

SIEMENS

SINUMERIK

SINUMERIK 840D sl / 828D Arbeitsvorbereitung

Programmierhandbuch

Gültig für

Steuerung
SINUMERIK 840D sl / 840DE sl
SINUMERIK 828D

Software
CNC-Software

Version
4.5

02/2012

6FC5398-2BP40-3AA0


Vorwort


Flexible NC-Programmierung	1
Datei- und Programmverwaltung	2
Schutzbereiche	3
Spezielle Wegbefehle	4
Koordinatentransformationen (Frames)	5
Transformationen	6
Werkzeugkorrekturen	7
Bahnverhalten	8
Achskopplungen	9
Synchronaktionen	10
Pendeln	11
Stanzen und Nibbeln	12
Schleifen	13
Weitere Funktionen	14
Eigene Abspanprogramme	15
Zyklen extern programmieren	16
Tabellen	17
Anhang	A


Rechtliche Hinweise

Warnhinweiskonzept

Dieses Handbuch enthält Hinweise, die Sie zu Ihrer persönlichen Sicherheit sowie zur Vermeidung von Sachschäden beachten müssen. Die Hinweise zu Ihrer persönlichen Sicherheit sind durch ein Warndreieck hervorgehoben, Hinweise zu alleinigen Sachschäden stehen ohne Warndreieck. Je nach Gefährdungsstufe werden die Warnhinweise in abnehmender Reihenfolge wie folgt dargestellt.

 GEFAHR
bedeutet, dass Tod oder schwere Körperverletzung eintreten wird , wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

 WARNUNG
bedeutet, dass Tod oder schwere Körperverletzung eintreten kann , wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

 VORSICHT
mit Warndreieck bedeutet, dass eine leichte Körperverletzung eintreten kann, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

VORSICHT
ohne Warndreieck bedeutet, dass Sachschaden eintreten kann, wenn die entsprechenden Vorsichtsmaßnahmen nicht getroffen werden.

ACHTUNG
bedeutet, dass ein unerwünschtes Ergebnis oder Zustand eintreten kann, wenn der entsprechende Hinweis nicht beachtet wird.


Beim Auftreten mehrerer Gefährdungsstufen wird immer der Warnhinweis zur jeweils höchsten Stufe verwendet. Wenn in einem Warnhinweis mit dem Warndreieck vor Personenschäden gewarnt wird, dann kann im selben Warnhinweis zusätzlich eine Warnung vor Sachschäden angefügt sein.

Qualifiziertes Personal

Das zu dieser Dokumentation zugehörige Produkt/System darf nur von für die jeweilige Aufgabenstellung **qualifiziertem Personal** gehandhabt werden unter Beachtung der für die jeweilige Aufgabenstellung zugehörigen Dokumentation, insbesondere der darin enthaltenen Sicherheits- und Warnhinweise. Qualifiziertes Personal ist auf Grund seiner Ausbildung und Erfahrung befähigt, im Umgang mit diesen Produkten/Systemen Risiken zu erkennen und mögliche Gefährdungen zu vermeiden.

Bestimmungsgemäßer Gebrauch von Siemens-Produkten

Beachten Sie Folgendes:

 WARNUNG
Siemens-Produkte dürfen nur für die im Katalog und in der zugehörigen technischen Dokumentation vorgesehenen Einsatzfälle verwendet werden. Falls Fremdprodukte und -komponenten zum Einsatz kommen, müssen diese von Siemens empfohlen bzw. zugelassen sein. Der einwandfreie und sichere Betrieb der Produkte setzt sachgemäßen Transport, sachgemäße Lagerung, Aufstellung, Montage, Installation, Inbetriebnahme, Bedienung und Instandhaltung voraus. Die zulässigen Umgebungsbedingungen müssen eingehalten werden. Hinweise in den zugehörigen Dokumentationen müssen beachtet werden.

Marken

Alle mit dem Schutzrechtsvermerk ® gekennzeichneten Bezeichnungen sind eingetragene Marken der Siemens AG. Die übrigen Bezeichnungen in dieser Schrift können Marken sein, deren Benutzung durch Dritte für deren Zwecke die Rechte der Inhaber verletzen kann.

Haftungsausschluss

Wir haben den Inhalt der Druckschrift auf Übereinstimmung mit der beschriebenen Hard- und Software geprüft. Dennoch können Abweichungen nicht ausgeschlossen werden, so dass wir für die vollständige Übereinstimmung keine Gewähr übernehmen. Die Angaben in dieser Druckschrift werden regelmäßig überprüft, notwendige Korrekturen sind in den nachfolgenden Auflagen enthalten.

Vorwort

SINUMERIK-Dokumentation

Die SINUMERIK-Dokumentation ist in folgende Kategorien gegliedert:

- Allgemeine Dokumentation
- Anwender-Dokumentation
- Hersteller/Service-Dokumentation

Weiterführende Informationen

Unter dem Link www.siemens.com/motioncontrol/docu finden Sie Informationen zu folgenden Themen:

- Dokumentation bestellen / Druckschriftenübersicht
- Weiterführende Links für den Download von Dokumenten
- Dokumentation online nutzen (Handbücher/Informationen finden und durchsuchen)

Bei Fragen zur Technischen Dokumentation (z. B. Anregungen, Korrekturen) senden Sie bitte eine E-Mail an folgende Adresse:

docu.motioncontrol@siemens.com

My Documentation Manager (MDM)

Unter folgendem Link finden Sie Informationen, um auf Basis der Siemens Inhalte eine OEM-spezifische Maschinen-Dokumentation individuell zusammenstellen:

www.siemens.com/mdm

Training

Informationen zum Trainingsangebot finden Sie unter:

- www.siemens.com/sitrain
SITRAIN - das Training von Siemens für Produkte, Systeme und Lösungen der Automatisierungstechnik
- www.siemens.com/sinutrain
SinuTrain - Trainingssoftware für SINUMERIK

FAQs

Frequently Asked Questions finden Sie in den Service&Support Seiten unter Produkt Support. <http://support.automation.siemens.com>

SINUMERIK

Informationen zu SINUMERIK finden Sie unter folgendem Link:

www.siemens.com/sinumerik

Zielgruppe

Die vorliegende Druckschrift wendet sich an:

- Programmierer
- Projektueure

Nutzen

Das Programmierhandbuch befähigt die Zielgruppe, Programme und Software-Oberflächen zu entwerfen, zu schreiben, zu testen und Fehler zu beheben.

Standardumfang

In der vorliegenden Programmieranleitung ist die Funktionalität des Standardumfangs beschrieben. Ergänzungen oder Änderungen, die durch den Maschinenhersteller vorgenommen werden, werden vom Maschinenhersteller dokumentiert.

Es können in der Steuerung weitere, in dieser Dokumentation nicht erläuterte Funktionen ablauffähig sein. Es besteht jedoch kein Anspruch auf diese Funktionen bei der Neulieferung bzw. im Servicefall.

Ebenso enthält diese Dokumentation aus Gründen der Übersichtlichkeit nicht sämtliche Detailinformationen zu allen Typen des Produkts und kann auch nicht jeden denkbaren Fall der Aufstellung, des Betriebes und der Instandhaltung berücksichtigen.

Technical Support

Landesspezifische Telefonnummern für technische Beratung finden Sie im Internet unter <http://www.siemens.com/automation/service&support>

Informationen zu Struktur und Inhalt

Programmierhandbuch "Grundlagen" und "Arbeitsvorbereitung"

Die Beschreibungen zur NC-Programmierung sind auf zwei Handbücher verteilt:

1. Grundlagen

Das Programmierhandbuch "Grundlagen" dient dem Maschineneinsteiger und setzt entsprechende Kenntnisse für Bohr-, Fräs- und Drehbearbeitungen voraus. An einfachen Programmierbeispielen werden die auch nach DIN 66025 bekannten Befehle und Anweisungen erläutert.

2. Arbeitsvorbereitung

Das Programmierhandbuch "Arbeitsvorbereitung" dient dem Technologen mit Kenntnissen über die gesamten Programmiermöglichkeiten. Die SINUMERIK-Steuerung ermöglicht mit einer speziellen Programmiersprache die Programmierung eines komplexen Werkstückprogramms (z. B. Freiformflächen, Kanalkoordinierung, ...) und erleichtert dem Technologen eine aufwendige Programmierung.

Verfügbarkeit der beschriebenen NC-Sprachelemente

Alle im vorliegenden Handbuch beschriebenen NC-Sprachelemente stehen für SINUMERIK 840D sl zur Verfügung. Die Verfügbarkeit bezüglich SINUMERIK 828D ist der Tabelle "Anweisungen: Verfügbarkeit bei SINUMERIK 828D (Seite 806)" zu entnehmen.

Inhaltsverzeichnis

	Vorwort	3
1	Flexible NC-Programmierung	17
1.1	Variablen	17
1.1.1	Systemvariable	18
1.1.2	Vordefinierte Anwendervariablen: Rechenparameter (R).....	21
1.1.3	Vordefinierte Anwendervariablen: Link-Variablen.....	23
1.1.4	Definition von Anwendervariablen (DEF).....	25
1.1.5	Redefinition von Systemvariablen, Anwendervariablen und NC-Sprachbefehlen (REDEF).....	31
1.1.6	Attribut: Initialisierungswert.....	34
1.1.7	Attribut: Grenzwerte (LLI, ULI).....	37
1.1.8	Attribut: Physikalische Einheit (PHU).....	39
1.1.9	Attribut: Zugriffsrechte (APR, APW, APRP, APWP, APRB, APWB)	41
1.1.10	Übersicht definierbarer und redefinierbarer Attribute.....	46
1.1.11	Definition und Initialisierung von Feldvariablen (DEF, SET, REP)	47
1.1.12	Definition und Initialisierung von Feldvariablen (DEF, SET, REP): Weitere Informationen	52
1.1.13	Datentypen.....	55
1.1.14	Explizite Datentypwandlungen (AXTOINT, INTTOAX).....	56
1.1.15	Vorhandensein einer Variablen prüfen (ISVAR).....	57
1.1.16	Attributwerte/Datentyp lesen (GETVARPHU, GETVARAP, GETVARLIM, GETVARDFT, GETVARTYP)	59
1.2	Indirekte Programmierung	65
1.2.1	Indirekte Programmierung von Adressen	65
1.2.2	Indirekte Programmierung von G-Codes	68
1.2.3	Indirekte Programmierung von Positionsattributen (GP).....	69
1.2.4	Indirekte Programmierung von Teileprogrammzeilen (EXECSTRING).....	72
1.3	Rechenfunktionen	73
1.4	Vergleichs- und logische Operationen.....	75
1.5	Genauigkeitskorrektur bei Vergleichsfehlern (TRUNC).....	77
1.6	Minimum, Maximum und Bereich von Variablen (MINVAL, MAXVAL, BOUND).....	79
1.7	Priorität der Operationen.....	81
1.8	Mögliche Typenkonvertierungen.....	82
1.9	Stringoperationen.....	83
1.9.1	Typenkonvertierung nach STRING (AXSTRING).....	84
1.9.2	Typenkonvertierung von STRING (NUMBER, ISNUMBER, AXNAME)	85
1.9.3	Verkettung von Strings (<<).....	86
1.9.4	Wandlung in Klein-/Großbuchstaben (TOLOWER, TOUPPER).....	88
1.9.5	Länge eines Strings bestimmen (STRLEN).....	89
1.9.6	Zeichen/String in String suchen (INDEX, RINDEX, MINDEX, MATCH).....	89
1.9.7	Auswahl eines Teilstrings (SUBSTR)	91
1.9.8	Lesen und Schreiben von einzelnen Zeichen.....	92
1.9.9	String formatieren (SPRINT).....	93

1.10	Programmsprünge und -verzweigungen.....	102
1.10.1	Rücksprung auf Programmanfang (GOTOS).....	102
1.10.2	Programmsprünge auf Sprungmarken (GOTOB, GOTOF, GOTO, GOTOC)	103
1.10.3	Programmverzweigung (CASE ... OF ... DEFAULT ...)	106
1.11	Programmteilwiederholung (REPEAT, REPEATB, ENDLABEL, P)	108
1.12	Kontrollstrukturen	115
1.12.1	Programmschleife mit Alternative (IF, ELSE, ENDIF)	116
1.12.2	Endlos-Programmschleife (LOOP, ENDLOOP).....	118
1.12.3	Zählschleife (FOR ... TO ..., ENDFOR).....	119
1.12.4	Programmschleife mit Bedingung am Schleifenanfang (WHILE, ENDWHILE)	121
1.12.5	Programmschleife mit Bedingung am Schleifenende (REPEAT, UNTIL).....	122
1.12.6	Programmbeispiel mit verschachtelten Kontrollstrukturen.....	123
1.13	Programmkoordinierung (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM)	124
1.14	Interruptroutine (ASUP).....	130
1.14.1	Funktion einer Interruptroutine	130
1.14.2	Interruptroutine erstellen	131
1.14.3	Interruptroutine zuordnen und starten (SETINT, PRIO, BLSYNC).....	132
1.14.4	Zuordnung einer Interruptroutine deaktivieren/reactivieren (DISABLE, ENABLE)	134
1.14.5	Zuordnung einer Interruptroutine löschen (CLRINT)	135
1.14.6	Schnellabheben von der Kontur (SETINT LIFTFAST, ALF)	136
1.14.7	Verfahrrichtung beim Schnellabheben von der Kontur	139
1.14.8	Bewegungsablauf bei Interruptroutinen	142
1.15	Achstausch, Spindeltausch (RELEASE, GET, GETD)	143
1.16	Achse einem anderen Kanal übergeben (AXTOCHAN)	148
1.17	Maschinendaten wirksam setzen (NEWCONF).....	150
1.18	Datei schreiben (WRITE)	151
1.19	Datei löschen (DELETE)	156
1.20	Zeilen in Datei lesen (READ)	158
1.21	Vorhandensein einer Datei prüfen (ISFILE).....	161
1.22	Datei-Informationen auslesen (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO)	163
1.23	Checksummenberechnung über ein Feld (CHECKSUM).....	166
1.24	Aufrunden (ROUNDUP).....	168
1.25	Unterprogrammtechnik.....	169
1.25.1	Allgemeines.....	169
1.25.1.1	Unterprogramm	169
1.25.1.2	Unterprogrammnamen	170
1.25.1.3	Schachtelung von Unterprogrammen	171
1.25.1.4	Suchpfad	172
1.25.1.5	Formal- und Aktualparameter	172
1.25.1.6	Parameterübergabe	173
1.25.2	Definition eines Unterprogramms	175
1.25.2.1	Unterprogramm ohne Parameterübergabe.....	175
1.25.2.2	Unterprogramm mit Parameterübergabe Call-by-Value (PROC)	176
1.25.2.3	Unterprogramm mit Parameterübergabe Call-by-Reference (PROC, VAR)	178
1.25.2.4	Modale G-Funktionen sichern (SAVE).....	180

1.25.2.5	Einzeilsatzbearbeitung unterdrücken (SBLOF, SBLON).....	181
1.25.2.6	Aktuelle Satzanzeige unterdrücken (DISPLOF, DISPLO, ACTBLOCNO).....	187
1.25.2.7	Unterprogramme mit Vorbereitung kennzeichnen (PREPRO)	190
1.25.2.8	Unterprogrammrücksprung M17	191
1.25.2.9	Unterprogrammrücksprung RET	192
1.25.2.10	Parametrierbarer Unterprogrammrücksprung (RET ...).....	193
1.25.3	Aufruf eines Unterprogramms.....	199
1.25.3.1	Unterprogrammaufruf ohne Parameterübergabe	199
1.25.3.2	Unterprogrammaufruf mit Parameterübergabe (EXTERN)	201
1.25.3.3	Anzahl der Programmwiederholungen (P).....	203
1.25.3.4	Modaler Unterprogrammaufruf (MCALL)	204
1.25.3.5	Indirekter Unterprogrammaufruf (CALL)	206
1.25.3.6	Indirekter Unterprogrammaufruf mit Angabe des auszuführenden Programmteils (CALL BLOCK ... TO ...).....	207
1.25.3.7	Indirekter Aufruf eines in ISO-Sprache programmierten Programms (ISOCALL)	209
1.25.3.8	Unterprogramm mit Pfadangabe und Parametern aufrufen (PCALL)	210
1.25.3.9	Suchpfad bei Unterprogrammaufrufen erweitern (CALLPATH)	211
1.25.3.10	Externes Unterprogramm abarbeiten (840D sl) (EXTCALL)	213
1.25.3.11	Externes Unterprogramm abarbeiten (828D) (EXTCALL).....	217
1.26	Makrotechnik (DEFINE ... AS)	221
2	Datei- und Programmverwaltung	225
2.1	Programmspeicher.....	225
2.2	Arbeitsspeicher (CHANDATA, COMPLETE, INITIAL).....	230
3	Schutzbereiche.....	233
3.1	Festlegung der Schutzbereiche (CPROTDEF, NPROTDEF).....	233
3.2	Schutzbereiche aktivieren/deaktivieren (CPROT, NPROT)	237
3.3	Überprüfung auf Schutzbereichsverletzung, Arbeitsfeldbegrenzung und Softwarelimits (CALCPOSI).....	241
4	Spezielle Wegbefehle	249
4.1	Codierte Positionen anfahren (CAC, CIC, CDC, CACP, CACN).....	249
4.2	Spline-Interpolation (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL)	250
4.3	Spline-Verbund (SPLINEPATH)	261
4.4	NC-Satz-Kompression (COMPON, COMPCURV, COMPCAD, COMPOF).....	262
4.5	Polynom-Interpolation (POLY, POLYPATH, PO, PL).....	265
4.6	Einstellbarer Bahnbezug (SPATH, UPATH).....	271
4.7	Messen mit schaltendem Taster (MEAS, MEAW).....	274
4.8	Axiales Messen (MEASA, MEAWA, MEAC) (Option).....	277
4.9	Spezielle Funktionen für den OEM-Anwender (OMA1 ... OMA5, OEMIPO1, OEMIPO2, G810 ... G829)	287
4.10	Vorschubreduzierung mit Eckenverzögerung (FENDNORM, G62, G621).....	288

4.11	Programmierbares Bewegungsendekriterium (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA).....	289
5	Koordinatentransformationen (Frames)	293
5.1	Koordinatentransformation über Framevariable	293
5.1.1	Vordefinierte Framevariable (\$P_BFRAME, \$P_IFRAME, \$P_PFRAME, \$P_ACTFRAME)....	295
5.2	Framevariablen/Frames Werte zuweisen	299
5.2.1	Direkte Werte zuweisen (Achswert, Winkel, Maßstab).....	299
5.2.2	Framekomponenten lesen und verändern (TR, FI, RT, SC, MI).....	302
5.2.3	Verknüpfung von kompletten Frames	303
5.2.4	Definition neuer Frames (DEF FRAME)	305
5.3	Grob- und Feinverschiebung (CFINE, CTRANS)	306
5.4	Externe Nullpunktverschiebung	308
5.5	Preset-Verschiebung mit PRESETON	309
5.6	Frame-Berechnung aus 3 Messpunkten im Raum (MEAFRAME)	311
5.7	NCU-globale Frames	315
5.7.1	Kanalspezifische Frames (\$P_CHBFR, \$P_UBFR).....	316
5.7.2	Im Kanal wirksame Frames.....	317
6	Transformationen.....	323
6.1	Allgemeine Programmierung der Transformationsarten.....	323
6.1.1	Orientierungsbewegungen bei den Transformationen.....	326
6.1.2	Übersicht der Orientierungstransformation TRAORI	329
6.2	Drei-, Vier- und Fünf-Achs-Transformation (TRAORI).....	331
6.2.1	Allgemeine Zusammenhänge Kardanischer Werkzeugkopf.....	331
6.2.2	Drei, Vier, und Fünf- Achs-Transformation (TRAORI)	334
6.2.3	Varianten der Orientierungsprogrammierung und Grundstellung (ORIRESET).....	336
6.2.4	Programmierung der Werkzeugorientierung (A..., B..., C..., LEAD, TILT).....	338
6.2.5	Stirnfräsen (A4, B4, C4, A5, B5, C5)	345
6.2.6	Bezug der Orientierungsachsen (ORIWKS, ORIMKS).....	346
6.2.7	Programmierung der Orientierungsachsen (ORIAxes, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2).....	348
6.2.8	Orientierungsprogrammierung entlang einer Kegelmantelfläche (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO).....	351
6.2.9	Orientierungsvorgabe zweier Kontaktpunkte (ORICURVE, PO[XH]=, PO[YH]=, PO[ZH]=)	355
6.3	Orientierungspolynome (PO[Winkel], PO[Koordinate]).....	357
6.4	Drehungen der Werkzeugorientierung (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA)	359
6.5	Bahnrelative Orientierungen	362
6.5.1	Orientierungsarten relativ zur Bahn	362
6.5.2	Bahnrelative Drehung der Werkzeugorientierung (ORIPATH, ORIPATHS, Drehwinkel).....	363
6.5.3	Bahnrelative Interpolation der Werkzeugdrehung (ORIROTC, THETA)	365
6.5.4	Glättung des Orientierungsverlaufs (ORIPATHS A8=, B8=, C8=).....	367
6.6	Komprimierung der Orientierung (COMPON, COMPCURV, COMPCAD)	368
6.7	Glättung des Orientierungsverlaufs (ORISON, ORISOF).....	371
6.8	Kinematische Transformation	373

6.8.1	Fräsbearbeitung an Drehteilen (TRANSMIT)	373
6.8.2	Zylindermanteltransformation (TRACYL).....	377
6.8.3	Schräge Achse (TRAANG)	385
6.8.4	Schräge Achse programmieren (G5, G7)	388
6.9	Kartesisches PTP-Fahren.....	390
6.9.1	PTP bei TRANSMIT	395
6.10	Randbedingungen bei der Anwahl einer Transformation	399
6.11	Transformation abwählen (TRAFOOF).....	400
6.12	Verkettete Transformationen (TRACON, TRAFOOF)	401
7	Werkzeugkorrekturen	403
7.1	Korrekturspeicher.....	403
7.2	Additive Korrekturen	407
7.2.1	Additive Korrekturen anwählen (DL).....	407
7.2.2	Verschleiß- und Einrichtewerte festlegen (\$TC_SCPxy[t,d], \$TC_ECPxy[t,d])	409
7.2.3	Additive Korrekturen löschen (DELDEL).....	410
7.3	Werkzeugkorrektur - Sonderbehandlung.....	411
7.3.1	Werkzeuglängen spiegeln.....	413
7.3.2	Vorzeichenbewertung Verschleiß	414
7.3.3	Koordinatensystem der aktiven Bearbeitung (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS).....	415
7.3.4	Werkzeuglänge und Ebenenwechsel	418
7.4	Online-Werkzeugkorrektur (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF).....	419
7.5	Aktivierung 3D-Werkzeugkorrekturen (CUT3DC..., CUT3DF...)	424
7.5.1	Aktivierung von 3D-Werkzeugkorrekturen (CUT3DC, CUT3DF, CUT3DFS, CUT3DFF, ISD)	424
7.5.2	3D-Werkzeugkorrektur: Umfangfräsen, Stirnfräsen.....	426
7.5.3	3D-Werkzeugkorrektur: Werkzeugformen und Werkzeugdaten für Stirnfräsen	428
7.5.4	3D-Werkzeugkorrektur: Korrektur auf der Bahn, Bahnkrümmung, Eintauchtiefe (CUT3DC, ISD)	429
7.5.5	3D-Werkzeugkorrektur: Innenecken/Außenecken und Schnittpunktverfahren (G450/G451)	431
7.5.6	3D-Werkzeugkorrektur: 3D-Umfangsfräsen mit Begrenzungsflächen.....	433
7.5.7	3D-Werkzeugkorrektur: Berücksichtigung einer Begrenzungsfläche (CUT3DCC, CUT3DCCD)	433
7.6	Werkzeugorientierung (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST)	437
7.7	Freie D-Nummernvergabe, Schneidenummer.....	443
7.7.1	Freie D-Nummernvergabe, Schneidenummer (Adresse CE)	443
7.7.2	Freie D-Nummernvergabe: D-Nummern prüfen (CHKDNO)	444
7.7.3	Freie D-Nummernvergabe: D-Nummern umbenennen (GETDNO, SETDNO)	445
7.7.4	Freie D-Nummernvergabe: T-Nummer zur vorgegebenen D-Nummer ermitteln (GETACTTD)	446
7.7.5	Freie D-Nummernvergabe: D-Nummern ungültig setzen (DZERO)	446
7.8	Werkzeugträgerkinematik	447
7.9	Werkzeuglängenkorrektur für orientierbare Werkzeugträger (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ).....	452

7.10	Online-Werkzeuglängenkorrektur (TOFFON, TOFFOF)	455
7.11	Schneidendaten-Modifikation bei drehbaren Werkzeugen (CUTMOD).....	458
8	Bahnverhalten	463
8.1	Tangentialsteuerung (TANG, TANGON, TANGOF, TLIFT, TANGDEL)	463
8.2	Vorschubverlauf (FNORM, FLIN, FCUB, FPO)	470
8.3	Beschleunigungsverhalten	475
8.3.1	Beschleunigungsmodus (BRISK, BRISKA, SOFT, SOFTA, DRIVE, DRIVEA)	475
8.3.2	Beeinflussung der Beschleunigung bei Folgeachsen (VELOLIMA, ACCLIMA, JERKLIMA)....	478
8.3.3	Aktivierung von Technologie-spezifischen Dynamikwerten (DYNNORM, DYNPOS, DYNROUGH, DYNSEMIFIN, DYNFINISH)	480
8.4	Fahren mit Vorsteuerung (FFWON, FFWOF).....	482
8.5	Programmierbare Konturgenauigkeit (CPRECON, CPRECOF).....	483
8.6	Programmablauf mit Vorlaufspeicher (STOPFIFO, STARTFIFO, FIFOCTRL, STOPRE)	485
8.7	Bedingt unterbrechbare Programmabschnitte (DELAYFSTON, DELAYFSTOF).....	488
8.8	Programmstelle für SERUPRO verhindern (IPTRLOCK, IPTRUNLOCK).....	493
8.9	Wiederanfahren an Kontur (REPOSA, REPOSL, REPOSQ, REPOSHA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN).....	495
8.10	Beeinflussung der Bewegungsführung	505
8.10.1	Prozentuale Ruckkorrektur (JERKLIM).....	505
8.10.2	Prozentuale Geschwindigkeitskorrektur (VELOLIM)	506
8.10.3	Programmbeispiel für JERKLIM und VELOLIM	508
8.11	Programmierbare Kontur-/Orientierungstoleranz (CTOL, OTOL, ATOL).....	509
8.12	Toleranz bei G0-Bewegungen (STOLF)	512
8.13	Satzwechselverhalten bei aktiver Kopplung (CPBC).....	514
9	Achskopplungen	515
9.1	Mitschleppen (TRAILON, TRAILOF).....	515
9.2	Kurventabellen (CTAB)	520
9.2.1	Kurventabellen definieren (CTABDEF, CATBEND).....	521
9.2.2	Vorhandensein einer Kurventabelle prüfen (CTABEXISTS).....	527
9.2.3	Kurventabellen löschen (CTABDEL).....	528
9.2.4	Kurventabellen gegen Löschen und Überschreiben sperren (CTABLOCK, CTABUNLOCK)	529
9.2.5	Kurventabellen: Tabelleneigenschaften ermitteln (CTABID, CTABISLOCK, CTABMEMTYP, CTABPERIOD)	530
9.2.6	Kurventabellenwerte lesen (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX).....	532
9.2.7	Kurventabellen: Ressourcennutzung prüfen (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL)	537
9.3	Axiale Leitwertkopplung (LEADON, LEADOF)	539
9.4	Elektronisches Getriebe (EG)	545
9.4.1	Elektronisches Getriebe definieren (EGDEF)	545

