit hoher Geschwindigkeit
```

PRGNO

Program No.

FUNKTION

Legt fest, ob die Auswahl der Programmnummer am Remote-Anschluß (REMOTE2) in zyklischer Reihenfolge (auf- bzw. absteigend) oder durch Eingabe eines Binärcodes erfolgen soll.

FORMAT

PRGNO | 0 |
| 1 |

0: zyklisch (auf- bzw. absteigend)
1: Binärcode
Der Standardwert ist 1.

BESCHREIBUNG

Ist die Robotersteuerung über den Remote-Anschluß (REMOTE3) mit einer Ablaufsteuerung verbunden, können Sie verschiedene Funktionen der Robotersteuerung zur Remote-Steuerung nutzen, indem spezielle Signalsequenzen über die Ablaufsteuerung an den Remote-Anschluß (REMOTE3) gesendet werden. Eine dieser Funktionen ist die Auswahl von Programmnummern.

Information zur Einstellung von REMOTE3 erhalten Sie in der Dokumentation der Robotersteuerung und der des SPEL-Editors.

Die Auswahl der Programmnummer erfolgt entweder zyklisch (auf- bzw. absteigend) oder durch Binärcodes.

Soll die Auswahl zyklisch erfolgen, geben Sie den Parameter 0 an. In diesem Fall stehen Ihnen die Programmnummern 00 bis 64 zur Verfügung. Bei der Auswahl der Programmnummern werden Eingang 2^0 und 2^1 von REMOTE3 zur Eingabe der zyklischen Signale verwendet. Dabei werden die Signale wie folgt zugeordnet:

Pin-Nummer	Signal	Eingang	Funktion
6	Auswahl der Programmnummer für Eingang 2^0	Eingang 4	Aufsteigend
7	Auswahl der Programmnummer für Eingang 2^1	Eingang 5	Absteigend

Soll die Auswahl über Binärcodes erfolgen, geben Sie den Parameter 1 an. In diesem Fall stehen Ihnen die Programmnummern 00 bis 15 zur Verfügung. Bei der Auswahl der Programmnummern werden Eingang 2^0 bis 2^3 von REMOTE3 zur Eingabe der binär-codierten Signale verwendet. Dabei werden die Signale wie in der folgenden Tabelle dargestellt, zugeordnet.

Die Einstellung mit dem Befehl PRGNO bleibt auch über das Ausschalten hinaus wirksam. Bei Ausführung des Befehls VERINIT wird die Einstellung auf den Standardwert 1 (Binärcodes) zurückgesetzt.

Programm-nummer	Auswahl der Programmnummer für Eingang			
	2^0	2^1	2^2	2^3
00	0	0	0	0
01	0	0	0	1
02	0	0	1	0
03	0	0	1	1
04	0	1	0	0
05	0	1	0	1
06	0	1	1	0
07	0	1	1	1
08	1	0	0	0
09	1	0	0	1
10	1	0	1	0
11	1	0	1	1
12	1	1	0	0
13	1	1	0	1
14	1	1	1	0
15	1	1	1	1

VERWANDTE BEFEHLE DSW( )

PRGSIZE

Program Size (Programmgröße)

FUNKTION Definiert bzw. zeigt die Größe des Programmbereichs an.

FORMAT `PRGSIZE {[Bereichsgröße]}`

BESCHREIBUNG Mit Hilfe des Befehls PRGSIZE können Sie die Größe des Programmbereichs im Hauptspeicher auf die angegebene Größe (Bereichsgröße) einstellen.

Wird keine Bereichsgröße angegeben, wird mit dem Befehl PRGSIZE die aktuelle Größe des Programmbereichs angezeigt.

Nach der Eingabe des Befehls PRGSIZE wird die folgende Eingabeaufforderung angezeigt:

```
>Program, Point, Backup Variable, Object all clear -> OK?
```

Zur Ausführung des Befehls geben Sie Y (= Yes) bzw. y ein.

Zum Abbruch des Befehls geben Sie N (= No) bzw. n ein.

Durch die Ausführung des Befehls PRGSIZE werden sowohl der Programmbereich als auch der Bereich für die Punktdaten sowie der Variablen- und der Objektbereich im Hauptspeicher gelöscht. Daher sollten Sie folgendes berücksichtigen:

Erstellen Sie **vor der Änderung** eine Sicherungskopie der Programm- und Positionsdatendateien im Hauptspeicher.

Geben Sie **nach der Änderung** die Backup-Variablen erneut ein.

Da die Größe des Hauptspeichers in der Regel nicht veränderlich ist, wird bei einer Vergrößerung des Programmbereichs automatisch die Größe des Objektbereichs entsprechend reduziert. Aus diesem Grund sollte die Vergrößerung des Programmbereichs auf ein Minimales beschränkt werden.

VERWANDTE BEFEHLE PNTSIZE, LIBSIZE, FREE, SYS

BEISPIEL

```
>PRGSIZE &H10000
>PRGSIZE
 65536
>
```

Setzt den Programmbereich auf 64 KB (&H: hexadezimales Präfix)

Die folgende Abbildung erläutert die Zuordnung des Hauptspeichers (Mapping) unter SPEL III. Die Befehle zur Änderung der einzelnen Bereiche werden auf der rechten Seite aufgeführt.

Bereich	Standardwert		anzuwendender Befehl
	*1	*2	
Quellprogramm	128 k	64 k	↑ PRGSIZE ↓
Positionspunkte	200 Punktdaten		↑ PNTSIZE ↓
Backup-Variablen	10 Variablen, 0,5 k		↑ LIBSIZE ↓
Objektprogramm	240 k	104 k	↑ Abhängig von den o.g. Einstellungen ↓
Gesamt	ca. 374 k	ca. 174 k	

*1 Diese Standardwerte gelten für den Fall, daß die Steuerung über eine optionale RAM-Erweiterung verfügt und die Größe des Hauptspeichers bzw. der RAM-Disk über den DIP-Schalter DSW2 auf der Hauptplatine eingestellt ist.

*2 Diese Werte gelten für alle übrigen Fälle.

PRINT

> S

FUNKTION Zeigt Daten auf dem Anzeigegerät der Programmiereinheit an.

FORMAT

```
PRINT |[numerische Variable]  |{,|[numerische Variable]    |}n
      |[Zeichenkettenvariable]|  |[Zeichenkettenvariable]  |
      |"[Zeichenkette]"       |  |"[Zeichenkette]"         |
      |[Funktion]             |  |[Funktion]               |
```

BESCHREIBUNG Zeigt die angegebenen Daten auf dem Anzeigegerät der Programmiereinheit an.

VERWANDTE BEFEHLE INPUT

BEISPIEL

```
>PRINT A                Zeigt den Wert der Variablen A an
>PRINT "TEST"           Zeigt den Schriftzug "TEST" an
>PRINT "B=",B           Zeigt in einer Zeile "B=" sowie den Wert der
                        Variablen B an
 B= 9
>
```

PRINT

> S

FUNKTION Gibt Daten an eine der Kommunikationsschnittstellen aus.

FORMAT

```
PRINT #[Portnr],|[numerischer Wert]       |{,|[numerischer Wert]     |}n
                |[numerische Variable]    |  |[numerische Variable]  |
                |"[Zeichenkette]"         |  |"[Zeichenkette]"       |
                |[Zeichenkettenvarariable]|  |[Zeichenkettenvariable]|
```

Die Portnummer muß eine ganze Zahl von 20 bis 24 sein.

BESCHREIBUNG Gibt numerische Werte und Zeichenketten an die durch die Portnummer definierte Kommunikationsschnittstelle aus.

Portnummer und Schnittstelle sind wie folgt zugeordnet:

Portnummer	Schnittstelle
#20, #21	Standardmäßige RS-232C-Schnittstelle
#22, #23	Zusätzliche RS-232C-Schnittstelle
#24	REMOTE1 (OPU, Bediengerät)

VERWANDTE BEFEHLE INPUT #, CONFIG

BEISPIEL

```
>PRINT #20,5          Sendet den numerischen Wert 5 an Port 20
>PRINT #20,A          Sendet den Inhalt der Variablen A an Port 20
>PRINT #21,"PORT"     Sendet die Zeichenkette "PORT" an Port 21
>PRINT #21,A,5        Sendet den Inhalt der Variablen A sowie den
                      numerischen Wert 5 an Port 21
>PRINT #20,NAME$      Sendet den Inhalt der Zeichenkettenvariablen
                      NAME$ an Port 20
>
```

PULSE

> S

FUNKTION

Führt eine gleichzeitige PTP-Bewegung aller vier Achsen entsprechend der festgelegten Pulse-Werte aus bzw. zeigt die aktuellen Positionspulsewerte an.

FORMAT

(1)
```
PULSE [Pulsewert 1. Achse],[Pulsewert 2. Achse],~
      [Pulsewert 3. Achse],[Pulsewert 4. Achse]~
     {[!parallel zu verarbeitende Anweisung!]}
```

Der Pulsebereich für die erste bis vierte Achse ist der durch den Befehl RANGE definierte Bereich (ganze Zahl).

(2) PULSE

BESCHREIBUNG

(1) Führt eine gleichzeitige PTP-Bewegung aller vier Achsen entsprechend der festgelegten Pulse-Werte in nicht entsprechend festgelegten orthogonalen (rechtwinkligen) Koordinatenwerten aus.

(2) Zeigt die aktuellen Positionspulsewerte für alle vier Achsen wie folgt an:

[Pulsewert 1. Achse]	[Pulsewert 2. Achse]
[Pulsewert 3. Achse]	[Pulsewert 4. Achse]

Informationen zum 0-Pulsewert sowie zur +/- Ausrichtung einer Achse erhalten Sie in der Dokumentation zum jeweiligen Manipulatorarm. Diese Werte sind abhängig vom verwendeten Modell.

Der Befehl PULSE sollte in erster Linie bei der Wartung des Roboters eingesetzt werden.

Anders als beim Befehl JUMP werden durch den PULSE-Befehl alle vier Achsen gleichzeitig bewegt, einschließlich der Z-Achse, die in der Bewegung zur Zielposition angehoben bzw. wieder abgesenkt wird. Daher sollten Sie bei Verwendung des PULSE-Befehls besonders darauf achten, daß sich im Verfahrweg keine Hindernisse befinden und sich die Roboterhand frei bewegen kann.

VERWANDTE BEFEHLE

SPEED, ACCEL, ! ... !, RANGE

BEISPIEL

```
>PULSE 16000,10000,-100,10
```

Bewegt die 1., 2., 3. bzw. 4. Achse an die Positionen, die den Werten 16000, 10000, -100 bzw. 10 entsprechen

```
>PULSE
16000   10000
 -100    -100
>
```

Zeigt die Pulsewerte an, die den aktuellen Positionen der 1. bis 4. Achse entsprechen

QP

> S

Quick Pause (Schnellpause)

FUNKTION

Schaltet den Schnellpause-Modus ein bzw. aus oder zeigt den aktuell eingeschalteten Modus an.

FORMAT

```
QP {|ON |}
   |OFF|
```

Standardmäßig ist der Modus eingeschaltet (ON).

BESCHREIBUNG

Dieser Befehl steuert das Verhalten des Roboters, wenn während der Abarbeitung eines Befehls zur Armbewegung entweder der Taster PAUSE gedrückt wird oder ein Pause-Signal am REMOTE3-Anschluß der Roboter-Steuerung eingeht. Abhängig von der gewählten Einstellung unterbricht der Roboter sofort die Armwegung (Schnellpause-Modus ist eingeschaltet) oder unterbricht erst, nachdem der Bewegungsbefehl ausgeführt wurde (Schnellpause-Modus ist ausgeschaltet).

Als Schnellpause bezeichnet man das unmittelbare Abbremsen und Anhalten des Manipulatorarms.

Bei der Einstellung QP ON wird der Modus für die Schnellpause eingeschaltet; bei der Einstellung QP OFF wird der Modus ausgeschaltet. Wird der Befehl ohne einen Parameter eingegeben, wird der aktuelle Status des Schnellpause-Modus angezeigt.

Bei einem Softare-Reset bleibt der Status des Schnellpause-Modus unverändert.

Bei Aus- und Wiedereinschalten des Roboters wird der Modus standardmäßig eingeschaltet.

Wird die Schutzabschrankung geöffnet, unterbricht der Roboter die Armbewegung sofort, auch wenn der Schnell-Pause ausgeschaltet ist (QP OFF).

Im TEACH-Modus unterbricht der Roboter die Tätigkeit auch, wenn Sie die Taste ESC an der Programmiereinheit drücken; jedoch erst nach Beendigung der Bewegung und nicht unmittelbar, unabhägig davon, ob der Schnellpause-Modus ein- oder ausgeschaltet ist.

BEISPIEL

```
>QP ON      Schaltet den Schnellpause-Modus ein
.
.
.
>QP
QP ON       Zeigt den Schnellpause-Modus an.
            In diesem Fall ist er eingeschaltet.
>_
```

QUIT

> S

FUNKTION Beendet laufende Tasks bzw. kurzfristig unterbrochene Tasks.

FORMAT `QUIT ![Tasknummer]`

Die Tasknummer muß eine ganze Zahl zwischen 1 und 16 sein.

BESCHREIBUNG Beendet laufende Tasks bzw. kurzfristig unterbrochene Tasks.

Als Bestandteil einer Task kann der QUIT-Befehl zwar andere Tasks beenden, jedoch nicht die Task, in der er eingebunden ist. Zur Beendigung der Task, in der der QUIT-Befehl eingebunden ist, verwenden Sie den Befehl END.

VERWANDTE BEFEHLE RUN, XQT, HALT, RESUME

BEISPIEL

```
10 FUNCTION MAIN
20 XQT !2 BLINK1          Startet die Task "BLINK1" als 2. Task
30 XQT !3 BLINK2          Startet die Task "BLINK2" als 3. Task
40 WAIT 10
50 QUIT !2;QUIT !3        Beendet Task 2 und Task 3
60 FEND
70'
80 FUNCTION BLINK1        Diese Task blinkt 10 Sekunden lang in
                          Intervallen von 0,2 Sekunden, bis Zeile 50
                          diese Task wieder stoppt.
90 LOOP1:
100 ON 1;WAIT 0.2
110 OFF 1;WAIT 0.2
120 GOTO LOOP1
130 FEND
140 FUNCTION BLINK2       Diese Task blinkt 10 Sekunden lang in
                          Intervallen von 0,5 Sekunden, bis Zeile 50
                          diese Task wieder stoppt.
150 LOOP2:
160 ON 2;WAIT 0.5
170 OFF 2;WAIT 0.5
180 GOTO LOOP2
190 FEND
```

RANGE

> S

FUNKTION

Definiert den zulässigen Verfahrbereich jeder Achse in Pulsen oder zeigt die derzeit zulässigen Bereiche an.

FORMAT

(1)
```
RANGE [unterer Pulsewert 1. Achse],[oberer Pulsewert 1. Achse]~
      [unterer Pulsewert 2. Achse],[oberer Pulsewert 2. Achse]~
      [unterer Pulsewert 3. Achse],[oberer Pulsewert 3. Achse]~
      [unterer Pulsewert 4. Achse],[oberer Pulsewert 4. Achse]
```

(2) RANGE

BESCHREIBUNG

(1) Definiert den zulässigen Verfahrbereich jeder Achse durch Festlegung einer unteren und oberen Grenze in Pulsen.

Der Wert für den unteren Bereich darf den für den oberen Bereich nicht übersteigen, da ansonsten ein Fehler erfolgt und eine Bewegung des Manipulatorarms nicht ausgeführt werden kann.

Der durch den RANGE-Befehl definierte Verfahrbereich bleibt über das Ausschalten des Roboters hinaus gültig, ebenso bei Ausführung des VERINIT-Befehls.

(2) Zeigt die aktuellen Verfahrbereiche wie folgt an:

[Unterer Bereich der 1. Achse]	[Oberer Bereich der 1. Achse]
[Unterer Bereich der 2. Achse]	[Oberer Bereich der 2. Achse]
[Unterer Bereich der 3. Achse]	[Oberer Bereich der 3. Achse]
[Unterer Bereich der 4. Achse]	[Oberer Bereich der 4. Achse]

Informationen über die maximal zulässigen Verfahrbereiche entnehmen Sie bitte der Dokumentation zum jeweiligen Manipulatorarm, da diese abhängig vom verwendeten Modell sind.

VERWANDTE BEFEHLE

XYLIM, VER

BEISPIEL

```
>RANGE 0,32000,0,32224,-10000,0,-40000,40000
>RANGE
     0   32000          Der Verfahrbereich der 1. Achse liegt
                        zwischen 0 und 32000 Pulsen
     0   32224          Der Verfahrbereich der 2. Achse liegt
                        zwischen 0 und 32224 Pulsen
-10000       0          Der Verfahrbereich der 3. Achse liegt
                        zwischen -10000 und 0 Pulsen
-40000   40000          Der Verfahrbereich der 4. Achse liegt
                        zwischen -40000 und 40000 Pulsen
```

RENAME, REN, NAME

Re-name (Namen ändern)

FUNKTION Ändert den Namen einer Datei, d.h., benennt eine Datei um.

FORMAT

```
RENAME {[Laufwerk]}{[Pfad]}[aktueller Dateiname] [neuer Dateiname]
```

Der Dateiname muß mit der Dateinamenerweiterung angegeben werden.

BESCHREIBUNG Ändert den Namen der angegebenen Datei [aktueller Dateiname] in den angegebenen Dateinamen [neuer Dateiname] um.

Wird keine Laufwerksbezeichnung angegeben, sucht die Funktion RENAME die angegebene Datei auf dem aktuellen Laufwerk. Bei Auslassung der Pfadangabe sucht die Funktion im aktuellen Verzeichnis.

Eine Angabe des Laufwerks und /oder des Pfads ist für den neuen Dateinamen nicht möglich. Das bedeutet, die umbenannte Datei befindet sich im selben Pfad wie vorher die ursprüngliche Datei.

Die Angabe des aktuellen Dateinamens sowie des neuen Dateinamens ist zwingend erforderlich.

Existiert im angegebenen Pfad bereits eine Datei mit dem neu zu vergebenden Dateinamen wird der Befehl nicht ausgeführt.

Bei Verwendung des Befehls RENAME dürfen auch Platzhalter eingesetzt werden. Wenn Sie Platzhalter im aktuellen Dateinamen verwenden, werden bei Ausführung des Befehls RENAME alle entsprechenden Dateien umbenannt. Wollen Sie beispielsweise alle Dateien mit dem Namen A nach B umbenennen, ohne jedoch die jeweilige Dateinamenerweiterung zu ändern, geben Sie folgenden Befehl ein:

```
RENAME A.* B.*
```

Wenn Sie Platzhalter im neuen Dateinamen verwenden, werden die Zeichen aus dem bestehenden Dateinamen im neuen Dateinamen an derselben Position wieder übernommen. Wollen Sie beispielsweise alle Dateien mit dem Namen TEST.* nach TEXT.* umbenennen, geben Sie folgenden Befehl ein:

```
RENAME TEST.* ??X?.*
```

Wenn Sie die Datei mit dem neuen Namen auf einem anderen Laufwerk und/oder in einem anderen Verzeichnis erstellen wollen, verwenden Sie den Befehl COPY.

VERWANDTE BEFEHLE COPY

BEISPIEL

```
RENAME A*.* B*.*
RENAME B:TEST.* ??X?.*
RENAME B:A.PRG B.PRG
```

RENDIR

Re-name directory (Verzeichnis umbenennen)

FUNKTION Ändert einen Verzeichnisnamen.

FORMAT `RENDIR {[Laufwerk]}{[Pfad]}[aktueller Verzeichnisname] [neuer Name]`

BESCHREIBUNG Ändert den Namen des angegebenen Verzeichnisses [aktueller Verzeichnisname] in den angegebenen Verzeichnisnamen [neuer Verzeichnisname] um.

Wird keine Laufwerksbezeichnung angegeben, sucht die Funktion RENDIR das angegebene Verzeichnis auf dem aktuellen Laufwerk. Bei Auslassung der Pfadangabe sucht die Funktion das Verzeichnis im aktuellen Verzeichnis.

Eine Angabe des Laufwerks und /oder des Pfads ist für den neuen Verzeichnisnamen nicht erlaubt. Das bedeutet, das umbenannte Verzeichnis befindet sich im selben Pfad wie vorher das ursprüngliche Verzeichnis.

Die Angabe des aktuellen Verzeichnisnamens sowie des neuen Verzeichnisnamens ist zwingend erforderlich.

Existiert im angegebenen Pfad bereits ein Verzeichnis mit dem neu zu vergebenden Verzeichnisnamen wird der Befehl nicht ausgeführt.

Die Verwendung von Platzhaltern ist nicht erlaubt.

VERWANDTE BEFEHLE DIR

BEISPIEL

```
RENDIR B:\BAK USR2
```

RENUM

Re-number (Neu numerieren)

FUNKTION Numeriert die Programmzeilen neu.

FORMAT `RENUM {[erste Zeilennummer]}{,[Steigerungswert]}`

Die folgende Gleichung ist zu beachten: (erste Zeilennummer)+(Gesamtzahl der Programmzeilen)x(Steigerungswert) ≤ 32767

BESCHREIBUNG Führt eine Neunumerierung der Programmzeilen durch. Dabei wird die angegebene erste Zeilennummer der ersten Programmzeile zugeordnet, alle weiteren Programmzeilen werden entsprechend dem angegebenen Steigerungswert neu numeriert.

Wird die erste Zeilennummer nicht angegeben, wird die erste Programmzeile mit der Zeilennummer 10 numeriert. Wird kein Steigerungswert angegeben, wird der Wert automatisch auf 10 gesetzt.

Wie bereits in der Gleichung dargestellt, müssen die Werte so gewählt werden, daß die erste Zeilennummer plus der Gesamtzahl der Programmzeilen multipliziert mit dem Steigerungswert die Zahl 32767 nicht übersteigt.

Der Befehl RENUM numeriert auch Verweise auf Zeilen innerhalb der Befehle GOTO, GOSUB und ONERR.

VERWANDTE BEFEHLE LIST

BEISPIEL

```
10 HOME
20 JUMP P10
30 JUMP P20
40 GOTO 20
>RENUM 100,20
```

Numeriert die erste Zeile mit 100, alle weiteren mit einem Steigerungswert von 20

```
>
>LIST
100 HOME
120 JUMP P10
140 JUMP P20
160 GOTO 120
>_

>RENUM
```

Numeriert die erste Zeile mit 10, alle weiteren mit einem Steigerungswert von 10

```
>
```

RESET

FUNKTION Setzt die Robotersteuerung zurück.

FORMAT RESET

BESCHREIBUNG Mit dem Befehl RESET wird folgendes zurückgesetzt:

Status des Notaus (emergency stop)
Fehlerstatus (mit Ausnahme bestimmter Fehler)
Ausgänge (alle Ausgänge aus)
SPEED, SPEEDS (auf Standardwert zurückgesetzt)
ACCEL, ACCLES (auf Standardwert zurückgesetzt)
LIMZ-Parameter (auf 0 zurückgesetzt)
FINE (auf Standardwert zurückgesetzt)
LP OFF (Low-Power-Modus eingeschaltet, LP ON)

Die o.g. Komponenten und Werte (z.B. Servo-Fehler, Status des Notaus) können nur mit Hilfe des RESET-Befehls zurückgesetzt werden. Führen Sie in einem solchen Fall zuerst den Befehl RESET und anschließend alle im weiteren erforderlichen Schritte aus.

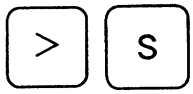
Beispiel: Nach einem Notaus müssen Sie zuerst sicherstellen, daß eine Bedienung des Roboters ohne Gefahr möglich ist. Führen Sie dann den Befehl RESET aus und anschließend den Befehl MOTOR ON.

VERWANDTE BEFEHLE ON, OFF, SPEED, ACCEL, LIMZ, SPEEDS, ACCELS, MOTOR ON

BEISPIEL

```
>RESET
```

RESUME



FUNKTION Setzt die Ausführung einer durch HALT kurzfristig unterbrochenen Task fort.

FORMAT `RESUME ![Tasknummer]`

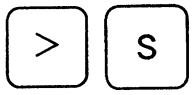
Die Tasknummer muß eine ganze Zahl von 1 bis 16 sein.

VERWANDTE BEFEHLE RUN, XQT, HALT, QUIT

BEISPIEL

```
10 FUNCTION MAIN
20 XQT  !2 BLINK          Startet die Task "BLINK" als 2. Task
30 LOOP:
40 WAIT 3                 Task 1 wartet 3 Sekunden (Timerfunktion)
                          (Task 2 läuft noch)
50 HALT !2                Unterbricht Task 2 (nach Zeile 40 sind
                          3 Sekunden abgelaufen)
60 WAIT 3                 Task 1 wartet 3 Sekunden (Timerfunktion)
                          (Task 2 wurde unterbrochen)
70 RESUME !2              Setzt Task 2 fort (nach Zeile 60 sind
                          3 Sekunden abgelaufen)
80 GOTO LOOP
90 FEND
100 '                     ab hier beginnt Task 2
110 FUNCTION BLINK        Anzeige blinkt (an/aus) in Intervallen von
                          0,2 Sekunden. Das Blinken wiederum wird in
                          Intervallen von je 3 Sekunden ein- und wieder
                          ausgeschaltet.
                          In diesem Beispiel ist nicht eindeutig bestimmt,
                          ob die Anzeige bei Ausführung der Zeile 50
                          ein- oder ausgeschaltet ist, da HALT !2 die
                          Task BLINK an jeder Stelle zwischen
                          Zeile 120 und Zeile 170 anhalten kann.
120 LOOP1:
130 ON 1
140 WAIT 0.2
150 OFF 1
160 WAIT 0.2
170 GOTO LOOP1
180 FEND
```

REVERSE



FUNKTION Schaltet den Modus für die invertierte Textanzeige für den angegebenen Bereich ein.

FORMAT `REVERSE [x-Spalte],[y-Zeile],[Zeichenanzahl]`

Für den Wert x muß eine ganze Zahl von 1 bis 32, für den Wert y eine ganze Zahl von 1 bis 8 und für die Zeichenanzahl eine ganze Zahl von 1 bis 32 angegeben werden.

BESCHREIBUNG Schaltet den Modus für die invertierte Textanzeige für den angegebenen Bereich, ausgehend von der (x, y) Position, ein.

Die Zeichenausgabe auf der Bedieneinheit erfolgt als dunkle Schrift auf hellem Hintergrund. Mit REVERSE wird der Anzeigemodus für den angegebenen Bereich verändert (die Schriftzeichen werden hell auf dunklem Untergrund dargestellt).

Ist der angegebene Bereich länger als eine Zeile, verschiebt sich dieser zusätzliche Bereich nicht in die nächste Zeile, sondern kehrt an den Zeilenanfang zurück und zeigt die Zeichen in invertierter Form an.
Werden Zeichen in dem definierten Bereich ausgegeben, werden auch diese zeichen invertiert dargestellt.

VERWANDTE BEFEHLE NORMAL, CHARSIZE, CLS, CURSOR, OPUNIT, OPU PRINT

BEISPIEL