9.4.2	Elektronisches Getriebe einschalten (EGON, EGONSYN, EGONSYNE).....	547
9.4.3	Elektronisches Getriebe ausschalten (EGOFS, EGOFC)	551
9.4.4	Definition eines Elektronischen Getriebes löschen (EGDEL).....	552
9.4.5	Umdrehungsvorschub (G95) / Elektronisches Getriebe (FPR)	552
9.5	Synchronspindel	553
9.5.1	Synchronspindel: Programmierung (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC).....	554
9.6	Master/Slave-Kopplung (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS).....	565
10	Synchronaktionen	569
10.1	Definition einer Synchronaktion	569
11	Pendeln	571
11.1	Asynchrones Pendeln (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB)	571
11.2	Über Synchronaktionen gesteuertes Pendeln (OSCILL).....	577
12	Stanzen und Nibbeln	585
12.1	Aktivierung, Deaktivierung	585
12.1.1	Stanzen und Nibbeln ein oder aus (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC)	585
12.2	Automatische Wegaufteilung	590
12.2.1	Wegaufteilung bei Bahnachsen	593
12.2.2	Wegaufteilung bei Einzelachsen.....	595
13	Schleifen	597
13.1	Schleifenspezifische Werkzeugüberwachung im Teileprogramm (TMON, TMOF).....	597
14	Weitere Funktionen	599
14.1	Achsfunktionen (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL).....	599
14.2	Umschaltbare Geometrieachsen (GEOAX).....	602
14.3	Achscontainer (AXCTSWE, AXCTSWED, AXCTSWEC).....	606
14.4	Warten auf gültige Achsposition (WAITENC)	608
14.5	Programmierbare Parametersatzumschaltung (SCPARA).....	610
14.6	Vorhandenen NC-Sprachumfang prüfen (STRINGIS).....	612
14.7	Fenster aus dem Teileprogramm interaktiv aufrufen (MMC).....	616
14.8	Programmlaufzeit / Werkstückzähler	618
14.8.1	Programmlaufzeit / Werkstückzähler (Übersicht)	618
14.8.2	Programmlaufzeit.....	618
14.8.3	Werkstückzähler	622
14.9	Process DataShare - Ausgabe auf ein externes Gerät/Datei (EXTOPEN, WRITE, EXTCLOSE).....	623
14.10	Alarmer (SETAL).....	633
14.11	Erweitertes Stillsetzen und Rückziehen (ESR).....	635
14.11.1	NC-geführtes ESR	637
14.11.1.1	NC-geführtes Rückziehen (POLF, POLFA, POLFMASK, POLFMLIN).....	637

14.11.1.2	NC-geführtes Stillsetzen	641
14.11.2	Antriebsautarkes ESR.....	642
14.11.2.1	Antriebsautarkes Stillsetzen projektieren (ESRS)	642
14.11.2.2	Antriebsautarkes Rückziehen projektieren (ESRR).....	643
15	Eigene Abspannprogramme	645
15.1	Unterstützende Funktionen für das Abspannen.....	645
15.2	Konturtabelle erstellen (CONTPRON)	646
15.3	Codierte Konturtabelle erstellen (CONTDCON)	652
15.4	Schnittpunkt zwischen zwei Konturelementen ermitteln (INTERSEC)	656
15.5	Konturelemente einer Tabelle satzweise abfahren (EXECTAB)	658
15.6	Kreisdaten berechnen (CALCDAT).....	659
15.7	Konturaufbereitung ausschalten (EXECUTE).....	661
16	Zyklen extern programmieren	663
16.1	Technologische Zyklen	663
16.1.1	Einleitung.....	663
16.1.2	Bohren, Zentrieren - CYCLE81.....	665
16.1.3	Bohren, Plansenken - CYCLE82	666
16.1.4	Reiben - CYCLE85.....	667
16.1.5	Tieflochbohren - CYCLE83	668
16.1.6	Ausdrehen - CYCLE86.....	670
16.1.7	Gewindebohren ohne Ausgleichsfutter - CYCLE84.....	671
16.1.8	Gewindebohren mit Ausgleichsfutter - CYCLE840.....	674
16.1.9	Bohrgewinde fräsen - CYCLE78.....	676
16.1.10	Beliebige Positionen- CYCLE802	678
16.1.11	Lochreihe - HOLES1	680
16.1.12	Gitter oder Rahmen - CYCLE801	681
16.1.13	Lochkreis - HOLES2	682
16.1.14	Planfräsen - CYCLE61.....	683
16.1.15	Rechtecktasche fräsen - POCKET3	685
16.1.16	Kreistasche fräsen - POCKET4	687
16.1.17	Rechteckzapfen fräsen - CYCLE76.....	690
16.1.18	Kreiszapfen fräsen - CYCLE77.....	692
16.1.19	Mehrkant - CYCLE79.....	694
16.1.20	Längsnut - SLOT1	696
16.1.21	Kreisnut - SLOT2	699
16.1.22	Offene Nut fräsen - CYCLE899	701
16.1.23	Langloch - LONGHOLE	703
16.1.24	Gewindefräsen - CYCLE70.....	705
16.1.25	Gravurzyklus - CYCLE60.....	707
16.1.26	Konturaufruf - CYCLE62	709
16.1.27	Bahnfräsen - CYCLE72.....	710
16.1.28	Konturtasche vorbohren - CYCLE64	713
16.1.29	Konturtasche fräsen - CYCLE63.....	714
16.1.30	Abspannen - CYCLE951.....	717
16.1.31	Einstich - CYCLE930	719
16.1.32	Freistich Formen - CYCLE940.....	722
16.1.33	Gewindedrehen - CYCLE99.....	725

16.1.34	Gewindekette - CYCLE98	728
16.1.35	Abstich - CYCLE92	731
16.1.36	Konturstechen - CYCLE952	733
16.1.37	Schwenken - CYCLE800	737
16.1.38	High Speed Settings - CYCLE832	740
17	Tabellen.....	743
17.1	Anweisungen.....	743
17.2	Anweisungen: Verfügbarkeit bei SINUMERIK 828D	806
17.3	Aktuelle Sprache im HMI	827
A	Anhang	829
A.1	Liste der Abkürzungen	829
A.2	Dokumentationsübersicht	838
	Glossar	841
	Index.....	865

Flexible NC-Programmierung

1.1 Variablen

Durch die Verwendung von Variablen, insbesondere in Verbindung mit Rechenfunktionen und Kontrollstrukturen, können Teileprogramme und Zyklen extrem flexibel gestaltet werden. Vom System werden drei unterschiedliche Arten von Variablen zur Verfügung gestellt:

- Systemvariablen

Systemvariablen sind im System definierte und dem Anwender zur Verfügung gestellte Variablen mit einer fest vorgegebenen Bedeutung. Sie werden auch von der Systemsoftware gelesen und geschrieben. Beispiel: Maschinendaten

Die Bedeutung einer Systemvariablen ist vom System fest vorgegeben. Die Eigenschaften können aber in geringem Umfang vom Anwender durch Redefinition noch angepasst werden. Siehe "Redefinition von Systemvariablen, Anwendervariablen und NC-Sprachbefehlen (REDEF) (Seite 31)"

- Anwendervariablen

Anwendervariablen sind Variablen, deren Bedeutung dem System nicht bekannt ist und vom System auch nicht ausgewertet werden. Die Bedeutung wird ausschließlich durch den Anwender festgelegt.

Anwendervariablen sind unterteilt in:

- Vordefinierte Anwendervariablen

Vordefinierte Anwendervariablen sind im System bereits definierte Variablen, deren Anzahl über spezifische Maschinendaten vom Anwender nur noch parametrisiert werden muss. Die Eigenschaften dieser Variablen können vom Anwender weitestgehend angepasst werden. Siehe "Redefinition von Systemvariablen, Anwendervariablen und NC-Sprachbefehlen (REDEF) (Seite 31)".

- Anwenderdefinierte Variablen

Anwenderdefinierte Variablen sind Variablen, die ausschließlich vom Anwender definiert und vom System erst zur Laufzeit angelegt werden. Ihre Anzahl, Datentyp, Sichtbarkeit und alle weiteren Eigenschaften werden ausschließlich durch den Anwender festgelegt.

Siehe "Definition von Anwendervariablen (DEF) (Seite 25)"

1.1.1 Systemvariable

Systemvariablen sind im System vordefinierte Variablen, die in Teileprogrammen und Zyklen Zugriff auf die aktuelle Parametrierung der Steuerung sowie auf Maschinen-, Steuerungs- und Prozesszustände ermöglichen.

Vorlaufvariablen

Als Vorlaufvariablen werden Systemvariablen bezeichnet, die im Kontext des Vorlaufs, d. h. zum Zeitpunkt der Interpretation des Teileprogrammsatzes, in dem die Systemvariable programmiert ist, gelesen und geschrieben werden. Vorlaufvariablen lösen keinen Vorlaufstopp aus.

Hauptlaufvariablen

Als Hauptlaufvariablen werden Systemvariablen bezeichnet, die im Kontext des Hauptlaufs, d. h. zum Zeitpunkt der Ausführung des Teileprogrammsatzes, in dem die Systemvariable programmiert ist, gelesen oder geschrieben werden. Hauptlaufvariablen sind:

- Systemvariablen, die in Synchronaktionen programmiert werden können (Lesen/Schreiben)
- Systemvariablen, die im Teileprogramm programmiert werden können und Vorlaufstopp auslösen (Lesen/Schreiben)
- Systemvariablen, die im Teileprogramm programmiert werden können und deren Wert im Vorlauf ermittelt, aber erst im Hauptlauf geschrieben wird (Hauptlauf-synchron: nur Schreiben)

Präfix-Systematik

Zur besonderen Kennzeichnung von Systemvariablen ist dem Namen im Normalfall ein Präfix vorangestellt, der sich aus dem \$-Zeichen, gefolgt von einem oder zwei Buchstaben und einem Unterstrich, zusammensetzt:

\$ + 1. Buchstabe	Bedeutung: Datenart
Systemvariablen, die im Vorlauf gelesen / geschrieben werden	
\$M	Maschinendaten ¹⁾
\$S	Settingdaten, Schutzbereiche ¹⁾
\$T	Werkzeugverwaltungsdaten
\$P	Programmierte Werte
\$C	Zyklusvariablen der ISO-Hüllzyklen
\$O	Optionsdaten
R	R-Parameter (Rechenparameter) ²⁾
Systemvariablen, die im Hauptlauf gelesen / geschrieben werden	
\$\$M	Maschinendaten ¹⁾
\$\$S	Settingdaten ¹⁾
\$A	Aktuelle Hauptlaufdaten
\$V	Servo-Daten
\$R	R-Parameter (Rechenparameter) ²⁾
¹⁾ Ob Maschinen- und Settingdaten als Vor- oder Hauptlaufvariable behandelt werden, hängt davon ab, ob sie mit einem oder zwei einem \$-Zeichen geschrieben werden. Die Schreibweise ist anwendungsspezifisch frei wählbar. ²⁾ Bei der Verwendung eines R-Parameters im Teileprogramm / Zyklus als Vorlaufvariable wird kein Präfix geschrieben, z. B. R10. Bei der Verwendung in einer Synchronaktion als Hauptlaufvariable wird als Präfix ein \$-Zeichen geschrieben, z. B. \$R10.	

2. Buchstabe	Bedeutung: Sichtbarkeit
N	NCK-globale Variable (N CK)
C	kanalspezifische Variable (C hannel)
A	achsspezifische Variable (A xis)

Randbedingungen

Ausnahmen in der Präfix-Systematik

Folgende Systemvariablen weichen von der oben genannten Präfix-Systematik ab:

- \$TC_...: Der 2. Buchstabe C verweist hier nicht auf kanalspezifische, sondern auf Werkzeughalter-spezifische Systemvariablen (TC = Tool Carrier)
- \$P_ ...: Kanalspezifische Systemvariablen

Verwendung von Maschinen- und Settingdaten in Synchronaktionen

Bei der Verwendung von Maschinen- und Settingdaten in Synchronaktionen kann durch den Präfix bestimmt werden, ob das Maschinen- oder Settingdatum vorlauf- oder hauptlaufsynchron gelesen/geschrieben wird.

Bleibt das Datum während der Bearbeitung unverändert, kann vorlaufsynchron gelesen werden. Der Präfix des Maschinen- oder Settingdatums wird dazu mit einem \$-Zeichen geschrieben:

```
ID=1 WHENEVER $AA_IM[z] < $SA_OSCILL_REVERSE_POS2[Z]-6 DO $AA_OVR[X]=0
```

Wird das Datum während der Bearbeitung verändert, muss hauptlaufsynchron gelesen / geschrieben werden. Der Präfix des Maschinen- oder Settingdatums wird dazu mit zwei \$-Zeichen geschrieben:

```
ID=1 WHENEVER $AA_IM[z] < $$SA_OSCILL_REVERSE_POS2[Z]-6 DO $AA_OVR[X]=0
```

Hinweis

Schreiben von Maschinendaten

Beim Schreiben eines Maschinen- oder Settingdatums ist darauf zu achten, dass die aktive Zugriffsstufe beim Ausführen des Teileprogramms / Zyklus den Schreibzugriff erlaubt und die Wirksamkeit des Datums "IMMEDIATE" ist.

Literatur

Eine Auflistung der Eigenschaften aller Systemvariablen findet sich in:

Listenhandbuch Systemvariablen

Siehe auch

Variablen (Seite 17)

1.1.2 Vordefinierte Anwendervariablen: Rechenparameter (R)

Funktion

Rechenparameter oder R-Parameter sind vordefinierte Anwendervariablen mit der Bezeichnung R, definiert als Feld vom Datentyp REAL. Aus historischen Gründen ist für R-Parameter neben der Schreibweise mit Feldindex z. B. $R_{[10]}$, auch die Schreibweise ohne Feldindex z. B. R_{10} , erlaubt.

Bei der Verwendung in Synchronaktionen muss der Buchstabe \$ vorangestellt werden, z. B. $\$R_{10}$.

Syntax

Bei Verwendung als Vorlaufvariable:

$R_{\langle n \rangle}$

$R_{[\langle \text{Ausdruck} \rangle]}$

Bei Verwendung als Hauptlaufvariable:

$\$R_{\langle n \rangle}$

$\$R_{[\langle \text{Ausdruck} \rangle]}$

Bedeutung

R:	Bezeichner bei Verwendung als Vorlaufvariable, z. B. im Teileprogramm
$\$R$:	Bezeichner bei Verwendung als Hauptlaufvariable z. B. in Synchronaktionen
Typ:	REAL
Wertebereich:	Bei nicht-exponentieller Schreibweise: $\pm (0.000\ 0001 \dots 9999\ 9999)$ Hinweis: Es sind maximal 8 Dezimalstellen erlaubt Bei exponentieller Schreibweise: $\pm (1 \cdot 10^{-300} \dots 1 \cdot 10^{+300})$ Hinweis: <ul style="list-style-type: none"> • Schreibweise: $\langle \text{Mantisse} \rangle \text{EX} \langle \text{exponent} \rangle$ z.B. 8.2EX-3 • Es sind maximal 10 Zeichen einschließlich Vorzeichen und Dezimalpunkt erlaubt.

1.1 Variablen

<n>: Nummer des R-Parameters
 Typ: INT
 Wertebereich: 0 - MAX_INDEX
Hinweis
 MAX_INDEX ergibt sich aus der parametrisierten Anzahl an R-Parametern:
 MAX_INDEX = (MD28050 \$MN_MM_NUM_R_PARAM) - 1

<Ausdruck>: Feldindex
 Als Feldindex kann ein beliebiger Ausdruck angegeben werden, solange das Ergebnis des Ausdrucks in den Datentyp INT gewandelt werden kann (INT, REAL, BOOL, CHAR)

Beispiel

Zuweisungen an R-Parameter und Verwendung von R-Parametern in mathematischen Funktionen:

Programmcode	Kommentar
R0=3.5678	; Zuweisung im Vorlauf
R[1]=-37.3	; Zuweisung im Vorlauf
R3=-7	; Zuweisung im Vorlauf
\$R4=-0.1EX-5	; Zuweisung im Hauptlauf: R4 = -0.1 * 10 ⁻⁵
\$R[6]=1.874EX8	; Zuweisung im Hauptlauf: R6 = 1.874 * 10 ⁸
R7=SIN(25.3)	; Zuweisung im Vorlauf
R[R2]=R10	; Indirekte Adressierung über R-Parameter
R[(R1+R2)*R3]=5	; Indirekte Adressierung über math. Ausdruck
X=(R1+R2)	; Verfahre Achse X auf die Position die sich aus der Summe von R1 und R2 ergibt
Z=SQRT(R1*R1+R2*R2)	; Verfahre Achse Z auf Position Quadratwurzel(R1 ² + R2 ²)

Siehe auch

Variablen (Seite 17)

1.1.3 Vordefinierte Anwendervariablen: Link-Variablen

Funktion

Über Link-Variablen können im Rahmen der Funktion "NCU-Link" zyklisch Daten zwischen NCUs, die in einem Netzwerk miteinander verbunden sind, ausgetauscht werden. Sie ermöglichen dabei einen Datenformat-spezifischen Zugriff auf den Link-Variablen-Speicher. Der Link-Variablen-Speicher wird sowohl bezüglich der Größe und als auch der Datenstruktur vom Anwender / Maschinenhersteller anlagenspezifisch festgelegt.

Link-Variablen sind systemglobale Anwendervariablen, die bei projektierter Link-Kommunikation von allen NCUs des Link-Verbundes in Teileprogrammen und Zyklen gelesen und geschrieben werden können. Im Gegensatz zu globalen Anwendervariablen (GUD) können Link-Variablen auch in Synchronaktionen verwendet werden.

Bei Anlagen ohne aktiven NCU-Link können Link-Variablen Steuerungs-lokal neben den globalen Anwendervariablen (GUD) als zusätzliche globale Anwendervariablen verwendet werden.

Syntax

```
$A_DLB [<Index>]
$A_DLW [<Index>]
$A_DLD [<Index>]
$A_DLR [<Index>]
```

Bedeutung

\$A_DLB:	Link-Variable für Datenformat BYTE (1 Byte) Datentyp: UINT Wertebereich: 0 ... 255
\$A_DLW:	Link-Variable für Datenformat WORD (2 Bytes) Datentyp: INT Wertebereich: -32768 ... 32767
\$A_DLD:	Link-Variable für Datenformat DWORD (4 Bytes) Datentyp: INT Wertebereich: -2147483648 ... 2147483647
\$A_DLR:	Link-Variable für Datenformat REAL (8 Bytes) Datentyp: REAL Wertebereich: $\pm(2,2 \cdot 10^{-308} \dots 1,8 \cdot 10^{+308})$

1.1 Variablen

<Index>: Adressindex in Byte, gerechnet vom Anfang des Link-Variablen-Speichers
 Datentyp: INT
 Wertebereich: 0 - MAX_INDEX

Hinweis

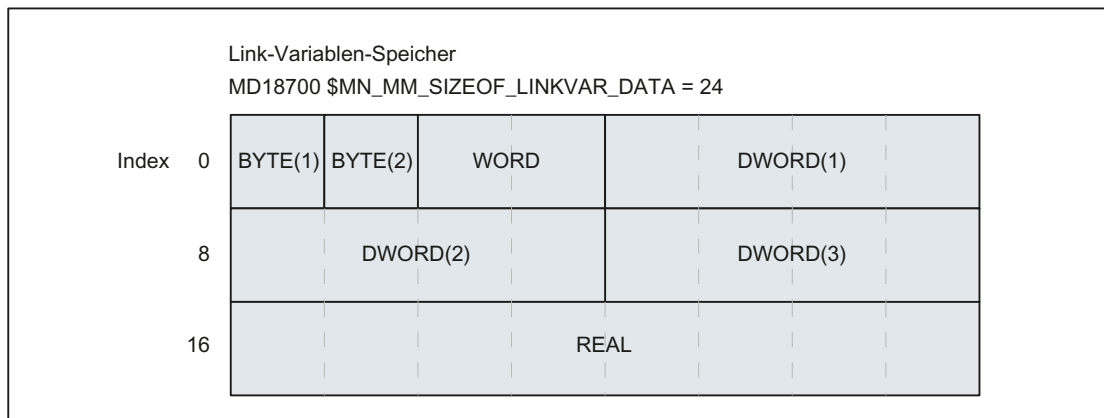
- MAX_INDEX ergibt sich aus der parametrisierten Größe des Link-Variablen-Speichers: $MAX_INDEX = (MD18700 \$MN_MM_SIZEOF_LINKVAR_DATA) - 1$
- Es dürfen nur Indizes programmiert werden, so dass die im Link-Variablen-Speicher adressierten Bytes auf einer Datenformatgrenze liegen \Rightarrow
 $Index = n * Bytes$, mit $n = 0, 1, 2, \dots$
 - \$A_DLB[i]: i = 0, 1, 2, ...
 - \$A_DLW[i]: i = 0, 2, 4, ...
 - \$A_DLD[i]: i = 0, 4, 8, ...
 - \$A_DLR[i]: i = 0, 8, 16, ...

Beispiel

In der Automatisierungsanlage sind 2 NCUs (NCU1 und NCU2) vorhanden. An NCU1 ist Maschinenachse AX2 angeschlossen, die als Link-Achse von NCU2 verfahren wird.

NCU1 schreibt zyklisch den Stromistwert (\$VA_CURR) der Achse AX2 in den Link-Variablen-Speicher. NCU2 liest zyklisch den per Link-Kommunikation übertragenen Stromistwert und zeigt bei Überschreitung des Grenzwertes Alarm 61000 an.

Die Datenstruktur im Link-Variablen-Speicher ist im folgenden Bild dargestellt. Der Stromistwert wird über den REAL-Wert übertragen.



NCU1

NCU1 schreibt in einer statischen Synchronaktion zyklisch im IPO-Takt den Stromistwert der Achse AX2 über die Link-Variable \$A_DLR[16] in den Link-Variablen-Speicher.

Programmcode

```
N111 IDS=1 WHENEVER TRUE DO $A_DLR[16]=$VA_CURR[AX2]
```


NCU2

NCU2 liest in einer statischen Synchronaktion zyklisch im IPO-Takt den Stromwert der Achse AX2 über die Link-Variable \$A_DLR[16] aus dem Link-Variablen-Speicher. Ist der Stromwert größer als 23.0 A, wird der Alarm 61000 angezeigt.

Programmcode

```
N222 IDS=1 WHEN $A_DLR[16] > 23.0 DO SETAL(61000)
```

Siehe auch

Variablen (Seite 17)

1.1.4 Definition von Anwendervariablen (DEF)

Funktion

Mit dem Befehl `DEF` können sie eigene Variablen definieren und mit Werten belegen. In Abgrenzung zu den Systemvariablen werden diese als anwenderdefinierte Variablen oder Anwendervariablen (User Data) bezeichnet.

Entsprechend dem Gültigkeitsbereich, d. h. dem Bereich in dem die Variable sichtbar ist, gibt es folgende Kategorien von Anwendervariablen:

- Lokale Anwendervariablen (LUD)

Lokale Anwendervariablen (LUD) sind Variablen, die in einem Teileprogramm definiert sind, das zum Zeitpunkt der Abarbeitung nicht das Hauptprogramm ist. Sie werden beim Aufruf des Teileprogramms angelegt und mit dem Ende des Teileprogramms bzw. NC-Reset gelöscht. Auf LUD kann nur innerhalb des Teileprogramms zugegriffen werden, in dem sie definiert sind.

- Programmglobale Anwendervariablen (PUD)

Programmglobale Anwendervariablen (PUD) sind Variablen, die in einem als Hauptprogramm verwendeten Teileprogramm definiert sind. Sie werden mit Teileprogrammstart angelegt und mit Teileprogrammende bzw. NC-Reset gelöscht. Auf PUD kann im Hauptprogramm und in allen Unterprogrammen zugegriffen werden.

- Globale Anwendervariablen (GUD)

Globale Anwendervariablen (GUD) sind NC- bzw. Kanal-globale Variablen, die in einem Datenbaustein (SGUD, MGUD, UGUD, GUD4 ... GUD9) definiert sind und auch über Power On hinaus erhalten bleiben. Auf GUD kann in allen Teileprogrammen zugegriffen werden.

Anwendervariablen müssen vor ihrer Verwendung (Lesen / Schreiben) definiert worden sein. Folgende Regeln sind dabei zu beachten:

- GUD müssen in einer Definitionsdatei, z. B. `_N_DEF_DIR/_M_SGUD_DEF`, definiert werden.
- PUD und LUD müssen im Definitionsteil eines Teileprogramms definiert werden.
- Die Datendefinition muss in einem eigenen Satz erfolgen.
- Pro Datendefinition darf nur ein Datentyp verwendet werden.
- Pro Datendefinition können mehrere Variable des gleichen Datentyps definiert werden.

Syntax

LUD und PUD

DEF <Typ> <Phys_Einheit> <Grenzwerte> <Name>[<Wert_1>, <Wert_2>, <Wert_3>]=<Init_Wert>

GUD

DEF <Bereich> <VL_Stop> <Zugriffsrechte> <Typ> <Phys_Einheit> <Grenzwerte> <Name>[<Wert_1>, <Wert_2>, <Wert_3>]=<Init_Wert>

Bedeutung

DEF:	Befehl zur Definition von Anwendervariablen GUD, PUD, LUD
<Bereich>:	Gültigkeitsbereich, nur relevant für GUD:
	NCK: NC-globale Anwendervariable
	CHAN: Kanal-globale Anwendervariable
<VL_Stop>:	Vorlaufstop, nur relevant für GUD (optional)
	SYNR: Vorlaufstop beim Lesen
	SYNW: Vorlaufstop beim Schreiben
	SYNRW: Vorlaufstop beim Lesen/Schreiben
<Zugriffsrechte>:	Schutzstufe für das Lesen / Schreiben von GUD über Teileprogramm oder BTSS (optional)
	APRP <Schutzstufe>: Lesen: Teileprogramm
	APWP <Schutzstufe>: Schreiben: Teileprogramm
	APRB <Schutzstufe>: Lesen: BTSS
	APWB <Schutzstufe>: Schreiben: BTSS
	<Schutzstufe>: Wertebereich: 0 ... 7
	Siehe "Attribut: Zugriffsrechte (APR, APW, APRP, APWP, APRB, APWB) (Seite 41)"

<Typ>:	Datentyp:
	INT: Ganzzahliger Wert mit Vorzeichen
	REAL: Real-Zahl (LONG REAL nach IEEE)
	BOOL: Wahrheitswert TRUE (1) / FALSE (0)
	CHAR: ASCII-Zeichen
	STRING [<MaxLänge>]: Zeichenkette definierter Länge
	AXIS: Achs-/Spindelbezeichner
	FRAME: Geometrische Angaben für eine statische Koordinatentransformation
	Siehe "Datentypen (Seite 55)"
<Phys_Einheit>:	Physikalische Einheit (optional)
	PHU <Einheit>: physikalische Einheit
	Siehe "Attribut: Physikalische Einheit (PHU) (Seite 39)"
<Grenzwerte>:	unterer / oberer Grenzwert (optional)
	LLI <Grenzwert>: unterer Grenzwert (lower limit)
	ULI <Grenzwert>: oberer Grenzwert (upper limit)
	Siehe "Attribut: Grenzwerte (LLI, ULI) (Seite 37)"
<Name>:	Name der Variablen
	Hinweis
	<ul style="list-style-type: none"> • Maximal 31 Zeichen • Die beiden ersten Zeichen müssen ein Buchstabe und/oder ein Unterstrich sein. • Das "\$"-Zeichen ist für Systemvariablen reserviert und darf nicht verwendet werden.
[<Wert_1>, <Wert_2>, <Wert_3>]:	Angabe der Feldgrößen für 1- bis max. 3-dimensionale Feldvariablen (optional)
	Zur Initialisierung von Feldvariablen siehe "Definition und Initialisierung von Feldvariablen (DEF, SET, REP) (Seite 47)"
<Init_Wert>:	Initialisierungswert (optional)
	siehe "Attribut: Initialisierungswert (Seite 34)"
	Zur Initialisierung von Feldvariablen siehe "Definition und Initialisierung von Feldvariablen (DEF, SET, REP) (Seite 47)"

Beispiele

Beispiel 1: Definitionen von Anwendervariablen im Datenbaustein für Maschinenhersteller

Programmcode

```
%_N_MGUD_DEF ; GUD-Baustein: Maschinenhersteller
$PATH=/_N_DEF_DIR
DEF CHAN REAL PHU 24 LLI 0 ULI 10 STROM_1, STROM_2
; Beschreibung
; Definition zweier GUD: STROM_1, STROM_2
; Gültigkeitsbereich: Kanalweit
; Datentyp: REAL
; VL-Stop: nicht programmiert => Defaultwert = kein VL-Stop
; Phys. Einheit: 24 = [A]
; Grenzwerte: Low = 0.0, High = 10.0
; Zugriffsrechte: nicht programmiert => Defaultwert = 7 = Schüsselschalterstellung 0
; Initialisierungswert: nicht programmiert => Defaultwert = 0.0

DEF NCK REAL PHU 13 LLI 10 APWP 3 APRP 3 APWB 0 APRB 2 ZEIT_1=12, ZEIT_2=45
; Beschreibung
; Definition zweier GUD: ZEIT_1, ZEIT_2
; Gültigkeitsbereich: NCK-weit
; Datentyp: REAL
; VL-Stop: nicht programmiert => Defaultwert = kein VL-Stop
; Phys. Einheit: 13 = [s]
; Grenzwerte: Low = 10.0, High = nicht programmiert => obere Definitionsbereichsgrenze
; Zugriffsrechte:
; Teileprogramm: Schreiben/Lesen = 3 = Endanwender
; BTSS: Schreiben = 0 = Siemens, Lesen = 3 = Endanwender
; Initialisierungswert: ZEIT_1 = 12.0, ZEIT_2 = 45.0

DEF NCK APWP 3 APRP 3 APWB 0 APRB 3 STRING[5] GUD5_NAME = "COUNTER"
; Beschreibung
; Definition eines GUD: GUD5_NAME
; Gültigkeitsbereich: NCK-weit
; Datentyp: STRING, max. 5 Zeichen
; VL-Stop: nicht programmiert => Defaultwert = kein VL-Stop
; Phys. Einheit: nicht programmiert => Defaultwert = 0 = keine phys. Einheit
; Grenzwerte: nicht programmiert => Definitionsbereichsgrenzen: Low = 0, High = 255
; Zugriffsrechte:
; Teileprogramm: Schreiben/Lesen = 3 = Endanwender
; BTSS: Schreiben = 0 = Siemens, Lesen = 3 = Endanwender
; Initialisierungswert: "COUNTER"
M30
```

Beispiel 2: Programm-globale und -lokale Anwendervariablen (PUD / LUD)

Programmcode	Kommentar
PROC MAIN	; Hauptprogramm
DEF INT VAR1	; PUD-Definition
...	
SUB2	; Unterprogrammaufruf
...	
M30	

Programmcode	Kommentar
PROC SUB2	; Unterprogramm SUB2
DEF INT VAR2	; LUD-DEFINITION
...	
IF (VAR1==1)	; PUD lesen
VAR1=VAR1+1	; PUD lesen und schreiben
VAR2=1	; LUD schreiben
ENDIF	
SUB3	; Unterprogrammaufruf
...	
M17	

Programmcode	Kommentar
PROC SUB3	; Unterprogramm SUB3
...	
IF (VAR1==1)	; PUD lesen
VAR1=VAR1+1	; PUD lesen und schreiben
VAR2=1	; Fehler: LUD aus SUB2 nicht bekannt
ENDIF	
...	
M17	

Beispiel 3: Definition und Verwendung von Anwendervariablen vom Datentyp AXIS

Programmcode	Kommentar
DEF AXIS ABSZISSE	; 1. Geometrieachse
DEF AXIS SPINDLE	; Spindel
...	
IF ISAXIS(1) == FALSE GOTOF WEITER	
ABSZISSE = \$P_AXN1	
WEITER:	
...	
SPINDLE=(S1)	; 1. Spindel
OVRA [SPINDLE]=80	; Spindeloverride = 80%
SPINDLE=(S3)	; 3. Spindel

Randbedingungen

Globale Anwendervariablen (GUD)

Im Rahmen der Definition von globalen Anwendervariablen (GUD) sind folgende Maschinendaten zu berücksichtigen:

Nr.	Bezeichner: \$MN_	Bedeutung
11140	GUD_AREA_SAVE_TAB	zusätzliche Sicherung für GUD-Bausteine
18118 ¹⁾	MM_NUM_GUD_MODULES	Anzahl GUD-Dateien im aktiven Filesystem
18120 ¹⁾	MM_NUM_GUD_NAMES_NCK	Anzahl der globalen GUD-Namen
18130 ¹⁾	MM_NUM_GUD_NAMES_CHAN	Anzahl der kanalspez. GUD-Namen
18140 ¹⁾	MM_NUM_GUD_NAMES_AXIS	Anzahl der achsspez. GUD-Namen
18150 ¹⁾	MM_GUD_VALUES_MEM	Speicherplatz für globale GUD-Werte
18660 ¹⁾	MM_NUM_SYNACT_GUD_REAL	Anzahl projektierbare GUD Datentyp REAL
18661 ¹⁾	MM_NUM_SYNACT_GUD_INT	Anzahl projektierbare GUD Datentyp INT
18662 ¹⁾	MM_NUM_SYNACT_GUD_BOOL	Anzahl projektierbare GUD Datentyp BOOL
18663 ¹⁾	MM_NUM_SYNACT_GUD_AXIS	Anzahl projektierbare GUD Datentyp AXIS
18664 ¹⁾	MM_NUM_SYNACT_GUD_CHAR	Anzahl projektierbare GUD Datentyp CHAR
18665 ¹⁾	MM_NUM_SYNACT_GUD_STRING	Anzahl projektierbare GUD Datentyp STRING

¹⁾ MD bei SINUMERIK 828D nur lesbar!