```
>REVERSE 10,2,3
```

RIGHT$( )

F

FUNKTION Gibt die äußerst rechts gelegenen Zeichen einer angegebenen Zeichenkette aus.

FORMAT

```
RIGHT$(|[Name d. Zeichenkettenvariablen]|,[Zeichenanzahl])
       |"[Zeichenkette]"                 |
```

BESCHREIBUNG Gibt die angegebene Anzahl der äußerst rechts gelegenen Zeichen einer angegebenen Zeichenkette aus. Die Zeichenkette kann durch eine Zeichenkettenvariable angegeben werden.

VERWANDTE BEFEHLE LEFT$( ), MID$( )

BEISPIEL

```
A$="EPSON"
>PRINT RIGHT$ ("ABCDE",2)
DE
>
>PRINT RIGHT$ (A$,4)
PSON
```

RMDIR, RD

Remove directory (Verzeichnis löschen)

FUNKTION Entfernt (löscht) ein leeres Unterverzeichnis.

FORMAT `RMDIR {[Laufwerk]}{[Pfad]}[Verzeichnisname]`

BESCHREIBUNG Entfernt (löscht) das angegebene Unterverzeichnis. Bevor Sie den Befehl RMDIR ausführen, müssen Sie alle Dateien aus dem Unterverzeichnis löschen.

Bei Auslassung der Laufwerksbezeichnung durchsucht die Funktion das aktuelle Laufwerk nach dem Unterverzeichnis; bei Auslassung des Pfadnamens sucht die Funktion im aktuellen Verzeichnis.

Wird kein Verzeichnisname angegeben, erfolgt ein Fehler.

Das aktuelle Verzeichnis (HOME-Verzeichnis) bzw. das übergeordnete Verzeichnis (PARENT-Verzeichnis) kann nicht gelöscht werden.

BEISPIEL

```
RD \USR2
```

ROPEN...CLOSE

S

Read Open (Öffnen zum Lesen)

FUNKTION Öffnet eine Datei, um deren Inhalt einzulesen.

FORMAT

```
ROPEN "[Dateiname]" AS #[Dateinummer]
     .
     .
     .
CLOSE
```

Der Dateiname muß mit der Dateinamenerweiterung angegeben werden. Die Dateinummer muß eine ganze Zahl von 30 bis 35 sein.

BESCHREIBUNG Öffnet die Datei mit dem angegebenen Dateinamen zum Lesen und kennzeichnet sie anhand der angegebenen Dateinummer. Mit Hilfe dieser Anweisung wird die gewünschte Datei geöffnet und die Daten werden eingelesen. Über den Befehl CLOSE wird die Datei wieder geschlossen und die kennzeichnende Dateinummer wieder freigegeben.

Eine Datei mit dem angegebenen Dateinamen muß bereits auf dem Laufwerk existieren.

Die Dateinummer kennzeichnet die Datei solange sie geöffnet ist, sie wird für die Eingabeanweisung zum Einlesen (INPUT #) und für den Befehl zum Schließen (CLOSE #) benötigt. Das bedeutet, solange die aktuelle Datei geöffnet ist, kann die Dateinummer nicht zur Kennzeichnung einer anderen Datei verwendet werden.

Es können maximal sechs Dateien gleichzeitig geöffnet sein. Sind jedoch die maximal erlaubten sechs Dateien gleichzeitig geöffnet, können die Befehle DLOAD und DMERGE nicht ausgeführt werden.

VERWANDTE BEFEHLE INPUT #, AOPEN, WOPEN

BEISPIEL

```
100 REAL DATA(200)
110 WOPEN "TEST.VAL" AS #30
120 FOR I=0 TO 100
130 PRINT #30, DATA(I)
140 NEXT
150 CLOSE #30
160 '
170 REOPEN "TEST.VAL" AS #30
180 FOR I=0 TO 100
190 INPUT #30, DATA(I)
200 NEXT
210 CLOSE #30
220 '
```

RSHIFT( )

F

Right Shift (Verschieben nach rechts)

FUNKTION Verschiebt die numerischen Daten (als Binärwert) nach rechts.

FORMAT `RSHIFT([numerische Daten],[Anzahl zu verschiebender Bits]`

BESCHREIBUNG Verschiebt die angegebenen numerischen Daten um die angegebene Anzahl von Bits nach rechts (in Richtung des wertniedrigsten Bits). An jede durch die Verschiebung freigewordene Stelle wird links eine 0 eingesetzt.

VERWANDTE BEFEHLE LSHIFT( ), NOT( )

BEISPIEL

```
>PRINT RSHIFT(4,2)
 1
>I=10
>PRINT RSHIFT(I,1)
 5
>
```

4 entspricht binär 0100, 2 bedeutet verschieben um 2 Bit

ergibt 0001, dies entspricht 1

R

RUN

FUNKTION Kompiliert ein Quellprogramm und führt es aus.

FORMAT

```
RUN {"{[Laufwerk]}{[Pfad]}[Dateiname]"}
```

Die Angabe einer Dateinamenerweiterung ist nicht erlaubt.

BESCHREIBUNG Kompiliert die angegebene Datei und führt sie aus.

Die Ausführung des Befehls RUN entspricht einer aufeinanderfolgenden Ausführung der Befehle COMPILE und XQT.

Werden keine Angaben zu Laufwerksbezeichnung, Pfadname und Dateiname gemacht, wird das Quellprogramm im Hauptspeicher kompiliert und ausgeführt.

Wenn ein Dateiname angegeben ist, wird die angegebene Datei entweder im Dateispeicher oder auf der Festplatte kompiliert und ausgeführt. Während des Kompilierungsvorgangs werden die folgenden zwei Dateien entweder im Dateispeicher oder auf der Festplatte angelegt:

Dateiname.OBJ
Dateiname.SYM

Unmittelbar vor der Programmausführung werden die folgenden Dateien in den Hauptspeicher geladen:

Dateiname.OBJ
Dateiname.SYM
Dateiname.PNT

Wenn diese Dateien in den Hauptspeicher geladen werden, werden die noch im Hauptspeicher befindlichen Positionsdaten überschrieben. Falls Sie diese Positionsdaten später noch benötigen, sollten Sie sie sichern, bevor Sie den Befehl RUN ausführen.

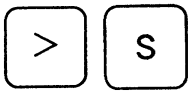
Wird kein Laufwerk angegeben, sucht die Funktion RUN auf dem aktuellen Laufwerk; wird kein Pfadname angegeben, sucht die Funktion im aktuellen Verzeichnis.

VERWANDTE BEFEHLE COMPILE, XQT, HALT, RESUME, QUIT

BEISPIEL

```
>RUN "B:TEST"
>
```

SEL



Select (Auswählen)

FUNKTION Wählt die Einstellungen für die TEACH-IN-Bewegungen und zeigt diese an.

FORMAT `SEL [Speicherplatznummer]`

Die Speicherplatznummer muß eine ganze Zahl von 0 bis 3 sein. Standardmäßig ist der Wert 0 eingestellt.

BESCHREIBUNG Wählt den Speicherplatz aus, in dem die jeweiligen Einstellungen für die TEACH-IN-Bewegungen der einzelnen Achsen gespeichert sind. Diese Einstellungen werden bei jedem Drücken einer TEACH-IN-Taste zur Bedienung des Roboters im Teach-Modus verwendet.

Die Einstellungen der TEACH-IN-Bewegungen für jeden Speicherplatz werden in der folgenden Form angezeigt:

[1. Wert] [2. Wert]
[3. Wert] [4. Wert]

Die auf den jeweiligen Speicherplätzen gespeicherten Einstellungen können über den Befehl SET geändert werden.

Die folgende Tabelle erläutert die für jede der vier Einstellungen gültigen TEACH-IN-Tasten, die Richtung und Achse der Armbewegung sowie die Einheit der TEACH-IN-Bewegung.

Die Zeichen + und - repräsentieren die Bewegungsrichtung.

TEACH-IN-Einstellungen für den Basis- und Werkzeugmodus

	1. Wert		2. Wert		3. Wert		4. Wert	
TEACH-IN-Taste	←	→	↑	↓	Z↑	Z↓	↶	↺
Basis	X-Achse		Y-Achse		Z-Achse		U-Achse	
Werkzeug	x-Achse		y-Achse		z-Achse		u-Achse	
Richtung	-	+	+	-	+	-	+	-
Einheit	mm						° (Grad)	

Die X-, Y-, Z- und U-Werte für den Basismodus entsprechen den X-, Y-, Z- und U-Werten im Roboter-Koordinatensystem.

Die x-, y-, z- und u-Werte für den Werkzeugmodus entsprechen den x-, y-, z- und u-Werten im Werkzeug-Koordinatensystem.

TEACH-IN-Einstellungen für den Joint-Modus

	1. Wert		2. Wert		3. Wert		4. Wert	
TEACH-IN-Taste	←	→	↑	↓	Z↑	Z↓	↶	↺
Achse	1. Achse		2. Achse		3. Achse		4. Achse	
Richtung	-	+	+	-	+	-	+	-
Einheit	mm für lineare Achsenbewegungen, ° (Grad) für Achsendrehungen						° (Grad)	

Weitere Informationen zu den jeweiligen Koordinaten-Modi für TEACH-IN-Bewegungen erhalten Sie im Benutzerhandbuch.

VERWANDTE BEFEHLE

SET

BEISPIEL

```
*SEL 1
     0.1  1
    10    5
*_
```

SELECT...SEND

S

Select ... Select End

FUNKTION

Definiert eine Verzweigungsformel sowie die entsprechenden Anweisungssequenzen zum Verzweigen.

FORMAT

```
SELECT [Formel]
  CASE [Feld];[Anweisung]
    .
    .
    .
  CASE [Feld];[Anweisung]
  {DEFAULT};[Anweisung]}
SEND
```

Es sind bis zu 20 Verschachtelungen erlaubt.

BESCHREIBUNG

Entspricht eines der CASE-Felder dem Ergebnis der SELECT-Formel, wird die entsprechende CASE-Anweisung ausgeführt. Danach wird die Programmsteuerung mit dem Befehl fortgesetzt, der dem Befehl SEND folgt.

Entspricht keines der CASE-Felder dem Ergebnis der SELECT-Formel, wird die mit DEFAULT definierte Anweisung ausgeführt und die Programmsteuerung wird mit dem Befehl fortgesetzt, der dem Befehl SEND folgt.

Wird die DEFAULT-Anweisung ausgelassen, erfolgt keine Ausführung und die Programmsteuerung wird mit dem Befehl fortgesetzt, der dem Befehl SEND folgt.

Die SELECT-Formel kann sowohl Konstanten als auch Variablen, Variablenformeln und logische Operatoren enthalten, die die Anweisungen AND, OR bzw. XOR verwenden.

Ein CASE-Feld kann sowohl Konstanten als auch Variablen enthalten.

Eine auf CASE folgende Anweisung kann Mehrfachanweisungen bzw. mehrzeilige Anweisungen enthalten.

BEISPIEL

```
110 FUNCTION MAIN
120 INTEGER I
130 FOR I=0 TO 10
140  SELECT I
150    CASE 0;OFF 1;ON 2;JUMP P1
160    CASE 3;ON 1;OFF 2
170             JUMP P2;MOVE P3;ON 3
180    CASE 7;ON 4
190    DEFAULT;ON 7
200  SEND
210 NEXT
220 FEND
```

SELRB

> S

Select Robot

FUNKTION Wählt ein Positionierungsgerät aus.

FORMAT `SELRB [Gerätenummer]`

Die Gerätenummer muß eine ganze Zahl von 0 bis 3 sein. Der Standardwert ist 1.

BESCHREIBUNG Dieser Befehl wird in einer Funktion (FUNCTION...FEND) verwendet, über die ein Positionierungsgerät, wie z.B. ein Manipulator oder ein RAIOC gesteuert wird und gibt an, welches Gerät die Funktion steuert.

Die jeweilige Gerätenummer entspricht der Adresse des entsprechenden Geräts.

1	Manipulatorarm
2	RAIOC an Adresse #2
3	RAIOC an Adresse #3

In der Haupttask und in Task 1 ist die Gerätenummer standardmäßig auf den Wert 1 eingestellt ("SELRB 1"), d.h., wenn über die Task 1 nur ein Manipulator gesteuert wird, ist eine SELRB-Anweisung nicht erforderlich. Werden jedoch mehrere Manipulatorarme gesteuert, müssen die jeweiligen Positionierungsgeräte mit Hilfe einer bzw. mehrerer SELRB-Anweisungen ausgewählt werden.

Wenn in einer Funktion ein Positionierungsgerät über eine SELRB-Anweisung ausgewählt wurde, gelten alle anschließenden Anweisungen zur Positionssteuerung für dieses Gerät, und zwar solange, bis eine andere SELRB-Anweisung folgt.

BEISPIEL

```
10 FUNCTION MAIN
20 XQT !2 ROBOT
30 FEND
40 '
50 FUNCTION ROBOT
60 SELRB 1
70 JUMP P1
   .
   .
   .
500 FEND
```

SENSE

> S

FUNKTION Definiert und zeigt die Bedingung an, die zu einem Stop einer mit SENSE gekennzeichneten JUMP-Bewegung über dem Zielpunkt führt.

FORMAT (1)
```
SENSE [Bedingung]
  .
  .
  .
JUMP [Positionsangabe] SENSE
```

Die folgenden Funktionen und Operatoren können innerhalb einer Bedingung verwendet werden:

Funktionen:	SW, IN (die Angabe eines Ausgangs oder eines Merkers ist möglich)
Operatoren:	AND, OR, XOR, +, *
Sonstige:	Runde Klammern zur Festlegung von Prioritäten für Operationen sowie Variablen

(2)
```
SENSE
```

BESCHREIBUNG (1) SENSE-Befehl:
Definiert die SENSE-Bedingung.
Die SENSE-Bedingung muß mindestens eine Ausgangs- oder Merkerfunktion beinhalten. Verwenden Sie innerhalb einer SENSE-Bedingung nur die zuvor erwähnten Operatoren. (Wird ein anderer Operator verwendet, erfolgt daraufhin kein Fehler, sondern eine unkontrollierte Armbewegung.

Bei Verwendung von Variablen werden die jeweiligen Werte während der Ausführung des SENSE-Befehls berechnet.

Die mehrfache Verwendung des SENSE-Befehls ist erlaubt, wobei die jeweils letzte SENSE-Angabe die aktuelle bleibt, bis sie außer Kraft gesetzt wird.

Beim Einschalten des Roboters ist die SENSE-Bedingung wie folgt:

SENSE SW(0)=1

Verwenden Sie die Funktion JS(0) bzw. STAT(1), um zu überprüfen, ob die SENSE-Bedingung erfüllt ist.

JUMP-Befehl mit SENSE-Bedingung:
Das System überprüft, ob die aktuelle SENSE-Bedingung erfüllt ist. Wenn die Bedingung erfüllt ist, stoppt der Roboter über der Zielposition.

(2) Zeigt die aktuelle SENSE-Bedingung an.
Da das Format zur Anzeige der SENSE-Bedingung geändert wird, um die Reihenfolge der Operationen klar darzustellen, kann die Anzeige vom ursprünglichen Eingabeformat abweichen.

VERWANDTE BEFEHLE JUMP, SW( ), STAT(1), JS(0)

BEISPIEL

```
1000 SENSE SW(1)=0
```
Definiert die SENSE-Bedingung

```
1010 JUMP P1 SENSE
```
Stoppt über der Zielposition, wenn die aktuelle SENSE-Bedingung (Zeile 1000) erfüllt ist

```
1020 SENSE SW(1)=1 AND SW($1)=1
```
Definiert eine andere SENSE-Bedingung

```
1030 JUMP P2 SENSE
```
Stoppt über der Zielposition, wenn die aktuelle SENSE-Bedingung (Zeile 1020) erfüllt ist

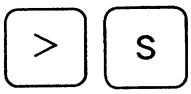
```
1040 FOR I=1 TO 2
1050   SENSE SW(I)=1
```
(Iteration 1) Definiert die SENSE-Bedingung (abhängig von Eingang 1)
(Iteration 2) Definiert die SENSE-Bedingung (abhängig von Eingang 2)

```
1060   JUMP PI SENSE
1070 NEXT
1080 I=5
1090 JUMP P4 SENSE
```
Stoppt über der Zielposition, wenn die aktuelle SENSE-Bedingung (Zeile 1050, Iteration 2) erfüllt ist

```
1110 JUMP P6 SENSE
```
Stoppt über der Zielposition, wenn die aktuelle SENSE-Bedingung (Zeile 1050, Iteration 2) erfüllt ist

```
>SENSE SW(1)=1 AND SW($1)=1
>SENSE
.      SW(1)=1 AND SW($1)=1
>
>SENSE SW(0) OR SW(1) AND SW($1)
>SENSE
.     (SW(0) OR SW(1)) AND SW($1)
>
```

SET



FUNKTION Definiert die Werte für die TEACH-IN-Bewegungen.

FORMAT `SET [Speicherplatznr],[1.Wert],[2. Wert],[3. Wert],[4. Wert]`

Die Speicherplatznummer muß eine ganze Zahl von 0 bis 3 sein. Der Wert für die TEACH-IN-Bewegung kann in Einheiten von 0,001 festgelegt werden.

BESCHREIBUNG Definiert die TEACH-IN-Bewegungen der einzelnen Achsen. Diese Bewegungen werden bei jedem Drücken einer TEACH-IN-Taste zur Bedienung des Roboters im Teach-Modus ausgeführt. Die Einstellungen für diese Bewegungen werden auf sogenannten Speicherplätzen gespeichert. Bei Definition einer TEACH-IN-Bewegung muß die Nummer des jeweiligen Speicherplatzes angegeben werden.

Um eine TEACH-IN-Bewegung einer Achse in die entgegengesetzte Richtung ausführen zu lassen, verwenden Sie ein Minus-Zeichen (-).

Die folgende Tabelle erläutert die für jede der vier Einstellungen gültigen TEACH-IN-Tasten sowie die Standardeinstellung für jeden Speicherplatz.

	1. Wert		2. Wert		3. Wert		4. Wert	
TEACH-IN-Taste	←	→	↑	↓	Z↑	Z↓	↶	↷
SEL 0	0,03		0,03		0,03		0,03	
SEL 1	0,1		0,1		0,1		0,1	
SEL 2	1		1		1		1	
SEL 3	10		10		10		10	

Um die Einstellungen des gewünschten Speicherplatz auszuwählen, geben Sie die Speicherplatznummer mit Hilfe des Befehls SEL an.

Informationen über die Richtung und Achse der Armbewegung beim Drücken einer TEACH-IN-Taste sowie über die Einheit der Bewegung erhalten Sie in der Beschreibung des SEL-Befehls.

Die mit SET definierten Werte bleiben auch über das Ausschalten des Roboters hinaus gültig. Bei Ausführung des VERINIT-Befehls werden sie auf die in der Tabelle aufgeführten Standardwerte zurückgesetzt.

VERWANDTE BEFEHLE SEL

BEISPIEL

```
>SET 1,10,5,3,3
```
Definiert die Werte für die TEACH-IN-Bewegung auf Speicherplatz 1

```
>SET 2,0.1,0.5,1,0
```
Definiert die Werte für die TEACH-IN-Bewegung auf Speicherplatz 2

SETENV

Set Environment

FUNKTION Definiert, löscht bzw. zeigt die Umgebungsvariable an.

FORMAT

```
SETENV {|COM={-V}{-L}                                          |}
       |PLI={/W}                                               |
       |XQT={{[Laufwerk]}[Pfad]{;{[Laufwerk]}[Pfad]}n}          |
       |PATH={{[Laufwerk]}[Pfad]{;{[Laufwerk]}[Pfad]}n}|
```

BESCHREIBUNG Definiert bzw. löscht Umgebungsvariablen und zeigt außerdem die eingestellten Werte an. In dem zuvor angegebenen Format werden die Einträge links vom Gleichheitszeichen (=) als Name der Umgebungsvariablen bezeichnet; der Inhalt der jeweiligen Variablen wird durch die Zeichenkette rechts vom Gleichheitszeichen festgelegt.

Die Umgebungsvariablen sind gültig bei der Ausführung der Befehle COMPILE und PLIST sowie bei der Ausführung einer Datei. Wenn Sie eine Umgebungsvariable definiert haben, können verschiedene Einstellungen, die normalerweise bei jeder Ausführung der genannten Befehle vorgenommen werden müssen, weggelassen werden.

Wenn eine Stapeldatei ausgeführt wird, muß der Pfad in der Umgebungsvariablen angegeben werden.

Wird nur der Befehl SETENV eingegeben, wird der Inhalt der aktuell gültigen Umgebungsvariablen angezeigt. Beim Einschalten des Roboters ist keine Umgebungsvariable definiert.

SETENV COM={-V}{-L}
Definiert die Kompilierungsbedingung. Bei Angabe des Parameters -V und/oder -L in der Umgebungsvariablen wird der Befehl COMPILE unter der angegebenen Bedingung ausgeführt, sobald Sie COMPILE eingeben. Wird nur "COM=" ohne weitere Parameterangabe eingegeben, wird die Definition der Umgebungsvariablen gelöscht. Da der Name der Umgebungsvariablen COM lautet, ist es nicht möglich, sie mit COMPILE zu beschreiben.
SETENV PLI={/W}
Definiert das Format für den Befehl PLIST. Wird der Parameter /W angegeben, wird der Befehl PLIST/W ausgeführt, wenn Sie PLIST eingeben. Wird nur "PLI=" ohne weitere Parameterangabe eingegeben, wird die Definition der Umgebungsvariablen gelöscht. Da der Name der Umgebungsvariablen PLI lautet, ist es nicht möglich, sie mit PLIST zu beschreiben.

SETENV XQT={{[Laufwerk]}[Pfad]{;{[Laufwerk]}[Pfad]}n}
Definiert den Pfad für die Ausführung einer Datei. Wenn Sie XQT "[Dateiname]" eingeben, sucht das System die entsprechende Datei in der Reihenfolge der angegebenen Pfade und führt sie anschließend aus. Sind in der Umgebungsvariablen keine Pfade definiert, sucht das System die Datei nur im aktuellen Verzeichnis auf dem aktuellen Laufwerk. Wird nur "XQT=" eingegeben, wird die Definition der Umgebungsvariablen gelöscht. **Hinweis:** Wenn der mit dem Befehl XQT "Dateiname" angegebene Pfad zu einer Datei von der Pfadangabe in der Umgebungsvariablen abweicht, wird die Angabe in der Umgebungsvariablen ignoriert.
SETENV PATH={{[Laufwerk]}[Pfad]{;{[Laufwerk]}[Pfad]}n}
Definiert den Pfad zur Ausführung einer Stapeldatei. Ist die Umgebungsvariable PATH definiert, sucht das System im angegebenen Pfad nach der angegebenen Datei und führt die Datei aus, wenn ihr Dateiname eingegeben wird. Wird nur "PATH=" eingegeben, wird die Definition der Umgebungsvariablen gelöscht. Der Pfad kann entweder über den Befehl SETENV oder über den Befehl PATH definiert werden. Ist PATH nicht definiert, kann die Stapeldatei nicht ausgeführt werden.

VERWANDTE BEFEHLE PATH, COMPILE, PLIST, XQT

BEISPIEL

```
>SETENV PLI=/W
>SETENV XQT=A:\;B:\
>SETENV
PLI=/W
XQT=A:\;B:\
>
>PLIST                    Verwendet das Format PLIST/W zur Anzeige
P000=  450.000,  400.000,    0.000,   0.000
P001=    1.325,  330.170   -58.200,   8.000  /L
P011= -200.000,  400.000, -100.000, 150.000/1/R
P013= -450.000,  200.000,    0.000,   0.000/2
P014=  450.000, -200.000,    0.000.   0.000/2
>
```

S

SETVER

Set VERsion

FUNKTION Lädt die verschiedenen Parameterdaten, die mit dem Befehl MKVER in einer Datei gespeichert wurden zurück in den entsprechenden Speicherbereich.

FORMAT SETVER

BESCHREIBUNG Dieser Befehl führt die Stapeldatei "MK#0.BAT" im aktuellen Verzeichnis auf dem aktuellen Laufwerk aus.

Diese Stapeldatei wird mit dem Befehl MKVER erstellt und enthält die Einstellungen, die mit den folgenden Befehlen gemacht wurden. Beachten Sie jedoch, daß der Inhalt der Datei davon abhängt, ob der Befehl MKVER mit oder ohne den Parameter "/A" eingegeben und ausgeführt wurde. Nähere Einzelheiten entnehmen Sie bitte den Erläuterungen zum Befehl MKVER.

CALPLS	HOFS	MCORDR	MCOFS
SEL	SET	ARCH	
TSPEED	TSPEEDS	HOMESET	HORDR
ARM	ARMSET	TOOL	TLSET
RANGE	XYLIM	BASE 0	
CONFIG	CONSOLE	SELDISK	WIDTH
PRGNO	OPUNIT	WEIGHT	MAXDEV
PRGSIZE	PNTSIZE	LIBSIZE	
Einstellungen über Software-Schalter		Einstellungen an REMOTE3	

PRG#0.PRG PNT#0.PNT LIB#0.BIN SYM#0.BIN OBJ#0.BIN	Befehle zum Laden dieser Dateien

Bei Ausführung des Befehls SETVER werden verschiedene Parameterdaten erneut definiert und die Backup-Variablen werden in den Hauptspeicher geladen.

Sind in der Stapeldatei auch Befehle zum Laden anderer Dateien beschrieben, werden auch diese Dateien in den Hauptspeicher geladen.

VERWANDTE BEFEHLE MKVER, VER

BEISPIEL Dieser Befehl ist beispielsweise sehr nützlich, wenn die MPU- oder SPU-Platine in der Robotersteuerung ausgetauscht werden soll. In diesem Fall können diese Daten dann problemlos auf die neue Platine geladen werden.

Sichern Sie die Daten, die sich auf der auszutauschenden Platine befinden in einer Datei.

```
>MKVER
```

Die folgenden Dateien werden angelegt:

MK#0.BAT
LIB#0.BIN
`CPU Data Backup`
`Backup Variable Backup`

Erstellen Sie eine Sicherungskopie von allen Dateien, die sich im Dateispeicher befinden, einschließlich der o.g. Dateien. Speichern Sie diese Sicherungskopien auf der Festplatte der Programmiereinheit. Verwenden Sie dazu die Taste **BACKUP**.

Bauen Sie die neue Platine in die Robotersteuerung ein.

Laden Sie alle Daten, die sich auf der Festplatte der Programmiereinheit befinden, in den Dateispeicher der Robotersteuerung. Drücken Sie dazu die Taste **RESTOR**.

Führen Sie den Befehl SETVER aus.