Programmglobale Anwendervariablen (PUD)

ACHTUNG
Sichtbarkeit von programmlokalen Anwendervariablen (PUD)
Im Hauptprogramm definierte programmlokale Anwendervariablen (PUD) sind nur dann auch in den Unterprogrammen sichtbar, wenn folgendes Maschinendatum gesetzt ist: MD11120 \$MN_LUD_EXTENDED_SCOPE = 1 Mit MD11120 = 0 sind die im Hauptprogramm definierten programmlokalen Anwendervariablen nur im Hauptprogramm sichtbar.

Kanalübergreifende Verwendung einer NCK-globalen Anwendervariablen vom Datentyp AXIS

Eine NCK-globale Anwendervariable vom Datentyp `AXIS`, die bei der Definition im Datenbaustein mit einem Achsbezeichner initialisiert wurde, kann nur dann in unterschiedlichen Kanälen der NC verwendet werden, wenn die Achse in diesen Kanälen die gleiche Kanalachsnummer hat.

Ist dies nicht der Fall, muss die Variable am Teileprogrammanfang geladen oder, wie im folgenden Beispiel, die Funktion `AXNAME(...)` verwendet werden.

Programmcode	Kommentar
DEF NCK STRING[5] ACHSE="X"	; Definition im Datenbaustein
...	
N100 AX[AXNAME(ACHSE)]=111 G00	; Verwendung im Teileprogramm

1.1.5 Redefinition von Systemvariablen, Anwendervariablen und NC-Sprachbefehlen (REDEF)

Funktion

Mit dem Befehl `REDEF` können die Attribute von Systemvariablen, Anwendervariablen und NC-Sprachbefehlen geändert werden. Grundvoraussetzung für eine Redefinition ist, dass sie zeitlich nach der entsprechenden Definition ausgeführt wird.

Bei einer Redefinition können nicht mehrere Attribute gleichzeitig geändert werden. Für jedes zu ändernde Attribut muss eine eigene `REDEF`-Anweisung programmiert werden.

Werden mehrere konkurrierende Attributänderungen programmiert, wird immer die letzte Änderung aktiv.

Redefinierbare Attribute

Siehe "Übersicht definierbarer und redefinierbarer Attribute (Seite 46)"

Lokale Anwendervariablen (PUD / LUD)

Für lokale Anwendervariablen (PUD / LUD) dürfen keine Redefinitionen vorgenommen werden.

Syntax

```
REDEF <Name> <VL_Stop>
REDEF <Name> <Phys_Einheit>
REDEF <Name> <Grenzwerte>
REDEF <Name> <Zugriffsrechte>
REDEF <Name> <Init_Zeitpunkt>
REDEF <Name> <Init_Zeitpunkt> <Init_Wert>
```

Bedeutung

<code>REDEF:</code>	Befehl zur Redefinition eines bestimmten Attributs von Systemvariablen, Anwendervariablen und NC-Sprachbefehlen
<code><Name>:</code>	Name einer bereits definierten Variablen oder eines NC-Sprachbefehls
<code><VL-Stop>:</code>	Vorlaufstop
	<code>SYNR:</code> Vorlaufstop beim Lesen
	<code>SYNW:</code> Vorlaufstop beim Schreiben
	<code>SYNRW:</code> Vorlaufstop beim Lesen/Schreiben

<Phys_Einheit>:	Physikalische Einheit PHU <Einheit>: physikalische Einheit Siehe "Attribut: Physikalische Einheit (PHU) (Seite 39)" Hinweis Nicht redefinierbar für: <ul style="list-style-type: none">• Systemvariablen• globale Anwenderdaten (GUD)• Datentypen: BOOL, AXIS, STRING, FRAME
<Grenzwerte>:	unterer / oberer Grenzwert LLI <Grenzwert>: unterer Grenzwert (lower limit) ULI <Grenzwert>: oberer Grenzwert (upper limit) Siehe "Attribut: Grenzwerte (LLI, ULI) (Seite 37)" Hinweis Nicht redefinierbar für: <ul style="list-style-type: none">• Systemvariablen• globale Anwenderdaten (GUD)• Datentypen: BOOL, AXIS, STRING, FRAME
<Zugriffsrechte>:	Zugriffsrechte für das Lesen / Schreiben über Teileprogramm oder BTSS APX <Schutzstufe>: Ausführen: NC-Sprachelement APRP <Schutzstufe>: Lesen: Teileprogramm APWP <Schutzstufe>: Schreiben: Teileprogramm APRB <Schutzstufe>: Lesen: BTSS APWB <Schutzstufe>: Schreiben: BTSS <Schutzstufe>: Wertebereich: 0 ... 7 Siehe "Attribut: Zugriffsrechte (APR, APW, APRP, APWP, APRB, APWB) (Seite 41)"
<Init_Zeitpunkt>:	Zeitpunkt zu dem die Variable reinitialisiert wird INIPO: Power On INIRE: Hauptprogrammende, NC-Reset oder Power On INICF: NewConfig oder Hauptprogrammende, NC-Reset oder Power On PRLOC: Hauptprogrammende, NC-Reset nach lokaler Änderung oder Power On Siehe "Attribut: Initialisierungswert (Seite 34)"
<Init_Wert>:	Initialisierungswert Bei Redefinition des Initialisierungswerts muss immer auch ein Initialisierungszeitpunkt (siehe <Init_Zeitpunkt>) angegeben werden. Siehe "Attribut: Initialisierungswert (Seite 34)" Zur Initialisierung von Feldvariablen siehe "Definition und Initialisierung von Feldvariablen (DEF, SET, REP) (Seite 47)" Hinweis Nicht redefinierbar für: <ul style="list-style-type: none">• Systemvariablen, ausgenommen Settingdaten

Beispiel

Redefinitionen der Systemvariable \$TC_DPC1 im Datenbaustein für Maschinenhersteller

Programmcode

```
%_N_MGUD_DEF ; GUD-Baustein: Maschinenhersteller
$PATH=/_N_DEF_DIR
REDEF $TC_DPC1 APWB 2 APWP 3 ; 1)
REDEF $TC_DPC1 PHU 21 ; 2)
REDEF $TC_DPC1 LLI 0 ULI 200 ; 3)
REDEF $TC_DPC1 INIPO (100, 101, 102, 103) ; 4)
; 1) Zugriffsrecht Schreiben: BTSS = Schutzstufe 2, Teileprogramm = Schutzstufe 3
; Hinweis
; Bei Verwendung von ACCESS-Dateien muss die Redefinition der Zugriffsrechte von
; _N_MGUD_DEF nach _N_MACCESS_DEF verlagert werden
; 2) Physikalische Einheit = [ % ]
; 3) Grenzwerte: unterer Grenzwert = 0, oberer Grenzwert = 200
; 4) Die Feldvariable wird bei PowerOn mit den vier Werten initialisiert
M30
```

Randbedingungen

Granularität

Eine Redefinition bezieht sich immer auf die gesamte, durch ihren Namen eindeutig gekennzeichnete Variable. Es ist nicht möglich z. B. bei Feldvariablen für einzelne Feldelemente unterschiedliche Attributwerte zuzuweisen.

1.1.6 Attribut: Initialisierungswert

Definition (_{DEF}) von Anwendervariablen

Bei der Definition kann für folgende Anwendervariablen ein Initialisierungswert vorgegeben werden:

- globale Anwendervariablen (GUD)
- programmglobale Anwendervariablen (PUD)
- lokale Anwendervariablen (LUD)

Redefinition (_{REDEF}) von System- und Anwendervariablen

Bei der Redefinition kann für folgende Variablen ein Initialisierungswert vorgegeben werden:

- Systemdaten
 - Settingdaten
- Anwenderdaten
 - R-Parameter
 - Synchronaktionsvariable (\$AC_MARKER, \$AC_PARAM, \$AC_TIMER)
 - Synchronaktions-GUD (SYG_xy[], mit x=R, I, B, A, C, S und y=S, M, U, 4, ..., 9)
 - EPS-Parameter
 - Werkzeugdaten-OEM
 - Magazindaten-OEM
 - globale Anwendervariablen (GUD)

Reinitialisierungszeitpunkt

Bei der Redefinition kann der Zeitpunkt angegeben werden, zu dem die Variable reinitialisiert, d. h. wieder auf den Initialisierungswert gesetzt werden soll:

- `INIPO` (Power On)
Die Variable wird bei Power On reinitialisiert.
- `INIRE` (Reset)
Die Variable wird bei NC-Reset, BAG-Reset, Teileprogrammende (`M02 / M30`) oder Power On reinitialisiert.
- `INICF` (NewConfig)
Die Variable wird bei NewConf-Anforderung über HMI, Teileprogramm-Befehl `NEWCONFIG` oder NC-Reset, BAG-Reset, Teileprogrammende (`M02 / M30`) oder Power On reinitialisiert.
- `PRLOC` (programmlokale Änderung)
Die Variable wird nur dann bei NC-Reset, BAG-Reset oder Teileprogrammende (`M02 / M30`) reinitialisiert, wenn sie im Rahmen des aktuellen Teileprogramms verändert worden ist.

Das Attribut `PRLOC` darf nur in Zusammenhang mit programmierbaren Settingdaten (siehe folgende Tabelle) verwendet werden.

Tabelle 1- 1 Programmierbare Settingdaten

Nummer	Bezeichner	G-Befehl ¹⁾
42000	\$SC_THREAD_START_ANGLE	SF
42010	\$SC_THREAD_RAMP_DISP	DITS / DITE
42400	\$SA_PUNCH_DWELLTIME	PDELAYON
42800	\$SA_SPIND_ASSIGN_TAB	SETMS
43210	\$SA_SPIND_MIN_VELO_G25	G25
43220	\$SA_SPIND_MAX_VELO_G26	G26
43230	\$SA_SPIND_MAX_VELO_LIMS	LIMS
43300	\$SA_ASSIGN_FEED_PER_REV_SOURCE	FPRAON
43420	\$SA_WORKAREA_LIMIT_PLUS	G26
43430	\$SA_WORKAREA_LIMIT_MINUS	G25
43510	\$SA_FIXED_STOP_TORQUE	FXST
43520	\$SA_FIXED_STOP_WINDOW	FXSW
43700	\$SA_OSCILL_REVERSE_POS1	OSP1
43710	\$SA_OSCILL_REVERSE_POS2	OSP2
43720	\$SA_OSCILL_DWELL_TIME1	OST1
43730	\$SA_OSCILL_DWELL_TIME2	OST2
43740	\$SA_OSCILL_VELO	FA
43750	\$SA_OSCILL_NUM_SPARK_CYCLES	OSNSC
43760	\$SA_OSCILL_END_POS	OSE
43770	\$SA_OSCILL_CTRL_MASK	OSCTRL
43780	\$SA_OSCILL_IS_ACTIVE	OS
43790	\$SA_OSCILL_START_POS	OSB
1) mit diesem G-Befehl wird das Settingdatum angesprochen		

Randbedingungen

Initialisierungswert: globale Anwendervariablen (GUD)

- Für globale Anwendervariable (GUD) mit dem Gültigkeitsbereich `NCK` kann als Initialisierungszeitpunkt nur `INIPO` (Power On) vorgegeben werden.
- Für globale Anwendervariablen (GUD) mit dem Gültigkeitsbereich `CHAN` kann als Initialisierungszeitpunkt neben `INIPO` (Power On) auch `INIRE` (Reset) oder `INICF` (NewConfig) vorgegeben werden.
- Bei globalen Anwendervariablen (GUD) mit dem Gültigkeitsbereich `CHAN` und Initialisierungszeitpunkt `INIRE` (Reset) oder `INICF` (NewConfig) werden bei NC-Reset, BAG-Reset und NewConfig die Variablen nur in den Kanälen neu initialisiert, in denen die genannten Ereignisse ausgelöst wurden.

Initialisierungswert: Datentyp FRAME

Für Variablen vom Datentyp `FRAME` darf kein Initialisierungswert angegeben werden. Variablen vom Datentyp `FRAME` werden implizit immer mit dem Defaultframe initialisiert.

Initialisierungswert: Datentyp CHAR

Für Variablen vom Datentyp `CHAR` kann statt des ASCII-Codes (0...255) auch das entsprechende ASCII-Zeichen in Anführungszeichen programmiert werden, z.B. "A"

Initialisierungswert: Datentyp STRING

Bei Variablen vom Datentyp `STRING` muss die Zeichenkette in Anführungszeichen gesetzt werden z.B.: `...= "MASCHINE_1"`

Initialisierungswert: Datentyp AXIS

Für Variablen vom Datentyp `AXIS` muss bei erweiterter Adressschreibweise der Achsbezeichner in Klammern gesetzt werden, z.B.: `...=(X3)`

Initialisierungswert: Systemvariable

Für Systemvariable können durch Redefinition keine anwenderspezifischen Initialisierungswerte vorgegeben werden. Die Initialisierungswerte der Systemvariablen sind vom System fest vorgegeben. Durch Redefinition kann aber der Zeitpunkt (`INIRE`, `INICF`) zu dem die Systemvariable reinitialisiert wird geändert werden.

Impliziter Initialisierungswert: Datentyp AXIS

Für Variablen vom Datentyp `AXIS` wird folgender implizite Initialisierungswert verwendet:

- Systemdaten: "erste Geometrieachse"
- Synchronaktions-GUD (Bezeichnung: SYG_A*), PUD, LUD:
Achsbezeichner aus Maschinendatum: MD20082
`$MC_AXCONF_CHANAX_DEFAULT_NAME`

Impliziter Initialisierungswert: Werkzeug- und Magazindaten

Für Werkzeug- und Magazindaten können Initialisierungswerte über folgendes Maschinendatum vorgegeben werden: MD17520 `$MN_TOOL_DEFAULT_DATA_MASK`

ACHTUNG
Synchronisation
Die Synchronisation von Ereignissen die eine Reinitialisierung einer globalen Variable auslösen mit dem Lesen dieser Variable an anderer Stelle, liegt ausschließlich in der Verantwortung des Anwenders / Maschinenherstellers.

Siehe auch

Variablen (Seite 17)

1.1.7 Attribut: Grenzwerte (LLI, ULI)

Ein oberer und unterer Grenzwert des Definitionsbereichs kann nur für folgende Datentypen vorgegeben werden:

- INT
- REAL
- CHAR

Definition (DEF) von Anwendervariablen: Grenzwerte und implizite Initialisierungswerte

Wird bei der Definition einer Anwendervariablen von einem der oben genannten Datentypen kein expliziter Initialisierungswert definiert, wird die Variable auf den impliziten Initialisierungswert des Datentyps gesetzt:

- INT: 0
- REAL: 0.0
- CHAR: 0

Liegt der implizite Initialisierungswert außerhalb des durch die programmierten Grenzwerte festgelegten Definitionsbereichs, wird die Variable mit dem Grenzwert initialisiert, der dem impliziten Initialisierungswert am nächsten liegt:

- impliziter Initialisierungswert < unterer Grenzwert (LLI) \Rightarrow
Initialisierungswert = unterer Grenzwert
- impliziter Initialisierungswert > oberer Grenzwert (ULI) \Rightarrow
Initialisierungswert = oberer Grenzwert

Beispiele:

Programmcode	Kommentar
DEF REAL GUD1	; unterer Grenzwert = Definitionsbereichsgrenze ; oberer Grenzwert = Definitionsbereichsgrenze ; kein Initialisierungswert programmiert ; => impliziter Initialisierungswert = 0.0
DEF REAL LLI 5.0 GUD2	; unterer Grenzwert = 5.0 ; oberer Grenzwert = Definitionsbereichsgrenze ; => Initialisierungswert = 5.0
DEF REAL ULI -5 GUD3	; unterer Grenzwert = Definitionsbereichsgrenze ; oberer Grenzwert = -5.0 ; => Initialisierungswert = -5.0

Redefinition (REDEF) von Anwendervariablen: Grenzwerte und aktuelle Istwerte

Werden bei der Redefinition der Grenzwerte einer Anwendervariablen diese so geändert, dass der aktuelle Istwert außerhalb des neuen Definitionsbereichs liegt, erfolgt ein Alarm und die Grenzwerte werden nicht übernommen.

Hinweis

Redefinition (REDEF) von Anwendervariablen

Bei der Redefinition der Grenzwerte einer Anwendervariablen ist auf das konsistente Ändern der folgenden Werte zu achten:

- Grenzwerte
 - Istwert
 - Initialisierungswert beim Redefinieren und beim automatischen Reinitialisieren aufgrund von INIPO, INIRE oder INICF
-

Siehe auch

Variablen (Seite 17)

1.1.8 Attribut: Physikalische Einheit (PHU)

Eine physikalische Einheit kann nur für Variablen von folgende Datentypen vorgegeben werden:

- INT
- REAL

Programmierbare physikalische Einheiten (PHU)

Die Angabe der physikalische Einheit erfolgt als Festkommazahl: PHU <Einheit>

Folgende physikalische Einheiten können programmiert werden:

<Einheit>	Bedeutung	Physikalische Einheit
0	keine physikalische Einheit	-
1	Linear- oder Winkel-Position ¹⁾²⁾	[mm], [inch], [Grad]
2	Linear-Position ²⁾	[mm], [inch]
3	Winkel-Position	[Grad]
4	Linear- oder Winkel-Geschwindigkeit ¹⁾²⁾	[mm/min], [inch/min], [U/min]
5	Linear-Geschwindigkeit ²⁾	[mm/min]
6	Winkel-Geschwindigkeit	[U/min]
7	Linear- oder Winkel-Beschleunigung ¹⁾²⁾	[m/s ²], [inch/s ²], [U/s ²]
8	Linear-Beschleunigung ²⁾	[m/s ²], [inch/s ²]
9	Winkel-Beschleunigung	[U/s ²]
10	Linear- oder Winkel-Ruck ¹⁾²⁾	[m/s ³], [inch/s ³], [U/s ³]
11	Linear-Ruck ²⁾	[m/s ³], [inch/s ³]
12	Winkel-Ruck	[U/s ³]
13	Zeit	[s]
14	Lageregler-Verstärkung	[16.667/s]
15	Umdrehungsvorschub ²⁾	[mm/U], [inch/U]
16	Temperaturkompensation ¹⁾²⁾	[mm], [inch]
18	Kraft	[N]
19	Masse	[kg]
20	Trägheitsmoment ³⁾	[kgm ²]
21	Prozent	[%]
22	Frequenz	[Hz]
23	Spannung	[V]
24	Strom	[A]
25	Temperatur	[°C]
26	Winkel	[Grad]
27	KV	[1000/min]
28	Linear- oder Winkel-Position ³⁾	[mm], [inch], [Grad]
29	Schnittgeschwindigkeit ²⁾	[m/min], [feet/min]
30	Umfangsgeschwindigkeit ²⁾	[m/s], [feet/s]

<Einheit>	Bedeutung	Physikalische Einheit
31	Widerstand	[Ohm]
32	Induktivität	[mH]
33	Drehmoment ³⁾	[Nm]
34	Drehmomentkonstante ³⁾	[Nm/A]
35	Stromreglerverstärkung	[V/A]
36	Drehzahlreglerverstärkung ³⁾	[Nm/(rad*s)]
37	Drehzahl	[U/min]
42	Leistung	[kW]
43	Strom, klein	[µA]
46	Drehmoment, klein ³⁾	[µNm]
48	Promille	-
49	-	[Hz/s]
65	Durchfluss	[l/min]
66	Druck	[bar]
67	Volumen ³⁾	[cm ³]
68	Streckenverstärkung ³⁾	[mm/(V*min)]
69	Streckenverstärkung Kraftregler	[N/V]
155	Gewindesteigung ³⁾	[mm/U], [inch/U]
156	Gewindesteigungsänderung ³⁾	[mm/U / U], [inch/U / U]
1) Die physikalische Einheit ist abhängig vom Achstyp: Linear- oder Rundachse		
2) Maßsystem-Umschaltung G70/G71(inch/metrisch) Nach einer Umschaltung des Grundsystems (MD10240 \$MN_SCALING_SYSTEM_IS_METRIC) mit G70/G71 erfolgt bei Schreib/Lesezugriffen auf längenbehaftete System- und Anwendervariablen keine Umrechnung der Werte (Istwert, Defaultwert und Grenzwerte) G700/G710(inch/metrisch) Nach einer Umschaltung des Grundsystems (MD10240 \$MN_SCALING_SYSTEM_IS_METRIC) mit G700/G710 erfolgt bei Schreib/Lesezugriffen auf längenbehaftete System- und Anwendervariablen eine Umrechnung der Werte (Istwert, Defaultwert und Grenzwerte)		
3) Die Variable wird nicht automatisch in das aktuelle Maßsystem der NC (inch/metrisch) umgerechnet. Die Umrechnung liegt ausschließlich in der Verantwortung des Anwenders / Maschinenherstellers.		

Hinweis

Ebenenüberlauf durch Formatumrechnung

Das interne Ablageformat für alle Anwendervariablen (GUD / PUD / LUD) mit längenbehafteten physikalischen Einheiten ist metrisch. Eine exzessive Verwendung derartiger Variablen im Hauptlauf des NCK, z. B. in Synchronaktionen, kann bei einer Maßsystemumschaltung zu einem Rechenzeitüberlauf der Interpolatorebene, Alarm 4240, führen.

ACHTUNG
Kompatibilität von Einheiten
Bei der Verwendung von Variablen (Zuweisung, Vergleich, Berechnung etc.) erfolgt keine Prüfung auf Kompatibilität der beteiligten Einheiten. Eine gegebenenfalls erforderliche Umrechnung liegt ausschließlich in der Verantwortung des Anwenders / Maschinenherstellers.

Siehe auch

Variablen (Seite 17)

1.1.9 Attribut: Zugriffsrechte (APR, APW, APRP, APWP, APRB, APWB)

Den Zugriffsrechten entsprechen folgende bei der Programmierung anzugebende Schutzstufen:

Zugriffsrecht	Schutzstufe
Kennwort System	0
Kennwort Maschinenhersteller	1
Kennwort Service	2
Kennwort Endanwender	3
Schlüsselschalter Stellung 3	4
Schlüsselschalter Stellung 2	5
Schlüsselschalter Stellung 1	6
Schlüsselschalter Stellung 0	7

Definition (DEF) von Anwendervariablen

Zugriffsrechte (APR... / APW...) können für folgende Variablen definiert werden:

- globale Anwenderdaten (GUD)

Redefinition (**REDEF**) von System- und Anwendervariablen

Zugriffsrechte (**APR...** / **APW...**) können für folgende Variablen redefiniert werden:

- Systemdaten
 - Maschinendaten
 - Settingdaten
 - FRAME
 - Prozessdaten
 - Spindelsteigungsfehlerkompensation (EEC)
 - Durchhangkompensation (CEC)
 - Quadrantenfehlerkompensation (QEC)
 - Magazindaten
 - Werkzeugdaten
 - Schutzbereiche
 - orientierbare Werkzeugträger
 - kinematische Ketten
 - 3D-Schutzbereiche
 - Arbeitsfeldbegrenzung
 - ISO-Werkzeugdaten
- Anwenderdaten
 - R-Parameter
 - Synchronaktionsvariable (**\$AC_MARKER**, **\$AC_PARAM**, **\$AC_TIMER**)
 - Synchronaktions-GUD (**SYG_xy[]**, mit x=R, I, B, A, C, S und y=S, M, U, 4, ..., 9)
 - EPS-Parameter
 - Werkzeugdaten-OEM
 - Magazindaten-OEM
 - globale Anwendervariablen (GUD)

Hinweis

Bei der Redefinition kann das Zugriffsrecht auf eine Variable zwischen der niedrigsten Schutzstufe 7 und der eigenen Schutzstufe, z. B. 1 (Maschinenhersteller), frei vergeben werden.

Redefinition (**REDEF**) von NC-Sprachbefehlen

Das Zugriffs- bzw. Ausführungsrecht (**APX**) kann für folgende NC-Sprachbefehle redefiniert werden:

- G-Funktionen / Wegbedingungen

Literatur:

Programmierhandbuch Grundlagen; Kapitel: G-Funktionen / Wegbedingungen

- Vordefinierte Funktionen

Literatur:

Programmierhandbuch Grundlagen; Kapitel: Vordefinierte Funktionen

- Vordefinierte Unterprogrammaufrufe

Literatur:

Programmierhandbuch Grundlagen; Kapitel: Vordefinierte Unterprogrammaufrufe

- Anweisung **DO** bei Synchronaktionen

- Programmbezeichner von Zyklen

Der Zyklus muss in einem Zyklenverzeichnis abgelegt sein und eine **PROC**-Anweisung enthalten.