```
>SETVER
```

SFREE

> S

Servo Free

FUNKTION Schaltet die Motoren frei.

FORMAT `SFREE {[Achsennummer]{,[Achsennummer]}3}`

BESCHREIBUNG Schaltet die Motoren frei. Dieser Zustand wird "Servo free" genannt. Dieser Befehl wird bei der manuellen Positionierung (Teaching) oder in Situationen verwendet, in denen die Stromzufuhr zu einer bestimmten Achse unterbrochen werden muß.

Wird keine Achsennummer angegeben, werden alle vier Achsen freigeschaltet; bei Angabe einer oder mehrerer Achsennummern wird nur jeweils die angegebene Achse freigeschaltet.

Da bei der dritten Achse die elektrische Bremse aktiviert wird, kann diese Achse nicht von Hand bewegt werden, selbst wenn sie im Befehl SFREE definiert wurde. Eine manuelle Positionierung der dritten Achse ist nur möglich, wenn Sie den Schalter zum Lösen der Bremse für die dritte Achse während der Positionierung gedrückt halten.

Bei Ausführung des SFREE-Befehls werden die Werte der folgenden Befehle zurückgesetzt:

SPEED	Standardwert des jeweiligen Modells
ACCEL	Standardwert des jeweiligen Modells
LIMZ	0

Um eine Achse wieder einzuschalten, führen Sie einen der Befehle SLOCK oder MOTOR ON aus.

Erhält der Roboter im SFREE-Zustand einen Befehl zur Ausführung einer Bewegung, erfolgt eine Fehlermeldung. Um eine solche Fehlermeldung bei Ausführung eines Bewegungsbefehls zu vermeiden, schalten Sie Bit 5 des Software-Schalters SS6 ein.

VERWANDTE BEFEHLE SLOCK, MOTOR

BEISPIEL

Beispiel 1

```
SFREE

>SLOCK
>SFREE 1,2
>SLOCK 1,2
```

Beispiel 2

```
100 JUMP P1:Z0

110 SWITCH 1,&H10
120 SFREE 1,2
130 GO P1
140 SLOCK 1,2
```

Führt eine Montage eines Bauteils durch. Bei der Vertikal-Bewegung (GO P1) sind Achse 1 und Achse 2 freigeschaltet.

SGN( )

F

Sign

FUNKTION Gibt das Vorzeichen des angegebenen numerischen Wertes aus.

FORMAT `SGN([numerischer Wert])`

BESCHREIBUNG Gibt das Vorzeichen des angegebenen numerischen Wertes aus.

Ist der numerische Wert positiv, wird folgendes ausgegeben:	1
Ist der numerische Wert 0, wird folgendes ausgegeben:	0
Ist der numerische Wert negativ, wird folgendes ausgegeben:	-1

VERWANDTE BEFEHLE ABS( ), INT( ), SQR( )

BEISPIEL

```
>PRINT SGN(0.55)
1
>PRINT SGN(-0.55)
-1
>
```

S

SIN ( )

F

Sine (Sinus)

FUNKTION Gibt den Sinuswert des angegebenen Winkels aus.

FORMAT `SIN([Radiantenwert])`

BESCHREIBUNG Bei Angabe eines Radiantenwertes wird der Sinuswert des angegebenen Winkels ausgegeben. Winkelangaben in Grad müssen mit Hilfe der folgenden Gleichung in Radiantenwerte (rad) umgewandelt werden.

rad = Grad * π/180 (π = 3,141593)

VERWANDTE BEFEHLE COS( ), TAN( ), ATAN( ), ATAN2( )

BEISPIEL

```
>PRINT SIN(0.55)
.5226872
```
Zeigt einen Sinuswert von 0,55 rad an

```
>PRINT SIN(30*3.141953/180)
.5
```
Zeigt einen Sinuswert von 30 Grad an

```
>A=30*3.141953/180

>PRINT SIN(A)
.5
>
```
Zeigt mit Hilfe einer Variablen einen Sinuswert von 30 Grad an

SLOCK

> S

Servo Lock

FUNKTION Schaltet eine Achse wieder ein (aus einem "Servo-free"-Zustand).

FORMAT `SLOCK {[Achsennummer]{,[Achsennummer]}3}`

Die Achsennummer muß eine ganze Zahl von 1 bis 4 sein.

BESCHREIBUNG Schaltet eine Achse, die zuvor durch den Befehl SFREE freigeschaltet worden war, wieder ein.

Wird keine Achsennummer angegeben, werden alle vier Achsen wieder eingeschaltet.

Bei Angabe einer oder mehrerer Achsennummern werden die jeweiligen Achsen wieder eingeschaltet.

Bei Ausführung des SLOCK-Befehls werden die Werte der folgenden Befehle zurückgesetzt:

SPEED	Standardwert des jeweiligen Modells
ACCEL	Standardwert des jeweiligen Modells
LIMZ	0

Wenn die Z-Achse wieder eingeschaltet wird, löst sich automatisch die elektrische Bremse.

Um alle Achsen wieder einzuschalten, können Sie anstelle des Befehls SLOCK auch den Befehl MOTOR ON verwenden.

Wird der Befehl SLOCK ausgeführt obwohl die Stromzufuhr zu den Motoren unterbrochen ist (durch den Befehl MOTOR OFF), erfolgt eine Fehlermeldung.

VERWANDTE BEFEHLE SFREE, MOTOR

BEISPIEL

Beispiel
```
>SFREE
>SLOCK
>SFREE 1,2
>SLOCK 1,2
```

Beispiel
```
100 JUMP P1:Z0
110 SWITCH 1,&H10
120 SFREE 1,2
130 GO P1
140 SLOCK 1,2
```

Führt eine Montage eines Bauteils durch. Bei der Vertikal-Bewegung (GO P1) sind Achse 1 und Achse 2 freigeschaltet

SPACE$( )

F

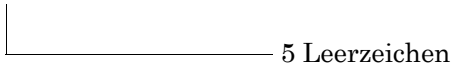
FUNKTION Gibt eine Zeichenkette mit der angegebenen Anzahl von Leerzeichen aus.

FORMAT `SPACE$([Anzahl])`

BESCHREIBUNG Gibt eine Zeichenkette mit der angegebenen Anzahl von Leerzeichen aus.

BEISPIEL

```
>PRINT "AB"+SPACE$(5)+"CD"
AB_ _ _ _ _CD
```



SPEED

> S

FUNKTION Definiert und zeigt die Geschwindigkeit für eine PTP-Bewegung an.

FORMAT (1) `SPEED [Geschwindigkeit]{,[Geschwindigkeit Z-Achsen-Aufwärtsbewegung],~`
`[Geschwindigkeit Z-Achsen-Abwärtsbewegung]}`

Für jeden Geschwindigkeitswert muß eine ganze Zahl von 1 bis 100 (%) angegeben werden. Informationen zum Standardwert erhalten Sie in der jeweiligen Dokumentation zum Manipulatorarm.

(2) SPEED

BESCHREIBUNG (1) Definiert die Geschwindigkeit, mit der alle PTP-Bewegungen (definiert durch die Befehle GO, JUMP, PULSE usw.) ausgeführt werden. Für die Angabe der Geschwindigkeit sind nur ganze Zahlen von 1 bis 100 erlaubt, wobei dieser Wert die prozentuale Geschwindigkeit ausgehend von der Maximalgeschwindigkeit wiedergibt. Die definierten Geschwindigkeitswerte für die Auf- bzw. Abwärtsbewegung der Z-Achse gelten nur für den JUMP-Befehl. Wurden keine Werte definiert, wird für beide Geschwindigkeiten automatisch der Standardwert eingesetzt.

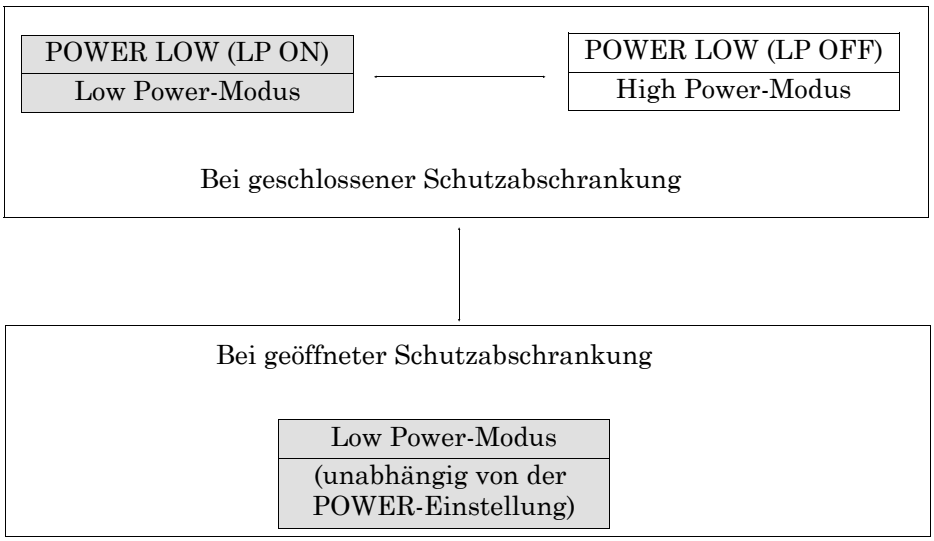
(2) Zeigt die aktuell eingestellten SPEED-Werte wie folgt an:
[Geschwindigkeit]
[Aufwärtsbewegung der Z-Achse] [Abwärtsbewegung der Z-Achse]

Bei Ausführung der folgenden Befehle bzw. Aktionen wird der SPEED-Wert automatisch auf den Standardwert zurückgesetzt:

Einschalten der Stromzufuhr
Modus-Umschaltung (TEACH/AUTO)
Software-Reset
MOTOR ON
SFREE, SLOCK
VERINIT
Drücken der Taste **STOP**
Drücken der Tastenkombination **Strg** + **C**

Wenn sich der Roboter im TEACH-Modus befindet, unterscheidet sich die tatsächliche Bewegungsgeschwindigkeit im Low Power-Modus von der im High Power-Modus.

Motor-Power-Status	Tatsächliche Bewegungsgeschwindigkeit
Low-Power-Status	Standardwert des Befehls SPEED SPEED-Wert } der niedrigere Wert
High-Power-Status	SPEED-Wert



Wird eine höhere Geschwindigkeit benötigt, schalten Sie den High Power-Modus über den Befehl POWER HIGH bzw. LP OFF ein und schließen Sie die Schutzabschrankung. Ist die Schutzabschrankung geöffnet, werden die Einstellungen für die Geschwindigkeit auf die Standardwerte zurückgesetzt.

Wird der Befehl SPEED ausgeführt während sich der Roboter im Low Power-Modus befindet, wird die folgende Meldung angezeigt. Die in der Meldung angezeigte Zahl gibt den SPEED-Standardwert an, der für die einzelnen Modelle unterschiedlich sein kann. Im folgenden Beispiel wird angezeigt, daß sich der Manipulator nicht mit der durch den SPEED-Befehl definierten Geschwindigkeit (80) bewegt, sondern mit der Standardgeschwindigkeit (5), da der Low-Power-Modus eingeschaltet ist.

```
>SPEED 80
Low Power State : SPEED ist begrenzt auf 5
>
>SPEED
Low Power State : SPEED ist begrenzt auf 5
      80
      80       80

>
```

VERWANDTE BEFEHLE

POWER (LP), TSPEED, ACCEL, GO, JUMP, PASS, PULSE

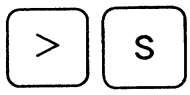
BEISPIEL

```
>SPEED 100,100,50
```

Nur die Geschwindigkeit für die Abwärtsbewegung der Z-Achse ist mit 50 (%) definiert

```
>SPEED
   100
   100    50>_
```

SPEEDS



FUNKTION

Definiert bzw. zeigt die Geschwindigkeit für die CP-Bewegung (Continuous Path) an.

FORMAT

(1) `SPEEDS [Wert für die Geschwindigkeit]`

Für den Geschwindigkeitswert muß eine ganze Zahl von 1 bis 1120 (mm/s) angegeben werden. Informationen zum Standardwert erhalten Sie in der jeweiligen Dokumentation zum Manipulatorarm.

(2) `SPEEDS`

BESCHREIBUNG

(1) Definiert die Geschwindigkeit der Manipulatorhand in mm/s, mit der alle CP-Bewegungen (definiert durch die Befehle ARC, MOVE, CVMOVE usw.) ausgeführt werden.

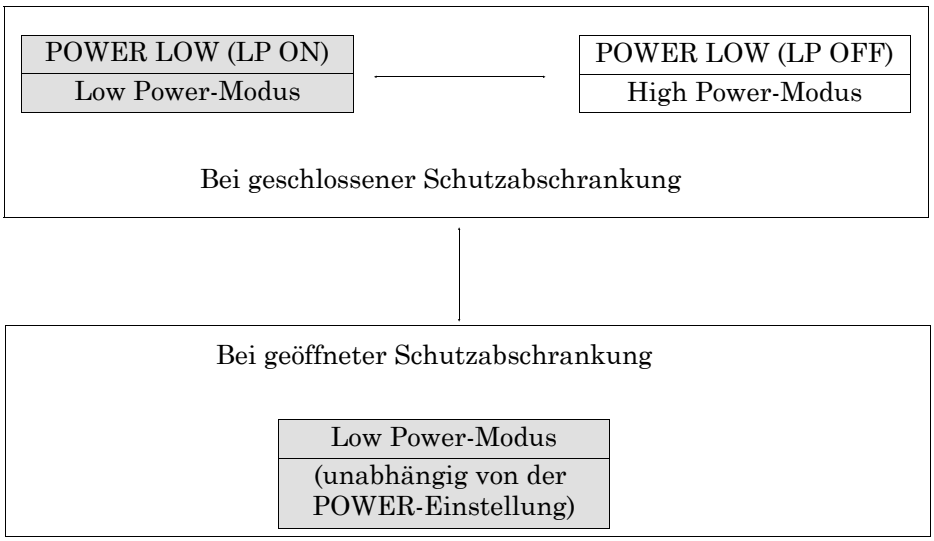
(2) Zeigt den aktuell eingestellten SPEEDS-Wert an:

Bei Ausführung der folgenden Befehle bzw. Aktionen wird der SPEEDS-Wert automatisch auf den Standardwert zurückgesetzt:

Einschalten der Stromzufuhr
Modus-Umschaltung (TEACH/AUTO)
Software-Reset
MOTOR ON
SFREE, SLOCK
VERINIT
Drücken der Taste **STOP**
Drücken der Tastenkombination **Strg** + **C**

Wenn sich der Roboter im TEACH-Modus befindet, unterscheidet sich die tatsächliche Bewegungsgeschwindigkeit im Low-Power-Modus von der im High Power-Modus.

Motor-Power-Status	Tatsächliche Bewegungsgeschwindigkeit
Low-Power-Status	Standardwert des Befehls SPEEDS SPEEDS-Wert } der niedrigere Wert
High-Power-Status	SPEED-Wert



Wird eine höhere Geschwindigkeit benötigt, schalten Sie den High Power-Modus über den Befehl POWER HIGH bzw. LP OFF ein und schließen Sie die Schutzabschrankung. Ist die Schutzabschrankung geöffnet, werden die Einstellungen für die Geschwindigkeit auf die Standardwerte zurückgesetzt.

Wird der Befehl SPEEDS ausgeführt während sich der Roboter im Low-Power-Modus befindet, wird die folgende Meldung angezeigt. Die in der Meldung angezeigte Zahl gibt den SPEEDS-Standardwert an, der für die einzelnen Modelle unterschiedlich sein kann. Im folgenden Beispiel wird angezeigt, daß sich der Manipulator nicht mit der durch den SPEEDS-Befehl definierten Geschwindigkeit (800) bewegt, sondern mit der Standardgeschwindigkeit (50), da der Low-Power-Modus eingeschaltet ist.

```
>SPEEDS 800
Low Power State : SPEEDS ist begrenzt auf 50
>
>SPEEDS
Low Power State : SPEEDS ist begrenzt auf 50
      800

>
```

VERWANDTE BEFEHLE

POWER (LP), TSPEEDS, ACCELS, MOVE, CMOVE, ARC, CARC, CVMOVE

BEISPIEL

```
100 SPEEDS 200        Definiert die Geschwindigkeit (200 mm/s)
110 MOVE P1
120 MOVE P20
130 SPEEDS 100        Definiert die Geschwindigkeit (100 mm/s)
140 MOVE P30
```

SQR( )

F

Square Root

FUNKTION Gibt die Quadratwurzel aus.

FORMAT `SQR([numerischer Wert])`

BESCHREIBUNG Gibt die Quadratwurzel des angegebenen numerischen Wertes aus.

VERWANDTE BEFEHLE ABS( ), SGN( ), INT( )

BEISPIEL

```
>PRINT SQR(2)
 1.414214
>
```

S

STAT( )

F

Status

FUNKTION

Gibt den Status der Robotersteuerung oder eines anderen an die RS-232C-Schnittstelle angeschlossenen Kontrollers aus.

FORMAT

```
STAT([Adresse])
```

Die Adresse für die Robotersteuerung selbst ist 0 oder 1, für einen an die RS-232C-Schnittstelle angeschlossenen Kontroller muß die Adresse eine ganze Zahl (Portnummer) von 20 bis 23 sein.

BESCHREIBUNG

Wird als Adresse der Wert 0 oder 1 angegeben, gibt das System nach dem Befehl STAT( ) eine ganze Zahl von 4 Byte aus, die den Status der Robotersteuerung selbst wiedergibt. Nähere Informationen dazu erhalten Sie in Tabelle 1 auf der folgenden Seite.

Wird als Adresse eine ganze Zahl von 20 bis 23 (Nummer eines der RS-232C-Anschlüsse) angegeben, gibt das System nach dem Befehl STAT( ) eine ganze Zahl von 3 Byte aus, die den Status des an die RS-232C-Schnittstelle angeschlossenen Kontrollers wiedergibt. Nähere Informationen dazu erhalten Sie in Tabelle 2 auf der folgenden Seite.

Die in Tabelle 1 und 2 dargestellten nicht definierten Bits müssen mit dem Operator AND oder einem ähnlichen Operator maskiert werden.

BEISPIEL

```
>PRINT (STAT(0) AND &HOFF)
 6
>PRINT (STAT(20) AND &H030000)/65536
 1

>_
```

Zeigt den Status der Taskausführung für die Robotersteuerung selbst an

Nur die Tasks 2 und 3 werden ausgeführt

Zeigt den Status (Ausführung/Pause/Fehler) des an der RS-232C-Schnittstelle angeschlossenen Kontrollers an

Adresse	Bit	Status der Robotersteuerung, angezeigt durch ein ON-Bit
0	0 bis 15	Task 1 wird ausgeführt (XQT) oder ist unterbrochen (HALT) bis Task 16 wird ausgeführt (XQT) oder ist unterbrochen (HALT)
	16	Task-Ausführung
	17	Pause
	18	Fehler
	19	nicht definiert (Version 4.2) | TEACH-Modus (Version 5.2)
	20	Not-Aus
	21	Low-Power-Modus eingeschaltet (POWER LOW bzw. LP ON)
	22	nicht definiert (Version 4.2) | Sicherheitstür geschlossen (Version 5.2)
	23	nicht definiert (Version 4.2) | Aktivierungsschalter geöffnet (Version 5.2)
	24 bis 31	nicht definiert
1	0	Protokolleintragung, daß der Roboter nach Erfüllung einer Bedingung in einer JUMP...SENSE-Anweisung über der Zielposition angehalten hat. (Diese Eintragung wird gelöscht, sobald eine weitere JUMP-Anweisung ausgeführt wird.)
	1	Protokolleintragung, daß der Roboter nach Erfüllung einer Bedingung in einer GO/JUMP/MOVE...TILL-Anweisung in einer Zwischenposition angehalten hat. (Diese Eintragung wird gelöscht, sobald eine weitere GO/JUMP/MOVE...TILL-Anweisung ausgeführt wird.)
	2	Protokolleintragung, daß ein MCAL-Befehl bzw. eine MCAL-Anweisung ausgeführt wird.
	3	Protokolleintragung, daß der Roboter nach Erfüllung einer Bedingung in einer TRAP-Anweisung in einer Zwischenposition angehalten hat.
	4	Stromzufuhr zu den Motoren eingeschaltet (MOTOR ON)
	5 bis 7	nicht definiert
	8	Motor der 4. Achse läuft
	9	Motor der 3. Achse läuft
	10	Motor der 2. Achse läuft
	11	Motor der 1. Achse läuft
	12 bis 31	nicht definiert

Tabelle 1 Statusanzeige, wenn mit dem Befehl STAT( ) die Adreßnummer 0 oder 1 angegeben wurde.

Bit	Kontrollerstatus, angezeigt durch ein ON-Bit
0 bis 15	Merker 0 ist ein bis Merker 15 ist ein
16	Task-Ausführung
17	Pause
18	Fehler
19 bis 23	nicht definiert

Tabelle 2 Statusanzeige, wenn mit dem Befehl STAT( ) eine RS-232-C-Portnummer angegeben wurde.

STR$( )

F

FUNKTION Gibt den angegebenen numerischen Wert als numerische Zeichenkette aus.

FORMAT `STR$([numerischer Wert])`

BESCHREIBUNG Gibt den angegebenen numerischen Wert als numerische Zeichenkette aus.

Bei positiven numerischen Werten wird bei der Ausgabe der numerischen Zeichenkette eine Leerstelle vor die erste Ziffer und eine weitere Leerstelle hinter der letzten Ziffer eingefügt.

Bei negativen numerischen Werten wird bei der Ausgabe der numerischen Zeichenkette ein Minus-Zeichen (-) vor die erste Ziffer und eine Leerstelle hinter der letzten Ziffer eingefügt.

BEISPIEL

```
>PRINT STR$(5)+""+STR$(6)
_5_ _ _6_                          _ stellt eine Leerstelle dar
>
10 FUNCTION MAIN
20 STRING LETTER$;LETTER$=""
30 INTEGER I
40 FOR I=1 TO 5
50 LETTER$=LETTER$+STR$(I)         Fügt eine numerische Zeichenkette
                                   an LETTER$ an
60 NEXT I
70 PRINT LETTER$
80 FEND
>RUN
COMPILE END
_1_ _2_ _3_ _4_ _5_                _ stellt eine Leerstelle dar
>
```

STRING

FUNKTION Definiert eine Zeichenkettenvariable.

FORMAT `STRING [Name d. Zeichenkettenvariablen]{,[Name d. Zeichenkettenvariablen]}n`

BESCHREIBUNG Definiert eine Zeichenkettenvariable. Das letzte Zeichen der Zeichenkettenvariablen muß ein $-Zeichen sein, um diese Variable von einer numerischen Variablen zu unterscheiden.

Die Zeichenkettenvariable darf maximal 8 Zeichen einschließlich des $-Zeichens sein. In einer Zeichenkettenvariablen können bis zu 79 Zeichen als Daten eingesetzt werden.

Die folgenden Operatoren können in Verbindung mit einer Zeichenkettenvariablen verwendet werden:

+	kombiniert Zeichenkettenvariablen
=	vergleicht Zeichenkettenvariablen; gilt als erfüllt, wenn die beiden Zeichenkettenvariablen identisch sind.
< >	vergleicht Zeichenkettenvariablen; gilt als erfüllt, wenn sich die beiden Zeichenketten in mindestens einem Zeichen unterscheiden.

VERWANDTE BEFEHLE ASC( ), CHR$( ), LEFT$( ), LEN( ), MID$( ), RIGHT$( ), SPACE$( ), STR$( ), VAL( )

BEISPIEL

```
10 FUNCTION LETTER
20 STRING LETTER$
30 INTEGER LETTER
40 LETTER$="Letter Count="
50 LETTER=LEN(LETTER$)
60 PRINT LETTER$,LETTER
70 FEND


RUN
COMPILE END
Letter Count=13


10 FUNCTION LETTER
20 STRING LETTER$
30 INPUT LETTER$
40 IF LETTER$="A" THEN GOSUB JOB1
50 IF LETTER$="B" THEN GOSUB JOB2
60 GOTO 30
70 JOB1:
 .
 .
 .
```

SW( )

F

Switch

FUNKTION Gibt den Status eines Eingangs aus.

FORMAT `SW([Nummer des Eingangs])`

Die Nummer für den Eingang muß eine ganze Zahl von 0 bis 127 sein.

BESCHREIBUNG Gibt den Status des angegebenen Eingangs entweder mit 0 oder mit 1 aus.

0: AUS
1: EIN

VERWANDTE BEFEHLE WAIT, IF...THEN

BEISPIEL

```
80 A=SW(0)
```
Speichert den Status des Eingangs 0 in A

```
90 IF A=0 THEN GOTO 200
100 IF SW(0)=1 AND SW(127)=0 THEN GOTO START
110 WAIT SW(0) OR SW (1) AND SW($1)
```
Identisch mit: WAIT (SW(0) OR SW(1))=1 AND SW($1)=1

```
 .
200 WAIT SW(5)
```
Identisch mit der Angabe WAIT SW(5)=1, wartet bis der Status des Eingangs 5 EIN ist

SW($)

F

Switch$

FUNKTION Gibt den Status eines Merkers aus.

FORMAT `SW($[Bitnummer])`

Die Bitnummer muß eine ganze Zahl von 0 bis 511 sein.

BESCHREIBUNG Gibt den Status des angegebenen Merkers entweder mit 0 oder mit 1 aus.

0: AUS
1: EIN

VERWANDTE BEFEHLE ON $, OFF $, WAIT, IF...THEN

BEISPIEL

```
>A=SW($18)
```

Speichert 1 in A, wenn Merker 18 EIN ist bzw. speichert 0 in A, wenn Merker 18 AUS ist.

S

SYS

S

System

FUNKTION Definiert Backup-Variablen.

FORMAT `SYS [Typendefinition] [Variablenname]{,[Variablenname]}n`

BESCHREIBUNG Definiert die Backup-Variablen und speichert diese im Backup-Variablen-Bereich des Hauptspeichers.

Das Sichern der Backup-Variablen beim Ausschalten der Robotersteuerung erfolgt batteriegepuffert, d.h., die Variablen werden bei einer Stromunterbrechung gesichert. Eine definierte Backup-Variable kann solange verwendet werden, bis der Speicherbereich der Backup-Variablen durch den Befehl CLRLIB oder einen ähnlichen Befehl gelöscht wird.

Sie können die folgenden Variablentypen definierten:

BYTE	1 Byte (-128 bis +127)	Ganze Zahl
INTEGER	2 Bytes (-32768 bis +32767)	
LONG	4 Bytes (-2147483648 bis +2147483647)	
REAL	4 Bytes, 7 Ziffern	Reelle Zahl
DOUBLE	8 Bytes, 14 Ziffern	

Sie können mehrere Variablennamen in einer einzigen SYS-Anweisung zur Typendefinition angeben. In diesem Fall muß jedoch jede Definition eines neuen Variablentyps auf einer neuen Zeile beginnen.

Wenn Sie ein Programm benutzen wollen, in dem bestimmte Variablen als Backup-Variablen angenommen werden, müssen Sie diese Backup-Variablen vor dem Kompilierungsvorgang definieren, ansonsten werden diese Variablen nicht wie Backup-Variablen verarbeitet.

Die Deklarierung von Backup-Variablen erfolgt durch ein einfaches **separates** Programm. Die einmalige Ausführung des Programms genügt. Mehrmaliges Ausführen und die damit verbundene Backup-Variablen-Definition löst eine Fehlermeldung aus.

Bei Ausführung des Befehls CLRLIB werden alle Backup-Variablen gelöscht. Dabei sollten Sie beachten, daß der gesamte Backup-Variablen-Bereich gelöscht wird und Sie keine Möglichkeit haben, nur bestimmte Variablen anzugeben.

VERWANDTE BEFEHLE LIBRARY, CLRLIB, LIBSIZE

BEISPIEL

```
>10 FUNCTION B_UP
>20 SYS INTEGER V(50)
>30 SYS REAL A(10)
>40 FEND
>RUN
>
```

Dieses Programm definiert die Variablen V(50) und A(10) als Backup-Variablen.
Nach RUN (COMPILE + XQT) wird ein spezieller Speicherbereich reserviert.
Das Programm kann dann gelöscht werden.

SYSINIT

System Initialize (System initialisieren)

FUNKTION Initialisiert den Hauptspeicher.

FORMAT SYSINIT

BESCHREIBUNG Initialisiert die Größenzuordnungen im Hauptspeicher und setzt sie auf die Standardwerte gemäß der folgenden Tabelle zurück.

Bereich	Standardeinstellungen	
	Nr. 1	Nr. 2
Quellprogramm	64 k	128 k
Punktdaten	200 Punktdaten	
Backup-Variablen	10 Variablen, 0,5 k	
Objektprogramm	104 K	240 K
Gesamt	184 k	374 k

Standardeinstellungen
- Nr. 2 Gilt, wenn die Robotersteuerung mit einem optionalen RAM-Speicher erweitert wurde und die Größen für den Hauptspeicher und den Dateispeicher durch DIP-Schalter SD2 auf der MPU-Platine eingestellt wurden.
- Nr. 1 Gilt für alle anderen Fälle

Der Befehl SYSINIT wird verwendet, um den Hauptspeicher zu initialisieren, wenn ein Fehler bei der Überprüfung des Hauptspeichers auftritt. Durch diesen Befehl werden alle Daten im Hauptspeicher gelöscht. Daher sollten Sie alle benötigten Programm- und Punktdaten im Hauptspeicher sichern, d.h., ein Backup erstellen, bevor Sie den Befehl SYSINIT ausführen.

Nach Ausführung des Befehls können Sie die Größen der einzelnen Bereiche, falls erforderlich, neu definieren. Die Backup-Variablen müssen dann erneut registriert werden.

VERWANDTE BEFEHLE PRGSIZE PNTSIZE LIBSIZE SYS

TAN( )

F

Tangent (Tangenswert)

FUNKTION Gibt den Tangenswert des angegebenen Winkels aus.

FORMAT `TAN([Radiantenwert])`

BESCHREIBUNG Gibt den Tangenswert des angegebenen Winkels aus, wenn dieser als Radiantenwert definiert wird.

Winkelwerte in der Maßeinheit Grad müssen anhand der folgenden Gleichung in Radiantenwerte umgerechnet werden:

Radiant = Grad * π/180 (π = 3,141593)

VERWANDTE BEFEHLE SIN( ), COS( ), ATAN( ), ATAN2( )

BEISPIEL

```
>PRINT TAN(0.55)                    'Zeigt einen Wert von 0,55 rad. an
.6131052
>PRINT TAN(30*3.141593/180)         Zeigt einen Tangenswert von 30 ° an
.5773503
>A=30*3.141593/180                  Zeigt einen Tangenswert von 30 °
                                    unter Verwendung einer Variablen an
>PRINT TAN(A)
.5773503
>
```

T

TGO

> S

Tool-coordinate Go

FUNKTION Führt eine PTP-Bewegung im gewählten Werkzeug-Koordinatensystem aus.

FORMAT `TGO [Positionsangabe] {TILL} {![Parallelanweisung]!}`

BESCHREIBUNG Führt eine PTP-Bewegung im gewählten Werkzeug-Koordinatensystem aus.

Mit Hilfe einer TILL-Bedingung wird der TGO-Befehl beendet, indem der Manipulator abgebremst wird und an einer Zwischenposition anhält, sobald die aktuelle TILL-Bedingung erfüllt ist.

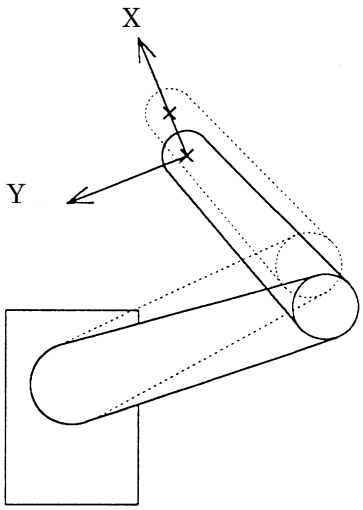
VERWANDTE BEFEHLE P=, ! ... !, TOOL, SPEED, ACCEL, TILL, TMOVE

BEISPIEL

```
>TGO 100,0,0,0
```

Bewegt das derzeit ausgewählte Werkzeug um 100 mm in horizontaler Richtung, also entlang der Werkzeug-Koordinaten-X-Achse

```
>_
TGO P1
>
```



Beispiel eines Werkzeug-Koordinatensystems

TILL

> S

FUNKTION

Definiert bzw. zeigt die Eingabebedingung an, durch die - wenn erfüllt - die aktuell ausgeführte Bewegung (gesteuert durch die Befehle JUMP, GO oder MOVE) beendet wird, indem der Manipulator abgebremst wird und in einer Zwischenposition stoppt.

FORMAT

(1)
```
TILL [Eingangsbedingung(en)]
   .
   .
   .
|JUMP| [Positionsangabe] TILL
|GO  |
|MOVE|
```

Die folgenden Funktionen bzw. Operatoren können in der Eingabebedingung verwendet werden:

Funktionen:	SW, IN (sowohl die Verwendung von I/O-Angaben als auch von Merkern ist erlaubt)
Operatoren:	AND, OR, XOR, +, *
Sonstige:	Runde Klammern zur Festlegung von Prioritäten für Operationen sowie Variablen.

(2)
```
|GO  | [Positionsangabe] TILL SW([Eingang]){=|0|}
|MOVE|                                     |1|
```

(3)
```
TILL
```

BESCHREIBUNG

(1) TILL-Befehl:
Definiert die TILL-Bedingung
Die TILL-Bedingung muß mindestens eine Eingangs- oder Merkerfunktion beinhalten. Verwenden Sie innerhalb einer TILL-Bedingung nur die zuvor erwähnten Operatoren. (Wird ein anderer Operator verwendet, erfolgt daraufhin kein Fehler, sondern eine unkontrollierte Armbewegung.) Bei Verwendung von Variablen werden die jeweiligen Werte während der Ausführung des TILL-Befehls berechnet. Die mehrfache Verwendung des TILL-Befehls ist erlaubt, wobei die jeweils letzte TILL-Angabe die aktuelle bleibt, bis sie außer Kraft gesetzt wird.

JUMP, GO bzw. MOVE mit TILL-Bedingung:
Überprüft, ob die aktuelle TILL-Bedingung erfüllt ist. Wird die Bedingung erfüllt, wird der Bewegungsbefehl beendet, indem der Manipulator abgebremst wird und an einer Zwischenposition stoppt. Während einer Abwärtsbewegung der Z-Achse bei einem JUMP-Befehl wird die Erfüllung der TILL-Bedingung nicht überprüft. Nähere Informationen erhalten Sie in der Beschreibung des JUMP-Befehls.

Beim Einschalten des Roboters ist die TILL-Bedingung wie folgt:

```
TILL SW(0)=1
```
Eingang 0 ist eingeschaltet

Verwenden Sie die Funktion STAT(1), um zu überprüfen, ob die TILL-Bedingung erfüllt ist.

(2) GO bzw. MOVE mit TILL-Bedingung, SW(Eingangs)-Bedingung und Eingangsbedingung (0 bzw. 1):
Überprüft, ob die Eingangsbedingung erfüllt ist. Wenn die Bedingung erfüllt ist, wird der Bewegungsbefehl beendet, indem der Manipulator abgebremst wird und an einer Zwischenposition stoppt.

GO bzw. MOVE mit TILL-Bedingung und SW(Eingangs)-Bedingung, jedoch ohne Eingangsbedingung:
Die Eingangsbedingung wird auf den Standardwert 1 gesetzt. Ist der angegebene Eingang eingeschaltet, wird der Bewegungsbefehl beendet, indem der Manipulator abgebremst wird und an einer Zwischenposition stoppt.

(3) Zeigt die aktuelle TILL-Bedingung an:
Da das Format zur Anzeige der TILL-Bedingung geändert wird, um die Reihenfolge der Operationen klar darzustellen, kann die Anzeige vom ursprünglichen Eingabeformat abweichen.

VERWANDTE BEFEHLE JUMP, GO, MOVE, SW( ), STAT(1)

BEISPIEL

```
1000 TILL SW(1)=0

1010 GO P1 TILL


1020 TILL SW(1)=1 AND SW($1)=1

1030 MOVE P2 TILL

1040 MOVE P3 TILL

1050 FOR I=1 TO 2
1060   TILL SW(I)=1

1070   MOVE PI TILL




1080 NEXT
1090 I=5
1100 GO P4 TILL

1110 MOVE P5 TILL SW(10)=1
1020 MOVE P6 TILL

>TILL SW(1)=1 AND SW($1)=1
>TILL
       SW(1)=1 AND SW($1)=1
>TILL SW(0) OR SW(1) AND SW($1)
>TILL
      (SW(0) OR SW(1)) AND SW($1)
>
```

Definiert die TILL-Bedingung

Stoppt, wenn die aktuelle TILL-Bedingung (Zeile 1000) erfüllt ist

Definiert die TILL-Bedingung

Stoppt, wenn die aktuelle TILL-Bedingung (Zeile 1020) erfüllt ist
Stoppt, wenn die aktuelle TILL-Bedingung (Zeile 1020) erfüllt ist

(Iteration 1) Definiert die TILL-Bedingung
(Iteration 2) Definiert die TILL-Bedingung
(Iteration 1) Stoppt, wenn die aktuelle TILL-Bedingung (Zeile 1060, Iteration 1) erfüllt ist
(Iteration 2) Stoppt, wenn die aktuelle TILL-Bedingung (Zeile 1060, Iteration 2) erfüllt ist

Stoppt, wenn die aktuelle TILL-Bedingung (Zeile 1060, Iteration 2) erfüllt ist
Stoppt, wenn Eingang 10 ein ist
Stoppt, wenn die aktuelle TILL-Bedingung (Zeile 1060, Iteration 2) erfüllt ist

TIME

FUNKTION Stellt die aktuelle Uhrzeit ein bzw. zeigt sie an.

FORMAT

(1) `TIME [Stunde]:[Minute]:[Sekunde]|A|`
`|P|`

(2) `TIME`

BESCHREIBUNG

(1) Stellt die aktuelle Uhrzeit ein. Dieser Wert wird bei internen Protokoll- und Archivierungsvorgängen verwendet.

Bei der Angabe des Formats können Sie eine 12-Stunden-Anzeige festlegen (durch Hinzufügen des Parameters A für die Zeit vor dem Mittag bzw. durch den Parameter P für die Zeit nach dem Mittag) oder eine 24-Stunden-Anzeige. Bei Eingabe der Uhrzeit 11:59:59 oder früher ohne einen Parameter nimmt das System automatisch den Parameter A an.

(2) Zeigt die aktuelle Uhrzeit im 12-Stunden-Format (mit a oder p) an.

VERWANDTE BEFEHLE DATE

BEISPIEL

```
>TIME
The current time is 10:15:32a

>TIME 1:05:00P
>TIME
The current time is 1:05:15p
>
```

T

TIME ( )

F

FUNKTION Gibt die Betriebsstunden der Robotersteuerung aus.

FORMAT `TIME([Auswahl der Anzeigeeinheit])`

Als Anzeigeeinheit kann entweder 0, 1, oder 2 angegeben werden.

BESCHREIBUNG Gibt die gesamten Betriebsstunden der Robotersteuerung als ganze Zahl aus.

0: Stunden
1: Minuten
2: Sekunden

Wenn Sie als Anzeigeeinheit 2, also Sekunden, angeben, besteht die angezeigte Zahl u.U. aus vielen Ziffern. Wenn Sie also den durch TIME ( ) ausgegebenen Wert in einer Variablen speichern wollen, sollten Sie den Variablentyp zuvor als eine ganze Zahl von 4 Byte (LONG) definieren.

VERWANDTE BEFEHLE HOUR

BEISPIEL

```
>LONG A,B,C      Definiert A, B und C als ganze Zahl von 4 Byte
>A=TIME(0)       Speichert die Zeit in Stunden in der Variablen A
>B=TIME(1)       Speichert die Zeit in Minuten in der Variablen B
>C=TIME(2)       Speichert die Zeit in Sekunden in der Variablen C
>PRINT A
100              Betriebszeit der Steuerung beträgt 100 Stunden
>PRINT B
6000             Betriebszeit der Steuerung beträgt 6.000 Minuten
>PRINT C
360000           Betriebszeit der Steuerung beträgt 360.000 Sekunden
>
```

TLSET

> S

Tool Set (Werzeugeinstellung)

FUNKTION Definiert das Werkzeug-Koordinatensystem bzw. zeigt es an.

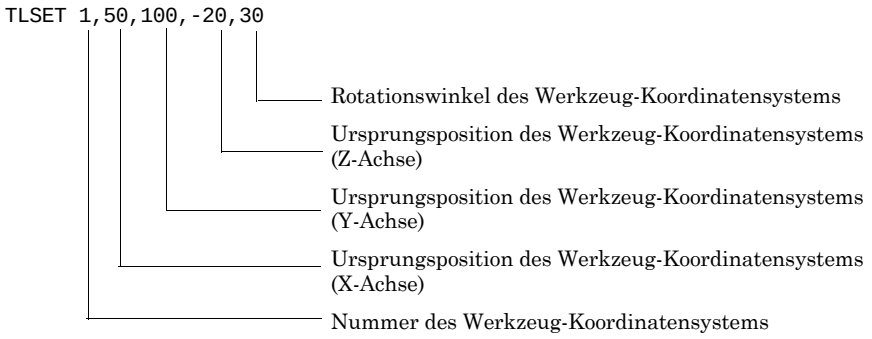
FORMAT (1) TLSET [Nr Werkzeug-Koordinatensystem],[Position]

Die Nummer des Werkzeug-Koordinatensystems muß eine ganze Zahl zwischen 1 und 3 sein.

(2) TLSET

BESCHREIBUNG (1) Definiert die Werkzeug-Koordinatensysteme Tool 1, Tool 2 bzw. Tool 3. Dazu geben Sie den Ursprung des Werkzeug-Koordinatensystems sowie den Rotationswinkel im Verhältnis zum Werkzeug-Koordinatensystem Tool 0 (Hand-Koordinatensystem) an.

Beispiel:




```
TLSET 2,P10+X20
```

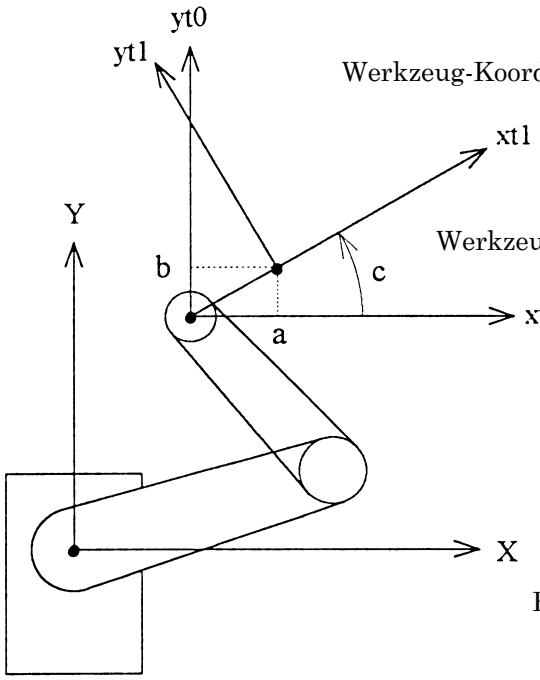
In diesem Fall wurden nur die Koordinatenwerte für die X-U-Achse von P10 definiert; Armattribute sowie die Nummer des lokalen Koordinatensystems wurden ignoriert.

(2) Zeigt alle aktuellen Werkzeug-Koordinatensysteme einschließlich Tool 0 an.

Zur Auswahl eines bestimmten Werkzeug-Koordinatensystems verwenden Sie den Befehl TOOL.

Die TLSET-Werte bleiben über das Ausschalten des Roboters hinaus gültig.

Der Ursprung des Werkzeug-Koordinatensystems Tool 0 ist der Drehpol des Roboters. Bei einer Rotation um diesen Punkt dreht sich das Koordinatensystem Tool 0 und damit auch die Koordinatensysteme von Tool 1, Tool 2 und Tool 3.



Wird die vierte Achse des Roboters mit 0 ° definiert, verläuft das Werkzeug-Koordinatensystem Tool 0 (xt0-yt0) parallel zum Roboter-Koordinatensystem. Dies wird in der obigen Abbildung verdeutlicht. In dieser Situation, d.h., wenn das Werkzeug wie dargestellt auf der vierten Achse installiert ist, ist es sinnvoll das Koordinatensystem Tool 1 zu definieren (xt1-yt1), und zwar mit dem Rotationswinkel c des Koordinatensystems Tool 0.

Bei der Ausführung des Befehls VERINIT werden die Werkzeug-Koordinatensysteme Tool 1, Tool 2 und Tool 3 gelöscht. Das Werkzeug-Koordinatensystem Tool 0 kann jedoch nicht verändert werden.

VERWANDTE BEFEHLE

TOOL, TGO, TMOVE, VERINIT

BEISPIEL

```
>TLSET 1,100,0,0,0
```
Definiert das Werkzeug-Koordinatensystem Tool 1 (plus 100 mm in X-Richtung ausgehend vom Hand-Koordinatensystem)

```
>TOOL 1
```
Wählt das Werkzeug-Koordinatensystem Tool 1 aus

```
>TGO P5
>
```

TMOUT

> S

Time Out (Zeitüberschreitung)

FUNKTION Definiert das Intervall für die Unterbrechung beim WAIT-Befehl.

FORMAT `TMOUT [Intervall]`

Der Wert für das Intervall kann zwischen 0 und 32767 (Sekunden) liegen.

BESCHREIBUNG Definiert die Dauer der Unterbrechung, die bei Ausführung des WAIT-Befehls gemacht wird, bis die entsprechende WAIT-Bedingung erfüllt ist. Die Angabe des Wertes erfolgt in der Einheit Sekunden. Stellt das System nach Ablauf dieses Intervalls fest, daß die WAIT-Bedingung nicht erfüllt ist, erfolgt ein Timeout-Fehler.

Wird das Timeout-Intervall mit 0 definiert, wird die Timeout-Funktion nicht ordnungsgemäß ausgeführt. Dementsprechend wartet das System bei Ausführung des WAIT-Befehls unbegrenzt lange, bis die WAIT-Bedingung erfüllt ist.

VERWANDTE BEFEHLE SW( ), SW($), IN( ), IN($)

BEISPIEL

```
10 TMOUT 60
 .
 .
 .
100 WAIT SW(0)=1
```

Das Timeout-Intervall ist auf 60 Sekunden festgelegt

Wird SW(0) nicht innerhalb von 60 Sekunden gleich ein, erfolgt eine Fehlermeldung; diese kann mit ONERR verarbeitet werden

T

TMOVE

> S

Tool-coordinate Move (Bewegung im Werkzeug-Koordinatensystem)

FUNKTION Führt eine linearinterpolierte Bewegung innerhalb des ausgewählten Werkzeug-Koordinatensystems aus.

FORMAT `TMOVE [Positionsangabe] {TILL} {![Parallelanweisung]!}`

BESCHREIBUNG Führt eine linearinterpolierte Bewegung innerhalb des ausgewählten Werkzeug-Koordinatensystems aus.

Mit Hilfe einer TILL-Bedingung kann die TMOVE-Bewegung so gesteuert werden, daß der Manipulatorarm abgebremst und an einer Zwischenposition angehalten wird, wenn die aktuelle TILL-Bedingung erfüllt ist.

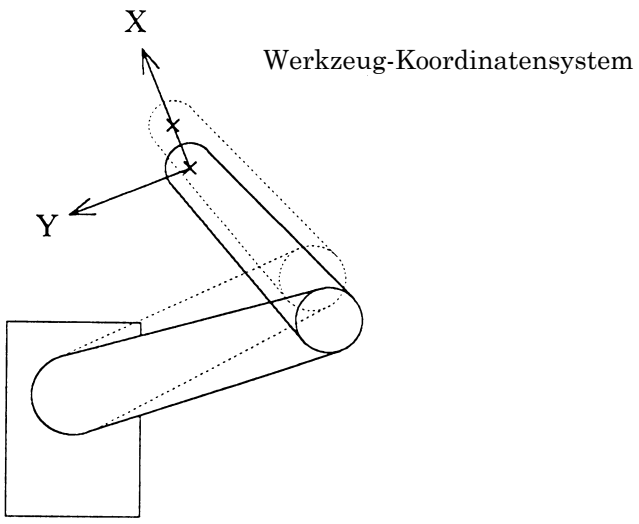
VERWANDTE BEFEHLE P=, ! ... !, SPEEDS, ACCELS, TILL, TGO

BEISPIEL

```
>TMOVE 100,0,0,0
```

Bewegt das derzeit ausgewählte Werkzeug innerhalb des Werkzeug-Koordinatensystems um 100 mm in horizontaler Richtung (entlang der X-Achse)

```
>_
TMOVE P1
>_
```



Werkzeug-Koordinatensystem

TMR( )

F

Timer

FUNKTION Timer-Funktion

FORMAT TMR([Timer-Nummer])

Die Timer-Nummer muß eine ganze Zahl zwischen 0 und 15 sein.

Ausgegebene Werte: 0 bis 21
474
836,47 (&H7FFFFFFF/100).

BESCHREIBUNG Gibt die abgelaufene Zeit aus; die Mindesteinheit beträgt 0,01 Sekunden.

Sie können maximal 16 Timer mit einer Nummer von 0 bis 15 definieren.

Definierte Timer können mit Hilfe des Befehls TMRESET zurückgesetzt werden.

VERWANDTE BEFEHLE TMRESET

BEISPIEL

```
100 TMRESET 0                Setzt Timer 0 zurück
110 FOR I=1 TO 10            Führt eine Operation 10mal aus
120    GOSUB CYCL
130 NEXT
140 PRINT TMR(0)/10          Berechnet die Zykluszeit und zeigt die Zeit
                             für einen Zyklus nach 10 Durchläufen als
                             Durchschnittswert an
```

T

TMRESET

> S

Timer Reset (Timer zurücksetzen)

FUNKTION Setzt einen definierten Timer zurück.

FORMAT `TMRESET([Timer-Nummer])`

Die Timer-Nummer muß eine ganze Zahl zwischen 0 und 15 sein.

BESCHREIBUNG Setzt den anhand der Timer-Nummer angegebenen Timer zurück und startet ihn gleichzeitig wieder.

Es können maximal 16 Timer zwischen 0 und 15 definiert sein.

Mit Hilfe des Befehls TMR( ) können Sie die Timer-Zeit (abgelaufene Zeit) anzeigen lassen.

VERWANDTE BEFEHLE TMR( )

BEISPIEL

```
100 TMRESET 0              Setzt Timer 0 zurück
110 FOR I=1 TO 10
120    GOSUB CYCL
130 NEXT
140 PRINT TMR(0)/10        Zeigt 1/10 der abgelaufenen Zeit an
```

TOFF

S

Trace Off (Anzeige aus)

FUNKTION Unterdrückt die Anzeige der Programmzeilennummern.

FORMAT `TOFF {![Task-Nummer]}`

BESCHREIBUNG Unterdrückt die Anzeige der Zeilennummern eines Programms.

Wird keine Task-Nummer angegeben, unterdrückt der Befehl TOFF die Anzeige der Zeilennummern nur bei den Tasks, für die der Befehl TOFF ausgeführt wurde.

VERWANDTE BEFEHLE TON

TON

> S

Trace On (Anzeige ein)

FUNKTION Zeigt die Zeilennummern eines Programms an.

FORMAT `TON {![Task-Nummer],}[Nummer des Ausgabegerätes]`

Die Task-Nummer muß eine ganze Zahl zwischen 1 und 16 sein (Standardwert ist 1); die Nummer des Ausgabegerätes muß eine ganze Zahl zwischen 0 und 3 sein (Standardwert ist 1).

BESCHREIBUNG Zeigt während der Programmausführung die Zeilennummer der angegebenen Task auf dem angegebenen Gerät an.

Wird keine Task-Nummer angegeben, zeigt der Befehl TON nur die Zeilennummern einer Task an, für die der Befehl TON ausgeführt wurde.

Die Ausgabegeräte werden wie folgt definiert:

0	Keine Anzeige
1	Anzeige im LED-Display der Steuerung
2	Anzeige an der Bedieneinheit
3	Anzeige im LED-Display der Steuerung und an der Bedieneinheit

Da bei Angabe mehrerer Tasks auch die Zeilennummern dieser verschiedenen Tasks gleichzeitig angezeigt werden, sollten Sie immer nur eine Task angeben.

Die Standardeinstellung für den Befehl TON ist TON!1,1. Das bedeutet, bei der Standardeinstellung werden die Programmzeilen der Task 1 im LED-Display der Steuerung angezeigt.

Bei einem Abbruch des Programms wird der TON-Befehl auf seine Standardeinstellung zurückgesetzt.

VERWANDTE BEFEHLE TOFF

BEISPIEL

```
>XQT !2, MAIN        Erforderlich, um die angegebene Task zu starten
>TON !2,2
[150]
[160]
   .
   .
   .
```

TOOL

> S

FUNKTION Wählt ein Werkzeug-Koordinatensystem aus und zeigt es an.

FORMAT

(1) `TOOL [Nummer des Werkzeug-Koordinatensystems]`

Die Nummer des Werkzeug-Koordinatensystems muß eine ganze Zahl von 0 bis 3 sein; der Standardwert beträgt 0.

(2) TOOL

BESCHREIBUNG

(1) Wählt das angegebene Werkzeug-Koordinatensystem aus.

(2) Zeigt die Nummer des derzeit ausgewählten Werkzeug-Koordinatensystems an.

Bei Angabe von Tool 0 wird das Standard-Werkzeug-Koordinatensystem (Hand-Koordinatensystem) ausgewählt.

Die Auswahl des Werkzeug-Koordinatensystems bleibt auch über das Ausschalten des Roboters hinaus gültig. Bei Ausführung des VERINIT-Befehls wird das Standard-Werkzeug-Koordinatensystem (Tool 0) ausgewählt.

VERWANDTE BEFEHLE TLSET, TGO, TMOVE

BEISPIEL