Zugriffsrechte bezüglich Teileprogrammen und Zyklen (**APRP**, **APWP**)

Die unterschiedlichen Zugriffsrechte haben für den Zugriff in einem Teileprogramm bzw. Zyklus folgende Auswirkungen:

- **APRP 0 / APWP 0**
 - beim Abarbeiten des Teileprogramms muss das System-Kennwort gesetzt sein
 - der Zyklus muss im Verzeichnis **_N_CST_DIR** (System) abgelegt sein
 - für das Verzeichnis **_N_CST_DIR** muss im MD11160 **\$MN_ACCESS_EXEC_CST** das Ausführungsrecht auf System eingestellt sein
- **APRP 1 / APWP 1 bzw. APRP 2 / APWP 2**
 - beim Abarbeiten des Teileprogramms muss das Maschinenhersteller- bzw. Service-Kennwort gesetzt sein
 - der Zyklus muss im Verzeichnis **_N_CMA_DIR** (Maschinenhersteller) oder **_N_CST_DIR** abgelegt sein
 - für die Verzeichnisse **_N_CMA_DIR** bzw. **_N_CST_DIR** müssen in den Maschinendaten MD11161 **\$MN_ACCESS_EXEC_CMA** bzw. MD11160 **\$MN_ACCESS_EXEC_CST** die Ausführungsrechte mindestens auf Maschinenhersteller eingestellt sein

1.1 Variablen

- $APRP\ 3 / APWP\ 3$
 - beim Abarbeiten des Teileprogramms muss das Endanwender-Kennwort gesetzt sein
 - der Zyklus muss im Verzeichnis $_N_CUS_DIR$ (Anwender), $_N_CMA_DIR$ oder $_N_CST_DIR$ abgelegt sein
 - für die Verzeichnisse $_N_CUS_DIR$, $_N_CMA_DIR$ bzw. $_N_CST_DIR$ müssen in den Maschinendaten MD11162 $\$MN_ACCESS_EXEC_CUS$, MD11161 $\$MN_ACCESS_EXEC_CMA$ bzw. MD11160 $\$MN_ACCESS_EXEC_CST$ die Ausführungsrechte mindestens auf Endanwender eingestellt sein
- $APRP\ 4\dots7 / APWP\ 4\dots7$
 - beim Abarbeiten des Teileprogramms muss Schlüsselschalterstellung 3 ... 0 eingestellt sein
 - der Zyklus muss im Verzeichnis $_N_CUS_DIR$, $_N_CMA_DIR$ oder $_N_CST_DIR$ abgelegt sein
 - für die Verzeichnisse $_N_CUS_DIR$, $_N_CMA_DIR$ bzw. $_N_CST_DIR$ müssen in den Maschinendaten MD11162 $\$MN_ACCESS_EXEC_CUS$, MD11161 $\$MN_ACCESS_EXEC_CMA$ bzw. MD11160 $\$MN_ACCESS_EXEC_CST$ die Ausführungsrechte mindestens auf die entsprechende Schlüsselschalterstellung eingestellt sein

Zugriffsrechte bezüglich BTSS ($APRB$, $APWB$)

Die Zugriffsrechte ($APRB$, $APWB$) beschränken den Zugriff auf System- und Anwendervariablen über BTSS für alle Systemkomponenten (HMI, PLC, externe Rechner, EPS-Dienste, etc.) gleichermaßen.

Hinweis

HMI-lokale Zugriffsrechte

Bei Änderungen von Zugriffsrechten von Systemdaten muss darauf geachtet werden, dass diese konsistent zu den über HMI-Mechanismen festgelegten Zugriffsrechten erfolgt.

Zugriffsattribute APR / APW

Aus Kompatibilitätsgründen werden die Attribute APR und APW implizit auf die Attribute $APRP / APRB$ und $APWP / APWB$ abgebildet:

- $APR\ x \Rightarrow APRP\ x\ APRB\ x$
- $APW\ y \Rightarrow APWP\ y\ APWB\ y$

Zugriffsrechte über ACCESS-Dateien einstellen

Bei der Verwendung von ACCESS-Dateien für die Vergabe von Zugriffsrechten, dürfen Redefinitionen von Zugriffsrechten für Systemdaten, Anwenderdaten und NC-Sprachbefehlen nur noch in diesen ACCESS-Dateien programmiert werden. Eine Ausnahme bilden globale Anwenderdaten (GUD). Für diese muss, falls dies erforderlich erscheint, die Redefinition der Zugriffsrechte weiterhin in den entsprechenden Definitionsdateien programmiert werden.

Für einen durchgehenden Zugriffsschutz müssen die Maschinendaten für die Ausführungsrechte und den Zugriffsschutz der entsprechenden Verzeichnisse konsistenten angepasst werden.

Es ergibt sich folgende prinzipielle Vorgehensweise:

- Erstellen der benötigten Definitionsdateien:
 - `_N_DEF_DIR/_N_SACCESS_DEF`
 - `_N_DEF_DIR/_N_MACCESS_DEF`
 - `_N_DEF_DIR/_N_UACCESS_DEF`
- Parametrieren des Schreibrechts für die Definitionsdateien auf den für die Redefinition erforderlichen Wert:
 - `MD11170 $MN_ACCESS_WRITE_SACCESS`
 - `MD11171 $MN_ACCESS_WRITE_MACCESS`
 - `MD11172 $MN_ACCESS_WRITE_UACCESS`
- Für Zugriffe auf geschützte Elemente aus Zyklen heraus müssen die Ausführungs- und Schreibrechte der Zyklenverzeichnisse `_N_CST_DIR`, `_N_CMA_DIR` und `_N_CST_DIR` angepasst werden:

Ausführungsrechte

- `MD11160 $MN_ACCESS_EXEC_CST`
- `MD11161 $MN_ACCESS_EXEC_CMA`
- `MD11162 $MN_ACCESS_EXEC_CUS`

Schreibrechte

- `MD11165 $MN_ACCESS_WRITE_CST`
- `MD11166 $MN_ACCESS_WRITE_CMA`
- `MD11167 MN_ACCESS_WRITE_CUS`

Das Ausführungsrecht muss mindestens auf die gleiche Schutzstufe wie die höchste Schutzstufe des verwendeten Elements gesetzt werden.

Das Schreibrecht muss mindestens auf die gleiche Schutzstufe wie das Ausführungsrecht gesetzt werden.

- Die Schreibrechte der HMI-lokalen Zyklenverzeichnisse müssen auf die gleiche Schutzstufe wie die der NC-lokalen Zyklenverzeichnisse gesetzt werden.

Literatur:

Bedienhandbuch

Unterprogrammaufrufe in ACCESS-Dateien

Für die weitere Strukturierung des Zugriffsschutzes können in den ACCESS-Dateien auch Unterprogramme (Kennung SPF oder MPF) aufgerufen werden. Die Unterprogramme erben dabei die Ausführungsrechte der aufrufenden ACCESS-Datei.

Hinweis

In den ACCESS-Dateien können nur die Zugriffsrechte redefiniert werden. Alle anderen Attribute müssen weiterhin in den entsprechenden Definitionsdateien programmiert bzw. redefiniert werden.

Siehe auch

Variablen (Seite 17)

1.1.10 Übersicht definierbarer und redefinierbarer Attribute

Die folgenden Tabellen zeigen bei welchen Datenarten welche Attribute definiert (DEF) und/oder redefiniert (REDEF) werden können.

Systemdaten

Datenart	Init.Wert	Grenzwerte	phys. Einheit	Zugriffsrechte
Maschinendaten	---	---	---	REDEF
Settingdaten	REDEF	---	---	REDEF
FRAME-Daten	---	---	---	REDEF
Prozessdaten	---	---	---	REDEF
Spindelsteigungsfehlerkomp. (EEC)	---	---	---	REDEF
Durchhangkompensation (CEC)	---	---	---	REDEF
Quadrantenfehlerkompensation (QEC)	---	---	---	REDEF
Magazindaten	---	---	---	REDEF
Werkzeugdaten	---	---	---	REDEF
Schutzbereiche	---	---	---	REDEF
orientierbare Werkzeugträger	---	---	---	REDEF
kinematische Ketten	---	---	---	REDEF
3D-Schutzbereiche	---	---	---	REDEF
Arbeitsfeldbegrenzung	---	---	---	REDEF
ISO-Werkzeugdaten	---	---	---	REDEF

Anwenderdaten

Datenart	Init.Wert	Grenzwerte	phys. Einheit	Zugriffsrechte
R-Parameter	REDEF	REDEF	REDEF	REDEF
Synchronaktionsvariable (\$AC_...)	REDEF	REDEF	REDEF	REDEF
Synchronaktions-GUD (SYG_...)	REDEF	REDEF	REDEF	REDEF
EPS-Parameter	REDEF	REDEF	REDEF	REDEF
Werkzeugdaten-OEM	REDEF	REDEF	REDEF	REDEF
Magazindaten-OEM	REDEF	REDEF	REDEF	REDEF
globale Anwendervariablen (GUD)	DEF / REDEF	DEF	DEF	DEF / REDEF
lokale Anwendervariablen (PUD / LUD)	DEF	DEF	DEF	---

Siehe auch

Variablen (Seite 17)

1.1.11 Definition und Initialisierung von Feldvariablen (DEF, SET, REP)

Funktion

Eine Anwendervariable kann als 1- bis maximal 3-dimensionales Feld (Array) definiert werden:

- 1-dimensional: DEF <Datentyp> <Variablenname> [<n>]
- 2-dimensional: DEF <Datentyp> <Variablenname> [<n>, <m>]
- 3-dimensional: DEF <Datentyp> <Variablenname> [<n>, <m>, <o>]

Hinweis

Anwendervariable vom Datentyp STRING können maximal als 2-dimensionales Feld definiert werden.

Datentypen

Anwendervariable können als Felder für folgende Datentypen definiert werden: BOOL, CHAR, INT, REAL, STRING, AXIS, FRAME

Wertzuweisung an Feldelemente

Wertzuweisungen an Feldelemente können zu folgenden Zeitpunkten vorgenommen werden:

- bei der Felddefinition (Initialisierungswerte)
- während des Programmablaufs

Wertzuweisung können dabei erfolgen über:

- explizite Angabe eines Feldelements
- explizite Angabe eines Feldelements als Startelement und Angabe einer Werteliste (`SET`)
- explizite Angabe eines Feldelements als Startelement und Angabe eines Wertes und der Häufigkeit seiner Wiederholung (`REP`)

Hinweis

Anwendervariablen vom Datentyp `FRAME` können keine Initialisierungswerte zugewiesen werden.

Syntax (`DEF`)

```
DEF <Datentyp> <Variablenname> [<n>, <m>, <o>]
DEF  STRING [<Stringlänge>] <Variablenname> [<n>, <m>]
```

Syntax (`DEF...=SET...`)

Verwendung einer Werteliste:

- bei der Definition:

```
DEF <Datentyp> <Variablenname> [<n>, <m>, <o>] = SET(<Wert1>, <Wert2>, ...)
```

gleichbedeutend mit:

```
DEF <Datentyp> <Variablenname> [<n>, <m>, <o>] = (<Wert1>, <Wert2>, ...)
```

Hinweis

Bei der Initialisierung über eine Werteliste ist die Angabe von `SET` optional.

- bei einer Wertzuweisung:

```
<Variablenname> [<n>, <m>, <o>] = SET(<WERT1>, <Wert2>, ...)
```

Syntax (`DEF...=REP...`)

Verwendung eines Werte mit Wiederholung

- bei der Definition:

```
DEF <Datentyp> <Variablenname> [<n>, <m>, <o>] = REP(<Wert>)
```

```
DEF <Datentyp> <Variablenname> [<n>, <m>, <o>] = REP(<Wert>, <Anzahl_Feldelemente>)
```

- bei einer Wertzuweisung:

```
<Variablenname> [<n>, <m>, <o>] = REP(<Wert>)
```

```
<Variablenname> [<n>, <m>, <o>] = REP(<Wert>, <Anzahl_Feldelemente>)
```


Bedeutung

DEF:	Befehl zur Definition von Variablen
<Datentyp>:	Datentyp der Variablen
	Wertebereich:
	<ul style="list-style-type: none"> • bei Systemvariablen: BOOL, CHAR, INT, REAL, STRING, AXIS • bei GUD- oder LUD-Variablen: BOOL, CHAR, INT, REAL, STRING, AXIS, FRAME
<Stringlänge>:	Maximale Anzahl der Zeichen beim Datentyp STRING
<Variablenname>:	Variablenname
[<n>, <m>, <o>]:	Feldgrößen bzw. Feldindizes
<n>:	Feldgröße bzw. Feldindex für 1. Dimension
	Typ: INT (bei Systemvariablen auch AXIS)
	Wertebereich: Max. Feldgröße: 65535 Feldindex: $0 \leq n \leq 65534$
<m>:	Feldgröße bzw. Feldindex für 2. Dimension
	Typ: INT (bei Systemvariablen auch AXIS)
	Wertebereich: Max. Feldgröße: 65535 Feldindex: $0 \leq m \leq 65534$
<o>:	Feldgröße bzw. Feldindex für 3. Dimension
	Typ: INT (bei Systemvariablen auch AXIS)
	Wertebereich: Max. Feldgröße: 65535 Feldindex: $0 \leq o \leq 65534$
SET:	Wertzuweisung über die angegebenen Werteliste
(<Wert1>, <Wert2>, ...):	Werteliste
REP:	Wertzuweisung über den angegebenen <Wert>
<Wert>:	Wert, mit dem die Feldelemente bei der Initialisierung mit REP beschrieben werden sollen.
<Anzahl_Feldelemente>:	Anzahl der Feldelemente, die mit dem angegebenen <Wert> beschrieben werden sollen. Für die restlichen Feldelemente gilt abhängig vom Zeitpunkt:
	<ul style="list-style-type: none"> • Initialisierung bei der Felddefinition: → Die restlichen Feldelemente werden mit Null beschrieben • Zuweisung während des Programmlaufs: → Die aktuellen Werte der Feldelemente bleiben unverändert.
	Ist der Parameter nicht programmiert, werden alle Feldelemente mit <Wert> beschrieben.
	Ist der Parameter gleich Null, gilt abhängig vom Zeitpunkt:
	<ul style="list-style-type: none"> • Initialisierung bei der Felddefinition: → Alle Elemente werden mit Null vorbelegt • Zuweisung während des Programmlaufs: → Die aktuellen Werte der Feldelemente bleiben unverändert.

Feldindex

Die implizite Reihenfolge der Feldelemente z.B. bei einer Wertzuweisung über SET oder REP erfolgt durch Iteration der Feldindizes von rechts nach links.

Beispiel: Initialisierung eines 3-dimensionalen Feldes mit 24 Feldelementen:

```
DEF INT FELD[2,3,4] = REP(1,24)
  FELD[0,0,0] = 1      1. Feldelement
  FELD[0,0,1] = 1      2. Feldelement
  FELD[0,0,2] = 1      3. Feldelement
  FELD[0,0,3] = 1      4. Feldelement
  ...
  FELD[0,1,0] = 1      5. Feldelement
  FELD[0,1,1] = 1      6. Feldelement
  ...
  FELD[0,2,3] = 1      12. Feldelement
  FELD[1,0,0] = 1      13. Feldelement
  FELD[1,0,1] = 1      14. Feldelement
  ...
  FELD[1,2,3] = 1      24. Feldelement
```

entsprechend:

```
FOR n=0 TO 1
  FOR m=0 TO 2
    FOR o=0 TO 3
      FELD[n,m,o] = 1
    ENDFOR
  ENDFOR
ENDFOR
```

Beispiel: Initialisierung kompletter Variablenfelder

Aktuelle Belegung siehe Abbildung.

Programmcode

```

N10 DEF REAL FELD1 [10,3]=SET(0,0,0,10,11,12,20,20,20,30,30,30,40,40,40,)
N20 FELD1 [0,0]=REP(100)
N30 FELD1 [5,0]=REP(-100)
N40 FELD1 [0,0]=SET(0,1,2,-10,-11,-12,-20,-20,-20,-30, , , , -40,-40,-50,-60,-70)
N50 FELD1 [8,1]=SET(8.1,8.2,9.0,9.1,9.2)

```

Feldindex				2					
[1,2]	N10: Initialisierung bei Definition			N20/N30: Initialisierung mit identischem Wert			N40/N50: Initialisierung mit verschiedenen Werten		
	0	1	2	0	1	2	0	1	2
0	0	0	0	100	100	100	0	1	2
1	10	11	12	100	100	100	-10	-11	-12
2	20	20	20	100	100	100	-20	-20	-20
3	30	30	30	100	100	100	-30	0	0
4	40	40	40	100	100	100	0	-40	-40
5	0	0	0	-100	-100	-100	-50	-60	-70
6	0	0	0	-100	-100	-100	-100	-100	-100
7	0	0	0	-100	-100	-100	-100	-100	-100
8	0	0	0	-100	-100	-100	-100	8.1	8.2
9	0	0	0	-100	-100	-100	9.0	9.1	9.2
	Die Feldelemente [5,0] bis [9,2] wurden mit dem Defaultwert (0.0) initialisiert.						Die Feldelemente [3,1] bis [4,0] wurden mit dem Defaultwert (0.0) initialisiert. Die Feldelemente [6,0] bis [8,0] wurden nicht verändert.		
1									

Siehe auch

Definition und Initialisierung von Feldvariablen (DEF, SET, REP): Weitere Informationen (Seite 52)

Variablen (Seite 17)

1.1.12 Definition und Initialisierung von Feldvariablen (DEF, SET, REP): Weitere Informationen

Weitere Informationen (SET)

Initialisierung bei der Definition

- Es werden, beginnend beim 1. Feldelement, so viele Feldelemente mit den Werten aus der Werteliste initialisiert, wie Elemente in der Werteliste programmiert sind.
- Feldelemente ohne explizit angegebene Werte in der Werteliste (Lücken in der Werteliste) werden mit 0 belegt.
- Bei Variablen vom Datentyp AXIS sind Lücken in der Werteliste nicht zugelassen.
- Enthält die Werteliste mehr Werte als Feldelemente definiert sind, wird ein Alarm angezeigt.

Wertzuweisung im Programmablauf

Bei der Wertzuweisung im Programmablauf gelten die oben bei der Definition beschriebenen Regeln. Zusätzlich gibt es folgende Möglichkeiten:

- Als Elemente in der Werteliste sind auch Ausdrücke erlaubt.
- Die Wertzuweisung beginnt bei dem programmierten Feldindex. Hierdurch lassen sich gezielt Teilfelder mit Werten belegen.

Beispiel:

Programmcode	Kommentar
DEF INT FELD[5,5]	; Felddefinition
FELD[0,0]=SET(1,2,3,4,5)	; Wertzuweisung an die ersten 5 Feldelemente [0,0] - [0,4]
FELD[0,0]=SET(1,2, , ,5)	; Wertzuweisung mit Lücke an die ersten 5 Feldelemente [0,0] - [0,4], Feldelemente [0,2] und [0,3] = 0
FELD[2,3]=SET(VARIABLE,4*5.6)	; Wertzuweisung mit Variable und Ausdruck ab Feldindex [2,3]: [2,3] = VARIABLE [2,4] = 4 * 5.6 = 22.4

Weitere Informationen (REP)

Initialisierung bei der Definition

- Alle oder die optional angegebene Anzahl an Feldelementen werden mit dem angegebenen Wert (Konstante) initialisiert.
- Variablen vom Datentyp FRAME können nicht initialisiert werden.

Beispiel:

Programmcode	Kommentar
DEF REAL varName[10]=REP(3.5,4)	; Felddefinition und Feldelemente [0] bis [3] mit Wert 3,5 initialisieren

Wertzuweisung im Programmablauf

Bei der Wertzuweisung im Programmablauf gelten die oben bei der Definition beschriebenen Regeln. Zusätzlich gibt es folgende Möglichkeiten:

- Als Elemente in der Werteliste sind auch Ausdrücke erlaubt.
- Die Wertzuweisung beginnt bei dem programmierten Feldindex. Hierdurch lassen sich gezielt Teilfelder mit Werten belegen.

Beispiele:

Programmcode	Kommentar
DEF REAL varName[10]	; Felddefinition
varName[5]=REP(4.5,3)	; Feldelemente [5] bis [7] = 4,5
R10=REP(2.4,3)	; R-Parameter R10 bis R12 = 2,4
DEF FRAME FRM[10]	; Felddefinition
FRM[5]=REP(CTRANS(X,5))	; Feldelemente [5] bis [9] = CTRANS(X,5)

Weitere Informationen (allgemein)

Wertzuweisungen an axiale Maschinendaten

Axiale Maschinendaten haben prinzipiell einen Feldindex vom Datentyp `AXIS`. Bei Wertzuweisungen an ein axiales Maschinendatum mittels `SET` oder `REP` wird dieser Feldindex ignoriert bzw. nicht durchlaufen.

Beispiel: Wertzuweisung an Maschinendatum MD36200 `$MA_AX_VELO_LIMIT`

```
$MA_AX_VELO_LIMIT[1,AX1]=SET(1.1, 2.2, 3.3)
```

Entspricht:

```
$MA_AX_VELO_LIMIT[1,AX1]=1.1
```

```
$MA_AX_VELO_LIMIT[2,AX1]=2.2
```

```
$MA_AX_VELO_LIMIT[3,AX1]=3.3
```

ACHTUNG

Wertzuweisungen an axiale Maschinendaten

Bei Wertzuweisungen an axiale Maschinendaten mittels `SET` oder `REP` wird der Feldindex vom Datentyp `AXIS` ignoriert bzw. nicht durchlaufen.

Speicherbedarf

Datentyp	Speicherbedarf pro Element
BOOL	1 Byte
CHAR	1 Byte
INT	4 Bytes
REAL	8 Bytes
STRING	(Stringlänge + 1) Bytes
FRAME	~ 400 Bytes, abhängig von Achszahl
AXIS	4 Bytes

Siehe auch

Definition und Initialisierung von Feldvariablen (DEF, SET, REP) (Seite 47)

1.1.13 Datentypen

Folgende Datentypen stehen in der NC zur Verfügung:

Datentyp	Bedeutung	Wertebereich
INT	ganzzahliger Wert mit Vorzeichen	-2147483648 ... +2147483647
REAL	Real-Zahl (LONG REAL nach IEEE)	$\pm(\sim 2,2 \cdot 10^{-308} \dots \sim 1,8 \cdot 10^{308})$
BOOL	Wahrheitswert TRUE (1) und FALSE (0)	1, 0
CHAR	ASCII-Zeichen	ASCII-Code 0 ... 255
STRING	Zeichenkette definierter Länge	maximal 200 Zeichen (keine Sonderzeichen)
AXIS	Achs-/Spindelbezeichner	Kanalachsbezeichner
FRAME	Geometrische Angaben für eine statische Koordinatentransformation (Verschieben, Drehen, Skalieren, Spiegeln)	---

Implizite Datentypwandlungen

Folgende Datentypwandlungen sind möglich und werden bei Zuweisungen und Parameterübergaben implizit vorgenommen:

von ↓ / nach →	REAL	INT	BOOL
REAL	x	o	&
INT	x	x	&
BOOL	x	x	x

x: ohne Einschränkungen möglich
o: Datenverlust durch Überschreitung des Wertebereichs möglich ⇒ Alarm;
Rundung: Nachkommawert $\geq 0,5$ ⇒ aufrunden, Nachkommawert $< 0,5$ ⇒ abrunden
&: Wert $\neq 0$ ⇒ TRUE, Wert $= 0$ ⇒ FALSE

Siehe auch

Variablen (Seite 17)

1.1.14 Explizite Datentypwandlungen (AXTOINT, INTTOAX)

Funktion

Mit den vordefinierten Funktionen AXTOINT und INTTOAX kann der Datentyp einer Achsvariablen explizit gewandelt werden.

Typwandlung AXIS → INT

Syntax:

<Ergebnis>=AXTOINT (<Wert>)

Bedeutung:

<Ergebnis>:	INT-Darstellung der Achsvariablen (≙ Achsindex <n>)	
	Im Fehlerfall:	
	= 7	NO_AXIS, d. h. <Wert> enthält den Wert "keine Achse"
	= -1	<Wert> ist kein Achsname vom Typ AXIS
AXTOINT:	AXTOINT wandelt den Datentyp einer Achsvariablen von AXIS zu INT	
<Wert>:	Geometrieachsname (MD20060 \$MC_AXCONF_GEOAX_NAME_TAB[<n>]) oder	
	Kanalachsname (MD20080 \$MC_AXCONF_CHANAX_NAME_TAB[<n>]) oder	
	Maschinenachsname (MD10000 \$MN_AXCONF_MACHAX_NAME_TAB[<n>])	
	Datentyp:	AXIS

Typwandlung INT → AXIS

Syntax:

<Ergebnis>=INTTOAX (<Wert>)

<Ergebnis>:	AXIS-Darstellung der Achsvariablen (≙ Achsname)	
	Im Fehlerfall:	
	= NO_AXIS	<Wert> enthält den Wert "keine Achse"?
	= -1	<Wert> ist ein INT-Wert, zu dem es keinen Achsnamen vom Typ AXIS gibt?
INTTOAX:	INTTOAX wandelt den Datentyp einer Achsvariablen von INT zu AXIS	
<Wert>:	INT-Wert der Achsvariablen	
	Wertebereich:	0 - 32

Beispiel

Siehe Beispiel zu GETVARDFT unter "Attributwerte/Datentyp lesen (GETVARPHU, GETVARAP, GETVARLIM, GETVARDFT, GETVARTYP) (Seite 59)".

1.1.15 Vorhandensein einer Variablen prüfen (ISVAR)

Funktion

Mit der vordefinierten Funktion ISVAR kann geprüft werden, ob eine System-/Anwendervariable (z. B. Maschinendatum, Settingdatum, Systemvariable, allgemeine Variable wie GUD) im NCK bekannt ist.

Syntax

`<Ergebnis>=ISVAR(<Variable>)`

Der Übergabeparameter `<Variable>` kann wie folgt aufgebaut sein:

- dimensionslose \$-Variable: `$<Bezeichner>`
- eindimensionale \$-Variable ohne Feldindex: `$<Bezeichner> []`
- eindimensionale \$-Variable mit Feldindex: `$<Bezeichner> [<n>]`
- zweidimensionale \$-Variable ohne Feldindex: `$<Bezeichner> [,]`
- zweidimensionale \$-Variable mit Feldindex: `$<Bezeichner> [<n>, <m>]`

Bedeutung

<code><Ergebnis></code> :	Rückgabewert	
	Datentyp:	BOOL
	Wertebereich:	1 Variable vorhanden 0 Variable unbekannt
ISVAR:	Prüft, ob System-/Anwendervariable im NCK bekannt ist	
<code><Bezeichner></code> :	Name der System- /Anwendervariablen	
	Datentyp:	STRING
<code><n></code> :	Feldindex für erste Dimension	
	Datentyp:	INT
<code><m></code> :	Feldindex für zweite Dimension	
	Datentyp:	INT

Entsprechend dem Übergabeparameter werden folgende Prüfungen durchgeführt:

- Ist der Bezeichner vorhanden
- Ist es ein ein- oder zweidimensionales Feld
- Ist der Feldindex erlaubt

Nur wenn alle diese Prüfungen positiv sind, wird TRUE (1) zurückgeliefert. Wird nur eine Prüfung nicht erfüllt oder ist ein Syntaxfehler aufgetreten, dann wird dies mit FALSE (0) quittiert.

Beispiele

Programmcode	Kommentar
DEF INT VAR1	
DEF BOOL IS_VAR=FALSE	
N10 IS_VAR=ISVAR("VAR1")	; IS_VAR ist in diesem Fall TRUE.

Programmcode	Kommentar
DEF REAL VARARRAY[10,10]	
DEF BOOL IS_VAR=FALSE	
N10 IS_VAR=ISVAR("VARARRAY[,]")	; IS_VAR ist in diesem Fall TRUE, ist ein zweidimensionales Feld.
N20 IS_VAR=ISVAR("VARARRAY")	; IS_VAR ist TRUE, Variable existiert.
N30 IS_VAR=ISVAR("VARARRAY[8,11]")	; IS_VAR ist FALSE, Feldindex ist nicht erlaubt.
N40 IS_VAR=ISVAR("VARARRAY[8,8]")	; IS_VAR ist FALSE, "]" fehlt (Syntaxfehler).
N50 IS_VAR=ISVAR("VARARRAY[,8]")	; IS_VAR ist TRUE, Feldindex ist erlaubt.
N60 IS_VAR=ISVAR("VARARRAY[8,]")	; IS_VAR ist TRUE, Feldindex ist erlaubt.

Programmcode	Kommentar
DEF BOOL IS_VAR=FALSE	
N100 IS_VAR=ISVAR("\$MC_GCODE_RESET_VALUES[1]")	; Übergabeparameter ist ein Maschinendatum, IS_VAR ist TRUE.