```
>TOOL 1     Wählt das Werkzeug-Koordinatensystem Tool 1 aus.
            Dazu muß Tool 1 jedoch bereits mit Hilfe des Befehls TLSET
            definiert worden sein
>JUMP P1    Positioniert Werkzeug 1 auf Punkt P1
>TOOL 0     Wählt des Werkzeug-Koordinatensystem Tool 0 aus
>JUMP P1    Positioniert die Standard-Hand auf Punkt P1
```

T

TRAP

S

Nicht programmierter Sprung, Unterbrechungsbedingung

FUNKTION Definiert einen Interrupt-Vorgang.

FORMAT

(1)
```
TRAP [Trap-Nummer] {[Eingangsbedingung] |GOTO |[Zeilennummer]| |}
                                              |[Label        |
                                        |GOSUB|[Zeilennummer]| |
                                              |[Label]       |
                                        |CALL |[Funktionsname]|
```

Die Trap-Nummer muß eine ganze Zahl von 1 bis 4 sein.
Die folgenden Funktionen und Operatoren können innerhalb einer Eingangsbedingung verwendet werden:

Funktionen:	SW, IN (die Angabe eines Eingangs oder eines Merkers ist möglich)
Operatoren:	AND, OR, XOR, +, *
Sonstige:	Runde Klammern zur Festlegung von Prioritäten für Operationen sowie Variablen

(2)
```
TRAP  |EMERGENCY| {CALL [Funktionsname]}
      |ERROR    |
      |PAUSE    |
      |SGOPEN   |
      |SGCLOSE  |
```

BESCHREIBUNG

(1) Führt den durch einen der Befehle GOTO, GOSUB bzw. CALL definierten Interrupt-Vorgang aus, wenn die Eingangsbedinung erfüllt ist. Wurde ein Interrupt-Vorgang ausgeführt, werden die entsprechenden TRAP-Einstellungen gelöscht. Benötigen Sie denselben Interrupt-Vorgang nochmals, müssen Sie ihn erneut mit Hilfe von TRAP definieren.

Wird die Eingangsbedingung erfüllt, während eine andere durch CALL aufgerufene Funktion ausgeführt wird, werden die mit GOTO bzw. GOSUB definierten Interrupt-Vorgänge innerhalb des TRAP-Befehls nicht ausgeführt.

Bei Auslassung der Eingangsbedingung und aller darauffolgenden Parameter wird die angegebene TRAP-Einstellung gelöscht.

Bei Angabe von GOTO wird der aktuelle Befehl wie folgt ausgeführt und das System verzweigt anschließend zur angegebenen Zeilennummer bzw. zum angegebenen Label.
- Die Armbewegung wird sofort unterbrochen
- Der durch den Befehl WAIT bzw. INPUT definierte Status wird aufgehoben
- Andere Befehle werden erst ausgeführt, wenn der Vorgang abgeschlossen ist

Bei Angabe von GOSUB ist der Ablauf folgendermaßen:
Nach Ausführung eines Befehls wie unter GOTO beschrieben, verzweigt das System anschließend zur angegebenen Zeilennummer bzw. zum angegebenen Label und führt das darauffolgende Unterprogramm aus. Hat das System die dortigen Anweisungen abgearbeitet und erreicht die RETURN-Anweisung am Ende des Unterprogramms, setzt es die Programmausführung mit der Zeile fort, die auf GOSUB folgt.

Auch wenn während der Ausführung des Unterprogramms innerhalb des Interrupt-Vorgangs ein Fehler erfolgt, wird eine durch ONERR definierte Fehlerroutine nicht ausgeführt. Im Gegenteil, selbst wenn die Eingangsbedingung während der Ausführung einer Fehlerroutine erfüllt wird, wird das innerhalb von TRAP definierte Unterprogramm nicht ausgeführt. Die Verwendung der Befehle GOSUB bzw. CALL im Unterprogramm eines TRAP-Befehls ist nicht erlaubt.

Bei Angabe von CALL wird die angegebene Funktion ausgeführt. In diesem Fall bleibt die Steuerung bei der Task, die den TRAP-Befehl ausführt.

(2) Ist eine der Eingangsbedingungen der Notaus, ein Fehler, der Befehl PAUSE oder die Schutzabschrankung geöffnet/schließen, wird eine innerhalb des TRAP-Vorgangs mit CALL definierte Funktion vorrangig ausgeführt. Stellen Sie sicher, daß jede Funktion eines TRAP-Vorgangs durch END beendet wird. Verwenden Sie den Befehl XQT nicht innerhalb einer Funktion eines TRAP-Vorgangs.

Bei Auslassung von CALL und aller darauffolgenden Parameter wird die angegebene TRAP-Einstellung gelöscht.

Wird EMERGENCY angegeben, beendet das System bei Eingabe eines Notaus zuerst den laufenden Systemvorgang und führt dann die Funktion des TRAP-Vorgangs aus.
Wurde zuvor festgelegt, daß die Ein- bzw. Ausgänge bei Eingabe eines Notaus zurückgesetzt werden, können die Befehle ON, OFF bzw. OUT für Ein-/Ausgänge innerhalb der Funktion einer TRAP-Anweisung nicht ausgeführt werden. Wurde jedoch festgelegt, daß die Ein-/Ausgänge ihren Status auch bei einem Notaus beibehalten, können diese Befehle verwendet werden. Falls erforderlich, sollten Sie Bit 6 des Softwareschalters SS1 zuvor einschalten.

Wird ERROR angegeben, d.h., wird ein Fehler (einschließlich eines Fehlers innerhalb des Systems) ausgegeben, beendet das System zuerst den laufenden Systemfehlervorgang und führt dann die Funktion des TRAP-Vorgangs aus. Dies gilt jedoch nicht, wenn der Fehler direkt bei der Ausführung des Befehls ausgegeben wird.

Bei Angabe von PAUSE:
Wird PAUSE während der Ausführung eines Programms im AUTO-Modus angegeben, beendet das System zuerst den Pause-Status und führt dann die Funktion des TRAP-Vorgangs aus. Die Funktion des TRAP-Vorgangs wird jedoch nicht ausgeführt, wenn der Pause-Status durch Öffnen/Schließen der Schutzabschrankung verursacht wurde.

Bei Angabe von SGOPEN
Im TEACH-Modus
Wird die Schutzabschrankung während einer Programmausführung geöffnet, beendet das System zuerst den Pause-Status und führt dann die Funktion des TRAP-Vorgangs aus. Wurde die Schutzabschrankung geöffnet, nachdem durch die geschlossene Schutzabschrankung bei Programmausführung ein Pause-Status verusacht wurde, wird die Funktion des TRAP-Vorgangs ausgeführt.
Im AUTO-Modus
Wurde die Schutzabschrankung geöffnet, nachdem durch die geschlossene Schutzabschrankung bei Programmausführung ein Pause-Status verusacht wurde, wird die Funktion des TRAP-Vorgangs ausgeführt.

Bei Angabe von SGCLOSE
Im TEACH-Modus
Wird die Schutzabschrankung während einer Programmausführung geschlossen, beendet das System zuerst den Pause-Status und führt dann die Funktion des TRAP-Vorgangs aus. Wurde die Schutzabschrankung geschlossen, nachdem durch die geöffnete Schutzabschrankung bei Programmausführung ein Pause-Status verusacht wurde, wird die Funktion des TRAP-Vorgangs ausgeführt.
Im AUTO-Modus
Wurde die Schutzabschrankung geschlossen, nachdem durch die geöffnete Schutzabschrankung bei Programmausführung ein Pause-Status verusacht wurde, wird die Funktion des TRAP-Vorgangs ausgeführt.

VERWANDTE BEFEHLE

ONERR, GOTO, GOSUB, CALL, ERT( ), ERA( ), ERL( ), ERRMSG$( )

BEISPIELE

Beispiel 1: Benutzerdefiniertes Fehlerprogramm
Die Eingabe von SW(0) wird als benutzerdefinierte Fehlereingabe betrachtet.

```
100 FUNCTION MAIN
110 TRAP 1 SW(0)=1 GOTO ERROR Definiert TRAP
 .
 .
 .
500 ERROR:
510 ON 31                                  Schaltet z.B. die rote
                                           Fehlermeldelampe ein
520 PRINT #20, "Fehler wird ausgegeben."
530 END
540 FEND
```

Beispiel 2: Verwendung wie beim Multitasking

```
100 FUNCTION MAIN
110 TRAP 2 SW($0)=1 OR SW($1)=1 CALL FEEDER
 .
 .
 .
540 FEND
 .
 .
 .
700 FUNCTION FEEDER
710 IF SW($0)=1 THEN OFF $0;ON #2,$0
720 IF SW($1)=1 THEN OFF $1;ON #3,$1
730 FEND
```

Beispiel 3: Scheintätigkeit des Programms (Dummy-Aktion)
Wenn Eingang SW(31) eingeschaltet wird, wird der WAIT-Status zwingend aufgehoben und das System verzweigt zum TRAP-Unterprogramm.

```
110 FUNCTION MAIN
120 TRAP 1 SW(31)=1 GOSUB SKP_WAIT
 .
 .
 .
200 WAIT SW(0)=1
210 WAIT SW(1)=1
 .
 .
 .
700 SKP_WAIT:
710 IF (STAT(1) AND &H0008)=1 THEN END
720 WAIT SW(31)=0
730 TRAP 1 SW(31)=1 GOSUB SKP_WAIT
740 RETURN
750 '
760 FEND
```

Die TRAP-Bedingung wird während der Armbewegung erfüllt. Sie wird nicht wiederholt

Definiert die TRAP-Bedingung erneut

Beispiel 4: ERROR ist angegeben

```
10 FUNCTION MAIN
20 TRAP ERROR CALL MSGOUT
 .
 .
 .
999 FEND
1000 FUNCTION MSGOUT
1010 PRINT #20, ERRMSG$(ERR(0))
1020 END
1030 FEND
```

TSPEED

> S

Teach Mode Speed (Geschwindigkeit im Teach-Modus)

FUNKTION

Definiert die maximal zulässige Geschwindigkeit für eine PTP-Bewegung im TEACH-Modus bzw. zeigt sie an.

FORMAT

(1) `TSPEED [Geschwindigkeitswert]`

Der Geschwindigkeitswert muß eine ganze Zahl von 1 bis 100 (%) sein. Der Standardwert ist 100.

(2) `TSPEED`

BESCHREIBUNG

(1) Definiert die im TEACH-Modus maximal zulässige Geschwindigkeit für eine PTP-Bewegung (über einen der Befehle GO, JUMP, PULSE o.ä. definiert).

Befindet sich der Roboter im Low-Power-Modus, d.h., die Stromzufuhr zum Roboter ist reduziert, beträgt die Geschwindigkeit für die PTP-Bewegung maximal die durch den Befehl TSPEED festgelegte Geschwindigkeit bzw. die SPEED-Standardgeschwindigkeit, unabhängig davon, welche Geschwindigkeit durch den Befehl SPEED direkt oder bereits zuvor definiert wurde.

Im normalen Modus beträgt die Geschwindigkeit für eine PTP-Bewegung maximal den derzeit gültigen TSPEED-Wert.

Die über den Befehl TSPEED festgelegte Geschwindigkeit bleibt über das Ausschalten des Roboters hinaus gültig.

Bei Ausführung des Befehls VERINIT wird die Geschwindigkeit auf den Standardwert 100 gesetzt.

Im AUTO-Modus wird der TSPEED-Wert ignoriert.

(2) Zeigt den derzeit gültigen TSPEED-Wert an.

VERWANDTE BEFEHLE

SPEED, POWER (LP)

BEISPIEL

```
>TSPEED 20
>COM
COMPILE END
>XQT
```

Die Geschwindigkeit für den JUMP-Befehl ist durch den TSPEED-Wert auf 20 % begrenzt.

```
>
>SPEED 100
>SPEED
       20
       20 20
```

Der über TSPEED definierte Wert hat Vorrang vor dem mit SPEED definierten Wert.

PROGRAMM

```
10 FUNCTION MAIN
20 SPEED 100
30 JUMP P1
40 JUMP P2
50 GOTO 20
60 FEND
```

TSPEEDS

> S

Teach Mode SPEED Straight

FUNKTION Definiert die maximal zulässige Geschwindigkeit für eine CP-Bewegung (Bahnbewegung) im TEACH-Modus bzw. zeigt sie an.

FORMAT

(1) `TSPEEDS [Geschwindigkeitswert]`

Der Geschwindigkeitswert muß eine ganze Zahl von 1 bis 1120 (mm/s) sein. Der Standardwert beträgt 1120.

(2) `TSPEEDS`

BESCHREIBUNG

(1) Definiert die im TEACH-Modus maximal zulässige Geschwindigkeit für eine CP-Bewegung (über einen der Befehle ARC, MOVE, CVMOVE o.ä. definiert).

Befindet sich der Roboter im Low-Power-Modus, d.h., die Stromzufuhr zum Roboter ist reduziert, beträgt die Geschwindigkeit für die CP-Bewegung maximal die durch den Befehl TSPEEDS festgelegte Geschwindigkeit bzw. die SPEEDS-Standardgeschwindigkeit, unabhängig davon, welche Geschwindigkeit durch den Befehl SPEEDS direkt oder bereits zuvor definiert wurde.

Die über den Befehl TSPEEDS festgelegte Geschwindigkeit bleibt über das Ausschalten des Roboters hinaus gültig.

Bei Ausführung des Befehls VERINIT wird die Geschwindigkeit auf den Standardwert 1120 mm/s gesetzt.

Im AUTO-Modus wird der TSPEEDS-Wert ignoriert.

(2) Zeigt den derzeit gültigen TSPEEDS-Wert an.

VERWANDTE BEFEHLE SPEEDS, POWER (LP)

BEISPIEL

```
>TSPEEDS 150
>SPEEDS 500
>SPEEDS
      150
```

Der über TSPEEDS definierte Wert hat Vorrang vor dem mit SPEEDS definierten Wert.

TSTAT

Task Status

FUNKTION Zeigt den Status einer Task an.

FORMAT TSTAT

BESCHREIBUNG Zeigt den aktuellen Status der Tasks 1 bis 16 an.

QUIT	Angehalten (Urzustand)
RUN	Ausführung läuft
HALT	Kurzzeitig unterbrochen
Zeile	Nummer der zuletzt ausgeführten Programmzeile (0 für Tasks, die nicht vom Programm ausgeführt wurden)

VERWANDTE BEFEHLE RUN, XQT, HALT, QUIT, RESUME

BEISPIEL

```
>TSTAT
Task     1     2     3     4     5     6     7    8
Status QUIT  QUIT  QUIT  QUIT  QUIT  QUIT  QUIT QUIT
Zeile  1250   180    60     0     0     0     0    0
>

Task     9    10    11    12    13    14    15   16
Status QUIT  QUIT  QUIT  QUIT  QUIT  QUIT  QUIT QUIT
Zeile     0   570     0     0     0     0     0    0
>
```

TW(0)

F

Timer Wait

FUNKTION Gibt den Status einer WAIT-Bedingung sowie das WAIT-Zeitintervall aus.

FORMAT `TW(0)`

Bei der Angabe in Klammern handelt es sich um die Ziffer 0 (Null).

BESCHREIBUNG Gibt den Status der vorangehenden WAIT-Bedingung sowie das WAIT-Zeitintervall durch eine der Ziffern 0 oder 1 aus.

Wenn die WAIT-Bedingung erfüllt wurde	0
Wenn das Zeitintervall abgelaufen ist	1

VERWANDTE BEFEHLE WAIT, TMOUT

BEISPIEL

```
100 WAIT SW(0)=1,5
```
Wartet 5 Sekunden, ob Eingang 0 eingeschaltet wird

```
110 IF TW(0)=1 THEN GOTO TIME_UP
```
Verzweigt nach Ablauf von 5 Sekunden zu TIME_UP

T

TYPE

>

FUNKTION Zeigt den Inhalt einer Datei an.

FORMAT `TYPE {[Laufwerk]}{[Pfadangabe]}[Dateiname]`

Der Dateiname muß mit der Dateinamenerweiterung angegeben werden.

BESCHREIBUNG Zeigt den Inhalt der angegebenen Datei an. Stellen Sie sicher, daß Sie eine ASCII-Datei angeben, da nur solche Dateien angezeigt werden können. Der Befehl TYPE dient lediglich dazu, den Inhalt einer Datei anzuzeigen, nicht ihn zu editieren. Wenn Sie die Datei editieren wollen, verwenden Sie entsprechende Methode dazu.

ASCII-Datei

Quellprogramm (.PRG)
Punktdatendatei (.PNT)
Dateien, die im Editiermodus erstellt wurden
Dateien, die über den Befehl ROPEN bzw. WOPEN erstellt wurden

Binärdateien

Objektprogramm (.OBJ)
Symboltabelle (.SYM)
Dateien, die mit dem Befehl CURVE erstellt wurden

Geben Sie keine Binärdateien an.
Über den Befehl TYPE wird der Dateiinhalt nur angezeigt, die Datei wird nicht in den Hauptspeicher geladen. Dies bedeutet, daß das angezeigte Quellprogramm bzw. die angezeigten Punktdaten nicht mit den Daten im Hauptspeicher übereinstimmen.

VORSICHT: Achten Sie unbedingt darauf, daß Sie die ENTER-Taste nicht drücken, solange sich der Cursor auf dem angezeigten Dateiinhalt befindet. Durch Drücken der ENTER-Taste würde der angezeigte Inhalt einer Datei eingegeben, wodurch möglicherweise eine ungewollte Roboterbewegung erfolgt oder die Programm- bzw. Punktdaten im Hauptspeicher geändert werden.

BEISPIEL

```
>TYPE TEST.PRG
10 FUNCTION TEST
20 PRINT "Testprogramm"
30 FEND
```

VAL( )

F

Variable

FUNKTION Gibt den numerischen Wert der angegebenen numerischen Zeichenkette aus.

FORMAT

```
VAL(|[Name der Zeichenkettenvariablen]|)
    |     "[numerische Zeichenkette"  |
```

BESCHREIBUNG Gibt den numerischen Wert der angegebenen numerischen Zeichenkette aus.

VERWANDTE BEFEHLE STR$( )

BEISPIEL

```
>PRINT VAL("1234")
1234
>
```

V

VARIABLE

FUNKTION Zeigt Variablen an.

FORMAT VARIABLE {-A}

BESCHREIBUNG Zeigt die Variablen sowie deren Typ an, die im kompilierten Objektprogramm im Hauptspeicher verwendet werden.

Durch Eingabe von VARIABLE -A werden zusätzlich noch die Funktionsnamen angezeigt.

Der Befehl VARIABLE zeigt jedoch nicht die Backup-Variablen an. Verwenden Sie dazu den Befehl LIBRARY.

VERWANDTE BEFEHLE LIBRARY

BEISPIEL

```
10 FUNCTION MAIN
20 INTEGER I, J
30 REAL DATA
40 STRING DISPLAY$
50 FEND
```

```
>COMPILE
COMPILE END
>
>VARIABLE
INTEGER I
INTEGER J
REAL    DATA
CHR     DISPLAY$
>
VARIABLE -A
FUNCTION MAIN
INTEGER I
INTEGER J
REAL    DATA
CHR     DISPLAY$
>
```

CHR repräsentiert eine Zeichenkettenvariable

VER

Version

FUNKTION Zeigt die Systemdaten an.

FORMAT (1) VER

(2) VER {/A}

BESCHREIBUNG Zeigt die aktuellen Systemdaten an.

Bei Neuauslieferung des Roboters oder wenn die Systemdaten geändert wurden, sollten Sie die Systemdaten sichern. Verwenden Sie dazu den Befehl MKVER.

(1) Wenn Sie nur den Befehl VER eingeben, werden die folgenden Daten angezeigt. (Hierbei handelt es sich jedoch nur um ein Beispiel, da die tatsächlich angezeigten Daten abhängig sind vom verwendeten Modell bzw. vom Zustand.)

```
>VER
<VERSION>                    <Version von SPEL>
 VER 4.1
<ROM>                        <ROM-Version>
 MPUjs-41                      ROM auf der MPU-Platine
 DSPas-41                      ROM für DSP auf der MPU-Platine
 OPUaa-41                      ROM in der Bedieneinheit
 SCCaa-41                      Serieller Bus (SYS.BUS)
 ERSaa-41                      ROM auf einer zusätzlichen
                               RS-232C-Platine
<OPTIONS>                    <Installierte Optionen>
 OPU
 I/O 16-31
 SYS.BUS
 RS-232C.#22,#23
<CONSOLE>                    <AUTO-Mode-Bedieneinheit>
 OPERATING UNIT
<SD>                         <DIP-Schalter-Stellungen>
   8   F0                      SD1, SD2 auf der MPU-Platine
  00                           SD3 auf der MPU-Platine
<SS>                         <Status der Software-Schalter>
   0    0
   0    0
   0    0
<REMOTE>                     <Einstellung von REMOTE3>
   0    0
   0    0
<OPUNIT>                     <Modus der Bedieneinheit>
        0
<CONFIG>                     <Konfiguration für den RS-232C-Anschluß>
 #20   2  1  3  0
 #21   2  1  3  0
```

```
<SELRB>                          <Nummer des Positionierungsgerätes>
       1
<MAXDEF>                         <Max. Anzahl Peripheriegeräte am
       1                         seriellen Bus>
<MANIPULATOR>                    <Manipulator-Modell>
 SSR-H554BN
<M.CODE>                         <M.CODE>
   10302
<RANGE>                          <Zulässiger Verfahrbereich (Pulse)>
   -36409   364089                 Erste Achse
  -159289   159289                 Zweite Achse
   -92160        0                 Dritte Achse
  -172032   172032                 Vierte Achse
<HORDR>                          <Reihenfolge der Achsenbewegung beim
     02         0D                 HOME-Befehl>
     00         00
<HOFS>                           <Encoder-Offset-Pulse>
     1234     4567
        0        0
<MCORDR>                         <Reihenfolge der Achsenbewegung bei der
       02        0D                 Kalibrierung>
       00        00                 (Anzeige bei inkrementellem Roboter)
<MCOFS>                          <Offset-Werte für die Kalibrierung>
      04                           (Anzeige bei inkrementellem Roboter)
    6510      5368
    5662      3830
    -620      -295
<HTEST>                          <Werte für den HTEST-Befehl>
     -174       -11                (Anzeige bei inkrementellem Roboter)
        0        10
<WEIGHT> (kg,mm)                 <Werte für den WEIGHT-Befehl>
        2      225
<ARM>                            <Werte für Standard- und Zusatzarm>
arm0=225 0 0 325 0
<TOOL>                           <Werte für Standard- und
 tool0=0 0 0 0                   Zusatzwerkzeugkoordinaten>
<HOMESET>                        <Wert für die HOME-Position (Pulse)>
        0        0
        0        0
<CALPLS>                         <Werte für den CALPLS-Befehl>
    65523    43320
    -1550    21351
<FREE>                           <Freier Speicherbereich im Hauptspeicher>
 PRG = 65312                       Quellprogrammbereich
 VER = 9 , 458                     Backup-Variablen-Bereich
    ( 10 , 512)
OBJ =  99469                       Objektbereich
<MEMORY>                         <Kapazität der Bereiche im Hauptspeicher>
 PRGSIZE 65536                     Quellprogrammbereich
 PNTSIZE 125                       Verfügbare Anzahl von Positionsdaten
 LIBSIZE 10 ,  512                 Objektbereich
<PRGNO>                          <Art der Auswahl der Programmnummer
        1                          an REMOTE2>
<HOUR>                           <Gesamt-Betriebszeit der Steuerung>
      100
```

(2) Bei Eingabe von VER/A werden die Daten wie folgt angezeigt:

```
>VER/A
<DATE & TIME>
   1994.12.01/ 14:48:50
<VERSION>
 Ver 4.1
  .
  .
  .
<HOUR>
     100
<SELDISK>
        A:
<WIDTH>
     80 , 25
<ARCH>
arch0=30 30
arch1=40 40
arch2=50 50
arch3=60 60
arch4=70 70
arch5=80 80
arch6=90 90
<TSPEED>
       70
<TSPEEDS>
     1120
<XYLIM>
        0     0
        0     0
<LOCAL>
 local0 (p***:p***),(p***:p***)
```

<Datum und Uhrzeit bei Ausführung des Befehls VER/A>

} Identisch mit den VER-Daten

<Aktuelles Laufwerk>

Anzeigeformat für den FILES-Befehl

<Bogenform>

<TSPEED-Wert>

<TSPEEDS-Wert>

<Erlaubter Verfahrbereich in der XY-Ebene>

<Lokale Koordinaten>

VERWANDTE BEFEHLE VERINIT, MKVER, SETVER

VERINIT

Version Data Initialisation

FUNKTION Initialisiert die Systemdaten.

FORMAT VERINIT

BESCHREIBUNG Initialisiert die Systemdaten.

Durch Ausführung des Befehls VERINIT werden die Werte für jeden Befehl auf die vordefinierten Initialwerte zurückgesetzt.

Eine Aufzählung aller betroffenen Befehle und der entsprechenden Initialwerte finden Sie in der folgenden Liste.

Nach Eingabe von VERINIT erscheint die folgende Meldung:

```
Version data initialize  -- > OK?
?
```

Wenn Sie die Werte zurücksetzen wollen, geben Sie **Y** bzw. **y** ein.

Wenn Sie die aktuellen Werte erhalten wollen, geben Sie **N** bzw. **n** ein.

Dieser Befehl ist vor allem für Wartungszwecke gedacht. Der Befehl VERINIT setzt die Befehle in der o.g. Weise zurück. Wenn Sie nicht mit diesen Werten arbeiten wollen, müssen Sie sie nach Ausführung von VERINIT die gewünschten Werte entsprechend ändern.

VERWANDTE BEFEHLE VER, MKVER, SETVER

Befehl	Initialwert		
PRG.Nr.	00		
SELDISK CONFIG CONSOLE MAXDEV WIDTH PRGNO	A: 2, 1, 3, 0 (Alle RS-232C-Anschlüsse) OP 1 20, 23 1		
SEL	0		
SET	0, 0.03, 0.03, 0.03, 0.03		
	1, 0.1, 0.1, 0.1, 0.1		
	2, 1, 1, 1, 1		
	3, 10, 10, 10, 10		
SPEED ACCEL WEIGHT TSPEED TSPEEDS SPEEDS ACELLS FINE	Abhängig vom Manipulatortyp Abhängig vom Manipulatortyp Abhängig vom Manipulatortyp 100 1120 Abhängig vom Manipulatortyp Abhängig vom Manipulatortyp Abhängig vom Manipulatortyp		
LIMZ	0		
ARCH	0, 30, 30	1, 40, 40	2, 50, 50
	3, 60, 60	4, 70, 70	5, 80, 80
	6, 90, 90		
HOMESET HORDR MCORDR	Gelöscht Abhängig vom Manipulatortyp Abhängig vom Manipulatortyp		
XYLIM	0, 0, 0, 0		
ARM ARMSET TOOL TLSET LOCAL0 LOCAL1-15 BASE 0 BASE 1-15	0 Gelöscht (außer ARM 0) 0 Gelöscht (außer TOOL 0) Entspricht dem original Roboter-Koordinatensystem Gelöscht Entspricht dem original Roboter-Koordinatensystem Gelöscht		

VLOAD

> S

Variable Load (Variable laden)

FUNKTION Lädt die Daten einer Variablen.

FORMAT `VLOAD "{[Laufwerk]}{[Pfadname]}[Dateiname]"`

BESCHREIBUNG Lädt die Daten einer Variablendefinitionsdatei, die mit Hilfe des Befehls VSAVE erstellt wurde und ordnet den dort gespeicherten Wert der Variablen zu.

Wird die Laufwerksangabe ausgelassen, nimmt das System das aktuelle Laufwerk an; wird keine Pfadangabe gemacht, nimmt das System das aktuelle Verzeichnis an.

Stellen Sie sicher, daß der angegebene Dateiname mit dem identisch ist, den Sie bei der Speicherung der Variablendaten mit Hilfe des Befehls VSAVE verwendet haben. Haben Sie dabei eine Dateinamenerweiterung angegeben, müssen Sie den kompletten Dateinamen, also mit der Dateinamenerweiterung, hinter dem Befehl VLOAD eingeben.

Hinweis:
Um mit dem Befehl VLOAD korrekt arbeiten zu können, muß der Variablenname, auf den sich die Variablendefinitionsdatei bezieht bereits in der Symboltabelle im Hauptspeicher enthalten sein.

VERWANDTE BEFEHLE VSAVE

BEISPIEL