Programmcode	Kommentar
DEF BOOL IS_VAR=FALSE	
N10 IS_VAR=ISVAR("\$P_EP")	; IS_VAR ist in diesem Fall TRUE.
N20 IS_VAR=ISVAR("\$P_EP[X]")	; IS_VAR ist in diesem Fall TRUE.

1.1.16 Attributwerte/Datentyp lesen (GETVARPHU, GETVARAP, GETVARLIM, GETVARDFT, GETVARTYP)

Mit den vordefinierten Funktionen GETVARPHU, GETVARAP, GETVARLIM und GETVARDFT können die Attributwerte von System-/Anwendervariablen gelesen werden, mit GETVARTYP der Datentyp einer System-/Anwendervariablen.

Physikalische Einheit lesen

Syntax:

<Ergebnis>=GETVARPHU (<Name>)

Bedeutung:

<Ergebnis>:	Zahlenwert der physikalischen Einheit	
	Datentyp:	INT
	Wertebereich:	siehe Tabelle in "Attribut: Physikalische Einheit (PHU) (Seite 39)"
		Im Fehlerfall:
	- 2	Der angegebene <Name> ist keinem Systemparameter und keiner Anwendervariablen zugeordnet.
GETVARPHU:	Lesen der physikalischen Einheit einer System- /Anwendervariablen	
<Name>:	Name der System- /Anwendervariablen	
	Datentyp:	STRING

Beispiel:

Der NCK enthalte folgende GUD-Variable:

```
DEF CHAN REAL PHU 42 LLI 0 ULI 10000 electric
```

Programmcode	Kommentar
DEF INT result=0	
result=GETVARPHU("electric")	; Ermittle die physikalische Einheit der GUD-Variablen.
IF (result < 0) GOTO error	

Als Ergebnis wird der Wert 42 zurückgeliefert. Dies entspricht der physikalischen Einheit [kW].

Hinweis

Mit GETVARPHU lässt sich z. B. prüfen, ob bei einer Variablenzuweisung a = b beide Variablen die erwarteten physikalischen Einheiten haben.

Zugriffsrecht lesen

Syntax:

<Ergebnis>=GETVARAP (<Name>, <Zugriff>)

Bedeutung:

<Ergebnis>:	Schutzstufe für den angegebenen <Zugriff>		
	Datentyp:	INT	
	Wertebereich:	0 ... 7	Siehe "Attribut: Zugriffsrechte (APR, APW, APRP, APWP, APRB, APWB) (Seite 41)".
		Im Fehlerfall:	
		- 2	Der angegebene <Name> ist keinem Systemparameter und keiner Anwendervariablen zugeordnet.
- 3	falscher Wert für <Zugriff>		
GETVARAP:	Lesen des Zugriffsrechts auf eine System- /Anwendervariable		
<Name>:	Name der System- /Anwendervariablen		
	Datentyp:	STRING	
<Zugriff>:	Art des Zugriffs		
	Datentyp:	STRING	
	Wertebereich:	"RP":	Lesen über Teileprogramm
		"WP":	Schreiben über Teileprogramm
		"RB":	Lesen über BTSS
"WB":		Schreiben über BTSS	

Beispiel:

Programmcode	Kommentar
DEF INT result=0	
result=GETVARAP (" \$TC_MAP8 ", "WB")	; Ermittle den Zugriffsschutz für den Systemparameter "Magazinposition" bezüglich Schreiben über BTSS.
IF (result < 0) GOTOF error	

Als Ergebnis wird der Wert 7 zurückgeliefert. Dies entspricht der Schlüsselschalter-Stellung 0 (= kein Zugriffsschutz).

Hinweis

Mit GETVARAP kann z. B. ein Prüfprogramm realisiert werden, das die von der Applikation erwarteten Zugriffsrechte prüft.

Grenzwerte lesen

Syntax:

<Status>=GETVARLIM(<Name>, <Grenzwert>, <Ergebnis>)

Bedeutung:

<Status>:	Funktionsstatus		
	Datentyp:	INT	
	Wertebereich:	1	o. k.
		-1	kein Grenzwert definiert (bei Variablen vom Typ AXIS, STRING, FRAME)
		-2	Der angegebene <Name> ist keinem Systemparameter und keiner Anwendervariablen zugeordnet.
-3		falscher Wert für <Grenzwert>	
GETVARLIM:	Lesen des unteren/oberen Grenzwerts einer System- /Anwendervariablen		
<Name>:	Name der System- /Anwendervariablen		
	Datentyp:	STRING	
<Grenzwert>:	Gibt an, welcher Grenzwert ausgelesen werden soll		
	Datentyp:	CHAR	
	Wertebereich:	"L":	= unterer Grenzwert
		"U":	= oberer Grenzwert
<Ergebnis>:	Rückgabe des Grenzwerts		
	Datentyp:	VAR REAL	

Beispiel:

Programmcode	Kommentar
DEF INT state=0	
DEF REAL result=0	
state=GETVARLIM("\$MA_MAX_AX_VELO", "L", result)	; Ermittle den unteren Grenzwert für MD32000 \$MA_MAX_AX_VELO.
IF (result < 0) GOTO error	

Standardwert lesen

Syntax:

<Status>=GETVARDFT (<Name>, <Ergebnis> [, <Index_1>, <Index_2>, <Index_3>])

Bedeutung:

<Status>:	Funktionsstatus		
	Datentyp:	INT	
	Wertebereich:	1	o. k.
		-1	kein Standardwert verfügbar (z. B. weil <Ergebnis> den falschen Typ zu <Name> hat)
		-2	Der angegebene <Name> ist keinem Systemparameter und keiner Anwendervariablen zugeordnet.
		-3	falscher Wert für <Index_1>, Dimension kleiner eins (= kein Array = Skalar)
-4		falscher Wert für <Index_2>	
-5	falscher Wert für <Index_3>		
GETVARDFT:	Lesen des Standardwerts einer System- /Anwendervariablen		
<Name>:	Name der System-/Anwendervariablen		
	Datentyp:	STRING	
<Ergebnis>:	Rückgabe des Standardwerts		
	Datentyp:	VAR REAL (beim Lesen des Standardwerts von Variablen der Typen INT, REAL, BOOL, AXIS)	
		VAR STRING (beim Lesen des Standardwerts von Variablen der Typen STRING und CHAR)	
		VAR FRAME (beim Lesen des Standardwerts von Variablen des Typs FRAME)	
<Index_1>:	Index auf erste Dimension (optional)		
	Datentyp:	INT	
	Nicht programmiert bedeutet = 0		
<Index_2>:	Index auf zweite Dimension (optional)		
	Datentyp:	INT	
	Nicht programmiert bedeutet = 0		
<Index_3>:	Index auf dritte Dimension (optional)		
	Datentyp:	INT	
	Nicht programmiert bedeutet = 0		

Beispiel:

Programmcode	Kommentar
DEF INT state=0	
DEF REAL resultR=0	; Variable zur Aufnahme der Standardwerte der Typen INT, REAL, BOOL, AXIS.
DEF FRAME resultF=0	; Variable zur Aufnahme der Standardwerte des Typs FRAME
IF (GETVARTYP("\$MA_MAX_AX_VELO") <> 4) GOTO error	
state=GETVARDFT("\$MA_MAX_AX_VELO", resultR, AXTOINT(X))	; Ermittle den Standardwert der "X"-Achse. ; AXTOINT wandelt den Achsnamen "X" in den passenden Zugriffsindex.
IF (resultR < 0) GOTO error	
IF (GETVARTYP("\$TC_TP8") <> 3) GOTO error	
state=GETVARDFT("\$TC_TP8", resultR)	
IF (GETVARTYP("\$P_UBFR") <> 7) GOTO error	
state=GETVARDFT("\$P_UBFR", resultF)	

Datentyp lesen

Syntax:

<Ergebnis>=GETVARTYP (<Name>)

Bedeutung:

<Ergebnis>:	Datentyp der angegebenen System- /Anwendervariablen			
	Datentyp:	INT		
	Wertebereich:	1	= BOOL	
		2	= CHAR	
		3	= INT	
		4	= REAL	
		5	= STRING	
		6	= AXIS	
		7	= FRAME	
	Im Fehlerfall:			
< 0	Der angegebene <Name> ist keinem Systemparameter und keiner Anwendervariablen zugeordnet.			
GETVARTYP:	Lesen des Datentyps einer System- /Anwendervariablen			
<Name>:	Name der System- /Anwendervariablen			
	Datentyp:	STRING		

Beispiel:

Programmcode	Kommentar
DEF INT result=0	
DEF STRING name="R"	
result=GETVARTYP(name)	; Ermittle den Typ des R-Parameters.
IF (result < 0) GOTOF error	

Als Ergebnis wird der Wert 4 zurückgeliefert. Dies entspricht dem Datentyp REAL.

1.2 Indirekte Programmierung

1.2.1 Indirekte Programmierung von Adressen

Funktion

Bei der indirekten Programmierung von Adressen wird die erweiterte Adresse (<Index>) durch eine Variable geeigneten Typs ersetzt.

Hinweis

Die indirekte Programmierung von Adressen ist nicht möglich bei:

- N (Satznummer)
 - L (Unterprogramm)
 - Einstellbaren Adressen
(z. B. X[1] anstelle von X1 ist nicht zulässig)
-

Syntax

<ADRESSE> [<Index>]

Bedeutung

Element	Beschreibung
<ADRESSE> [...]:	Feste Adresse mit Erweiterung (Index)
<Index>:	Variable z. B. für Spindelnummer, Achse, ...

Beispiele

Beispiel 1: Indirekte Programmierung einer Spindelnummer

Direkte Programmierung:

Programmcode	Kommentar
S1=300	; Drehzahl 300 U/min für die Spindel mit Nummer 1.

Indirekte Programmierung:

Programmcode	Kommentar
DEF INT SPINU=1	; Definition der Variablen vom Typ INT und Wertzuweisung.
S[SPINU]=300	; Drehzahl 300 U/min für die Spindel, deren Nummer in der Variablen SPINU abgelegt ist (in diesem Beispiel die Spindel mit Nummer 1).

Beispiel 2: Indirekte Programmierung einer Achse

Direkte Programmierung:

Programmcode	Kommentar
FA[U]=300	; Vorschub 300 für die Achse "U".

Indirekte Programmierung:

Programmcode	Kommentar
DEF AXIS AXVAR2=U	; Definition einer Variablen vom Typ AXIS und Wertzuweisung.
FA[AXVAR2]=300	; Vorschub 300 für die Achse, deren Adressname in der Variablen mit dem Namen AXVAR2 abgelegt ist.

Beispiel 3: Indirekte Programmierung einer Achse

Direkte Programmierung:

Programmcode	Kommentar
\$AA_MM[X]	; Messtaster-Messwert (MKS) der Achse "X" lesen.

Indirekte Programmierung:

Programmcode	Kommentar
DEF AXIS AXVAR3=X	; Definition einer Variablen vom Typ AXIS und Wertzuweisung.
\$AA_MM[AXVAR3]	; Messtaster-Messwert (MKS) lesen für die Achse, deren Name in der Variablen AXVAR3 abgelegt ist.

Beispiel 4: Indirekte Programmierung einer Achse

Direkte Programmierung:

Programmcode	Kommentar
X1=100 X2=200	

Indirekte Programmierung:

Programmcode	Kommentar
DEF AXIS AXVAR1 AXVAR2	; Definition zweier Variablen vom Typ AXIS.
AXVAR1=(X1) AXVAR2=(X2)	; Zuweisung der Achsnamen.
AX[AXVAR1]=100 AX[AXVAR2]=200	; Verfahren der Achsen, deren Adressnamen in den Variablen mit den Namen AXVAR1 und AXVAR2 abgelegt sind.

Beispiel 5: Indirekte Programmierung einer Achse

Direkte Programmierung:

Programmcode
G2 X100 I20

Indirekte Programmierung:

Programmcode	Kommentar
DEF AXIS AXVAR1=X	; Definition einer Variablen vom Typ AXIS und Wertzuweisung.
G2 X100 IP[AXVAR1]=20	; Indirekte Programmierung der Mittelpunktsgabe für die Achse, deren Adressname in der Variablen mit dem Namen AXVAR1 abgelegt ist

Beispiel 6: Indirekte Programmierung von Feldelementen

Direkte Programmierung:

Programmcode	Kommentar
DEF INT FELD1[4,5]	; Definition von Feld 1.

Indirekte Programmierung:

Programmcode	Kommentar
DEFINE DIM1 AS 4	; Bei Felddimensionen müssen Feldgrößen als feste Werte angegeben werden.
DEFINE DIM2 AS 5	
DEF INT FELD[DIM1,DIM2]	
FELD[DIM1-1,DIM2-1]=5	

Beispiel 7: Indirekter Unterprogrammaufruf

Programmcode	Kommentar
CALL "L" << R10	; Aufruf des Programms, dessen Nummer in R10 steht (Stringverkettung).

1.2.2 Indirekte Programmierung von G-Codes

Funktion

Die indirekte Programmierung von G-Codes ermöglicht eine effektive Zyklenprogrammierung.

Syntax

G [<Gruppe>] = <Nummer>

Bedeutung

G [...]: G-Befehl mit Erweiterung (Index)
<Gruppe>: Index-Parameter: G-Funktionsgruppe
Typ: INT
<Nummer>: Variable für die G-Code-Nummer
Typ: INT oder REAL

Hinweis

Es können i. d. R. nur nicht-syntaxbestimmende G-Codes indirekt programmiert werden. Von den syntaxbestimmenden G-Codes sind nur die der G-Funktionsgruppe 1 möglich. Die syntaxbestimmenden G-Codes der G-Funktionsgruppen 2, 3 und 4 sind nicht möglich.

Hinweis

In der indirekten G-Code-Programmierung sind keine Arithmetik-Funktionen erlaubt. Eine notwendige Berechnung der G-Code-Nummer muss in einer eigenen Teileprogrammzeile vor der indirekten G-Code-Programmierung erfolgen.

Beispiele

Beispiel 1: Einstellbare Nullpunktverschiebung (G-Funktionsgruppe 8)

Programmcode	Kommentar
N1010 DEF INT INT_VAR	
N1020 INT_VAR=2	
...	
N1090 G[8]=INT_VAR G1 X0 Y0	; G54
N1100 INT_VAR=INT_VAR+1	; G-Code-Berechnung
N1110 G[8]=INT_VAR G1 X0 Y0	; G55

Beispiel 2: Ebenenanwahl (G-Funktionsgruppe 6)

Programmcode	Kommentar
N2010 R10=\$P_GG[6]	; Aktive G-Funktion der G-Funktions-Gruppe 6 lesen
...	
N2090 G[6]=R10	

Literatur

Informationen zu den G-Funktionsgruppen siehe:
Programmierhandbuch Grundlagen; Kapitel "G-Funktionsgruppen"

1.2.3 Indirekte Programmierung von Positionsattributen (GP)

Funktion

Positionsattribute, wie z. B. die inkrementelle oder absolute Programmierung der Achsposition, können in Verbindung mit dem Schlüsselwort `GP` indirekt als Variablen programmiert werden.

Anwendung

Die indirekte Programmierung von Positionsattributen findet Verwendung in **Ersetzungszyklen**, da hier folgender Vorteil gegenüber der Programmierung von Positionsattributen als Schlüsselwort (z. B. `IC`, `AC`, ...) besteht:

Durch die indirekte Programmierung als Variablen wird **keine** `CASE`-Anweisung benötigt, die über alle möglichen Positionsattribute verzweigt.

Syntax

```
<POSITIONIERBEFEHL> [<Achse/Spindel>]=GP (<Position>, <Positionsattribut>)  
<Achse/Spindel>=GP (<Position>, <Positionsattribut>)
```

Bedeutung

<POSITIONIERBEFEHL> []: Folgende Positionierbefehle können zusammen mit dem Schlüsselwort GP programmiert werden:
 POS, POSA, SPOS, SPOSA
 Außerdem möglich:

- alle im Kanal vorhandenen Achs-/Spindelbezeichner:
 <Achse/Spindel>
- variabler Achs-/Spindelbezeichner AX

<Achse/Spindel>: Achse/Spindel, die positioniert werden soll
 GP(): Schlüsselwort zur Positionierung
 <Position>: Parameter 1
 Achs-/Spindelposition als Konstante oder Variable
 <Positionsattribut>: Parameter 2
 Positionsattribut (z. B. Positionsanfahrmodus) als Variable (z. B. \$P_SUB_SPOSMODE) oder als Schlüsselwort (IC, AC, ...)

Die von den Variablen gelieferten Werte haben folgende Bedeutung:

Wert	Bedeutung	Zulässig bei:
0	Keine Änderung des Positionsattributs	
1	AC	POS, POSA, SPOS, SPOSA, AX, Achsadresse
2	IC	POS, POSA, SPOS, SPOSA, AX, Achsadresse
3	DC	POS, POSA, SPOS, SPOSA, AX, Achsadresse
4	ACP	POS, POSA, SPOS, SPOSA, AX, Achsadresse
5	ACN	POS, POSA, SPOS, SPOSA, AX, Achsadresse
6	OC	-
7	PC	-
8	DAC	POS, POSA, AX, Achsadresse
9	DIC	POS, POSA, AX, Achsadresse
10	RAC	POS, POSA, AX, Achsadresse
11	RIC	POS, POSA, AX, Achsadresse
12	CAC	POS, POSA
13	CIC	POS, POSA
14	CDC	POS, POSA
15	CACP	POS, POSA
16	CACN	POS, POSA

Beispiel

Bei einer aktiven Synchronspindelkopplung zwischen der Leitspindel S1 und der Folgespindel S2 wird durch den SPOS-Befehl im Hauptprogramm der folgende Ersetzungszyklus zur Positionierung der Spindeln aufgerufen.

Die Positionierung erfolgt über die Anweisung in N2230:
 SPOS [1]=GP (\$P_SUB_SPOSIT, \$P_SUB_SPOSMODE)
 SPOS [2]=GP (\$P_SUB_SPOSIT, \$P_SUB_SPOSMODE)

Die anzufahrende Position wird aus der Systemvariablen \$P_SUB_SPOSIT, der Positionsanfahrmodus wird aus der Systemvariablen \$P_SUB_SPOSMODE gelesen.

Programmcode	Kommentar
N1000 PROC LANG_SUB DISPLOF SBLOF	
...	
N2100 IF (\$P_SUB_AXFCT==2)	
N2110	; Ersetzung des SPOS / SPOSA / M19-Befehls bei aktiver Synchronspindelkopplung
N2185 DELAYFSTON	; Beginn Stopp-Delay-Bereich
N2190 COUPOF(S2,S1)	; Synchronspindelkopplung deaktivieren
N2200	; Leit- und Folgespindel positionieren
N2210 IF (\$P_SUB_SPOS==TRUE) OR (\$P_SUB_SPOSA==TRUE)	
N2220	; Spindel mit SPOS positionieren:
N2230 SPOS [1]=GP (\$P_SUB_SPOSIT, \$P_SUB_SPOSMODE)	
SPOS [2]=GP (\$P_SUB_SPOSIT, \$P_SUB_SPOSMODE)	
N2250 ELSE	
N2260	; Spindel mit M19 positionieren:
N2270 M1=19 M2=19	; Leit- und Folgespindel positionieren
N2280 ENDIF	
N2285 DELAYFSTOF	; Ende Stopp-Delay-Bereich
N2290 COUPON(S2,S1)	; Synchronspindelkopplung aktivieren
N2410 ELSE	
N2420	; Abfrage auf weitere Ersetzungen
...	
N3300 ENDIF	
...	
N9999 RET	

Randbedingungen

- In Synchronaktionen ist die indirekte Programmierung von Positionsattributen nicht möglich.

Literatur

Funktionshandbuch Grundfunktionen; BAG, Kanal, Programmbetrieb, Reset-Verhalten (K1), Kapitel: Ersetzung von NC-Funktionen durch Unterprogramme

1.2.4 Indirekte Programmierung von Teileprogrammzeilen (EXECSTRING)

Funktion

Mit dem Teileprogrammbehl `EXECSTRING` ist es möglich, eine zuvor erzeugte String-Variable als Teileprogrammzeile auszuführen.

Syntax

`EXECSTRING` wird in einer eigenen Teileprogrammzeile programmiert:
`EXECSTRING (<String-Variable>)`

Bedeutung

<code>EXECSTRING:</code>	Befehl zur Ausführung einer String-Variablen als Teileprogrammzeile
<code><String-Variable>:</code>	Variable vom Typ <code>STRING</code> , die die eigentlich auszuführende Teileprogrammzeile enthält

Hinweis

Mit `EXECSTRING` können alle Teileprogramm-Konstrukte abgesetzt werden, die im Programmteil eines Teileprogramms programmiert werden können. Ausgeschlossen sind damit `PROC-` und `DEF-`Anweisungen sowie generell die Verwendung in `INI-` und `DEF-`Dateien.

Beispiel

Programmcode	Kommentar
N100 DEF STRING[100] BLOCK	; Definition der String-Variablen zur Aufnahme der auszuführenden Teileprogrammzeile.
N110 DEF STRING[10] MFCT1="M7"	
...	
N200 EXECSTRING(MFCT1 << "M4711")	; Teileprogrammzeile "M7 M4711" ausführen.
...	
N300 R10=1	
N310 BLOCK="M3"	
N320 IF(R10)	
N330 BLOCK = BLOCK << MFCT1	
N340 ENDIF	
N350 EXECSTRING(BLOCK)	; Teileprogrammzeile "M3 M7" ausführen.

1.3 Rechenfunktionen

Funktion

Die Rechenfunktionen sind vorrangig für R-Parameter und Variable (oder Konstante und Funktionen) vom Typ REAL anwendbar. Zulässig sind auch die Typen INT und CHAR.

Operator / Rechenfunktion	Bedeutung
+	Addition
-	Subtraktion
*	Multiplikation
/	Division
	Achtung: (Typ INT)/(Typ INT)=(Typ REAL); Beispiel: 3/4 = 0.75
DIV	Division, für Variablentyp INT und REAL
	Achtung: (Typ INT)DIV(Typ INT)=(Typ INT); Beispiel: 3 DIV 4 = 0
MOD	Modulo-Division (nur für Typ INT) liefert Rest einer INT-Division Beispiel: 3 MOD 4 = 3
:	Kettungsoperator (bei FRAME-Variablen)
SIN ()	Sinus
COS ()	Cosinus
TAN ()	Tangens
ASIN ()	Arcussinus
ACOS ()	Arcuscosinus
ATAN2 (,)	Arcustangens2
SQRT ()	Quadratwurzel
ABS ()	Betrag
POT ()	2. Potenz (Quadrat)
TRUNC ()	ganzzahliger Teil Genauigkeiten bei Vergleichsbefehlen einstellbar mit TRUNC (siehe "Genauigkeitskorrektur bei Vergleichsfehlern (TRUNC) (Seite 77)")
ROUND ()	Runden auf ein Ganzzahliges
LN ()	natürlicher Logarithmus
EXP ()	Exponentialfunktion
MINVAL ()	kleinerer Wert zweier Variablen (siehe "Minimum, Maximum und Bereich von Variablen (MINVAL, MAXVAL, BOUND) (Seite 79)")
MAXVAL ()	größerer Wert zweier Variablen (siehe "Minimum, Maximum und Bereich von Variablen (MINVAL, MAXVAL, BOUND) (Seite 79)")

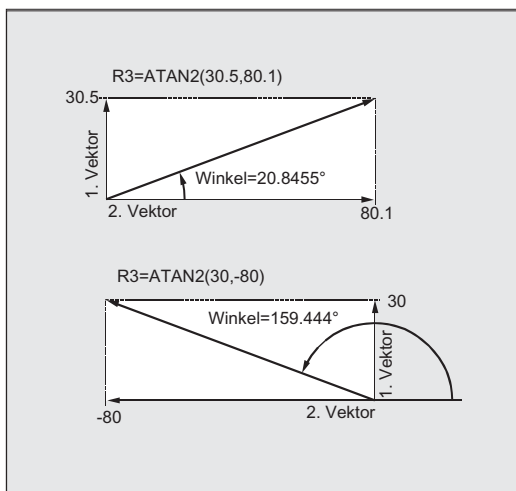
BOUND ()	Variablenwert, der im definierten Wertebereich liegt (siehe "Minimum, Maximum und Bereich von Variablen (MINVAL, MAXVAL, BOUND) (Seite 79)")
CTRANS ()	Verschiebung
CROT ()	Drehung
CSCALE ()	Maßstabsveränderung
CMIRROR ()	Spiegeln

Programmierung

Bei den Rechenfunktionen gilt die übliche mathematische Schreibweise. Prioritäten in der Abarbeitung werden durch runde Klammern gesetzt. Für die trigonometrischen und deren inverse Funktionen gilt die Gradangabe (rechter Winkel = 90°).

Beispiele

Beispiel 1: ATAN2



Die Rechenfunktion ATAN2 berechnet aus zwei aufeinander senkrecht stehenden Vektoren den Winkel des Summenvektors. Das Ergebnis liegt im Bereich von vier Quadranten (-180° < 0 < +180°). Basis für den Winkelbezug ist immer der 2. Wert in positiver Richtung.

Beispiel 2: Initialisierung kompletter Variablenfelder

Programmcode	Kommentar
R1=R1+1	; Neues R1 = altes R1 +1
R1=R2+R3 R4=R5-R6 R7=R8*R9	
R10=R11/R12 R13=SIN(25.3)	
R14=R1*R2+R3	; Punktrechnung geht vor Strichrechnung.
R14=(R1+R2)*R3	; Klammern werden zuerst berechnet.
R15=SQRT(POT(R1)+POT(R2))	; Innere Klammern werden zuerst aufgelöst: R15 = Quadratwurzel aus (R1+R2)
RESFRAME=FRAME1:FRAME2	; Mit dem Kettungsoperator werden Frames zu einem resultierenden Frame verknüpft oder den Frame-Komponenten Werte zugewiesen.
FRAME3=CTRANS(...):CROT(...)	

1.4 Vergleichs- und logische Operationen

Funktion

Vergleichsoperationen können z. B. zur Formulierung einer Sprungbedingung benutzt werden. Vergleichbar sind dabei auch komplexe Ausdrücke.

Die Vergleichsoperationen sind für Variable vom Typ `CHAR`, `INT`, `REAL` und `BOOL` anwendbar. Beim Typ `CHAR` wird der Codewert verglichen.

Bei den Typen `STRING`, `AXIS` und `FRAME` sind möglich: `==` und `<>`, die für Operationen vom Typ `STRING` auch in Synchronaktionen angewendet werden können.

Das Ergebnis von vergleichenden Operationen ist immer vom Typ `BOOL`.

Logische Operatoren dienen zur Verknüpfung von Wahrheitswerten.

Die logischen Operationen sind nur auf Variable vom Typ `BOOL` anwendbar. Über interne Typenkonvertierung sind sie auch auf die Datentypen `CHAR`, `INT`, und `REAL` anwendbar.

Bei den logischen (boolschen) Operationen gilt für die Datentypen `BOOL`, `CHAR`, `INT` und `REAL`:

- 0 entspricht: FALSE
- ungleich 0 entspricht: TRUE

Bitweise logische Operatoren

Mit den Variablen vom Typ `CHAR` und `INT` können auch bitweise logische Operationen vorgenommen werden. Gegebenenfalls erfolgt eine Typkonvertierung automatisch.

Programmierung

Vergleichsoperator	Bedeutung
<code>==</code>	gleich
<code><></code>	ungleich
<code>></code>	größer
<code><</code>	kleiner
<code>>=</code>	größer oder gleich
<code><=</code>	kleiner oder gleich

Logischer Operator	Bedeutung
<code>AND</code>	UND
<code>OR</code>	ODER
<code>NOT</code>	Negation
<code>XOR</code>	Exklusiv-ODER

Bitweise logischer Operator	Bedeutung
B_AND	bitweises UND
B_OR	bitweises ODER
B_NOT	bitweise Negation
B_XOR	bitweises Exklusiv-ODER

Hinweis

In arithmetischen Ausdrücken kann durch runde Klammern die Abarbeitungsreihenfolge aller Operatoren festgelegt und damit von den normalen Prioritätsregeln abgewichen werden.

Hinweis

Zwischen BOOLSCHEN Operanden und Operatoren müssen Zwischenräume geschrieben werden.

Hinweis

Der Operator `B_NOT` bezieht sich auf nur einen Operanden. Dieser steht nach dem Operator.

Beispiele

Beispiel 1: Vergleichsoperatoren

```
IF R10 >= 100 GOTOF ZIEL
```

oder

```
R11 = R10 >= 100  
IF R11 GOTOF ZIEL
```

Das Ergebnis des Vergleichs `R10 >= 100` wird zunächst in `R11` zwischengespeichert.

Beispiel 2: Logische Operatoren

```
IF (R10 < 50) AND ($AA_IM[X] >= 17.5) GOTOF ZIEL
```

oder

```
IF NOT R10 GOTOB START
```

`NOT` bezieht sich nur auf einen Operanden.

Beispiel 3: Bitweise logische Operatoren

```
IF $MC_RESET_MODE_MASK B_AND 'B10000' GOTOF ACT_PLANE
```

1.5 Genauigkeitskorrektur bei Vergleichsfehlern (TRUNC)

Funktion

Der TRUNC-Befehl schneidet den mit einem Genauigkeitsfaktor multiplizierten Operanden ab.

Einstellbare Genauigkeit bei Vergleichsbefehlen

Teileprogrammdateien vom Typ REAL werden intern im IEEE-Format mit 64 Bit dargestellt. Aufgrund dieser Darstellungsform können Dezimalzahlen ungenau abgebildet werden, die bei einem Vergleich mit ideal gerechneten Werten zu unerwarteten Ergebnissen führen können.

Relative Gleichheit

Damit die durch die Darstellungsform hervorgerufenen Ungenauigkeiten den Programmfluß nicht verfälschen, wird bei den Vergleichsbefehlen nicht auf absolute Gleichheit, sondern auf eine relative Gleichheit geprüft.

Syntax

Genauigkeitskorrektur bei Vergleichsfehlern

```
TRUNC (R1*1000)
```

Bedeutung

TRUNC: Abschneiden der Nachkommastellen

Berücksichtigte relative Gleichheit von 10^{-12} bei

- Gleichheit: (==)
- Ungleichheit: (<>)
- Größer-Gleich: (>=)
- Kleiner-Gleich: (<=)
- Größer/Kleiner: (><) mit absoluter Gleichheit
- Größer: (>)
- Kleiner: (<)

Kompatibilität

Aus Kompatibilitätsgründen kann die Prüfung auf relative Gleichheit bei (>) und (<) durch Setzen von Maschinendatum MD10280 \$MN_PROG_FUNCTION_MASK Bit0 = 1 deaktiviert werden.

Hinweis

Vergleiche mit Daten vom Typ REAL sind aus den genannten Gründen generell mit einer gewissen Ungenauigkeit behaftet. Bei nicht akzeptablen Abweichungen muss auf INTEGER-Rechnung ausgewichen werden, indem die Operanden mit einem Genauigkeitsfaktor multipliziert und danach mit TRUNC abgeschnitten werden.

Synchronaktionen

Das beschriebene Verhalten der Vergleichsbefehle gilt auch bei Synchronaktionen.

Beispiele**Beispiel 1: Genauigkeitsbetrachtungen**

Programmcode	Kommentar
N40 R1=61.01 R2=61.02 R3=0.01	; Zuweisung der Anfangswerte
N41 IF ABS(R2-R1) > R3 GOTOF FEHLER	; Sprung würde bisher ausgeführt werden
N42 M30	; Programmende
N43 FEHLER: SETAL(66000)	;
R1=61.01 R2=61.02 R3=0.01	; Zuweisung der Anfangswerte
R11=TRUNC(R1*1000) R12=TRUNC(R2*1000) R13=TRUNC(R3*1000)	; Genauigkeitskorrektur
IF ABS(R12-R11) > R13 GOTOF FEHLER	; Sprung wird nicht mehr ausgeführt
M30	; Programmende
FEHLER: SETAL(66000)	;

Beispiel 2: Quotient beider Operanden bilden und auswerten

Programmcode	Kommentar
R1=61.01 R2=61.02 R3=0.01	; Zuweisung der Anfangswerte
IF ABS((R2-R1)/R3) - 1 > 10EX-5 GOTOF FEHLER	; Sprung wird nicht ausgeführt
M30	; Programmende
FEHLER: SETAL(66000)	;

1.6 Minimum, Maximum und Bereich von Variablen (MINVAL, MAXVAL, BOUND)

Funktion

Mit den Befehlen `MINVAL` und `MAXVAL` können die Werte zweier Variablen miteinander verglichen werden. Als Ergebnis wird der kleinere Wert (bei `MINVAL`) bzw. größere Wert (bei `MAXVAL`) zurückgeliefert.

Mit dem Befehl `BOUND` kann geprüft werden, ob der Wert einer Prüfvariablen innerhalb eines definierten Wertebereichs liegt.

Syntax

```
<Kleinerer Wert>=MINVAL(<Variable1>,<Variable2>)
<Größerer Wert>=MAXVAL(<Variable1>,<Variable2>)
<Rückgabewert>=<BOUND>(<Minimum>,<Maximum>,<Prüfvariable>)
```

Bedeutung

<code>MINVAL:</code>	Ermittelt den kleineren Wert zweier Variablen (<code><Variable1></code> , <code><Variable2></code>)
<code><Kleinerer Wert>:</code>	Ergebnisvariable für den Befehl <code>MINVAL</code> Wird auf den kleineren Variablenwert gesetzt.
<code>MAXVAL:</code>	Ermittelt den größeren Wert zweier Variablen (<code><Variable1></code> , <code><Variable2></code>)
<code><Größerer Wert>:</code>	Ergebnisvariable für den Befehl <code>MAXVAL</code> Wird auf den größeren Variablenwert gesetzt.
<code>BOUND:</code>	Prüft, ob eine Variable (<code><Prüfvariable></code>) innerhalb eines definierten Wertebereichs liegt.
<code><Minimum>:</code>	Variable, die den Minimalwert des Wertebereichs definiert
<code><Maximum>:</code>	Variable, die den Maximalwert des Wertebereichs definiert
<code><Rückgabewert>:</code>	Ergebnisvariable für den Befehl <code>BOUND</code> Wenn der Wert der Prüfvariablen innerhalb des definierten Wertebereichs liegt, dann wird die Ergebnisvariable auf den Wert der Prüfvariablen gesetzt. Wenn der Wert der Prüfvariablen größer als der Maximalwert ist, dann wird die Ergebnisvariable auf den Maximalwert des Definitionsbereichs gesetzt. Wenn der Wert der Prüfvariablen kleiner als der Minimalwert ist, dann wird die Ergebnisvariable auf den Minimalwert des Definitionsbereichs gesetzt.

Hinweis

MINVAL, MAXVAL und BOUND können auch in Synchronaktionen programmiert werden.

Hinweis

Verhalten bei Gleichheit

Bei Gleichheit wird bei MINVAL/MAXVAL dieser gleiche Wert geliefert. Bei BOUND wird der Wert der zu prüfenden Variablen wieder zurückgegeben.

Beispiel

Programmcode	Kommentar
DEF REAL rVar1=10.5, rVar2=33.7, rVar3, rVar4, rVar5, rValMin, rValMax, rRetVar	
rValMin=MINVAL(rVar1,rVar2)	; rValMin wird auf den Wert 10.5 gesetzt.
rValMax=MAXVAL(rVar1,rVar2)	; rValMax wird auf den Wert 33.7 gesetzt.
rVar3=19.7	
rRetVar=BOUND(rVar1,rVar2,rVar3)	; rVar3 liegt innerhalb der Grenzen, rRetVar wird auf 19.7 gesetzt.
rVar3=1.8	
rRetVar=BOUND(rVar1,rVar2,rVar3)	; rVar3 liegt unterhalb der Minimumgrenze, rRetVar wird auf 10.5 gesetzt.
rVar3=45.2	
rRetVar=BOUND(rVar1,rVar2,rVar3)	; rVar3 liegt oberhalb der Maximumgrenze, rRetVar wird auf 33.7 gesetzt.

1.7 Priorität der Operationen

Funktion

Jedem Operator ist eine Priorität zugeordnet. Bei der Auswertung eines Ausdrucks werden stets die Operatoren höherer Priorität zuerst angewandt. Bei gleichrangigen Operatoren erfolgt die Auswertung von links nach rechts.

In arithmetischen Ausdrücken kann durch runde Klammern die Abarbeitungsreihenfolge aller Operatoren festgelegt und damit von den normalen Prioritätsregeln abgewichen werden.

Reihenfolge der Operatoren

Von der höchsten zur niedrigsten Priorität

1.	NOT, B_NOT	Verneinung, bitweise Verneinung
2.	*, /, DIV, MOD	Multiplikation, Division
3.	+, -	Addition, Subtraktion
4.	B_AND	bitweises UND
5.	B_XOR	bitweises exklusives ODER
6.	B_OR	bitweises ODER
7.	AND	UND
8.	XOR	exklusives ODER
9.	OR	ODER
10.	<<	Verkettung von Strings, Ergebnistyp STRING
11.	==, <>, >, <, >=, <=	Vergleichsoperatoren

Hinweis

Der Kettungsoperator ":" für Frames darf nicht mit anderen Operatoren in einem Ausdruck vorkommen. Eine Prioritätseinstufung für diesen Operator ist deshalb nicht erforderlich.

Beispiel If-Anweisung

```
If (otto==10) and (anna==20) gotof end
```

1.8 Mögliche Typenkonvertierungen

Funktion

Typkonvertierung bei Zuweisung

Der konstante Zahlenwert, die Variable oder der Ausdruck, der einer Variablen zugewiesen wird, muss mit dem Typ dieser Variablen verträglich sein. Ist dies gegeben, so wird bei der Zuweisung der Typ automatisch umgewandelt.

Mögliche Typkonvertierungen

nach	REAL	INT	BOOL	CHAR	STRING	AXIS	FRAME
von							
REAL	ja	ja*	ja ¹⁾	ja*	–	–	–
INT	ja	ja	ja ¹⁾	ja ²⁾	–	–	–
BOOL	ja	ja	ja	ja	ja	–	–
CHAR	ja	ja	ja ¹⁾	ja	ja	–	–
STRING	–	–	ja ⁴⁾	ja ³⁾	ja	–	–
AXIS	–	–	–	–	–	ja	–
FRAME	–	–	–	–	–	–	ja

Erklärungen

- * Bei Typumwandlung von REAL nach INT wird bei gebrochenem Wert ≥ 0.5 aufgerundet, ansonsten wird abgerundet (vgl. Funktion ROUND)
- 1) Wert $\neq 0$ entspricht TRUE, Wert $= 0$ entspricht FALSE
- 2) Wenn der Wert im zulässigen Zahlenbereich liegt
- 3) Wenn nur 1 Zeichen
- 4) Stringlänge 0 = >FALSE, ansonsten TRUE

Hinweis

Ist beim Konvertieren ein Wert größer als der Zielbereich, erfolgt eine Fehlermeldung.

Treten in einem Ausdruck gemischte Typen auf, so wird eine Typanpassung automatisch durchgeführt. Typumwandlungen sind auch in Synchronaktionen möglich, siehe Kapitel "Bewegungssynchronaktionen, Implizite Typwandlung".

1.9 Stringoperationen

Stringoperationen

Neben den klassischen Operationen "Zuweisung" und "Vergleich" sind folgende Stringoperationen möglich:

- Typenkonvertierung nach STRING (AXSTRING) (Seite 84)
- Typenkonvertierung von STRING (NUMBER, ISNUMBER, AXNAME) (Seite 85)
- Verkettung von Strings (<<) (Seite 86)
- Wandlung in Klein-/Großbuchstaben (TOLOWER, TOUPPER) (Seite 88)
- Länge eines Strings bestimmen (STRLEN) (Seite 89)
- Zeichen/String in String suchen (INDEX, RINDEX, MINDEX, MATCH) (Seite 89)
- Auswahl eines Teilstrings (SUBSTR) (Seite 91)
- Lesen und Schreiben von einzelnen Zeichen (Seite 92)
- String formatieren (SPRINT) (Seite 93)

Sonderbedeutung des 0-Zeichens

Das 0-Zeichen wird intern als Enderkennung eines Strings interpretiert. Wird ein Zeichen durch das 0-Zeichen ersetzt, wird der String damit verkürzt.

Beispiel:

Programmcode	Kommentar
DEF STRING[20] STRG="Achse . steht"	
STRG[6]="X"	
MSG(STRG)	; Liefert die Meldung "Achse X steht".
STRG[6]=0	
MSG(STRG)	; Liefert die Meldung "Achse".

1.9.1 Typenkonvertierung nach STRING (AXSTRING)

Funktion

Durch die Funktion "Typkonvertierung nach STRING" lassen sich Variablen unterschiedlichen Typs als Bestandteil einer Meldung (MSG) nutzen.

Erfolgt bei Verwendung des Operators << implizit für die Datentypen INT, REAL, CHAR und BOOL (siehe "Verkettung von Strings (<<) (Seite 86) ").

Ein INT-Wert wird in die normal lesbare Form umgewandelt. Bei REAL-Werten werden bis zu 10 Nachkommastellen angegeben.

Mit dem Befehl AXSTRING können Variable vom Typ AXIS nach STRING gewandelt werden.

Syntax

```
<STRING_ERG> = << <bel._Typ>  
<STRING_ERG> = AXSTRING(<Achsbezeichner>)
```

Bedeutung

<STRING_ERG>:	Variable für das Ergebnis der Typkonvertierung Typ: STRING
<bel._Typ>:	Variablen-Typen INT, REAL, CHAR, STRING und BOOL
AXSTRING:	Der Befehl AXSTRING liefert den angegebenen Achsbezeichner als String.
<Achsbezeichner>:	Variable für Achsbezeichner Typ: AXIS

Hinweis

FRAME-Variablen können nicht konvertiert werden.

Beispiele

Beispiel 1:

```
MSG ("Position:"<<$AA_IM[X])
```

Beispiel 2: AXSTRING

Programmcode	Kommentar
DEF STRING[32] STRING_ERG	
STRING_ERG=AXSTRING(X)	; STRING_ERG == "X"

1.9.2 Typenkonvertierung von STRING (NUMBER, ISNUMBER, AXNAME)

Funktion

Mit dem Befehl `NUMBER` wird von `STRING` nach `REAL` konvertiert. Die Konvertierbarkeit kann mit dem Befehl `ISNUMBER` überprüft werden.

Mit dem Befehl `AXNAME` wird ein String in den Datentyp `AXIS` konvertiert.

Syntax

```
<REAL_ERG>=NUMBER("<String>")
<BOOL_ERG>=ISNUMBER("<String>")
<AXIS_ERG>=AXNAME("<String>")
```

Bedeutung

<code>NUMBER:</code>	Der Befehl <code>NUMBER</code> liefert die durch den <code><String></code> dargestellte Zahl als <code>REAL</code> -Wert zurück.
<code><String>:</code>	Zu konvertierende Variable vom Typ <code>STRING</code>
<code><REAL_ERG>:</code>	Variable für das Ergebnis der Typkonvertierung mit <code>NUMBER</code> Typ: <code>REAL</code>
<code>ISNUMBER:</code>	Mit dem Befehl <code>ISNUMBER</code> kann überprüft werden, ob der <code><String></code> in eine gültige Zahl gewandelt werden kann.
<code><BOOL_ERG>:</code>	Variable für das Ergebnis der Abfrage mit <code>ISNUMBER</code> Typ: <code>BOOL</code> Wert: <code>TRUE</code> <code>ISNUMBER</code> liefert den Wert <code>TRUE</code> , wenn der <code><String></code> eine nach den Regeln der Sprache gültige <code>REAL</code> -Zahl darstellt. <code>FALSE</code> Liefert <code>ISNUMBER</code> den Wert <code>FALSE</code> , wird bei Aufruf von <code>NUMBER</code> mit dem gleichen <code><String></code> Alarm ausgelöst.
<code>AXNAME:</code>	Der Befehl <code>AXNAME</code> wandelt den angegebenen <code><String></code> in einen Achsbezeichner. Hinweis: Kann der <code><String></code> keinem projizierten Achsbezeichner zugeordnet werden, wird ein Alarm ausgelöst.
<code><AXIS_ERG>:</code>	Variable für das Ergebnis der Typkonvertierung mit <code>AXNAME</code> Typ: <code>AXIS</code>

Beispiel

Programmcode	Kommentar
DEF BOOL BOOL_ERG	
DEF REAL REAL_ERG	
DEF AXIS AXIS_ERG	
BOOL_ERG=ISNUMBER("1234.9876Ex-7")	; BOOL_ERG == TRUE
BOOL_ERG=ISNUMBER("1234XYZ")	; BOOL_ERG == FALSE
REAL_ERG=NUMBER("1234.9876Ex-7")	; REAL_ERG == 1234.9876Ex-7
AXIS_ERG=AXNAME("X")	; AXIS_ERG == X

1.9.3 Verkettung von Strings (<<)**Funktion**

Die Funktion "Verkettung von Strings" schafft die Möglichkeit, einen String aus einzelnen Bestandteilen zusammensetzen zu können.

Realisiert wird die Verkettung über den Operator "<<". Dieser Operator hat für alle Kombinationen der Basistypen CHAR, BOOL, INT, REAL und STRING als Zieltyp STRING. Eine eventuell notwendige Konvertierung wird nach den bestehenden Regeln vorgenommen.

Syntax

```
<bel._Typ> << <bel._Typ>
```

Bedeutung

<bel._Typ>: Variable vom Typ CHAR, BOOL, INT, REAL oder STRING

<< : Operator für die Verkettung von Variablen (<bel._Typ>) zu einer zusammengesetzten Zeichenkette (Typ STRING).

Dieser Operator ist auch alleinig als sog. "unäre" Variante verfügbar. So ist es möglich, eine explizite Typwandlung nach STRING auszuführen (nicht für FRAME und AXIS):

```
<< <bel._Typ>
```

Beispielsweise lässt sich so eine Meldung oder ein Kommando aus Textlisten zusammensetzen und Parameter (etwa ein Bausteinname) einfügen:

```
MSG(STRG_TAB[LOAD_IDX] <<BAUSTEIN_NAME)
```

VORSICHT

Die Zwischenergebnisse bei der Stringverkettung dürfen die maximale Stringlänge nicht überschreiten.

Hinweis

Die Typen FRAME und AXIS können nicht zusammen mit dem Operator "<<" verwendet werden.

Beispiele

Beispiel 1: Verkettung von Strings

Programmcode	Kommentar
DEF INT IDX=2	
DEF REAL VALUE=9.654	
DEF STRING[20] STRG="INDEX:2"	
IF STRG=="Index:"<<IDX GOTOF NO_MSG	
MSG("Index:"<<IDX<<"/Wert:"<<VALUE)	; Anzeige: "Index:2/Wert:9.654"
NO_MSG:	

Beispiel 2: Explizite Typkonvertierung mit <<

Programmcode	Kommentar
DEF REAL VALUE=3.5	
<<VALUE	; Die angegebene Variable vom Typ REAL wird in den Typ STRING konvertiert.

1.9.4 Wandlung in Klein-/Großbuchstaben (TOLOWER, TOUPPER)

Funktion

Die Funktion "Wandlung in Klein-/Großbuchstaben" erlaubt es, alle Buchstaben einer Zeichenkette in eine einheitliche Darstellung zu wandeln.

Syntax

```
<STRING_ERG>=TOUPPER("<String>")  
<STRING_ERG>=TOLOWER("<String>")
```

Bedeutung

TOUPPER:	Mit dem Befehl <code>TOUPPER</code> werden alle Buchstaben einer Zeichenkette in Großbuchstaben umgewandelt.
TOLOWER:	Mit dem Befehl <code>TOLOWER</code> werden alle Buchstaben einer Zeichenkette in Kleinbuchstaben umgewandelt.
<String>:	Zeichenkette, die umgewandelt werden soll Typ: STRING
<STRING_ERG>:	Variable für das Ergebnis der Umwandlung Typ: STRING

Beispiel

Da es auch möglich ist, Benutzereingaben an der Bedienoberfläche anzustoßen, kann eine einheitliche Darstellung mit Klein- oder Großbuchstaben erreicht werden:

```
Programmcode  
DEF STRING [29] STRG  
...  
IF "LEARN.CNC"==TOUPPER(STRG) GOTOF LOAD_LEARN
```


1.9.5 Länge eines Strings bestimmen (STRLEN)

Funktion

Mit dem Befehl `STRLEN` ist es möglich, die Länge einer Zeichenkette zu bestimmen.

Syntax

```
<INT_ERG>=STRLEN ("<STRING>")
```

Bedeutung

<code>STRLEN:</code>	Mit dem Befehl <code>STRLEN</code> wird die Länge der angegebenen Zeichenkette bestimmt. Es wird die Anzahl der Zeichen zurückgegeben, die - vom Anfang der Zeichenkette an gezählt - keine 0-Zeichen sind.
<code><String>:</code>	Zeichenkette, deren Länge bestimmt werden soll Typ: <code>STRING</code>
<code><INT_ERG>:</code>	Variable für das Ergebnis der Bestimmung Typ: <code>INT</code>

Beispiel

Die Funktion im Zusammenhang mit dem Einzelzeichenzugriff ermöglicht es, das Ende einer Zeichenkette zu bestimmen:

```
Programmcode  
IF (STRLEN(BAUSTEIN_NAME)>10) GOTO F FEHLER
```

1.9.6 Zeichen/String in String suchen (INDEX, RINDEX, MINDEX, MATCH)

Funktion

Diese Funktionalität erlaubt es, einzelne Zeichen bzw. einen String in einem weiteren String zu suchen. Die Funktionsergebnisse geben an, an welcher Position des Strings das Zeichen/der String im zu untersuchenden String gefunden wurde.

Syntax

```
INT_ERG=INDEX (STRING, CHAR) ; Ergebnistyp: INT
INT_ERG=RINDEX (STRING, CHAR) ; Ergebnistyp: INT
INT_ERG=MINDEX (STRING, STRING) ; Ergebnistyp: INT
INT_ERG=MATCH (STRING, STRING) ; Ergebnistyp: INT
```

Semantik

Suchfunktionen: Sie liefern die Position im String (erster Parameter) zurück, wo die Suche erfolgreich war. Kann das Zeichen/der String nicht gefunden werden, wird der Wert -1 zurückgegeben. Das erste Zeichen hat dabei die Position 0.

Bedeutung

INDEX: sucht das als zweiten Parameter angegebene Zeichen (von vorne) im ersten Parameter.

RINDEX: sucht das als zweiten Parameter angegebene Zeichen (von hinten) im ersten Parameter.

MINDEX: entspricht der Funktion INDEX, außer, dass eine Liste von Zeichen (als String) übergeben wird, von denen der Index des ersten gefundenen Zeichens zurückgegeben wird.

MATCH: sucht einen String in einem String.

So lassen sich Strings nach bestimmten Kriterien zerlegen, etwa an Positionen mit Leerzeichen oder Pfadtrennzeichen ("/").

Beispiel

Zerlegen einer Eingabe in Pfad- und Bausteinnamen

Programmcode	Kommentar
DEF INT PFADIDX, PROGIDX	
DEF STRING[26] EINGABE	
DEF INT LISTIDX	
EINGABE = "/_N_MPF_DIR/_N_EXECUTE_MPF"	
LISTIDX = MINDEX (EINGABE, "M,N,O,P") + 1	; Als Wert in LISTIDX wird 3 zurückgeliefert; da "N" das erste Zeichen im Parameter EINGABE, aus der Auswahlliste von vorne, ist.
PFADIDX = INDEX (EINGABE, "/") +1	; damit gilt: PFADIDX = 1
PROGIDX = RINDEX (EINGABE, "/") +1	; damit gilt: PROGIDX = 12
	mit Hilfe der im nächsten Abschnitt eingeführten Funktion SUBSTR lässt sich die Variable EINGABE in die Komponenten ;"Pfad";und "Baustein" zerlegen:
VARIABLE = SUBSTR (EINGABE, PFADIDX, PROGIDX-PFADIDX-1)	; liefert dann "_N_MPF_DIR"
VARIABLE = SUBSTR (EINGABE, PROGIDX)	; liefert dann "_N_EXECUTE_MPF"

1.9.7 Auswahl eines Teilstrings (SUBSTR)

Funktion

Mit der Funktion SUBSTRING können beliebige Teile innerhalb eines Strings gelesen werden.

Syntax

```
<STRING_ERG>=SUBSTR(<String>,<Index>,<Länge>)
<STRING_ERG>=SUBSTR(<String>,<Index>)
```

Bedeutung

SUBSTR: Die Funktion liefert aus <String> einen Teilstring, ausgehend von <Index> mit der angegebenen <Länge>.
Ist der Parameter <Länge> nicht angegeben, liefert die Funktion einen Teilstring ausgehend von <Index> bis zum Stringende.

<Index>: Anfangsposition des Teilstrings innerhalb des Strings. Liegt die Anfangsposition hinter dem Stringende, wird ein Leerstring ("") zurückgegeben. Erstes Zeichen des Strings: Index = 0
Wertebereich: 0 ... (Stringlänge - 1)

<Länge>: Länge des Teilstrings. Wird eine zu große Länge angegeben, wird nur der Teilstring bis Stringende zurückgegeben.
Wertebereich: 1 ... (Stringlänge - 1)

Beispiel

Programmcode	Kommentar
DEF STRING[29] ERG	
; 1	
; 0123456789012345678	
ERG = SUBSTR("QUITTUNG: 10 bis 99", 10, 2)	; ERG == "10"
ERG = SUBSTR("QUITTUNG: 10 bis 99", 10)	; ERG == "10 bis 99"

1.9.8 Lesen und Schreiben von einzelnen Zeichen

Funktion

Innerhalb eines Strings können einzelne Zeichen gelesen und geschrieben werden.

Dabei sind folgende Randbedingungen zu beachten:

- nur bei anwenderdefinierten Variablen möglich, nicht bei Systemvariablen
- einzelne Zeichen eines String werden bei Unterprogrammaufrufen nur "call by value" übergeben

Syntax

```
<Zeichen>=<String> [<Index>]
<Zeichen>=<String_Array> [<Array_Index>, <Index>]
<String> [<Index>]=<Zeichen>
<String_Array> [<Array_Index>, <Index>]=<Zeichen>
```

Bedeutung

```
<String>:      Beliebiger String
<Zeichen>:     Variable vom Typ CHAR
<Index>:       Position des Zeichens innerhalb des Strings.
                Erstes Zeichen des Strings: Index = 0
                Wertebereich: 0 ... (Stringlänge - 1)
```

Beispiele

Beispiel 1: Variable Meldung

Programmcode	Kommentar
; 0123456789	
DEF STRING[50] MELDUNG = "Achse n hat Position erreicht"	
MELDUNG[6] = "X"	
MSG (MELDUNG)	; "Achse X hat Position erreicht"

Beispiel 2: Auswerten einer Systemvariablen

Programmcode	Kommentar
DEF STRING[50] STRG	; Zwischenspeicher für Systemvariable
...	
STRG = \$P_MMCA	; Systemvariable laden
IF STRG[0] == "E" GOTO ...	; Auswerten der Systemvariable

Beispiel 3: Parameterübergabe "call by value" und "call by reference"

Programmcode	Kommentar
; 0123456	
DEF STRING[50] STRG = "Achse X"	
DEF CHAR CHR	
...	
EXTERN UP_VAL(ACHSE)	; Definition Unterprogramm mit "call by value" Parameter
EXTERN UP_REF(VAR ACHSE)	; Definition Unterprogramm mit "call by ref." Parameter
...	
UP_VAL(STRG[6])	; Parameterübergabe "by value"
...	
CHR = STRG[6]	; Zwischenspeichern
UP_REF(CHR)	; Parameterübergabe "by reference"

1.9.9 String formatieren (SPRINT)**Funktion**

Mit der vordefinierten Funktion SPRINT können Zeichenketten formatiert und z. B. für die Ausgabe an externe Geräte aufbereitet werden (siehe auch "Process DataShare - Ausgabe auf ein externes Gerät/Datei (EXTOPEN, WRITE, EXTCLOSE) (Seite 623)").