```
I(0)=1
I(1)=2
I(2)=3
```

Das o.g. Datenfeld ist in einer Variablendefinitionsdatei mit dem Dateinamen IDATA.DAT enthalten. In diesem Fall kann die Variable I wie folgt definiert werden:

```
>PRINT I(0),I(1),I(2)
 0  0  0
>VLOAD "IDATA.DAT"
>PRINT I(0),I(1),I(2)
 1  2  3
>
```

VSAVE

Variable Save (Variable speichern)

FUNKTION Speichert Variablendaten.

FORMAT `VSAVE [Variablenname],"{[Laufwerk]}{[Pfad]}[Dateiname]"`

BESCHREIBUNG Speichert Daten, die einem bestimmten Variablennamen zugeordnet werden sollen unter dem angegebenen Dateinamen. Dadurch wird eine sogenannte Variablendefinitionsdatei erstellt.

Wird die Laufwerksangabe ausgelassen, nimmt das System das aktuelle Laufwerk an; wird keine Pfadangabe gemacht, nimmt das System das aktuelle Verzeichnis an.

Bei Angabe des Dateinamens können Sie eine beliebige Dateinamenerweiterung von maximal 3 Zeichen verwenden. Geben Sie keine Dateinamenerweiterung an, speichert das System nur den Dateinamen. Um jedoch die Variablendefinitionsdateien von anderen unterscheiden zu können, ist die Verwendung einer Dateinamenerweiterung empfehlenswert. Eine häufig verwendete Erweiterung ist beispielsweise ".DAT".

Sie können in einer Variablendefinitionsdatei die Daten von nur einer Variablen speichern.

Wenn Sie eine Datei ohne Dateinamenerweiterung mit Hilfe des Befehls KILL löschen wollen, müssen Sie nach dem Dateinamen unbedingt einen Punkt (.) eingeben, z.B.:

KILL "VARIABLE."

VERWANDTE BEFEHLE VLOAD

BEISPIEL

```
>I(0)=1;I(1)=2;I(2)=3
>VSAVE I, "IDATA.DAT"
>
```

Bei diesem Beispiel lautet der Inhalt der Datei IDATA.DAT wie folgt:

```
I(0)=1
I(1)=2
I(2)=3
```

WAIT

> S

FUNKTION

Definiert das Timer-Intervall und unterbricht die Programmausführung, bis die angegebene Bedingung erfüllt ist.

FORMAT

(1) `WAIT [Zeitintervall]`

(2) `WAIT [Eingangsbedingung]`

(3) `WAIT [Eingangsbedinung],[Zeitintervall]`

Das Timer-Intervall wird in der Einheit Sekunden angegeben, das Mindestintervall beträgt 0,01.

Die folgenden Funktionen und Operatoren können innerhalb einer Eingangsbedingung verwendet werden:

Funktionen:	SW( ), IN( ) (die Angabe eines Eingangs oder eines Merkers ist möglich)
Operatoren:	AND, OR, XOR, +, *
Sonstige:	Runde Klammern zur Festlegung von Prioritäten für Operationen sowie Variablen

BESCHREIBUNG

(1) WAIT mit Zeitintervall:
Legt das Zeitintervall fest. Nach Ablauf der angegebenen Zeit wird die Programmsteuerung mit dem nächsten Befehl fortgesetzt.

(2) WAIT mit Eingangsbedingung, jedoch ohne Zeitintervall:
Definiert die WAIT-Bedingung. Die Programmausführung wird unterbrochen, bis die WAIT-Bedingung erfüllt ist. Danach wird die Programmausführung mit dem nächsten Befehl fortgesetzt.

Wird die WAIT-Bedingung nach Ablauf des festgelegten Intervalls (mit Hilfe des Befehls TMOUT) nicht erfüllt, erfolgt eine Fehlermeldung.

(3) WAIT mit Eingangsbedingung und Zeitintervall:
Definiert die WAIT-Bedingung und das Zeitintervall. Das Programm wird mit dem nächsten Befehl fortgesetzt, wenn entweder die WAIT-Bedingung erfüllt wurde oder das Zeitintervall abgelaufen ist. Mit Hilfe des Befehls TW(0) können Sie feststellen, ob das Programm durch Erfüllung der WAIT-Bedingung oder nach Ablauf des Zeitintervalls fortgesetzt wurde.

VERWANDTE BEFEHLE

TMOUT, TW(0), SW( ), IN( ), DSW( )

BEISPIEL

```
>WAIT 1;ON 1
```
Wartet eine Sekunde und schaltet dann Ausgang 1 ein

```
>WAIT SW(1)=1
```
Wartet, bis Eingang 1 eingeschaltet ist

```
>WAIT SW(1)=1,5
```
Wartet 5 Sekunden, ob Eingang 1 eingeschaltet ist. Ist Eingang 1 nach Ablauf der 5 Sekunden nicht eingeschaltet, wird das Programm mit dem nächsten Befehl fortgesetzt

```
>WAIT SW(5)=1 AND SW(6)=1
```
Wartet, bis Eingang 5 und 6 eingeschaltet sind

```
100 CHK:
110   WAIT SW(1)=1,5
120   IF TW(0)=1 THEN GOSUB ERRPRC;GOTO CHK
```
Wird Eingang 1 nicht eingeschaltet, wird die Funktion ERRPRC ausgeführt und anschließend der Status von Eingang 1 erneut überprüft

```
200 ERRPRC:
210   PRINT "SW(2) einschalten, wenn OK"
220   WAIT SW(2)=1
230   RETURN
 .
 .
 .
```

Auswahl des Betriebsmodus mit Hilfe der Funktionstasten an der Bedieneinheit OPU-300

```
1000 OPUNIT 2
```
Modusauswahl

```
1010 OPU PRINT 1,3,"Bedienmodus auswählen."
```
Anzeige einer Meldung

```
1020 OPU PRINT 1,4,"F1:Normaler Betrieb"
1030 OPU PRINT 1,5,"F2:Dummy-Betrieb"
1040 OPU PRINT 1,6,"F4:Betriebsende"
1050 WAIT ( DSW(3) AND &B10110000 ) <> 0
```
Wartet auf Eingabe über eine Funktionstaste

```
1060 IF DSW(3) AND &B10000000 THEN GOTO F4KEY
1070 IF DSW(3) AND &B00100000 THEN GOTO F2KEY
1080 IF DSW(3) AND &B00010000 THEN GOTO F1KEY
```

Anmerkung:
Den Funktionstasten der Bedieneinheit sind Binärwerte zugeordnet. Diese können auch direkt abgefragt werden:

```
IF DSW(3)=16 THEN GOTO ANFANG
```
Sprung zu "ANFANG", wenn F1 gedrückt wird

WEIGHT

> S

FUNKTION Definiert und zeigt die Parameter an, die zur Berechnung der maximalen Beschleunigung bei einer PTP-Bewegung verwendet werden.

FORMAT

(1) `WEIGHT [Handgewicht]{,[Armlänge]}`

(2) `WEIGHT`

BESCHREIBUNG

(1) Definiert die Parameter, die zur Berechnung der maximalen Beschleunigung bei einer PTP-Bewegung verwendet werden.

Der Wert für das Handgewicht beinhaltet das Gewicht des Werkstücks und des Greifers.

Die Definition der Armlänge ist nur bei einem SCARA-Roboter notwendig. Hierbei handelt es sich um die Entfernung von der Mittellinie der Rotationsachse der zweiten Hand zum Schwerpunkt der Hand/Werkstück-Kombination.

Falls der aus den angegebenen Parametern berechnete Wert für das Gewicht des Werkstücks die erlaubte Gesamt-Nutzlast überschreitet, erfolgt eine Fehlermeldung.

(2) Zeigt die aktuellen WEIGHT-Parameter an.

Wenn das über den Befehl WEIGHT definierte Handgewicht geringer ist als das tatsächliche Gewicht des Werkstücks, kann dies zu übermäßigen Beschleunigungen bzw. Geschwindigkeitsverzögerungen führen und somit zu schweren Beschädigungen des Manipulators.

Die WEIGHT-Einstellungen bleiben auch über das Ausschalten des Roboters hinaus gültig. Bei Ausführung des Befehls VERINIT werden sie auf ihre Standardwerte zurückgesetzt. Die Standardwerte für den WEIGHT-Befehl sind abhängig vom verwendeten Manipulatorarm. Detaillierte Informationen dazu erhalten Sie in der Dokumentation zum Manipulatorarm.

VERWANDTE BEFEHLE ACCEL, VERINIT

BEISPIEL

```
>WEIGHT
>
```

WHILE...WEND

S

While...While End (Während...Während Ende)

FUNKTION

Führt die angegebenen Anweisungen aus, während die festgelegte Bedingung erfüllt ist.

FORMAT

```
WHILE [Bedingung]
  .
  .
  .
WEND
```

Es sind maximal 20 Verschachtelungen erlaubt. (Wie Sie Verschachtelungen angeben, erfahren Sie in den Erläuterungen zum Befehl #include.)

BESCHREIBUNG

Definiert die WHILE-Bedingung. Wenn diese erfüllt wird, werden die Anweisungen zwischen WHILE und WEND ausgeführt und anschließend wird die WHILE-Bedingung erneut geprüft. Dieser Vorgang (Ausführung der WHILE...WEND-Anweisungen, Prüfen der WHILE-Bedingung) wird solange wiederholt, wie die WHILE-Bedingung erfüllt ist.

Wird die WHILE-Bedingung nicht erfüllt, wird die Programmsteuerung mit dem auf WEND folgenden Befehl fortgesetzt. Wird die WHILE-Bedingung bei der ersten Abfrage nicht erfüllt, werden die Anweisungen zwischen WHILE und WEND nie ausgeführt.

Auf jede WHILE-Anweisung muß eine WEND-Anweisung folgen.

Es können bis zu 20 WHILE...WEND-Schleifen innerhalb einer WHILE...WEND-Schleife verschachtelt werden. Jeder WEND-Anweisung muß eine WHILE-Anweisung vorangehen; ist dies nicht der Fall, erfolgt eine Fehlermeldung.

Die WHILE-Bedingung kann jeden gültigen Operator enthalten.

BEISPIEL

```
100 I=1
110 WHILE I<60        Führt alle Anweisungen zwischen WHILE und
                      WEND aus, wenn I<60 ist
  .
  .
  .
300 I=I+2
310 WEND
```

WIDTH

FUNKTION Definiert die Anzahl der Zeichen pro Zeile und Zeilen pro Bildschirm, die auf dem Monitor der Programmiereinheit angezeigt werden.

FORMAT `WIDTH [Anzahl Zeichen/Zeile]{[,Anzahl Zeilen/Bildschirm]}`

Die Anzahl der Zeichen pro Zeile kann mit 20, 40 oder 80 angegeben werden. Der Standardwert beträgt 20. Die Anzahl der Zeilen pro Bildschirm kann mit einem Wert zwischen 4 und 23 angegeben werden. Der Standardwert beträgt 23.

BESCHREIBUNG Mit der Anzahl Zeichen pro Zeile legen Sie das Anzeigeformat bei Ausführung des Befehls FILES (Anzeige von Dateinamen) fest.

Die Anzahl der Zeilen pro Bildschirm legt fest, wieviel Zeilen bei Eingabe des Befehls DIR mit dem Parameter /P (seitenweise) angezeigt werden.

Die mit dem Befehl WIDTH definierten Werte bleiben über das Ausschalten des Roboters hinaus gültig, werden jedoch bei Ausführung des VERINIT-Befehls auf die Initialwerte zurückgesetzt.

VERWANDTE BEFEHLE FILES

BEISPIEL

```
>WIDTH 20
>FILES
AAAA PRG  1
AAAA OBJ  1
AAAA SYM  1
  space 59
>
>WIDTH 80
>FILES
AAAA PRG  AAAA OBJ  AAAA SYM
  space 59
>
```

WOPEN...CLOSE

S

Write Open (Öffnen zum Schreiben)

FUNKTION Öffnet eine Datei, um Daten hineinzuschreiben.

FORMAT

```
WOPEN "[Dateiname]" AS #[Dateinummer]
  .
  .
  .
CLOSE #[Dateinummer]
```

Der Dateiname muß mit der Dateinamenerweiterung angegeben werden. Die Dateinummer muß eine ganze Zahl von 30 bis 35 sein.

BESCHREIBUNG Öffnet die Datei mit dem angegebenen Dateinamen und kennzeichnet sie anhand der angegebenen Dateinummer. Mit Hilfe dieser Anweisung werden Daten in die Datei geschrieben. (Wenn Sie Daten an eine bestehende Datei anfügen wollen, verwenden Sie den Befehl AOPEN.) Über den Befehl CLOSE wird die Datei wieder geschlossen und die kennzeichnende Dateinummer wieder freigegeben.

Existiert der angegebene Dateiname nicht im aktuellen Pfad, wird durch den Befehl WOPEN eine neue Datei mit diesem Namen erstellt und die Daten werden in diese Datei geschrieben. Existiert die Datei bereits, werden alle Daten darin mit den neuen Daten überschrieben.

Die Dateinummer kennzeichnet die Datei solange sie geöffnet ist, sie wird für die Ausgabeanweisung zum Schreiben (PRINT #) und zum Schließen der Datei (CLOSE #) benötigt. Das bedeutet, solange die aktuelle Datei geöffnet ist, kann die Dateinummer nicht zur Kennzeichnung einer anderen Datei verwendet werden.

Es können maximal sechs Dateien gleichzeitig geöffnet sein. Sind jedoch die maximal erlaubten sechs Dateien gleichzeitig geöffnet, können die Befehle DLOAD und DMERGE nicht ausgeführt werden.

VERWANDTE BEFEHLE PRINT #, AOPEN, ROPEN

BEISPIEL

```
100 REAL DATA(200)
110 WOPEN "TEST.VAL" AS #30
120 FOR I=0 TO 100
130 PRINT #30, DATA(I)
140 NEXT
150 CLOSE #30
160 '
170 ROPEN "TEST.VAL" AS #30
180 FOR I=0 TO 100
190 INPUT #30,DATA(I)
200 NEXT
210 CLOSE #30
220 '
```

XQT

> S

Execute (Ausführen)

FUNKTION Führt ein Programm aus.

FORMAT

(1) `XQT {"{[Laufwerk]}{[Pfadangabe]}[Dateiname]"}`

Die Angabe einer Dateinamenerweiterung ist nicht erlaubt.

(2)
```
XQT ![Task-Nr.][Funktion]|[1. Zeilen-Nr.]-                |
                         |[1. Zeilen-Nr.]-[letzte Zeilen-Nr.]|
                         |              -[letzte Zeilen-Nr.]|
```

(3) `XQT !|Task-Nr.] [Funktion]`

BESCHREIBUNG Führt das angegebene Objektprogramm aus. Ein Objektprogramm muß zuerst durch die Kompilierung des Quellprogramms erstellt werden.

(1) Werden Laufwerk, Pfadangabe und Dateiname ausgelassen, wird das Programm ausgeführt, das sich im Hauptspeicher befindet.

In diesem Format kann XQT nicht als Anweisung verwendet werden.

Wird ein Dateiname angegeben, führt der Befehl XQT entweder das Programm im Dateispeicher oder auf der Festplatte (RAM) aus. In diesem Fall werden die folgenden Dateien in den Hauptspeicher geladen:

Dateiname.OBJ
Dateiname.SYM
Dateiname.PNT

Dementsprechend werden die Positionsdaten, die sich zuvor im Hauptspeicher befanden, gelöscht. Falls Sie diese Daten später noch benötigen, müssen Sie sie vor der Ausführung des Befehls XQT sichern.

Wird kein Laufwerk angeben, sucht das System die Datei auf dem aktuellen Laufwerk. Bei Auslassung der Pfadangabe sucht das System im aktuellen Verzeichnis.

(2) Führt die angegebene Funktion, die zuvor in den Hauptspeicher geladen wurde, als angegebene Task aus.

In diesem Format kann XQT nicht als Anweisung verwendet werden.

Bei Angabe einer oder mehrerer Zeilennummern erfolgt die Programmausführung wie folgt:

XQT ![Task-Nr.],[Funktion][1. Zeilen-Nr.]-
Alle Zeilennummern, beginnend mit der "1. Zeilen-Nr.", bis zum Ende des Programms werden ausgeführt
XQT ![Task-Nr.],[Funktion][1. Zeilen-Nr.]-[letzte Zeilen-Nr.]
Alle Zeilen von der "1. Zeilen-Nr." bis zur "letzten Zeilen-Nr." einschließlich werden ausgeführt
XQT ![Task-Nr.],[Funktion]-[letzte Zeilen-Nr.]
Alle Zeilen bis zur "letzten Zeilen-Nr." einschließlich werden ausgeführt

(3) Führt die angegebene Funktion als angegebene Task aus.

Im diesem Format kann XQT als Anweisung verwendet werden. Dazu muß die Anweisung in die Hauptfunktion (Main Function) integriert werden.

Im Low-Power-Modus (reduzierte Stromzufuhr) erscheint die folgende Meldung, um anzuzeigen, daß sich der Roboter im Low-Power-Modus befindet und sich mit niedriger Geschwindigkeit bewegt.

Low Power Mode (--> LP Command)

VERWANDTE BEFEHLE COMPILE, HALT, RESUME, QUIT, TSTAT, RUN, FUNCTION...FEND

XYLIM

> S

XY Limit

FUNKTION

Definiert oder zeigt den zulässigen Verfahrbereich in X- und Y-Richtung im Koordinatensystem an.

FORMAT

(1) `XYLIM [Unterer Bereich X-Achse],[oberer Bereich X-Achse],~`
`[Unterer Bereich Y-Achse],[oberer Bereich Y-Achse]`

(2) `XYLIM`

BESCHREIBUNG

(1) Definiert im Roboter-Koordinatensystem die X- und Y-Koordinaten für das untere und obere Bewegungslimit und legt damit den zulässigen Verfahrbereich fest.

Der über den Befehl XYLIM festgelegte Verfahrbereich gilt nur für die Zielpositionen bei Ausführung eines Bewegungsbefehls, nicht jedoch für den Verfahrweg von der Ausgangs- zur Zielposition. Das bedeutet, daß der Manipulatorarm den über XYLIM definierten zulässigen Bereich in der Bewegung verlassen kann.

Der über XYLIM definierte Verfahrbereich beeinflußt nicht die Ausführung der Befehle PULSE bzw. CVMOVE.

(2) Zeigt die aktuell gültigen XYLIM-Werte an.

Die XYLIM-Werte werden bei Ausführung des Befehls VERINIT oder durch Angabe von XYLIM 0,0,0,0 initialisiert. In beiden Fällen werden die Begrenzungen den Verfahrbereichs gelöscht.

VERWANDTE BEFEHLE

RANGE

BEISPIEL

```
>XYLIM                    Zeigt die aktuellen XYLIM-Werte an
       0    0
       0    0
>XYLIM -200,300,0,500     Definiert den Verfahrbereich des Roboters
                          -200 ≤ X ≤ 300
                          0 ≤ Y ≤ 500
```

ZEROFLG(0)

F

Zero Flag

FUNKTION Gibt den Status des Merkers (0) vor der letzten Statusänderung aus.

FORMAT ZEROFLG(0)

Bei dem Zeichen in Klammern handelt es sich um die Ziffer 0 (Null).

BESCHREIBUNG Gibt den Status des Merkers (0) vor der letzten Statusänderung aus.

Der Befehl ZEROFLG(0) dient der ausschließlichen Programmsteuerung und wird verwendet, um eine einzelne Ressource (z.B. einen RS-232C-Anschluß) während der Ausführung von Programmen mit Mehrfach-Tasks zentral zu nutzen.

VERWANDTE BEFEHLE ON $, OFF $

BEISPIEL

```
40 OFF $0
50 XQT !2, SUB
.
.
.
300 ON $0                          Fordert den RS-232C-Anschluß an
310 IF ZEROFLG(0)=1 THEN WAIT SW($0)=0;GOTO 300
320 PRINT #20,1
330 OFF $0                         Gibt den RS-232C-Anschluß frei
 .
 .
 .
1000 FUNCTION SUB
1010 ON $0                         Fordert den RS-232C-Anschluß an
1020 IF ZEROFLG(0)=1 THEN WAIT SW($0)=0;GOTO 1010
1030 FOR I=0 TO 100
1040 PRINT #20,I
1050 NEXT
1060 OFF $0
 .
 .
 .
```

Z

Fehlercode-Tabellen

Die vom System gemeldeten Fehler lassen sich in drei Kategorien unterteilen und werden in den Tabellen wie folgt gekennzeichnet:

Code	Fehler, die behoben werden können, ohne über den Befehl RESET ein Reset durchführen zu müssen.
Code	Fehler, die durch den Befehl RESET behoben werden sollten.
Code	Fehler, die behoben werden sollten, indem die Stromzufuhr aus- und wieder eingeschaltet wird.

Zusätzlich finden Sie in der Tabelle die bei jedem Fehler angezeigte Fehlermeldung.

Nummer	Meldung	Bedeutung	Maßnahme
0	Keine Fehler	Keine Fehler	
1	FOR, WHILE oder SELECT fehlt	① Die zur NEXT-Anweisung gehörende FOR-Anweisung fehlt.	① Fügen Sie eine FOR-Anweisung ein.
		② Die zur WEND-Anweisung gehörende WHILE-Anweisung fehlt.	② Fügen Sie eine WHILE-Anweisung ein.
		③ Die zur SEND-Anweisung gehörende SELECT-Anweisung fehlt.	③ Fügen Sie eine SELECT-Anweisung ein.
		④ Die zur ELSE, ENDIF gehörende IF-Anweisung fehlt.	④ Fügen Sie eine IF-Anweisung ein.
2	Syntax-Fehler o. undefinierte Angaben	① Syntaxfehler	① Korrigieren Sie die Syntax.
		② Der Befehl enthält eine nicht definierte Variable.	② Verwenden Sie die Variablen, die bereits kompiliert und eingetragen sind.
		③ Ein verwendetes Datenfeld ist nicht definiert.	③ Verwenden Sie die Variablen, die bereits kompiliert und eingetragen sind.
		④ Eine Batch-Datei sollte ohne Pfadangabe ausgeführt werden.	④ Geben Sie den Pfad zur Batch-Datei an.
3	RETURN wurde ohne GOSUB ausgeführt	Die GOSUB-Anweisung fehlt, obwohl eine RETURN-Anweisung existiert.	Definieren Sie eine GOSUB-Anweisung oder löschen Sie die RETURN-Anweisung.
4	zu viele GOTO,GOSUB oder Sprungmarken	① Das Programm enthält insgesamt 448 oder mehr GOTO- und GOSUB-Anweisungen.	① Reduzieren Sie die Anzahl der GOTO- bzw. GOSUB-Anweisungen.
		② Das Programm enthält 410 oder mehr Labels.	② Reduzieren Sie die Anzahl der Label.

Nummer	Meldung	Bedeutung	Maßnahme
5	zu großer numerischer Wert	① Die Parametereingabe ist zu lang.	
		② Die eingegebene Zahl überschreitet den festgelegten Bereich.	
		③ Das Argument ist anomal.	
		④ Es wurde ein nicht definierter Parameter verwendet.	
		⑤ Es wurde ein nicht definierter Befehl PALET verwendet.	
		⑥ Eine Null-Zeichenkette wurde definiert.	
6	zu viele Variablen oder falscher Wert	① Zahlen- oder Variablenüberlauf.	① Reduzieren Sie die Zahlen oder Variablen.
		② Die angegebene Adresse ist nicht definiert.	
7	zu viele GOSUB-RETURN Ebenen geöffnet	① Ein GOSUB...RETURN-Programm enthält zu viele Verschachtelungen.	① Reduzieren Sie die Verschachtelungen. Maximal sind 10 GOSUB..RETURN-Verschachtelungen erlaubt.
		② Das gespeicherte Quellprogramm ist zu groß.	② Lesen Sie die Informationen zum Befehl PRGSIZE.
8	fehlendes Ziel bei GOTO oder GOSUB	Eine über GOTO oder GOSUB aufgerufene Zeile existiert nicht.	Korrigieren Sie das Programm.
9	Feldvariable außerhalb der Definition	Der Index einer Datenfeldvariablen überschreitet die festgelegte Größe.	
10	Variable fehlerhaft definiert	① Der Variablen-, Label- oder Funktionname existiert bereits.	① Geben Sie einen anderen Namen an.
		② Ein reserviertes Wort wurde als Name für eine Variable, ein Label oder eine Funktion verwendet.	② Beachten Sie, daß reservierte Wörter nicht verwendet werden dürfen.
		③ Der Name für eine Variable, ein Label oder eine Funktion beginnt mit dem Buchstaben P.	③ Der Anfangsbuchstabe des Namens sollte nicht mit einem P beginnen.

Nummer	Meldung	Bedeutung	Maßnahme
11	Division durch 0/ undefinierte Palette	① Division durch Null (0).	Korrigieren Sie das Programm.
		② Die verwendete PALET-Funktion wurde nicht definiert.	
12	Befehl und Anweisung wurden vertauscht	Ein Befehl wurde als Anweisung verwendet oder umgekehrt eine Anweisung wurde als Befehl verwendet.	Korrigieren Sie das Programm.
13	Klammern links und rechts ungleich	Die Anzahl der Klammern auf der linken Seite "(" und auf der rechten Seite ")" stimmt nicht überein.	
14	Anzahl der Parameter nicht korrekt	Die Anzahl der Parameter stimmt nicht überein.	
15			
16	Programm ausserhalb FUNKTION-FEND	Eine Programmzeile befindet sich nicht innerhalb FUNCTION...FEND.	Achten Sie darauf, daß Sie die Programmzeilen nur zwischen FUNCTION und FEND einfügen.
17	FUNCTION innerhalb FUNCTION-FEND	Zwischen FUNCTION...FEND existiert eine mit FUNCTION definierte Funktion.	Definieren Sie FEND, so daß die FUNCTION-Funkion FEND entspricht.
18	Variable nicht am Zeilenanf. deklariert	Die Variablendefinition befindet sich nicht am Beginn einer Anweisungszeile	Nach Mehrfach-Anweisungen ist eine Variablendefinition nicht erlaubt. Wenn der Variablentyp unterschiedlich ist, müssen Sie die Anweisungszeile ändern.
19	80 oder mehr Zeichen in einer Zeile	Die Anzahl von Zeichen in einer Zeile beträgt mehr als 80.	In einer Zeile dürfen maximal 79 Zeichen stehen.
20	zu viele Verschachtelungen	Ein strukturiertes Programm enthält zu viele Verschachtelungsschleifen.	Reduzieren Sie die Anzahl der Verschachtelungs-schleifen. Es sind maximal 40 erlaubt.
21	Variabl. unterschiedlich (gross zu klein)	Beim Konvertieren einer Variablen erfolgte ein Datenüberlauf.	
22	Parameter ausserhalb erlaubter Grenzen	Die Tabelle der Parameterdefinitionen überschreitet den zulässigen Bereich.	Die erlaubte Zahl für den Befehl PALET liegt zwischen 0 und 15.
23	Zeile zu lang (nach COMPILE)	Zu viele Zeichen in einer Zeile (Zwischencode)	

Nummer	Meldung	Bedeutung	Maßnahme
24	Vermischung von Buchstaben und Zahlen	① Es wurde versucht, eine aus Zeichen und numerischen Werten bestehende Operation auszuführen.	
		② Daten-Tag-Fehler im Objektprogramm	
25	Umsetzungsfehler bei nummerischen Werten	① Fehler bei numerischer Variablen. Es wurden zu viele Ziffern eingegeben.	
		② Ein ASCII-Zeichen in einer INPUT-Anweisung konnte nicht in eine Ziffer konvertiert werden.	
26	Robotbefehl in Task 2-16, SELRB falsch	① Die angegebene Adresse existiert nicht.	① Die höchste Adreßangabe ist 19.
		② Ein Befehl zur Armbewegung wurde als Task 2 bis 16 ohne SELRB-Erklärung ausgeführt.	② Geben Sie die Task zur Armbewegung über den Befehl SELRB an.
27	angewählter E/A existiert nicht	① Der angegebene Eingang existiert nicht.	① Überprüfen Sie die Angabe des Eingangs.
		② Es wurde versucht, auf eine nicht geöffnete Datei zuzugreifen.	② Öffnen Sie die Datei bzw. überprüfen Sie den angegebenen Dateinamen.
28	NEXT, WEND oder SEND fehlt	① Die zur FOR-Anweisung gehörende NEXT-Anweisung fehlt.	① Geben Sie eine NEXT-Anweisung an.
		② Die zur WHILE-Anweisung gehörende WEND-Anweisung fehlt.	② Geben Sie eine WEND-Anweisung an.
		③ Die zur SELECT-Anweisung gehörende SEND-Anweisung fehlt.	③ Geben Sie eine SEND-Anweisung an.
		④ Die zur IF-Anweisung gehörende ENDIF-Anweisung fehlt.	④ Geben Sie eine ENDIF-Anweisung an.
		⑤ Die zur #ifdef-Anweisung gehörende #endif-Anweisung fehlt.	⑤ Geben Sie eine #endif-Anweisung an.
29	DMERG und DLOAD in Task 2-16 verwendet	Ein DMERGE- oder DLOAD-Befehl wurde als Task 2 bis 16 ausgeführt.	DMERGE- bzw. DLOAD-Befehle dürfen nur als Task 1 ausgeführt werden.

Nummer	Meldung	Bedeutung	Maßnahme
30	INPUT: Anzahl Daten/ Variablen ungleich	Die Anzahl der empfangenen Daten und die der Variablen für den Befehl INPUT stimmen nicht überein.	
31	Terminal-Kommunikation gestört	① Eine Kommunikation mit einem an die TEACH-Schnittstelle angeschlossenen PC ist nicht möglich.	① Überprüfen Sie die Kabelanschlüsse und aktivieren Sie die TEACH-Schnittstelle.
		② An die RS-232C-Schnittstelle angeschlossenen Geräte können nicht kommunizieren.	② Überprüfen Sie die Kabelanschlüsse und die Konfiguration der RS-232C-Schnittstelle.
32			
33	RS-232C Fehler (Puffer-überlauf)	Speicherüberlauf (Die Kapazität des Pufferspeichers ist erschöpft, dennoch werden weiter Daten gesandt.)	
34	RS-232C Fehler (parity/overrun/framing)	Bei der Kommunikation über die RS-232C-Schnittstelle erfolgte ein Paritäts-, Überlauf- oder Framing-Fehler.	
35			
36	RS-232C Fehler (mehr als 79 Zeichen)	Daten mit mehr als 80 Zeichen werden an die RS-232C-Schnittstelle gesandt.	Eine Zeile darf maximal 79 Zeichen enthalten.
37	RS-232C Fehler (overtime)	Zeitfehler bei der seriellen Kommunikation	
38			
39	Prog. enthält mehr als 70 FUNCTION-FEND	Die Anzahl der FUNCTION...FEND-Funktion beträgt mehr als 70.	Reduzieren Sie die Anzahl auf maximal 70.
40	aufgerufener Task existiert nicht	Die angegebene Task existiert nicht.	
41	aufgerufener Task ist bereits gestartet	Eine bereits gestartete Task kann nicht erneut gestartet werden.	
42	Speicher (wegen Fehlfunktion) zu klein	Ein Befehl kann aufgrund fehlender Speicher-kapazität oder einem Speicherfehler nicht ausgeführt werden.	
43			

Nummer	Meldung	Bedeutung	Maßnahme
44			
45	nicht erlaubte Anweisung ausgeführt	① Während eine Task ausgeführt wird, wurde ein nicht erlaubter Befehl ausgeführt.	① Drücken Sie die BREAK (STOP)-Taste oder betätigen Sie den RESET-Schalter.
		② Der Editiermodus wurde ausgewählt.	② Verlassen Sie den Editiermodus. (Lesen Sie dazu die Informationen zum EDIT-Befehl.)
46	24 Volt Anwenderstromversorgung gestört	Es liegt eine Störung bei der 24 Volt-Anwenderstromversorgung vor.	Überprüfen Sie den +24 Volt-Ausgang an der PSU-Einheit. Lesen Sie dazu die Informationen im Wartungshandbuch.
47			
48	kurzer Spannungs-einbruch/ Unterspannung	Momentaner Stromverlust oder Stromversorgungs-fehler.	
49	Batterie auf MPU-board ersetzen	Warnung - Zu niedrige Batteriespannung.	Sichern Sie sofort alle Daten, bevor sie verloren gehen. Lesen Sie dazu Abschnitt 3.6 im Handbuch zum SPEL-Editor. Tauschen Sie dann die Lithium-Batterie auf der MPU-Platine gegen eine neue aus. Lesen Sie dazu Abschnitt 4.8 im Wartungshandbuch.
50	falsch gewählter Befehlszusatz	Optionale Bezeichnung eines Befehls ungültig.	
51			
52			
53	Datei existiert nicht	Die angegebene Datei existiert nicht.	① Überprüfen Sie den Dateinamen.
			② Stellen Sie sicher, daß sich die angegebene Datei im Dateienspeicher befindet.
54			
55			
56			

Nummer	Meldung	Bedeutung	Maßnahme
57	Dateiname existiert bereits	Eine Datei mit dem angegebenen Namen existiert bereits.	Ändern Sie den Dateinamen oder speichern Sie die im Dateienspeicher befindliche Datei und löschen Sie sie anschließend aus dem Speicher.
58	Dateiname falsch oder existiert bereits	① Falscher Dateiname.	
		② Der Dateiname kann nicht mit Hilfe des NAME-Befehls geändert werden (Der Dateiname existiert bereits.)	
59			
60	kein freier Speicherplatz in Filememory	Keine freien Kapazitäten im Dateispeicher.	
61			
62	Datei kann nicht geöffnet werden	① Der Dateiname besteht aus mehr als 8 Zeichen.	
		② Datei kann nicht geöffnet werden.	
63	Diskette nicht bereit	Diskettenzugriff nicht möglich.	Stellen Sie sicher, daß die Diskette korrekt in das Diskettenlaufwerk eingelegt ist.
64			
65	Disketten-Lesefehler	Fehler beim Lesen der Diskette.	
66	Disketten-Schreibfehler	Fehler beim Schreiben auf die Diskette.	
67	falsches Disketten-laufwerk ausgewählt	Das falsche Disketten-laufwerk wurde ausgewählt.	Formatieren Sie die Diskette mit Hilfe des Befehls FORMAT.
68	schreibgeschützte Diskette	Versuch, Daten auf eine schreibgeschützte Diskette zu schreiben	
69			
70	File-Format ungültig	Ungültiges Dateiformat.	
71			
72	Prüfsummenfehler in Punktdaten	Prüfsummenfehler bei den Positionsdaten.	
73	Prüfsummenfehler im Programm	Prüfsummenfehler beim Quellprogramm.	
74	Prüfsummenfehler im Objektprogramm	Prüfsummenfehler beim Objektprogramm.	

Nummer	Meldung	Bedeutung	Maßnahme
75	undefinierte Variable oder Funktion	① Eine nicht definierte Variable oder Funktion wurde verwendet.	
		② Die angegebene Funktion wird nicht unterstützt.	
76	reservierter Befehl	Es wurden ungültige Befehle ausgeführt.	
77	Punktdefinition nicht korrekt	Falsche Punktnummer.	
78	undefinierter Punkt	Nicht definierte Positionsdaten sollen verwendet werden.	Definieren Sie den Punkt. Lesen Sie die Informationen zum Befehl PNTSIZE.
79			
80			
81			
82	interner Systemfehler	Systemfehler bei einem internen Vorgang.	Eliminieren Sie alle externen Störquellen. Lesen Sie Abschnitt 2.6 im Handbuch zur Steuerung. Sollte der Fehler häufiger auftreten, reparieren Sie die MPU-Platine oder tauschen Sie sie aus. Lesen Sie Abschnitt 4.8 im Wartungshandbuch.
83	zu viele Variablen verwendet	① Das Programm enthält 400 oder mehr Variablen.	① Es dürfen maximal 399 Variablen verwendet werden.
		② Es wurden zuviel Backup-Variablen angegeben.	② Lesen Sie die Informationen zum Befehl LIBSIZE.
84	Objekt-Datei nach COMPILE zu gross	Das kompilierte Objekt-programm ist zu groß.	Reduzieren Sie die Programmgröße.
85	Restore File ist zu lang.	Die über den Befehl RESTORE kopierte Datei ist zu groß.	
86	Speichertest-Fehler in Systemebene	Speicherprüffehler im Stystemarbeitsbereich.	Initialisieren Sie die MPU-Platine.
87	Speichertest-Fehler im File-Memory	Prüfsummenfehler bei der Datei.	Löschen Sie die fehlerhafte Datei über den Befehl DEL. Sollte der Fehler häufiger auftreten, müssen Sie die MPU-Platine auswechseln. Lesen Sie Abschnitt 4.8 im Wartungshandbuch.

Nummer	Meldung	Bedeutung	Maßnahme
88	I/O-bit ist bereits ein Remotefunktion	Der Eingang ist als REMOTE3 zugeordnet.	
89	Kommunikation I/O Board / CPU gestört	Kommunikationsfehler an Platine.	Führen Sie den RESET-Befehl aus.
90	Befehl in Parallel-processing unzulässig	In einer Parallelanweisung wurden ungültige Anweisungen verwendet.	
91	D-Parameter von internem Prozess fehlt	Bei einem internen Vorgang wurde kein D-Parameter ausgegeben.	
92	Türsicherheitskreis im Roboter gestört	Fehlfunktion in der Sicherungsleitung	
93	falsche Anzahl von D-Parametern	Die Anzahl der Parameter für den Befehl ist ungültig.	Die Anzahl der D-Parameter sollte maximal 5 betragen.
94	Wartezeit WAIT SW überschritten	Eine im WAIT SW-Befehl angegebene Bedingung wurde nicht in der festgelegten Zeit erfüllt.	
95	5 Volt Encoderstromversorgung gestört	5V für den Encoder sind nicht ordnungsgemäß.	Überprüfen Sie den +12 V-Ausgang an der PSU-Einheit. Lesen Sie Abschnitt 4.4 im Wartungshandbuch.
96	I/F board Kommuni-kationsfehler	Schnittstellenfehler.	
97	Speichertest-Fehler Systemparameter	Speicherfehler der Systemsteuerungs-parameter.	Eliminieren Sie alle externen Störquellen (s. Abschnitt 2.6 des Steuerunghandbuchs) und führen Sie VERINIT aus. Überprüfen Sie die Spannung der Lithium-Batterie auf der MPU-Platine. Tauschen Sie die Batterie aus, wenn die Spannung niedriger als 3,4 Volt ist. Sollte dieser Fehler häufiger auftreten, müssen Sie die MPU-Platine reparieren bzw. austauschen. Lesen Sie Abschnitt 4.8 im Wartungshandbuch.
98	Prüfsummenfehler in RAIOC	Parameterprüfsummen-fehler des RAIOC.	
99	Speichertestfehler-PROM	Speicherprüffehler des PROM.	Tauschen Sie das PROM aus.

Nummer	Meldung	Bedeutung	Maßnahme
100	Kommunikationsfehler (RAIOC etc.)	Gerätekommunikations-fehler (RAIOC etc.).	Überprüfen Sie die Geräteadresse, die Anschlüsse bzw. die MAXDEV-Einstellung.
101	Hardware-Fehler auf MPU-board	MPU-Fehler durch eine Fehlfunktion der Hardware.	Eliminieren Sie alle externen Störquellen (s. Abschnitt 2.6 des Steuerungshandbuchs) und führen Sie VERINIT aus. Sollte dieser Fehler häufiger auftreten, müssen Sie die MPU-Platine austauschen. Lesen Sie Abschnitt 4.8 im Wartungshandbuch.
102	Fehler in Stromversor-gung DC 7V o.DC 24V	Falsche Spannung des DC 7V bzw. DC 24V.	Überprüfen Sie den +12 Volt-Ausgang an der PSU-Einheit. Lesen Sie Abschnitt 4.4 des Wartungshandbuchs.
103	interner Systemfehler	Fehlfunktion der Robotersteuerung.	Eliminieren Sie externe Störquellen (s. Abschnitt 2.6 des Steuerungs-handbuchs). Sollte dieser Fehler häufiger auftreten, müssen Sie die MPU-Platine austauschen. Lesen Sie Abschnitt 4.8 im Wartungshandbuch.
104	Fehler in Stromversorgung DC 12V	Falsche Spannung von 12V.	Überprüfen Sie den ±12 V-Ausgang. Lesen Sie Abschnitt 4.4 des Wartungshandbuchs.
105	interner Kommunikationsfehler	Kommunikationsfehler bei internem Prozeß.	Eleminieren Sie externe Störquellen (s. Abschnitt 2.6 des Steuerungs-handbuchs). Sollte dieser Fehler häufiger auftreten, müssen Sie die MPU-Platine austauschen. Lesen Sie Abschnitt 4.8 im Wartungshandbuch.
106	interner Fehler (Buffer overflow)	Überlauf des Kommunikationsspeichers bei internem Prozeß.	Eleminieren Sie externe Störquellen (s. Abschnitt 2.6 des Steuerungs-handbuchs). Sollte dieser Fehler häufiger auftreten, müssen Sie die MPU-Platine austauschen. Lesen Sie Abschnitt 4.8 im Wartungshandbuch.

Nummer	Meldung	Bedeutung	Maßnahme
107	Prüfsummenfehler Robotermodell	Prüfsummenfehler bei den Robotermodelldaten.	Schalten Sie die Steuerung aus und wieder ein. Wird das Problem dadurch nicht behoben, wenden Sie sich an einen autorisierten EPSON-Händler.
108	Speichtestfehler	Fehler bei Speicherüberprüfung bei internem Prozeß.	Überprüfen Sie die Spannung der Lithium-Batterie auf der MPU-Platine. Tauschen Sie die Batterie aus, wenn die Spannung niedriger als 3,4 Volt ist und initialisieren Sie die Steuerung. Sollte dieser Fehler häufiger auftreten, müssen Sie die MPU-Platine reparieren bzw. austauschen. Lesen Sie Abschnitt 4.8 im Wartungshandbuch.
109	ungültiger Parameter	Anomaler Parameter, kann nicht angezeigt werden.	Schalten Sie die Steuerung aus und wieder ein und führen Sie den Befehl VERINIT aus. Sollte der Fehler danach nicht behoben sein, tauschen Sie die MPU-Platine aus. Lesen Sie Abschnitt 4.8 im Wartungshandbuch.
110	Fehler in Stromversorgung DC 24 V	Falsche Spannung von DC 24 V.	Überprüfen Sie den +24 V-Ausgang an der PSU-Einheit. Lesen Sie Abschnitt 4.4 des Wartungshandbuchs.
111	Coprozessor Fehler (falscher Code)	Koprozessor-Fehler (Codefehler).	Schalten Sie die Steuerung aus und wieder ein und führen Sie den Befehl VERINIT aus. Sollte der Fehler danach nicht behoben sein, tauschen Sie die MPU-Platine aus. Lesen Sie Abschnitt 4.8 im Wartungshandbuch.

Nummer	Meldung	Bedeutung	Maßnahme
112	Coprozessor Fehler (Überlauf)	Koprozessor-Fehler (Überlauf).	Schalten Sie die Steuerung aus und wieder ein und führen Sie den Befehl VERINIT aus. Sollte der Fehler danach nicht behoben sein, tauschen Sie die MPU-Platine aus. Lesen Sie Abschnitt 4.8 im Wartungshandbuch.
113	Coprozessor Fehler (Unterlauf)	Koprozessor-Fehler (Bereichsunterschreitung).	Schalten Sie die Steuerung aus und wieder ein und führen Sie den Befehl VERINIT aus. Sollte der Fehler danach nicht behoben sein, tauschen Sie die MPU-Platine aus. Lesen Sie Abschnitt 4.8 im Wartungshandbuch.
114	Coprozessor Fehler (Division durch 0)	Koprozessor-Fehler (Division durch Null).	Schalten Sie die Steuerung aus und wieder ein und führen Sie den Befehl VERINIT aus. Sollte der Fehler danach nicht behoben sein, tauschen Sie die MPU-Platine aus. Lesen Sie Abschnitt 4.8 im Wartungshandbuch.
115	Spannung/Temp.-Fehler in Motorpower	Unerlaubte Spannung oder Temperatur im Motor der Stromversorgungs-einheit.	Überprüfen Sie den Lüftungsventilator (reinigen Sie den Filter). Lesen Sie Abschnitt 4.12 des Wartungshandbuchs. Überprüfen Sie den Motorstromkreis. Lesen Sie Abschnitt 4.6 des Wartungshandbuchs.
116	Fehlfunktion Servo CPU dual port RAM	Fehlfunktion des Dual-Port-RAM-Speichers auf der Servo CPU.	Eleminieren Sie externe Störquellen (s. Abschnitt 2.6 des Steuerungs-handbuchs). Sollte dieser Fehler häufiger auftreten, müssen Sie die MPU-Platine austauschen. Lesen Sie Abschnitt 4.8 im Wartungshandbuch.
117			

Nummer	Meldung	Bedeutung	Maßnahme
118	Fehler in Stromversorgung DC-5V	Falsche Spannung bei DC 5 V.	Überprüfen Sie den -12 V-Ausgang an der PSU-Einheit. Lesen Sie Abschnitt 4.4 des Wartungshandbuchs.
119	<Hinweis>Arm wurde nach Poweroff bewegt	Der Manipulatorarm wurde nach Abschalten der Stromzufuhr zur Steuerung zu stark bewegt.	Schalten Sie die Steuerung aus und wieder ein oder führen Sie den Befehl RESET aus. Überprüfen Sie die Positionsdaten des Manipulatorarms. Sollten die Daten zu unterschiedlich sein, kalibrieren Sie jede Achse. Lesen Sie Abschnitt 3.12 des Wartungshandbuchs.
120	reservierter Befehl	Es wurden ungültige Befehle ausgeführt.	
121	Befehl während NotAus nicht ausführbar	Im Zustand des Not-Aus wurde ein ungültiger Befehl ausgeführt.	Heben Sie den Status des Not-Aus auf. (Einer von TEACH, REMOTE1 oder REMOTE2.)
122	Falscher Datentyp wird benutzt	Ein ungültiger Datentyp wurde verwendet.	
123	Befehl wird nicht unterstützt	Der angegebene Befehl wird nicht unterstützt.	
124	Nummerischer Wert Bereichsüberschreitung	Der numerische Wert ist außerhalb des zulässigen Bereichs.	
125	Arm erreicht Grenzen des Arbeitsbereiches	Der Manipulatorarm hat die Grenze des Verfahrbereichs erreicht.	
126	Arm erreicht XYLIM Grenzen	Der Manipulatorarm hat die Grenze des über XYLIM festgelegten Verfahrbereichs erreicht.	
127	gewählter Arm ist nicht definiert	Der über den Befehl ARM ausgewählte Zusatzarm ist nicht definiert.	① Verwenden Sie eine bereits definierte Armnummer ② Definieren Sie den Zusatzarm mit Hilfe des Befehls ARMSET.

Nummer	Meldung	Bedeutung	Maßnahme
128	gewähltes Tool ist nicht definiert	Das über den Befehl TOOL ausgewählte Werkzeug-Koordinaten-system ist nicht definiert.	① Verwenden Sie eine bereits definierte Nummer für ein Werkzeug-Koordinaten-system.
			② Definieren Sie das Werkzeug-Koordinaten-system mit Hilfe des Befehls TLSET.
129	LIMZ Fehler	LIMZ-Fehler.	Verkleinern Sie den gesetzten LIMZ-Wert für die aktuelle Position oder Zielposition.
130	unterschiedliche LOCAL Werte eingegeben	Es wurden unterschied-liche LOCAL-Attribute definiert.	Definieren Sie das lokale Koordinatensystem mit Hilfe des Befehls LOCAL.
131	gewähltes LOCAL ist nicht definiert	Das angegebene lokale Koordinatensystem ist nicht definiert.	① Geben Sie eine bereits definierte LOCAL-Nummer an.
			② Definieren Sie das lokale Koordinatensystem mit Hilfe des Befehls LOCAL.
132	HOFS Wert ausserhalb erlaubten Bereichs	Der HOFS-Wert liegt außerhalb des erlaubten Bereichs.	Der HOFS-Wert muß in einem Bereich von -40960 bis 40960 liegen.
133	nicht alle Achsen aktiv während HOME	Der HOME-Befehl wurde eingegeben, obwohl nicht alle Achsen eingeschaltet sind.	Schalten Sie alle Achsen mit Hilfe des SLOCK-Befehls ein, bevor Sie den HOME-Befehl ausführen.
134	Attribute wechseln in CP (L/R LOCAL)	Änderung eines Armattributs bei der Steuerung einer Bahnbewegung.	Bei der Steuerung einer Bahnbewegung können Armattribute nicht geändert werden.
135	SFREE ist für eine Achse gesperrt	Der Befehl SFREE wurde bei einer Achse aus-geführt, die nicht über diesen Befehl freige-schaltet werden kann.	Lesen Sie Abschnitt 9.2 des SPEL-Editor-Handbuchs.
136	Punkte bei CURVE unzureichend	Über den Befehl CURVE wurde eine nicht zulässige Anzahl (zuviel oder zu wenig) von Punktdaten definiert.	Lesen Sie die Informationen zum Befehl CURVE.
137	Attribute wechseln in CURVE (L/R LOCAL)	Die über den Befehl CURVE definierten Punktdaten enthalten einen Punkt mit abweichenden Armattributen.	Lesen Sie die Informationen zum Befehl CURVE.