Syntax

```
"<Ergebnis_String>"=SPRINT("<Format_String>",<Wert_1>,<Wert_2>,...,<Wert_n>)
```

Bedeutung

SPRINT:	Bezeichner für eine vordefinierte Funktion, die einen Wert vom Typ STRING liefert.
"<Format_String>":	Zeichenkette, die feste und variable Teile enthält. Die variablen Teile werden durch das Formatsteuerzeichen % und einer nachfolgenden Formatbeschreibung festgelegt.
< Wert_1>,< Wert_2>,...,< Wert_n>:	Wert in Form einer Konstanten oder NC-Variablen, der an der Stelle, an der das n-te Formatsteuerzeichen % steht, entsprechend der Formatbeschreibung in den <Format_String> eingefügt wird.
"<Ergebnis_String>":	Formatierte Zeichenkette (maximal 400 Bytes)

Verfügbare Formatbeschreibungen

%B:	<p>Wandlung in den String "TRUE", wenn der zu wandelnde Wert:</p> <ul style="list-style-type: none"> • ungleich 0 ist. • kein Leerstring ist (bei String-Werten). <p>Wandlung in den String "FALSE", wenn der zu wandelnde Wert:</p> <ul style="list-style-type: none"> • gleich 0 ist. • ein Leerstring ist. <p>Beispiel:</p> <pre>N10 DEF BOOL BOOL_VAR=1 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF BOOL_VAR:%B", BOOL_VAR)</pre> <p>Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "CONTENT OF BOOL_VAR:TRUE" beschrieben.</p>
%C:	<p>Wandlung in ein ASCII-Zeichen.</p> <p>Beispiel:</p> <pre>N10 DEF CHAR CHAR_VAR="X" N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF CHAR_VAR:%C", CHAR_VAR)</pre> <p>Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "CONTENT OF CHAR_VAR:X" beschrieben.</p>
%D:	<p>Wandlung in einen String mit einem ganzzahligen Wert (INTEGER).</p> <p>Beispiel:</p> <pre>N10 DEF INT INT_VAR=123 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF INT_VAR:%D", INT_VAR)</pre> <p>Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "CONTENT OF INT_VAR:123" beschrieben.</p>
%<m>D:	<p>Wandlung in einen String mit einem ganzzahligen Wert (INTEGER). Der String hat eine Mindestlänge von <m> Zeichen. Fehlende Stellen werden linksbündig mit Leerzeichen aufgefüllt.</p> <p>Beispiel:</p> <pre>N10 DEF INT INT_VAR=-123 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF INT_VAR:%6D", INT_VAR)</pre> <p>Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "CONTENT OF INT_VAR:xx-123" beschrieben ("x" steht im Beispiel stellvertretend für Leerzeichen).</p>
%F:	<p>Wandlung in einen String mit einer Dezimalzahl mit 6 Nachkommastellen. Die Nachkommastellen werden ggf. gerundet oder mit 0 aufgefüllt.</p> <p>Beispiel:</p> <pre>N10 DEF REAL REAL_VAR=-1.2341234EX+03 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%F", REAL_VAR)</pre> <p>Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "CONTENT OF REAL_VAR: -1234.123400" beschrieben.</p>

%<m>F:	<p>Wandlung in einen String mit einer Dezimalzahl mit 6 Nachkommastellen und einer Gesamtlänge von mindestens <m> Zeichen. Die Nachkommastellen werden ggf. gerundet oder mit 0 aufgefüllt. Fehlende Zeichen auf die Gesamtlänge <m> werden linksbündig mit Leerzeichen aufgefüllt.</p> <p>Beispiel:</p> <pre>N10 DEF REAL REAL_VAR=-1.23412345678EX+03 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%15F",REAL_VAR)</pre> <p>Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "CONTENT OF REAL_VAR: xxx-1234.123457" beschrieben ("x" steht im Beispiel stellvertretend für Leerzeichen).</p>
%.<n>F:	<p>Wandlung in einen String mit einer Dezimalzahl mit <n> Nachkommastellen. Die Nachkommastellen werden ggf. gerundet oder mit 0 aufgefüllt.</p> <p>Beispiel:</p> <pre>N10 DEF REAL REAL_VAR=-1.2345678EX+03 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%.3F",REAL_VAR)</pre> <p>Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "CONTENT OF REAL_VAR:-1234.568" beschrieben.</p>
%<m>.<n>F:	<p>Wandlung in einen String mit einer Dezimalzahl mit <n> Nachkommastellen und einer Gesamtlänge von mindestens <m> Zeichen. Die Nachkommastellen werden ggf. gerundet oder mit 0 aufgefüllt. Fehlende Zeichen auf die Gesamtlänge <m> werden linksbündig mit Leerzeichen aufgefüllt.</p> <p>Beispiel:</p> <pre>N10 DEF REAL REAL_VAR=-1.2341234567890EX+03 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%10.2F",REAL_VAR)</pre> <p>Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "CONTENT OF REAL_VAR:xx-1234.12" beschrieben ("x" steht im Beispiel stellvertretend für Leerzeichen).</p>
%E:	<p>Wandlung in einen String mit einer Dezimalzahl in Exponentialdarstellung. Die Mantisse wird normalisiert mit einer Vorkommastelle und 6 Nachkommastellen abgelegt. Die Nachkommastellen werden ggf. gerundet oder mit 0 aufgefüllt. Der Exponent beginnt mit dem Schlüsselwort "EX". Es folgt das Vorzeichen ("+" oder "-") und eine zwei- oder dreistellige Zahl.</p> <p>Beispiel:</p> <pre>N10 DEF REAL REAL_VAR=-1234.567890 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%E",REAL_VAR)</pre> <p>Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "CONTENT OF REAL_VAR:-1.234568EX+03" beschrieben.</p>

<p>%<m>E:</p>	<p>Wandlung in einen String mit einer Dezimalzahl in Exponentialdarstellung mit einer Gesamtlänge von mindestens <m> Zeichen. Fehlende Zeichen werden linksbündig mit Leerzeichen aufgefüllt. Die Mantisse wird normalisiert mit einer Vorkommastelle und 6 Nachkommastellen abgelegt. Die Nachkommastellen werden ggf. gerundet oder mit 0 aufgefüllt. Der Exponent beginnt mit dem Schlüsselwort "EX". Es folgt das Vorzeichen ("+" oder "-") und eine zwei- oder dreistellige Zahl.</p> <p>Beispiel:</p> <pre>N10 DEF REAL REAL_VAR=-1234.5 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%20E",REAL_VAR)</pre> <p>Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "CONTENT OF REAL_VAR:xxxxxx-1.234500EX+03" beschrieben ("x" steht im Beispiel stellvertretend für Leerzeichen)</p>
<p>%.<n>E:</p>	<p>Wandlung in einen String mit einer Dezimalzahl in Exponentialdarstellung. Die Mantisse wird normalisiert mit einer Vorkommastelle und <n> Nachkommastellen abgelegt. Die Nachkommastellen werden ggf. gerundet oder mit 0 aufgefüllt. Der Exponent beginnt mit dem Schlüsselwort "EX". Es folgt das Vorzeichen ("+" oder "-") und eine zwei- oder dreistellige Zahl.</p> <p>Beispiel:</p> <pre>N10 DEF REAL REAL_VAR=-1234.5678 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%.2E",REAL_VAR)</pre> <p>Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "CONTENT OF REAL_VAR:-1.23EX+03" beschrieben.</p>
<p>%<m>.<n>E:</p>	<p>Wandlung in einen String mit einer Dezimalzahl in Exponentialdarstellung mit einer Gesamtlänge von mindestens <m> Zeichen. Fehlende Zeichen werden linksbündig mit Leerzeichen aufgefüllt. Die Mantisse wird normalisiert mit einer Vorkommastelle und <n> Nachkommastellen abgelegt. Die Nachkommastellen werden ggf. gerundet oder mit 0 aufgefüllt. Der Exponent beginnt mit dem Schlüsselwort "EX". Es folgt das Vorzeichen ("+" oder "-") und eine zwei- oder dreistellige Zahl.</p> <p>Beispiel:</p> <pre>N10 DEF REAL REAL_VAR=-1234.5678 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%12.2E",REAL_VAR)</pre> <p>Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "CONTENT OF REAL_VAR:xx-1.23EX+03" beschrieben ("x" steht im Beispiel stellvertretend für Leerzeichen).</p>

%G:	<p>Wandlung in einen String mit einer Dezimalzahl je nach Wertebereich in Dezimal- oder Exponentialdarstellung: ist der darzustellende Wert betragsmäßig kleiner als 1.0EX-04 oder größer/gleich 1.0EX+06 wird die Exponentialdarstellung gewählt, sonst die Dezimaldarstellung. Es werden maximal sechs signifikante Stellen angezeigt, ggf. wird gerundet.</p> <p>Beispiel mit Dezimaldarstellung:</p> <pre>N10 DEF REAL REAL_VAR=1.234567890123456EX-04 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%G",REAL_VAR)</pre> <p>Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "CONTENT OF REAL_VAR:0.000123457" beschrieben.</p> <p>Beispiel mit Exponentialdarstellung:</p> <pre>N10 DEF REAL REAL_VAR=1.234567890123456EX+06 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%G",REAL_VAR)</pre> <p>Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "CONTENT OF REAL_VAR:1.23457EX+06" beschrieben.</p>
%<m>G:	<p>Wandlung in einen String mit einer Dezimalzahl je nach Wertebereich in Dezimal- oder Exponentialdarstellung (wie %G). Der String hat eine Gesamtlänge von mindestens <m> Zeichen. Fehlende Zeichen werden linksbündig mit Leerzeichen aufgefüllt.</p> <p>Beispiel mit Dezimaldarstellung:</p> <pre>N10 DEF REAL REAL_VAR=1.234567890123456EX-04 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%15G",REAL_VAR)</pre> <p>Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "CONTENT OF REAL_VAR:xxxx0.000123457" beschrieben ("x" steht im Beispiel stellvertretend für Leerzeichen).</p> <p>Beispiel mit Exponentialdarstellung:</p> <pre>N10 DEF REAL REAL_VAR=1.234567890123456EX+06 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%15G",REAL_VAR)</pre> <p>Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "CONTENT OF REAL_VAR:xxx1.23457EX+06" beschrieben ("x" steht im Beispiel stellvertretend für Leerzeichen).</p>
%.<n>G:	<p>Wandlung in einen String mit einer Dezimalzahl je nach Wertebereich in Dezimal- oder Exponentialdarstellung. Es werden maximal <n> signifikante Stellen angezeigt, ggf. wird gerundet. Ist der darzustellende Wert betragsmäßig kleiner als 1.0EX-04 oder größer/gleich 1.0EX(+<n>) wird die Exponentialdarstellung gewählt, sonst die Dezimaldarstellung.</p> <p>Beispiel mit Dezimaldarstellung:</p> <pre>N10 DEF REAL REAL_VAR=1.234567890123456EX-04 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%.3G",REAL_VAR)</pre> <p>Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "CONTENT OF REAL_VAR:0.000123" beschrieben.</p> <p>Beispiel mit Exponentialdarstellung:</p> <pre>N10 DEF REAL REAL_VAR=1.234567890123456EX+03 N20 DEF STRING[80] RESULT N30 RESULT = SPRINT("CONTENT OF REAL_VAR:%.3G",REAL_VAR)</pre> <p>Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "CONTENT OF REAL_VAR:1.23EX+03" beschrieben.</p>

<p>%<m>.<n>G:</p>	<p>Wandlung in einen String mit einer Dezimalzahl je nach Wertebereich in Dezimal- oder Exponentialdarstellung (wie %.<n>G). Der String hat eine Gesamtlänge von mindestens <m> Zeichen. Fehlende Zeichen werden linksbündig mit Leerzeichen aufgefüllt.</p> <p>Beispiel mit Dezimaldarstellung:</p> <pre>N10 DEF REAL REAL_VAR=1.234567890123456EX-04 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%12.4G",REAL_VAR)</pre> <p>Ergebnis: Die Stringvariable RESULT wird mit der Zeichenkette "CONTENT OF REAL_VAR:xxx0.0001235" beschrieben ("x" steht im Beispiel stellvertretend für Leerzeichen).</p> <p>Beispiel mit Exponentialdarstellung:</p> <pre>N10 DEF REAL REAL_VAR=1.234567890123456EX+04 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%12.4G",REAL_VAR)</pre> <p>Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "CONTENT OF REAL_VAR:xx1.235EX+06" beschrieben ("x" steht im Beispiel stellvertretend für Leerzeichen).</p>
<p>%.<n>P:</p>	<p>Wandlung eines REAL-Werts in einen INTEGER-Wert unter Berücksichtigung von <n> Nachkommastellen. Der INTEGER-Wert wird als 32-Bit-Binärzahl ausgegeben. Kann der zu wandelnde Wert nicht mit 32 Bit dargestellt werden, wird die Bearbeitung mit Alarm abgebrochen.</p> <p>Da eine mit der Formatanweisung %.<n>P erzeugte Byte-Folge auch binäre Nullen enthalten kann, entspricht der so erzeugte Gesamt-String nicht mehr den Konventionen des NC-Datentyps STRING. Er kann daher weder in einer Variablen vom Typ STRING gespeichert werden, noch mit den String-Befehlen der NC-Sprache weiter bearbeitet werden. Die einzig mögliche Verwendung ist die Parameterübergabe an den WRITE-Befehl mit Ausgabe an ein entsprechendes externes Gerät (siehe folgendes Beispiel).</p> <p>Sobald der <Format_String> eine Formatbeschreibung vom Typ %P enthält, wird der gesamte String, mit Ausnahme der mit %.<n>P generierten Binärzahl, entsprechend dem MD10750 \$MN_SPRINT_FORMAT_P_CODE im Zeichen-Code ASCII, ISO (DIN6024) oder EIA (RS244) ausgegeben. Wird ein nicht umsetzbares Zeichen programmiert, wird die Bearbeitung mit Alarm abgebrochen.</p> <p>Beispiel:</p> <pre>N10 DEF REAL REAL_VAR=123.45 N20 DEF INT ERROR N30 DEF STRING[20] EXT_DEVICE="/ext/dev/1" ... N100 EXTOPEN(ERROR,EXT_DEVICE) N110 IF ERROR <> 0 ... ; Fehlerhandling N200 WRITE(ERROR,EXT_DEVICE,SPRINT("INTEGER BINARY CODED:%.3P",REAL_VAR)) N210 IF ERROR <> 0 ... ; Fehlerhandling</pre> <p>Ergebnis: Der String "INTEGER BINARY CODED: 'H0001E23A'" wird an das Ausgabegerät /ext/dev/1 übertragen. Der hexadezimale Wert 0x0001E23A entspricht dem dezimalen Wert 123450.</p>

%<m> . <n>P:	<p>Wandlung eines REAL-Werts entsprechend der Einstellung im Maschinendatum MD10751 \$MN_SPRINT_FORMAT_P_DECIMAL in einen String mit:</p> <ul style="list-style-type: none"> • einer Ganzzahl von <m> + <n> Stellen oder • einer Dezimalzahl mit maximal <m> Vorkommastellen und exakt <n> Nachkommastellen. <p>Wie bei der Formatbeschreibung % . <n>P wird der gesamte String in dem durch MD10750 \$MN_SPRINT_FORMAT_P_CODE festgelegten Zeichen-Code abgelegt.</p> <p>Wandlung bei MD10751 = 0:</p> <p>Der REAL-Wert wird in einen String mit einer Ganzzahl von <m> + <n> Stellen gewandelt. Ggf. werden Nachkommastellen auf <n> Stellen gerundet oder mit 0 aufgefüllt. Fehlende Vorkommastellen werden mit Leerzeichen aufgefüllt. Das Minus-Vorzeichen wird linksbündig angefügt, an Stelle des Plus-Vorzeichens wird ein Leerzeichen abgesetzt.</p> <p>Beispiel:</p> <pre>N10 DEF REAL REAL_VAR=-123.45 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("PUNCHED TAPE FORMAT:%5.3P",REAL_VAR)</pre> <p>Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "PUNCHED TAPE FORMAT:-xx123450" beschrieben ("x" steht im Beispiel stellvertretend für Leerzeichen).</p> <p>Wandlung bei MD10751 = 1:</p> <p>Der REAL-Wert wird in einen String mit einer Dezimalzahl mit maximal <m> Vorkommastellen und exakt <n> Nachkommastellen gewandelt. Ggf. werden die Vorkommastellen abgeschnitten und die Nachkommastellen gerundet oder mit 0 aufgefüllt. Ist <n> gleich 0 entfällt auch der Dezimalpunkt.</p> <p>Beispiel:</p> <pre>N10 DEF REAL REAL_VAR1=-123.45 N20 DEF REAL REAL_VAR2=123.45 N30 DEF STRING[80] RESULT N40 RESULT=SPRINT("PUNCHED TAPE FORMAT:%5.3P VAR2:%2.0P", REAL_VAR1,REAL_VAR2)</pre> <p>Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "PUNCHED TAPE FORMAT:-123.450 VAR2:23" beschrieben.</p>
%S:	<p>Einfügen eines Strings.</p> <p>Beispiel:</p> <pre>N10 DEF STRING[16] STRING_VAR="ABCDEFGH" N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF STRING_VAR:%S",STRING_VAR)</pre> <p>Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "CONTENT OF STRING_VAR:ABCDEFGH" beschrieben.</p>
%<m>S:	<p>Einfügen eines Strings mit mindestens <m> Zeichen. Fehlende Stellen werden mit Leerzeichen aufgefüllt.</p> <p>Beispiel:</p> <pre>N10 DEF STRING[16] STRING_VAR="ABCDEFGH" N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF STRING_VAR:%10S",STRING_VAR)</pre> <p>Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "CONTENT OF STRING_VAR:xxxABCDEFGH" beschrieben ("x" steht im Beispiel stellvertretend für Leerzeichen).</p>

<p>% . <n>S:</p>	<p>Einfügen von <n> Zeichen eines Strings (beginnend mit dem ersten Zeichen). Beispiel: <pre>N10 DEF STRING[16] STRING_VAR="ABCDEFGH" N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF STRING_VAR:%.3S",STRING_VAR)</pre> Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "CONTENT OF STRING_VAR:ABC" beschrieben. </p>
<p>% <m> . <n>S:</p>	<p>Einfügen von <n> Zeichen eines Strings (beginnend mit dem ersten Zeichen). Die Gesamtlänge des erzeugten Strings hat mindestens < m> Zeichen. Fehlende Stellen werden mit Leerzeichen aufgefüllt. Beispiel: <pre>N10 DEF STRING[16] STRING_VAR="ABCDEFGH" N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF STRING_VAR:%10.5S", STRING_VAR)</pre> Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "CONTENT OF STRING_VAR:xxxxxABCDE" beschrieben ("x" steht im Beispiel stellvertretend für Leerzeichen). </p>
<p>%X:</p>	<p>Wandlung eines INTEGER-Werts in einen String mit Hexadezimal-Darstellung. Beispiel: <pre>N10 DEF INT INT_VAR='HA5B8' N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("INTEGER HEXADECIMAL:%X",INT_VAR)</pre> Ergebnis: Die String-Variable RESULT wird mit der Zeichenkette "INTEGER HEXADECIMAL:A5B8" beschrieben. </p>

Hinweis

Die Eigenschaft der NC-Sprache, bezüglich Bezeichnern und Schlüsselworten nicht zwischen Groß- und Kleinbuchstaben zu unterscheiden, gilt auch für die Formatbeschreibungen. Sie können daher ohne funktionalen Unterschied mit Klein- oder Großbuchstaben programmiert werden.

Kombinationsmöglichkeiten

Die folgende Tabelle gibt Auskunft darüber, welche NC-Datentypen mit welcher Formatbeschreibung kombiniert werden können. Es gelten die Regeln zur impliziten Datentypwandlung (siehe "Datentypen (Seite 55)").

	NC-Datentypen						
	BOOL	CHAR	INT	REAL	STRING	AXIS	FRAME
%B	+	+	+	+	+	-	-
%C	-	+	-	-	+	-	-
%D	+	+	+	+	-	-	-
%F	-	-	+	+	-	-	-
%E	-	-	+	+	-	-	-
%G	-	-	+	+	-	-	-
%S	-	+	-	-	+	-	-
%X	+	+	+	-	-	-	-
%P	-	-	+	+	-	-	-

Hinweis

Die Tabelle zeigt, dass die NC-Datentypen AXIS und FRAME nicht direkt in der SPRINT-Funktion verwendet werden können. Es ist aber möglich:

- den Datentyp AXIS mit der Funktion AXSTRING in einen String zu wandeln, der dann mit SPRINT weiterverarbeitet werden kann.
- die einzelnen Werte des Datentyps FRAME per Frame-Komponentenzugriff zu lesen. Dadurch bekommt man ein Datum vom Typ REAL, das mit SPRINT weiterverarbeitet werden kann.

1.10 Programmsprünge und -verzweigungen

1.10.1 Rücksprung auf Programmanfang (GOTOS)

Funktion

Mit dem Befehl `GOTOS` ist es möglich, zur Programmwiederholung an den Anfang eines Haupt- oder Unterprogramms zurückzuspringen.

Über Maschinendaten kann eingestellt werden, dass bei jedem Rücksprung auf den Programmanfang:

- die Programmlaufzeit auf "0" gesetzt wird.
- die Werkstückzählung um den Wert "1" erhöht wird.

Syntax

`GOTOS`

Bedeutung

`GOTOS`: Sprunganweisung mit Sprungziel Programmanfang.

Die Ausführung wird gesteuert über das NC/PLC-Nahtstellensignal:
DB21, ... DBX384.0 (Programmverzweigung steuern)

Wert: Bedeutung:

- | | |
|---|--|
| 0 | Kein Rücksprung auf den Programmanfang. Die Programmbearbeitung wird mit dem nächsten Teileprogrammsatz nach <code>GOTOS</code> fortgeführt. |
| 1 | Rücksprung auf den Programmanfang. Das Teileprogramm wird wiederholt. |

Randbedingungen

- `GOTOS` löst intern ein `STOPRE` (Vorlaufstopp) aus.
- Bei einem Teileprogramm mit Datendefinitionen (LUD-Variablen) wird mit `GOTOS` auf den ersten Programmsatz nach dem Definitionsabschnitt gesprungen, d. h. die Datendefinitionen werden nicht erneut ausgeführt. Die definierten Variablen behalten daher den im `GOTOS`-Satz erreichten Wert und werden nicht auf die im Definitionsabschnitt programmierten Standardwerte zurückgesetzt.
- In Synchronaktionen und Technologie-Zyklen steht der Befehl `GOTOS` nicht zur Verfügung.

Beispiel

Programmcode	Kommentar
N10 ...	; Programmanfang.
...	
N90 GOTOS	; Sprung an den Programmanfang.
...	

1.10.2 Programmsprünge auf Sprungmarken (GOTOB, GOTOF, GOTO, GOTOC)

Funktion

In einem Programm können Sprungmarken (Labels) gesetzt werden, auf die von anderen Stellen innerhalb desselben Programms mit dem Befehlen `GOTOF`, `GOTOB`, `GOTO` bzw. `GOTOC` gesprungen werden kann. Die Programmbearbeitung wird dann mit der Anweisung fortgesetzt, die unmittelbar nach der Sprungmarke folgt. Dadurch sind Verzweigungen innerhalb des Programms realisierbar.

Neben den Sprungmarken sind als Sprungziele auch Haupt- und Nebensatznummern möglich.

Wenn vor der Sprunganweisung eine Sprungbedingung (`IF ...`) formuliert ist, dann erfolgt der Programmsprung nur dann, wenn die Sprungbedingung erfüllt ist.

Syntax

```
GOTOB <Sprungziel>
IF <Sprungbedingung> = TRUE GOTOB <Sprungziel>
GOTOF <Sprungziel>
IF <Sprungbedingung> = TRUE GOTOF <Sprungziel>
GOTO <Sprungziel>
IF <Sprungbedingung> = TRUE GOTO <Sprungziel>
GOTOC <Sprungziel>
IF <Sprungbedingung> = TRUE GOTOC <Sprungziel>
```

Bedeutung

<code>GOTOB</code> :	Sprunganweisung mit Sprungziel in Richtung Programmanfang.						
<code>GOTOF</code> :	Sprunganweisung mit Sprungziel in Richtung Programmende.						
<code>GOTO</code> :	Sprunganweisung mit Sprungzielsuche. Die Suche erfolgt erst in Richtung Programmende, dann in Richtung Programmanfang.						
<code>GOTOC</code> :	Wirkung wie <code>GOTO</code> mit dem Unterschied, dass der Alarm 14080 "Sprungziel nicht gefunden" unterdrückt wird. Das bedeutet, dass die Programmbearbeitung im Falle einer ergebnislosen Sprungzielsuche nicht abgebrochen wird, sondern mit der auf den Befehl <code>GOTOC</code> folgenden Programmzeile fortgesetzt wird.						
<code><Sprungziel></code> :	Sprungzielparameter Mögliche Angaben sind: <table> <tr> <td><code><Sprungmarke></code>:</td> <td>Sprungziel ist die im Programm gesetzte Sprungmarke mit benutzerdefiniertem Namen: <code><Sprungmarke></code>:</td> </tr> <tr> <td><code><Satznummer></code>:</td> <td>Sprungziel ist eine Haupt- oder Nebensatznummer (z. B.: 200, N300)</td> </tr> <tr> <td>Variable vom Typ <code>STRING</code>:</td> <td>Variables Sprungziel. Die Variable steht für eine Sprungmarke oder eine Satznummer.</td> </tr> </table>	<code><Sprungmarke></code> :	Sprungziel ist die im Programm gesetzte Sprungmarke mit benutzerdefiniertem Namen: <code><Sprungmarke></code> :	<code><Satznummer></code> :	Sprungziel ist eine Haupt- oder Nebensatznummer (z. B.: 200, N300)	Variable vom Typ <code>STRING</code> :	Variables Sprungziel. Die Variable steht für eine Sprungmarke oder eine Satznummer.
<code><Sprungmarke></code> :	Sprungziel ist die im Programm gesetzte Sprungmarke mit benutzerdefiniertem Namen: <code><Sprungmarke></code> :						
<code><Satznummer></code> :	Sprungziel ist eine Haupt- oder Nebensatznummer (z. B.: 200, N300)						
Variable vom Typ <code>STRING</code> :	Variables Sprungziel. Die Variable steht für eine Sprungmarke oder eine Satznummer.						
<code>IF</code> :	Schlüsselwort zur Formulierung der Sprungbedingung. Die Sprungbedingung lässt alle Vergleichs- und logischen Operationen zu (Ergebnis: <code>TRUE</code> oder <code>FALSE</code>). Der Programmsprung wird ausgeführt, wenn das Ergebnis dieser Operation <code>TRUE</code> ist.						

Hinweis

Sprungmarken (Labels)

Sprungmarken stehen immer am Anfang eines Satzes. Wenn eine Programmnummer vorhanden ist, steht die Sprungmarke unmittelbar nach der Satznummer.

Für die Benennung von Sprungmarken gelten folgende Regeln:

- Anzahl an Zeichen:
 - mindestens 2
 - höchstens 32
 - Erlaubte Zeichen sind:
 - Buchstaben
 - Ziffern
 - Unterstriche
 - Die ersten beiden Zeichen müssen Buchstaben oder Unterstriche sein.
 - Nach dem Namen der Sprungmarke folgt ein Doppelpunkt (":").
-

Randbedingungen

- Sprungziel kann nur ein Satz mit Sprungmarke oder Satznummer sein, der **innerhalb** des Programms liegt.
- Eine Sprunganweisung ohne Sprungbedingung muss in einem separaten Satz programmiert werden. Bei Sprunganweisungen mit Sprungbedingungen gilt diese Einschränkung nicht. Hier können mehrere Sprunganweisungen in einem Satz formuliert werden.
- Bei Programmen mit Sprunganweisungen ohne Sprungbedingungen muss das Programmende $M2/M30$ nicht zwangsläufig am Programmende stehen.

Beispiele

Beispiel 1: Sprünge auf Sprungmarken

Programmcode	Kommentar
N10 ...	
N20 GOTOF Label_1	; Sprung in Richtung Programmende zur Sprungmarke "Label_1".
N30 ...	
N40 Label_0: R1=R2+R3	; Sprungmarke "Label_0" gesetzt.
N50 ...	
N60 Label_1:	; Sprungmarke "Label_1" gesetzt.
N70 ...	
N80 GOTOB Label_0	; Sprung in Richtung Programmanfang zur Sprungmarke "Label_0".
N90 ...	

Beispiel 2: Indirekter Sprung auf Satznummer

Programmcode	Kommentar
N5 R10=100	
N10 GOTOF "N"<<R10	; Sprung auf den Satz, dessen Satznummer in R10 steht.
...	
N90 ...	
N100 ...	; Sprungziel
N110 ...	
...	

Beispiel 3: Sprung auf variables Sprungziel

Programmcode	Kommentar
DEF STRING[20] ZIEL	
ZIEL = "Marke2"	
GOTOF ZIEL	; Sprung in Richtung Programmende zum variablen Sprungziel ZIEL.
Marke1: T="Bohrer1"	
...	
Marke2: T="Bohrer2"	; Sprungziel
...	

Beispiel 4: Sprung mit Sprungbedingung

Programmcode	Kommentar
N40 R1=30 R2=60 R3=10 R4=11 R5=50 R6=20	; Zuweisung der Anfangswerte.
N41 LA1: G0 X=R2*COS(R1)+R5 Y=R2*SIN(R1)+R6	; Sprungmarke LA1 gesetzt.
N42 R1=R1+R3 R4=R4-1	
N43 IF R4>0 GOTOB LA1	; Wenn Sprungbedingung erfüllt, dann Sprung in Richtung Programmanfang zur Sprungmarke LA1.
N44 M30	; Programmende

1.10.3 Programmverzweigung (CASE ... OF ... DEFAULT ...)

Funktion

Die CASE-Funktion bietet die Möglichkeit, den aktuellen Wert (Typ: INT) einer Variablen oder einer Rechenfunktion zu überprüfen und abhängig vom Ergebnis an unterschiedliche Stellen im Programm zu springen.