Nummer	Meldung	Bedeutung	Maßnahme
138	CURVE kann nicht generiert werden	Eine freie Kurve kann nicht mit Hilfe des Befehls CURVE ausgeführt werden.	Stellen Sie sicher, daß zwei aufeinanderfolgende Punkte sich nicht überlappen.
139	Start nach QUICK-PAUSE nicht möglich	Nach einer Schnellpause wurde versucht, eine über den Befehl CVMOVE gesteuerte Bewegung erneut zu starten.	Eine über CVMOVE gesteuerte Bewegung kann nach einer Schnellpause nicht erneut gestartet werden.
140	nur 4.Achse soll in CP bewegt werden	Bei der Steuerung einer Bahnbewegung wurde versucht, nur die 4. Achse zu bewegen.	
141	ARC-Punktabstand zu gering/gerade Linie	Bei der Definition eines ARC-Befehls befinden sich zwei Punkte zu dicht beieinander oder auf einer geraden Linie.	Lesen Sie die Informationen zum Befehl ARC.
142			
143	HOME Position nicht definiert	Die HOME-Position ist nicht definiert.	Definieren Sie die HOME-Position mit Hilfe des HOMESET-Befehls.
144	Anzahl der Parameter falsch	Unerlaubte Anzahl von Parametern für einen Befehl.	
145	Fehler in CP-file	Fehler in der CVMOVE-Befehlsdatei.	
146	Bewegung unter SFREE nicht möglich	Obwohl eine oder mehrere Achsen freigeschaltet waren (SFREE-Befehl), wurde eine Bewegungsbefehl ausgeführt.	Schalten Sie alle Achsen mit Hilfe des Befehls SLOCK ein.
147	BASE0-Parameter 4 u. 6 sind nicht gleich	Der Wert des 4. und 6. Parameters im Befehl BASE 0 ist unterschiedlich.	Lesen Sie die Informationen zum Befehl BASE.
148			
149	Befehl unter MOTOR ON nicht ausführbar	Der Befehl kann nicht bei eingeschalteter Stromzufuhr zu den Motoren ausgeführt werden (MOTOR ON).	
150	Befehl unter MOTOR OFF nicht ausführbar	Der Befehl kann nicht bei ausgeschalteter Stromzufuhr ausgeführt werden (MOTOR OFF).	

Nummer	Meldung	Bedeutung	Maßnahme
151	Position innerhalb FINE nicht erreichbar	Mit Hilfe des angegebenen FINE-Befehls kann eine Positionierung nicht engültig ausgeführt werden.	Überprüfen Sie die Verbindung zum Motorstromkreis. Stellen Sie sicher, daß die Motorstromeinheit die korrekte Spannung an jeden AC-Servomotor ausgibt. Lesen Sie Abschnitt 4.6 des Wartungshandbuchs. Überprüfen Sie den festen Sitz aller Sicherungsbolzen des Manipulatorarms. Überprüfen Sie das Untersetzungsgetriebe. Lesen Sie Abschnitt 3.6 des Wartungshandbuchs. Tauschen Sie den Motor, den AC-Servomotor oder die MPU-Platine aus. Lesen Sie dazu das Wartungshandbuch.
152	Überdrehzahl	Die Armbewegung überschreitet die maximale Geschwin-digkeit bzw. Beschleu-nigung.	
153	Verzögerungsstrecke bei CP zu kurz	Die Bahnbewegung kann bei der angegebenen Entfernung nicht abgebremst und angehalten werden.	① Stellen Sie sicher, daß die Strecke zum Abbremsen ausreicht. ② Eine Bahnbewegung darf nicht ohne Abbremsen ausgeführt werden.
154	Drehbereich 4.Achse überschritten	Die Bewegung der 4. Achse überschreitet den zulässigen Bewegungsbereich.	Es gibt einen Grenzwert für die Bewegung der 4. Achse bei Manipulator-armen mit einer Kugelumlaufspindel-ZU-Achseneinheit
155	Kommunikationsfehler mit Servo-CPU	Kommunkationsfehler mit der Servo-CPU	
156	Fehlfunktion der Servo CPU	Fehlfunktion der Servo-CPU	

Nummer	Meldung	Bedeutung	Maßnahme
157	Vorgabegeschwindigkeit nicht eingehalten	Der Manipulator hat sich mit einer unzulässigen Geschwindigkeit bewegt.	Schalten Sie die Steuerung aus und wieder ein oder geben Sie den Befehl RESET ein. Überprüfen Sie die Positionsdaten des Manipulatorarms. Sollten die Daten zu unterschiedlich sein, kalibrieren Sie jede Achse. Sollte dieser Fehler häufiger auftreten, tauschen Sie die MPU-Platine aus. Lesen Sie Abschnitt 4.8 des Wartungshandbuchs.
158			
159	Lesefehler Encoder	Lesefehler bei den Umdrehungsdaten des Encoders.	
		① Es gibt eine externe Störquelle.	① Eliminieren Sie die externe Störquelle. Lesen Sie Abschnitt 2.6 des Steuerungshandbuchs.
		② Keine Verbindung der Encoder-Signalleitung.	② Überprüfen Sie die Verbindung der Signalleitung zwischen der Steuerungsrückseite und Mother-Board und Servomotor. Wenn die LED-Anzeige ALARM (rot) am Servomotor leuchtet, sollten Sie auch die Signalleitungen im Manipulatorarm und das Signalkabel überprüfen.
		③ Die Spannung der Encoder-Stromversorgung ist anormal.	③ Überprüfen Sie die Spannung am Signalanschluß des Motors. Ist die Spannung nicht korrekt, überprüfen Sie die PSU-Platine und die Verkabelung. Wenn die LED-Anzeige (grün) am Servomotor dunkel ist, überprüfen Sie die Verbindung zwischen Signalanschluß und Servomotor.

Nummer	Meldung	Bedeutung	Maßnahme
159 Forts.		④ Sonstiges	④ Versuchen Sie folgende Maßnahmen: Überprüfen Sie die Verbindung zur MPU-Platine. Tauschen Sie U15 und U16 (SCC) auf der MPU-Platine aus. Tauschen Sie die MPU-Platine aus. Tauschen Sie den Servomotor aus. Tauschen Sie den Motor aus.
160	Unterbrechung der Encoder A-Phase	Unterbrechung der A-Phase des Encoders.	Siehe Beschreibung unter Fehler Nummer 159
161	Unterbrechung der Encoder B-Phase	Unterbrechung der B-Phase des Encoders.	
162	Unterbrechung der Encoder Z-Phase	Unterbrechung der Z-Phase des Encoders	
163	Unterbrechung der Encoder S-Phase	Unterbrechung der S-Phase des Encoders	
164	Encoderkommunikation gestört	Zeichenfehler beim absoluten Encoder.	
165	<ACHTUNG> Encoder wurden initialisiert	<HINWEIS> Der Encoder wurde initialisiert.	① Dieser Code wird angezeigt, wenn der Encoder initialisiert wird und die Stromzufuhr zur Steuerung eingeschaltet wird. Schalten Sie die Stromzuguhr zweimal aus und wieder ein.
			② Sollte dies nicht zutreffen, überprüfen Sie, ob die RES-Verkabelung des Encoders einen Kurzschluß auf einer anderen Leitung verursacht hat.
166	Servo Berechnung Überlauf	Überlauf bei der Servo-Berechnung.	
167	Hardware Felfunktion der Servo-CPU	Fehlfunktion der Hardware der Servo-CPU.	Tauschen Sie die MPU-Platine aus oder reparieren Sie sie. Lesen Sie Abschnitt 4.8 des Wartungshandbuchs.
168	Kommunikationsfehler Servo-CPU mit MPU	Kommunikationsfehler der Servo-CPU mit der Sub-CPU.	

Nummer	Meldung	Bedeutung	Maßnahme
169	Servo Überdrehzahl	Zu hohe Servo-Geschwindigkeit.	① Stellen Sie sicher, daß das Signalkabel korrekt angeschlossen ist.
			② Überprüfen Sie den Anschluß der Encoder-Leitung im Manipulator-arm.
			③ Überprüfen Sie die Kabelverbindung zwischen der Steuerungsrückseite und Mother-Board und Servomotor.
			④ Eliminieren Sie die externe Störquelle. Lesen Sie Abschnitt 2.6 des Steuerungs-handbuchs.
			⑤ Tauschen Sie ansonsten die MPU-Platine aus. Lesen Sie Abschnitt 4.8 des Wartungshandbuchs.
170	Servo Überlauf	Servo-Überlauf.	Führen Sie die Maßnah-men ① bis (11) durch:
		① Problem bei der Motorstromleitung.	① Überprüfen Sie die Motorstromleitung im Manipulatorarm und in der Steuerung.
		② Problem bei der Encoder-Signalleitung	② Wenn die Anzeige ALARM (rot) am Servomotor leuchtet, sollten Sie die folgenden Signalleitungen überprüfen: - im Manipulatorarm - im Signalkabel - zwischen der Steuerungsrückseite und Mother-Board und AC-Servomotor.
		③ Die Ausgabe der Motorstromeinheit an den Servomotor ist nicht korrekt.	③ Untersuchen Sie die Motorstromeinheit. Lesen Sie Abschnitt 4.6 des Wartungshandbuchs.

Nummer	Meldung	Bedeutung	Maßnahme
170 Forts.		④ Eine der Achsen wurde mit einer zusätzlichen Last belastet.	④ Überprüfen Sie die Bewegung aller Achsen, indem Sie jede Achse von Hand bewegen. Stellen Sie sicher, daß die Schmierung ausreichend ist bzw. tauschen Sie das Untersetzungsgetriebe aus. Lesen Sie Abschnitt 1.3 und 3.6 des Wartungshandbuchs.
		⑤ Der Synchronriemen ist nicht ordnungsgemäß gespannt.	⑤ Überprüfen Sie die Spannung des Synchronriemens. Spannen Sie ihn bzw. tauschen Sie ihn ggf. aus. Lesen Sie Abschnitt 3.8 des Wartungshandbuchs.
		⑥ Der Manipulatorarm sitzt fest.	⑥ Eliminieren Sie Faktoren wie Hindernisse.
		⑦ Die Bremse der Z-Achse kann nicht gelöst werden.	⑦ Kann die Bremse im Zustand MOTOR OFF nicht mit Hilfe des Schalters zum Lösen der Bremse gelöst werden, überprüfen Sie die Verdrahtung der Bremse. Kann die Bremse im Zustand MOTOR ON nicht gelöst werden, überprüfen Sie den Signalstromkreis der Bremssteuerung.
		⑧ Der Manipulatorarm ist gegen ein Hindernis gestoßen.	⑧ Achten Sie darauf, daß der Manipulatorarm nicht gegen Hindernisse stößt.
		⑨ Der Manipulatorarm hat sich unerwartet bewegt.	⑨ Führen Sie die unter Punkt ② aufgeführten Maßnahmen durch. Handelt es sich nicht um eines dieser Probleme, tauschen Sie den AC-Servomotor bzw. die MPU-Platine aus. Lesen Sie Abschnitt 4.7 bzw. 4.8 des Wartungshandbuchs.

Nummer	Meldung	Bedeutung	Maßnahme
170 Forts.		⑩ Es existiert eine externe Störquelle.	⑩ Eliminieren Sie externe Störquellen. Lesen Sie dazu Abschnitt 2.6 des Steuerungshandbuchs.
		11 Sonstiges	11 Tauschen Sie die MPU-Platine bzw. den AC-Servomotor aus. Lesen Sie dazu das Wartungshandbuch.
171	Prüfsummenfehler Servo CPU	Prüfsummenfehler bei der Kommunikation der Servo-CPU.	Eliminieren Sie externe Störquellen. Lesen Sie Abschnitt 2.6 des Wartungshandbuchs.
172	Überlast (Drehmoment)	Anormales Drehmoment der Motors.	Überprüfen Sie die Unterbrechung der Stromversorgung.
173	reduzierte Motor-Power durch Low Power mode	Der Robotor befindet sich im Low-Power-Modus. Die Motorenleistung ist begrenzt.	
174	Motordrehmoment überschritten	Überlastung des Motordrehmoments.	
175	Hardware der Servo-CPU fehlerhaft	Fehlfunktion der Hardware in Verbindung mit der Servo-CPU.	
176			
177			
178			
179	In Servo Adjustment Mode	Der Servo-Adjustment-Modus ist ausgewählt.	
180	Überhitzung/Überstrom einer Treiberstufe	Überhitzung einer Treiberstufe	
		r Wenn die rote LED-Anzeige an der Vorderseite des AC-Servomotors nicht leuchtet.	
		① Es existiert eine externe Störquelle.	① Eliminieren Sie die externe Störquelle. Lesen Sie Abschnitt 2.6 des Wartungshandbuchs. Wenn keine externe Störquelle existiert, tauschen Sie die MPU-Platine aus. Lesen Sie Abschnitt 4.8 des Wartungshandbuchs.

Nummer	Meldung	Bedeutung	Maßnahme
180 Forts.		r Wenn die rote LED-Anzeige an der Vorderseite des AC-Servomotors leuchtet.	
		② Das Gewicht von Hand und Werkstück übersteigt die erlaubte Nutzlast.	② Überprüfen Sie das tatsächliche Gewicht und sellen Sie den korrekten Wert über den Befehl WEIGHT ein.
		③ Die Bewegungsleistung des Manipulators ist zu hoch.	③ Reduzieren Sie die Arbeitsgeschwindigkeit des Roboters. Lesen Sie dazu die Informationen zu den Befehlen ACCEL, SPEED und WEIGHT.
		④ Eine der Achsen wurde mit einer zusätzlichen Last belastet.	④ Überprüfen Sie die Bewegung aller Achsen, indem Sie jede Achse von Hand bewegen. Stellen Sie sicher, daß die Schmierung ausreichend ist bzw. tauschen Sie das Untersetzungsgetriebe aus. Lesen Sie Abschnitt 1.3 und 3.6 des Wartungshandbuchs.
		⑤ Der Synchronriemen ist nicht ordnungsgemäß gespannt.	⑤ Überprüfen Sie die Spannung des Synchronriemens. Spannen Sie ihn bzw. tauschen Sie ihn ggf. aus. Lesen Sie Abschnitt 3.8 des Wartungshandbuchs.
		⑥ Die Lüfterkühlung funktioniert nicht.	⑥ Untersuchen Sie die Lüfterkühlung. Tauschen Sie sie ggf. aus. Lesen Sie dazu Abschnitt 4.12 des Wartungshandbuchs.
		⑦ Der Filter ist verstopft.	⑦ Reinigen Sie den Filter.
		⑧ Die Umgebungs-temperatur der Steuerung beträgt mehr als 40 °C.	⑧ Sorgen Sie für kühlere Umgebungstemperaturen am Standort der Steuerung.
		⑨ Die Motorstromleitung ist nicht geerdet oder hat einen Kurzschluß.	⑨ Überprüfen Sie die Motorstromleitung im Manipulatorarm und in der Steuerung.

Nummer	Meldung	Bedeutung	Maßnahme
180 Forts.		⑩ Sonstiges	⑩ Tauschen Sie den AC-Servomotor aus. Lesen Sie dazu Abschnitt 4.7 des Wartungshandbuchs.
181	Überlast einer Treiberstufe	Überlastung einer Treiberstufe.	
		r Wenn die rote LED-Anzeige an der Vorderseite des AC-Servomotors nicht leuchtet. ① Es existiert eine externe Störquelle.	① Eliminieren Sie die externe Störquelle. Lesen Sie Abschnitt 2.6 des Wartungshand-buchs. Wenn keine externe Störquelle existiert, tauschen Sie die MPU-Platine aus. Lesen Sie Abschnitt 4.8 des Wartungshandbuchs.
		r Wenn die rote LED-Anzeige an der Vorderseite des AC-Servomotors leuchtet. ② Das Gewicht von Hand und Werkstück übersteigt die erlaubte Nutzlast.	② Überprüfen Sie das tatsächliche Gewicht und sellen Sie den korrekten Wert über den Befehl WEIGHT ein.
		③ Die Bewegungsleistung des Manipulators ist zu hoch.	③ Reduzieren Sie die Arbeitsgeschwindigkeit des Roboters. Lesen Sie dazu die Informationen zu den Befehlen ACCEL, SPEED und WEIGHT.
		④ Eine der Achsen wurde mit einer zusätzlichen Last belastet.	④ Überprüfen Sie die Bewegung aller Achsen, indem Sie jede Achse von Hand bewegen. Stellen Sie sicher, daß die Schmierung ausreichend ist bzw. tauschen Sie das Untersetzungsgetriebe aus. Lesen Sie Abschnitt 1.3 und 3.6 des Wartungshandbuchs.

Nummer	Meldung	Bedeutung	Maßnahme
181 Forts.		⑤ Der Synchronriemen ist nicht ordnungsgemäß gespannt.	⑤ Überprüfen Sie die Spannung des Synchronriemens. Spannen Sie ihn bzw. tauschen Sie ihn ggf. aus. Lesen Sie Abschnitt 3.8 des Wartungshandbuchs.
		⑥ Der Manipulatorarm sitzt fest.	⑥ Eliminieren Sie Faktoren wie Hindernisse.
		⑦ Die Bremse der Z-Achse kann nicht gelöst werden.	⑦ Kann die Bremse im Zustand MOTOR OFF nicht mit Hilfe des Schalters zum Lösen der Bremse gelöst werden, überprüfen Sie die Verdrahtung der Bremse. Kann die Bremse im Zustand MOTOR ON nicht gelöst werden, überprüfen Sie den Signalstromkreis der Bremssteuerung.
		⑧ Der Manipulatorarm ist gegen ein Hindernis gestoßen.	⑧ Achten Sie darauf, daß der Manipulatorarm nicht gegen Hindernisse stößt.
		⑨ Die Motorstromleitung ist nicht geerdet oder hat einen Kurzschluß.	⑨ Überprüfen Sie die Motorstromleitung im Manipulatorarm und in der Steuerung.
		⑩ Sonstiges	⑩ Tauschen Sie den AC-Servomotor aus. Lesen Sie dazu Abschnitt 4.7 des Wartungshandbuchs.
182	Überstrom einer Treiberstufe	Die Treiberstufe hat eine Überdrehzahlfestgestellt.	
183	Motorwelle mechanisch blockiert	Die Treiberstufe hat eine blockierte Motorwelle festgestellt.	
		r Wenn die rote LED-Anzeige an der Vorderseite des AC-Servomotors nicht leuchtet.	

Nummer	Meldung	Bedeutung	Maßnahme
183 Forts.		① Es existiert eine externe Störquelle.	① Eliminieren Sie die externe Störquelle. Lesen Sie Abschnitt 2.6 des Wartungshandbuchs. Wenn keine externe Störquelle existiert, tauschen Sie die MPU-Platine aus. Lesen Sie Abschnitt 4.8 des Wartungshandbuchs.
		r Wenn die rote LED-Anzeige an der Vorderseite des AC-Servomotors leuchtet. ② Die Motorstromleitung ist nicht geerdet oder hat einen Kurzschluß.	② Überprüfen Sie die Motorstromleitung im Manipulatorarm und in der Steuerung.
		③ Eine der Achsen wurde mit einer zusätzlichen Last belastet.	③ Überprüfen Sie die Bewegung aller Achsen, indem Sie jede Achse von Hand bewegen. Stellen Sie sicher, daß die Schmierung ausreichend ist bzw. tauschen Sie das Untersetzungsgetriebe aus. Lesen Sie Abschnitt 1.3 und 3.6 des Wartungshandbuchs.
		④ Der Synchronriemen ist nicht ordnungsgemäß gespannt.	④ Überprüfen Sie die Spannung des Synchronriemens. Spannen Sie ihn bzw. tauschen Sie ihn ggf. aus. Lesen Sie Abschnitt 3.8 des Wartungshandbuchs.
		⑤ Der Manipulatorarm sitzt fest.	⑤ Eliminieren Sie Faktoren wie Hindernisse.
		⑥ Die Bremse der Z-Achse kann nicht gelöst werden.	⑥ Kann die Bremse im Zustand MOTOR OFF nicht mit Hilfe des Schalters zum Lösen der Bremse gelöst werden, überprüfen Sie die Verdrahtung der Bremse. Kann die Bremse im Zustand MOTOR ON nicht gelöst werden, überprüfen Sie den Signalstromkreis der Bremssteuerung.

Nummer	Meldung	Bedeutung	Maßnahme
183 Forts.		⑦ Der Manipulatorarm ist gegen ein Hindernis gestoßen.	⑦ Achten Sie darauf, daß der Manipulatorarm nicht gegen Hindernisse stößt.
		⑧ Sonstiges	⑧ Tauschen Sie den AC-Servomotor aus. Lesen Sie dazu Abschnitt 4.7 des Wartungshandbuchs.
184	Treiberstufe meldet fehlerhafte Bewegung	Die Treiberstufe hat eine unerlaubte Bewegung festgestellt.	
		r Wenn die rote LED-Anzeige an der Vorderseite des AC-Servomotors nicht leuchtet. ① Es existiert eine externe Störquelle.	① Eliminieren Sie die externe Störquelle. Lesen Sie Abschnitt 2.6 des Wartungshand-buchs. Wenn keine externe Störquelle existiert, tauschen Sie die MPU-Platine aus. Lesen Sie Abschnitt 4.8 des Wartungshandbuchs.
		r Wenn die rote LED-Anzeige an der Vorderseite des AC-Servomotors leuchtet. ② Problem bei der Encoder-Signalleitung	② Überprüfen Sie die folgenden Signal-leitungen: - im Manipulatorarm - im Signalkabel - zwischen der Steuerungsrückseite und Mother-Board und AC-Servomotor.
		③ Sonstiges	③ Tauschen Sie den Motor bzw. den AC-Servomotor aus. Lesen Sie dazu Abschnitt 35. bzw. 4.7 des Wartungshandbuchs.
185	Unterbrechung Encodersignal	Unterbrechung des Encodersignals.	Siehe Fehler Nummer 184
186	Fehlfunktion der Treiber CPU	Fehlfunktion der Treibersufen-CPU.	

Nummer	Meldung	Bedeutung	Maßnahme
186 Forts.		r Wenn die rote LED-Anzeige an der Vorderseite des AC-Servomotors nicht leuchtet.	
		① Es existiert eine externe Störquelle.	① Eliminieren Sie die externe Störquelle. Lesen Sie Abschnitt 2.6 des Wartungshandbuchs. Wenn keine externe Störquelle existiert, tauschen Sie die MPU-Platine aus. Lesen Sie Abschnitt 4.8 des Wartungshandbuchs.
		r Wenn die rote LED-Anzeige an der Vorderseite des AC-Servomotors leuchtet.	
		② Es gibt ein Problem bei der Kabelverbindung zwischen AC-Servomotor und Mother-Board.	② Überprüfen Sie diese Kabelverbindung.
		③ Es liegt ein Problem beim +24 Volt-Ausgang an der PSU-Einheit vor.	③ Stellen Sie fest, ob die POWER LED (grün) an der Vorderseite des AC-Servomotors leuchtet oder nicht. Leuchtet die LED nicht, überprüfen Sie den +24 Volt-Ausgang an der PSU-Einheit. Lesen Sie dazu die Informationen in Abschnitt 4.4 des Wartungshandbuchs.
		④ Der Synchronriemen ist nicht ordnungsgemäß gespannt.	④ Tauschen Sie den AC-Servomotor aus. Lesen Sie dazu Abschnitt 4.7 des Wartungshandbuchs.
187			
188		Die Treiberstufe hat ein unzlässiges Drehmoment/eine unzulässige Geshwindigkeit festgestellt.	
189		Das S-Phasen-Signal des Encoders wurde nicht in der zulässigen Zeit übertragen.	
190	Encoder überhitzt	Der Encoder ist überhitzt.	

Nummer	Meldung	Bedeutung	Maßnahme
191	Encoderüberdrehzahl	Die Drehzahl des Encoders ist zu hoch.	Überprüfen Sie, ob sich eine der Achsen beim Einschalten der Stromzufuhr bewegt. Wenn sich die Z-Achse bewegt, überprüfen Sie die Bremse dieser Achse.
192	Fehlfunktion eines Absolut-Sensors	Encoder-Datenfehler.	
193	Batteriefehler (Verteilerplatine)	Encoder-Batteriefehler.	① Überprüfen Sie die Spannung der Batterie auf der Signalrelais-Platine. Liegt die Spannung unter 2,8 V, tauschen Sie die Batterie aus.
			② Überprüfen Sie die Verbindung zwischen Motorencoder und Signalrelais-Platine.
194	Encoder Prüfsummenfehler	Prüfsummenfehler beim Encoder.	Kalibrieren Sie die Achse. Läßt sich das Problem dadurch nicht beheben, tauschen Sie den Motor aus. Lesen Sie dazu Abschnitt 3.5 des Wartungshandbuchs.
195	Encoder Back-Up Fehler	Backup-Fehler beim Encoder.	
		① Dieser Fehler wird nach dem Austausch eines Motors angezeigt.	① Lesen Sie Abschnitt 3.12 des Wartungshandbuchs.
		② Die Batteriespannung auf der Signalrelais-Platine ist nicht korrekt.	② Wenn die Spannung zu niedrig ist, tauschen Sie die Batterie aus.
		③ Die Spannung der Encoder-Stromversorgung (ENC +5 V) ist nicht korrekt.	③ Überprüfen Sie die Spannung am Signalanschluß in unmittelbarer Nähe des Motors. Lesen Sie dazu Abschnitt 4.5 des Wartungshandbuchs. Ist die Spannung zu niedrig oder nicht korrekt, überprüfen Sie die PSU-Platine und die Verkabelung.
		④ Sonstiges	④ Tauschen Sie den Motor aus. Lesen Sie dazu Abschnitt 3.5 des Wartungshandbuchs.

Nummer	Meldung	Bedeutung	Maßnahme
196			
197	Encoderdaten-Fehler (framing)	Framing-Fehler der Encoder-Daten.	
		① Dieser Fehler wird nach dem Austausch eines Motors angezeigt.	① Lesen Sie Abschnitt 3.12 des Wartungshandbuchs.
		② Es existiert eine starke externe Störungsquelle.	② Eliminieren Sie die externe Störquelle. Lesen Sie Abschnitt 2.6 des Wartungshandbuchs.
		③ Sonstiges	③ Versuchen Sie die folgenden Maßnahmen: Überprüfen Sie den Zustand der MPU-Platine. Tauschen Sie die MPU-Platine aus. Lesen Sie dazu Abschnitt 4.8 des Wartungshandbuchs. Tauschen Sie den Motor aus. Lesen Sie dazu Abschnitt 3.5 des Wartungshandbuchs.
198	Encoderdaten-Fehler (Überlauf)	Überlauffehler der Encoder-Daten.	
199	Encoderdaten-Fehler (Parität)	Paritätsfehler der Encoder-Daten.	
200			
230		Der Befehl MCORG wurde nicht ausgeführt.	
231		Der Befehl MCAL wurde nicht ausgehührt.	
232		Fehler bei der Home-Sensor-Erfassung.	Bewegen Sie den Manipulatorarm manuell und überprüfen Sie dabei, ob die Home-Sensor-LED an der Rückseite des Manipulatorarms aufleuchtet oder nicht. Leuchtet die LED nicht auf, führen Sie den Befehl MCORG aus. Leuchtet die LED auf, überprüfen Sie das Signalkabel.
233		Fehler bei der Erfassung des Encoder-Z-Phasen-Signals.	① Überprüfen Sie die Signalleitung der Z-Phase.

Nummer	Meldung	Bedeutung	Maßnahme
233 Forts.			② Liegt das Problem nicht bei der Leitung, tauschen Sie den Motor aus. Lesen Sie dazu Abschnitt 3.5 des Wartungshandbuchs.
234		Die Ausführung des Befehls MCORG wurde durch eine Schnellpause unterbrochen.	
235		Falsche Einstellung der Achse für die Kugelumlaufspindel.	Schalten Sie Bit 7 und 8 von SD2 auf der MPU-Platine ein.
236		Fehler bei der Erfassung der Encoder-Z-Phasen-Position.	Verbinden Sie den Manipulatorarm und die Steuerung mit demselben M.CODE. Ist der M.CODE für beide identisch, wenden Sie sich an einen autorisierten EPSON-Händler (halten Sie dabei die über den Befehl VER erhaltenen Informationen bereit).
237		Die HTEST-Daten sind unzulässig.	① Überprüfen Sie den M.CODE von Manipulatorarm und Steuerung. Sind die Werte unterschiedlich, verbinden Sie beide mit demselben M.CODE.
			② Hat sich Achse #4 in ausgeschaltetem Zustand mehr als 180 ° gedreht, schalten Sie die Stromzufuhr zur Steuerung aus und und bringen Sie Achse #4 zurück in die vorherige Position.
			③ Hat der Manipulatorarm einen Stoß erhalten, hat sich die Position u.U. mechanisch verschoben. Führen Sie den Befehl MCORG aus.
238			
239			
300	Unterverzeichnis existiert nicht	Das angegebene Verzeichnis exisitiert nicht.	

Nummer	Meldung	Bedeutung	Maßnahme
301	Verzeichnis existiert nicht	① Das angegebene Verzeichnis kann nicht gelöscht werden, da es nicht existiert.	
		② Das angegebene Verzeichnis kann nicht gelöscht werden, da es noch Dateien enthält.	
302	Verzeichnis kann nicht angelegt werden	Das Verzeichnis kann nicht angelegt werden.	① Das angegebene Verzeichnis exisitiert bereis.
			② Die Festplatte/Diskette ist voll.
303	Verzeichnis oder Datei existiert nicht	Das angegebene Verzeichnis/die angegebene Datei exisitiert nicht.	
304	Eingabe für DATE falsch	Die Eingabe zum DATE-Befehl ist nicht correkt.	
305	Eingabe für TIME falsch	Die Eingabe zum TIME-Befehl ist nicht korrekt.	
306	Gewähltes Laufwerk existiert nicht	Das angegebene Laufwerk exisitert nicht.	
307	Speicher für enviroment string zu klein	Nicht genügend Speicher für die Umgebungszeichenkette.	Der maximale Speicher für die Umgebungszeichenkette beträgt 512 Byte.
308	Speicher für Batch file zu klein	Die Stapeldatei ist zu groß.	Der maximale Speicher für die Stapeldatei beträgt 4 kByte.
309	File soll auf sich selbst kopiert werden	Eine Datei kann nicht auf sich selbst kopiert werden.	
310			