Syntax

```
CASE(<Ausdruck>) OF <Konstante_1> GOTOF <Sprungziel_1> <Konstante_2>
GOTOF <Sprungziel_2> ... DEFAULT GOTOF <Sprungziel_n>
```

Bedeutung

CASE:	Sprunganweisung
<Ausdruck>:	Variable oder Rechenfunktion
OF:	Schlüsselwort zur Formulierung der bedingten Programmverzweigungen
<Konstante_1>:	Erster angegebener konstanter Wert für die Variable oder Rechenfunktion Typ: INT
<Konstante_2>:	Zweiter angegebener konstanter Wert für die Variable oder Rechenfunktion Typ: INT
DEFAULT:	Für die Fälle, in denen die Variable oder Rechenfunktion keinen der angegebenen konstanten Werte annimmt, kann mit der Anweisung DEFAULT ein Sprungziel bestimmt werden. Hinweis: Falls die DEFAULT-Anweisung nicht programmiert ist, wird in diesen Fällen der auf die CASE-Anweisung folgende Satz zum Sprungziel.
GOTOF:	Sprunganweisung mit Sprungziel in Richtung Programmende. Statt GOTOF sind auch alle anderen GOTO-Befehle programmierbar (siehe Thema "Programmsprünge auf Sprungmarken").
<Sprungziel_1>:	Auf dieses Sprungziel wird verzweigt, wenn der Wert der Variablen oder Rechenfunktion der ersten angegebenen Konstanten entspricht. Das Sprungziel kann wie folgt angegeben werden: <Sprungmarke>: Sprungziel ist die im Programm gesetzte Sprungmarke mit benutzerdefiniertem Namen: <Sprungmarke> : <Satznummer>: Sprungziel ist eine Haupt- oder Nebensatznummer (z. B.: 200, N300) Variable vom Typ STRING: Variables Sprungziel. Die Variable steht für eine Sprungmarke oder eine Satznummer.

- <Sprungziel_2>: Auf dieses Sprungziel wird verzweigt, wenn der Wert der Variablen oder Rechenfunktion der zweiten angegebenen Konstanten entspricht.
- <Sprungziel_n>: Auf dieses Sprungziel wird verzweigt, wenn der Wert der Variablen keinen der angegebenen konstanten Werte annimmt.

Beispiel

Programmcode

```
...  
N20 DEF INT VAR1 VAR2 VAR3  
N30 CASE (VAR1+VAR2-VAR3) OF 7 GOTO Label_1 9 GOTO Label_2 DEFAULT GOTO Label_3  
N40 Label_1: G0 X1 Y1  
N50 Label_2: G0 X2 Y2  
N60 Label_3: G0 X3 Y3  
...
```

Die `CASE`-Anweisung aus `N30` definiert folgende Programmverzweigungsmöglichkeiten:

1. Wenn der Wert der Rechenfunktion $VAR1+VAR2-VAR3 = 7$, dann springe zu dem Satz mit Sprungmarkendefinition "Label_1" ($\rightarrow N40$).
2. Wenn der Wert der Rechenfunktion $VAR1+VAR2-VAR3 = 9$, dann springe zu dem Satz mit Sprungmarkendefinition "Label_2" ($\rightarrow N50$).
3. Wenn der Wert der Rechenfunktion $VAR1+VAR2-VAR3$ weder 7 noch 9 beträgt, dann springe zu dem Satz mit Sprungmarkendefinition "Label_3" ($\rightarrow N60$).

1.11 Programmteilwiederholung (REPEAT, REPEATB, ENDLABEL, P)

Funktion

Die Programmteilwiederholung ermöglicht die Wiederholung bereits geschriebener Programmteile innerhalb eines Programms in beliebiger Zusammensetzung.

Die zu wiederholenden Programmzeilen bzw. Programmbereiche werden durch Sprungmarken (Labels) gekennzeichnet.

Hinweis

Sprungmarken (Labels)

Sprungmarken stehen immer am Anfang eines Satzes. Wenn eine Programmnummer vorhanden ist, steht die Sprungmarke unmittelbar nach der Satznummer.

Für die Benennung von Sprungmarken gelten folgende Regeln:

- Anzahl an Zeichen:
 - mindestens 2
 - höchstens 32
 - Erlaubte Zeichen sind:
 - Buchstaben
 - Ziffern
 - Unterstriche
 - Die ersten beiden Zeichen müssen Buchstaben oder Unterstriche sein.
 - Nach dem Namen der Sprungmarke folgt ein Doppelpunkt (":").
-

Syntax

1. Einzelne Programmzeile wiederholen:

```
<Sprungmarke>: ...  
...  
REPEATB <Sprungmarke> P=<n>  
...
```

2. Programmbereich zwischen Sprungmarke und REPEAT-Anweisung wiederholen:

```
<Sprungmarke>: ...  
...  
REPEAT <Sprungmarke> P=<n>  
...
```

3. Bereich zwischen zwei Sprungmarken wiederholen:

```
<Start-Sprungmarke>: ...  
...  
<End-Sprungmarke>: ...  
...  
REPEAT <Start-Sprungmarke> <End-Sprungmarke> P=<n>  
...
```

Hinweis

Die REPEAT-Anweisung mit den beiden Sprungmarken zu klammern, ist nicht möglich. Wird die <Start-Sprungmarke> vor der REPEAT-Anweisung gefunden und wird die <End-Sprungmarke> nicht vor der REPEAT-Anweisung erreicht, dann wird die Wiederholung zwischen <Start-Sprungmarke> und REPEAT-Anweisung durchgeführt.

4. Bereich zwischen Sprungmarke und ENDLABEL wiederholen:

```
<Sprungmarke>: ...  
...  
ENDLABEL: ...  
...  
REPEAT <Sprungmarke> P=<n>  
...
```

Hinweis

Die REPEAT-Anweisung mit der <Sprungmarke> und dem ENDLABEL zu klammern, ist nicht möglich. Wird die <Sprungmarke> vor der REPEAT-Anweisung gefunden und wird ENDLABEL nicht vor der REPEAT-Anweisung erreicht, dann wird die Wiederholung zwischen <Sprungmarke> und REPEAT-Anweisung durchgeführt.

Bedeutung

REPEATB:	Befehl zum Wiederholen einer Programmzeile
REPEAT:	Befehl zum Wiederholen eines Programmbereichs
<Sprungmarke>:	Die <Sprungmarke> kennzeichnet: <ul style="list-style-type: none">• die zu wiederholende Programmzeile (bei REPEATB) bzw.• den Beginn des zu wiederholenden Programmbereichs (bei REPEAT) Die mit der <Sprungmarke> gekennzeichnete Programmzeile kann vor oder nach der REPEAT-/REPEATB-Anweisung stehen. Gesucht wird zunächst in Richtung Programmanfang. Wird die Sprungmarke in dieser Richtung nicht gefunden, dann wird in Richtung Programmende gesucht. <p>Ausnahme: Wenn der Programmbereich zwischen Sprungmarke und REPEAT-Anweisung wiederholt werden soll (siehe 2. unter Syntax), dann muss die mit der <Sprungmarke> gekennzeichnete Programmzeile vor der REPEAT-Anweisung stehen, da in diesem Fall nur in Richtung Programmanfang gesucht wird.</p> Enthält die Zeile mit der <Sprungmarke> weitere Anweisungen, so werden diese bei jeder Wiederholung erneut ausgeführt.
ENDLABEL:	Schlüsselwort, welches das Ende eines zu wiederholenden Programmbereichs markiert Enthält die Zeile mit dem ENDLABEL weitere Anweisungen, so werden diese bei jeder Wiederholung erneut ausgeführt. ENDLABEL kann mehrfach im Programm verwendet werden.
P:	Adresse zur Angabe der Wiederholungsanzahl
<n>:	Anzahl an Programmteilwiederholungen Typ: INT Der zu wiederholende Programmteil wird <n> mal wiederholt. Nach der letzten Wiederholung wird das Programm mit der auf die REPEAT-/REPEATB-Zeile folgenden Zeile fortgesetzt. <p>Hinweis: Ist kein P=<n> angegeben, wird der zu wiederholende Programmteil genau einmal wiederholt.</p>

Beispiele

Beispiel 1: Einzelne Programmzeile wiederholen

Programmcode	Kommentar
N10 POSITION1: X10 Y20	
N20 POSITION2: CYCLE(0,,9,8)	; Positionszyklus
N30 ...	
N40 REPEATB POSITION1 P=5	; Führe SATZ N10 fünfmal aus.
N50 REPEATB POSITION2	; Führe Satz N20 einmal aus.
N60 ...	
N70 M30	

Beispiel 2: Programmbereich zwischen Sprungmarke und REPEAT-Anweisung wiederholen

Programmcode	Kommentar
N5 R10=15	
N10 Begin: R10=R10+1	; Breite
N20 Z=10-R10	
N30 G1 X=R10 F200	
N40 Y=R10	
N50 X=-R10	
N60 Y=-R10	
N70 Z=10+R10	
N80 REPEAT BEGIN P=4	; Führe Bereich N10 bis N70 viermal aus.
N90 Z10	
N100 M30	

Beispiel 3: Bereich zwischen zwei Sprungmarken wiederholen

Programmcode	Kommentar
N5 R10=15	
N10 Begin: R10=R10+1	; Breite
N20 Z=10-R10	
N30 G1 X=R10 F200	
N40 Y=R10	
N50 X=-R10	
N60 Y=-R10	
N70 END: Z=10	
N80 Z10	
N90 CYCLE(10,20,30)	
N100 REPEAT BEGIN END P=3	; Führe Bereich N10 bis N70 dreimal aus.
N110 Z10	
N120 M30	

Beispiel 4: Bereich zwischen Sprungmarke und ENDLABEL wiederholen

Programmcode	Kommentar
N10 G1 F300 Z-10	
N20 BEGIN1:	
N30 X10	
N40 Y10	
N50 BEGIN2:	
N60 X20	
N70 Y30	
N80 ENDLABEL: Z10	
N90 X0 Y0 Z0	
N100 Z-10	
N110 BEGIN3: X20	
N120 Y30	
N130 REPEAT BEGIN3 P=3	; Führe Bereich N110 bis N120 dreimal aus.
N140 REPEAT BEGIN2 P=2	; Führe Bereich N50 bis N80 zweimal aus.
N150 M100	
N160 REPEAT BEGIN1 P=2	; Führe Bereich N20 bis N80 zweimal aus.
N170 Z10	
N180 X0 Y0	
N190 M30	

Beispiel 5: Fräsbearbeitung, Bohrposition mit verschiedenen Technologien bearbeiten

Programmcode	Kommentar
N10 ZENTRIERBOHRER()	; Zentrierbohrer einwechseln.
N20 POS_1:	; Bohrpositionen 1
N30 X1 Y1	
N40 X2	
N50 Y2	
N60 X3 Y3	
N70 ENDLABEL:	
N80 POS_2:	; Bohrpositionen 2
N90 X10 Y5	
N100 X9 Y-5	
N110 X3 Y3	
N120 ENDLABEL:	
N130 BOHRER()	; Bohrer wechseln und Bohrzyklus.
N140 GEWINDE(6)	; Gewindebohrer M6 einwechseln und Gewindezyklus.
N150 REPEAT POS_1	; Wiederhole Programmabschnitt ab POS_1 einmal bis ENDLABEL.
N160 BOHRER()	; Bohrer wechseln und Bohrzyklus.
N170 GEWINDE(8)	; Gewindebohrer M8 einwechseln und Gewindezyklus.
N180 REPEAT POS_2	; Wiederhole Programmabschnitt ab POS_2 einmal bis ENDLABEL.
N190 M30	

Weitere Informationen

- Programmteilwiederholung kann geschachtelt aufgerufen werden. Jeder Aufruf belegt eine Unterprogrammebene.
- Ist während der Bearbeitung einer Programmteilwiederholung `M17` oder `RET` programmiert, so wird die Programmteilwiederholung abgebrochen. Das Programm wird mit dem auf die `REPEAT`-Zeile folgenden Satz fortgesetzt.
- In der aktuellen Programm-Anzeige wird die Programmteilwiederholung als eigene Unterprogrammebene angezeigt.
- Wird während der Programmteil-Bearbeitung Ebenenabbruch ausgelöst, so wird das Programm nach dem Aufruf der Programmteilbearbeitung fortgesetzt.

Beispiel:

Programmcode	Kommentar
N5 R10=15	
N10 BEGIN: R10=R10+1	; Breite
N20 Z=10-R10	
N30 G1 X=R10 F200	
N40 Y=R10	; Ebenenabbruch
N50 X=-R10	
N60 Y=-R10	
N70 END: Z10	
N80 Z10	
N90 CYCLE(10,20,30)	
N100 REPEAT BEGIN END P=3	
N120 Z10	; Programmbearbeitung fortsetzen.
N130 M30	

- Kontrollstrukturen und Programmteilwiederholung können kombiniert genutzt werden. Es sollte jedoch keine Überschneidungen geben. Eine Programmteilwiederholung sollte innerhalb eines Kontrollstruktur-Zweigs liegen bzw. eine Kontrollstruktur innerhalb einer Programmteilwiederholung.
- Bei der Mischung von Sprüngen und Programmteilwiederholung werden die Sätze rein sequentiell abgearbeitet. Erfolgt z. B. ein Sprung aus einer Programmteilwiederholung, so wird solange bearbeitet, bis das programmierte Programmteilstück gefunden wird.

Beispiel:

Programmcode

```
N10 G1 F300 Z-10
N20 BEGIN1:
N30 X=10
N40 Y=10
N50 GOTOF BEGIN2
N60 ENDLABEL:
N70 BEGIN2:
N80 X20
N90 Y30
N100 ENDLABEL: Z10
N110 X0 Y0 Z0
N120 Z-10
N130 REPEAT BEGIN1 P=2
N140 Z10
N150 X0 Y0
N160 M30
```

Hinweis

Die REPEAT-Anweisung sollte hinter den Verfahrssätzen stehen.

1.12 Kontrollstrukturen

Funktion

Die Steuerung arbeitet die NC-Sätze standardmäßig in der programmierten Reihenfolge ab. Diese Reihenfolge kann durch die Programmierung von alternativen Programmblöcken und Programmschleifen variiert werden. Die Programmierung dieser Kontrollstrukturen erfolgt mit den Kontrollstrukturelementen (Schlüsselwörtern) `IF . . . ELSE`, `LOOP`, `FOR`, `WHILE` und `REPEAT`.

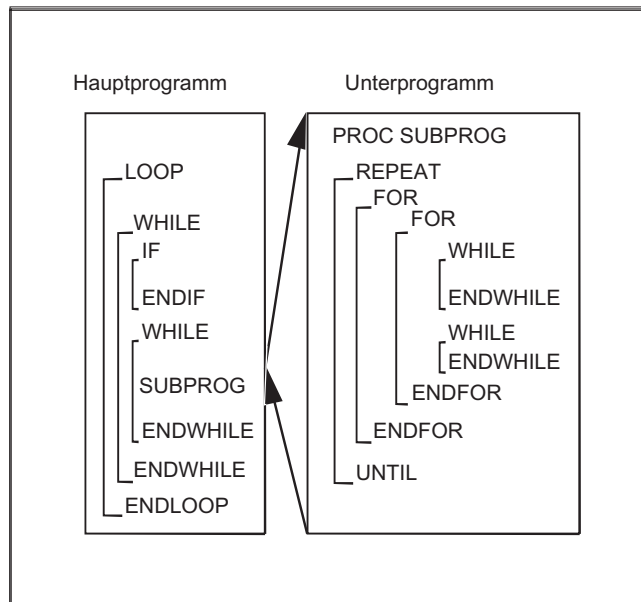
! VORSICHT
Kontrollstrukturen sind nur innerhalb des Anweisungsteils eines Programms möglich. Definitionen im Programmkopf können nicht bedingt oder wiederholt ausgeführt werden.
Schlüsselworte für Kontrollstrukturen dürfen ebenso wie Sprungziele nicht mit Makros überlagert werden. Eine Abprüfung bei der Makrodefinition findet nicht statt.

Wirksamkeit

Kontrollstrukturen gelten programm-lokal.

Schachtelungstiefe

Innerhalb jeder Unterprogrammebene ist eine Schachtelungstiefe von bis zu 16 Kontrollstrukturen möglich.



Laufzeitverhalten

Im standardmäßig aktiven Interpreterbetrieb kann durch Verwendung von Programmsprüngen ein schnellerer Programmablauf als mit Kontrollstrukturen erreicht werden.

In vorkompilierten Zyklen ist kein Unterschied zwischen Programmsprüngen und Kontrollstrukturen vorhanden.

Randbedingungen

- Sätze mit Kontrollstrukturelementen können nicht ausgeblendet werden.
- Sprungmarken (Labels) sind in Sätzen mit Kontrollstrukturelementen nicht erlaubt.
- Kontrollstrukturen werden interpretativ abgearbeitet. Bei Erkennen eines Schleifenendes wird unter Berücksichtigung der dabei gefundenen Kontrollstrukturen nach dem Schleifenanfang gesucht. Daher wird im Interpreterbetrieb die Blockstruktur eines Programms nicht komplett geprüft.
- Grundsätzlich empfiehlt sich, Kontrollstrukturen und Programmsprünge nicht gemischt zu verwenden.
- Bei Vorverarbeitung von Zyklen kann die korrekte Schachtelung von Kontrollstrukturen überprüft werden.

1.12.1 Programmschleife mit Alternative (IF, ELSE, ENDIF)

Funktion

Eine Konstruktion mit `IF` und `ELSE` wird verwendet, wenn die Programmschleife einen alternativen Programmblock enthalten soll: Wenn die `IF`-Bedingung erfüllt ist, dann wird der auf `IF` folgende Programmblock ausgeführt. Wenn die `IF`-Bedingung **nicht** erfüllt ist, dann wird der auf `ELSE` folgende alternative Programmblock ausgeführt.

Hinweis

Wenn keine Alternative erforderlich ist, dann kann eine IF-Schleife auch ohne `ELSE`-Anweisung und dem auf `ELSE` folgenden Programmblock programmiert werden.

Syntax

```
IF <Bedingung>  
...  
ELSE  
...  
ENDIF
```

Bedeutung

IF:	Leitet die IF-Schleife ein.
ELSE:	Leitet den alternativen Programmblock ein.
ENDIF:	Markiert das Ende der IF-Schleife und bewirkt Rücksprung auf den Schleifenanfang.
<Bedingung>:	Bedingung, die darüber entscheidet, welcher Programmblock durchlaufen wird.

Beispiel

Werkzeugwechselunterprogramm

Programmcode	Kommentar
PROC L6	; Werkzeugwechselroutine
N500 DEF INT TNR_AKTUELL	; Variable für aktive T-Nummer
N510 DEF INT TNR_VORWAHL	; Variable für vorgewählte T-Nummer
	; Aktuelles Werkzeug ermitteln
N520 STOPRE	
N530 IF \$P_ISTEST	; Im Programmtest-Betrieb wird ...
N540 TNR_AKTUELL = \$P_TOOLNO	; ... aus dem Programmkontext das "aktuelle" Werkzeug gelesen.
N550 ELSE	; Andernfalls wird ...
N560 TNR_AKTUELL = \$TC_MPP6[9998,1]	; ... das Werkzeug der Spindel ausgelesen.
N570 ENDIF	
N580 GETSELT(TNR_VORWAHL)	; T-Nummer des vorgewählten Werkzeugs auf der Spindel lesen.
N590 IF TNR_AKTUELL <> TNR_VORWAHL	; Wenn das vorgewählte Werkzeug noch nicht das aktuelle Werkzeug ist, dann ...
N600 G0 G40 G60 G90 SUPA X450 Y300 Z300 D0	; ... Werkzeugwechsellpunkt anfahren ...
N610 M206	; ... und Werkzeugwechsel ausführen.
N620 ENDIF	
N630 M17	

1.12.2 Endlos-Programmschleife (LOOP, ENDLOOP)

Funktion

Die Endlos-Schleife findet Verwendung in Endlos-Programmen. Am Schleifenende findet immer wieder der Rücksprung zum Schleifenanfang statt.

Syntax

```
LOOP  
...  
ENDLOOP
```

Bedeutung

LOOP: Leitet die Endlosschleife ein.
ENDLOOP: Markiert das Ende der Schleife und bewirkt Rücksprung auf den Schleifenanfang.

Beispiel

```
Programmcode  
...  
LOOP  
MSG("keine Werkzeugschneide aktiv")  
M0  
STOPRE  
ENDLOOP  
...
```

1.12.3 Zählschleife (FOR ... TO ..., ENDFOR)

Funktion

Die Zählschleife wird verwendet, wenn ein Arbeitsablauf mit einer festen Anzahl von Durchläufen wiederholt werden soll.

Syntax

```
FOR <Variable> = <Anfangswert> TO <Endwert>  
...  
ENDFOR
```

Bedeutung

FOR:	Leitet die Zählschleife ein.
ENDFOR:	Markiert das Ende der Schleife und bewirkt Rücksprung auf den Schleifenanfang, solange der Endwert der Zählung noch nicht erreicht ist.
<Variable>:	Zählvariable, die vom Anfangs- bis zum Endwert hochgezählt wird und sich bei jedem Durchlauf um den Wert "1" erhöht. Typ INT oder REAL Hinweis: Der Typ REAL wird genommen, wenn z. B. R-Parameter für eine Zählschleife programmiert werden. Ist die Zählvariable vom Typ REAL, wird ihr Wert auf einen ganzzahligen Wert gerundet.
<Anfangswert>:	Anfangswert der Zählung Bedingung: Der Anfangswert muss kleiner sein als der Endwert.
<Endwert>:	Endwert der Zählung

Beispiele

Beispiel 1: INTEGER-Variable oder R-Parameter als Zählvariable

INTEGER-Variable als Zählvariable:

Programmcode	Kommentar
DEF INT iVARIABLE1	
R10=R12-R20*R1 R11=6	
FOR iVARIABLE1= R10 TO R11	; Zählvariable = INTEGER-Variable
R20=R21*R22+R33	
ENDFOR	
M30	

R-Parameter als Zählvariable:

Programmcode	Kommentar
R11=6	
FOR R10=R12-R20*R1 TO R11	; Zählvariable = R-Parameter (Realvariable)
R20=R21*R22+R33	
ENDFOR	
M30	

Beispiel 2: Fertigung einer festen Teilstückzahl

Programmcode	Kommentar
DEF INT STUECKZAHL	; Definiert Variable vom Typ INT mit Namen "STUECKZAHL".
FOR STUECKZAHL = 0 TO 100	; Leitet die Zählschleife ein. Die Variable "STUECKZAHL" wird vom Anfangswert "0" bis zum Endwert "100" hochgezählt.
G01 ...	
ENDFOR	; Ende der Zählschleife.
M30	

1.12.4 Programmschleife mit Bedingung am Schleifenanfang (WHILE, ENDWHILE)

Funktion

Bei einer WHILE-Schleife steht die Bedingung am Schleifenanfang. Solange die Bedingung erfüllt ist, wird die WHILE-Schleife durchlaufen.

Syntax

```
WHILE <Bedingung>  
...  
ENDWHILE
```

Bedeutung

WHILE:	Leitet die Programmschleife ein.
ENDWHILE:	Markiert das Ende der Schleife und bewirkt Rücksprung auf den Schleifenanfang.
<Bedingung>:	Bedingung, die erfüllt sein muss, damit die WHILE-Schleife durchlaufen wird.

Beispiel

Programmcode	Kommentar
...	
WHILE \$AA_IW[BOHRACHSE] > -10	; Aufruf der WHILE-Schleife unter folgender Bedingung: der aktuelle WKS-Sollwert für die Bohrachse muss größer -10 sein.
G1 G91 F250 AX[BOHRACHSE] = -1	
ENDWHILE	
...	

1.12.5 Programmschleife mit Bedingung am Schleifenende (REPEAT, UNTIL)

Funktion

Bei einer REPEAT-Schleife steht die Bedingung am Schleifenende. Die REPEAT-Schleife wird einmal durchlaufen und solange wiederholt, bis die Bedingung erfüllt ist.

Syntax

```
REPEAT  
...  
UNTIL <Bedingung>
```

Bedeutung

REPEAT:	Leitet die Programmschleife ein.
UNTIL:	Markiert das Ende der Schleife und bewirkt Rücksprung auf den Schleifenanfang.
<Bedingung>:	Bedingung, die erfüllt sein muss, damit die REPEAT-Schleife nicht mehr durchlaufen wird.

Beispiel

Programmcode	Kommentar
...	
REPEAT	; Aufruf der REPEAT-Schleife.
...	
UNTIL ...	; Prüfung, ob Bedingung erfüllt ist.
...	

1.12.6 Programmbeispiel mit verschachtelten Kontrollstrukturen

Programmcode	Kommentar
LOOP	
IF NOT \$P_SEARCH	; kein Satzsuchlauf
G01 G90 X0 Z10 F1000	
WHILE \$AA_IM[X] <= 100	
G1 G91 X10 F500	; Bohrbild
Z-F100	
Z5	
ENDWHILE	
Z10	
ELSE	
MSG("Im Suchlauf wird nicht gebohrt")	
ENDIF	
\$A_OUT[1] = 1	; nächste Bohrplatte
G4 F2	
ENDLOOP	
M30	

1.13 Programmkoordinierung (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM)

Funktion

Ein Kanal kann sein eigenes Programm, unabhängig von anderen Kanälen, abarbeiten. Damit sind die ihm zeitweise zugeordneten Achsen und Spindeln über das Programm steuerbar.

Sind mehrere Kanäle an der Fertigung eines Werkstücks beteiligt, so kann eine Synchronisation der Programmabläufe erforderlich werden. Für diese Programmkoordinierung gibt es besondere Anweisungen (Kommandos).

Hinweis

Programmkoordinierung ist auch im eigenen Kanal möglich.

Voraussetzung

Die beteiligten Kanäle müssen zur **gleichen Betriebsartengruppe** (BAG) gehören.

Syntax

```
INIT (<Kanal-Nr.>, <Pfadangabe>, <Quittungsmodus>)  
START (<Kanal-Nr.>, <Kanal-Nr.>, ...)  
WAITM (<Marken-Nr.>, <Kanal-Nr.>, <Kanal-Nr.>, ...)  
WAITMC (<Marken-Nr.>, <Kanal-Nr.>, <Kanal-Nr.>, ...)  
WAITE (<Kanal-Nr.>, <Kanal-Nr.>, ...)  
SETM (<Marken-Nr.>, <Marken-Nr.>, ...)  
CLEARM (<Marken-Nr.>, <Marken-Nr.>, ...)
```

Hinweis

Die Anweisungen zur Programmkoordinierung müssen in eigenständigen Sätzen stehen.

Bedeutung

INIT:	Vordefinierte Prozedur zur Anwahl des NC-Programms, das im angegebenen Kanal abgearbeitet werden soll
<Kanal-Nr.>:	Nummer des Kanals Typ: INT
<Pfadangabe>:	Absoluter oder relativer Pfad zum NC-Programm Typ: STRING Absolute Pfadangabe: Ein absoluter Pfad muss nach folgendem Muster gebildet werden: aktuelles Verzeichnis/_N_<Name>_MPF <ul style="list-style-type: none"> • "aktuelles Verzeichnis" steht dabei für das angewählte Werkstückverzeichnis oder das Standardverzeichnis /_N_MPF_DIR. • Es ist der komplette Programmname anzugeben. Relative Pfadangabe: Bei relativer Pfadangabe gelten dieselben Regeln wie für Unterprogrammaufrufe.
<Quittungsmodus>:	Parameter vom Typ CHAR Werte: "N" ohne Quittung Die Programmbearbeitung wird nach Absenden des Kommandos fortgeführt. Der Absender wird nicht benachrichtigt, wenn das Kommando nicht erfolgreich ausgeführt werden kann. "S" synchrone Quittung Die Programmabarbeitung wird solange angehalten, bis die Empfängerkomponente das Kommando quittiert hat. Bei positiver Quittung wird der nächste Befehl abgearbeitet. Bei negativer Quittung wird eine Fehlermeldung ausgegeben.
	Hinweis: Entfällt die Angabe des Quittungsmodus, so erfolgt synchrone Quittung.

START:	<p>Vordefinierte Prozedur zum Starten der angewählten Programme in den anderen Kanälen</p> <p><Kanal-Nr.>, ...: Aufzählung der Kanalnummern Typ: INT</p>
WAITM:	<p>Vordefinierte Prozedur zum Warten auf das Erreichen einer Marke in den angegebenen Kanälen</p> <p>Im eigenen Kanal wird die angegebene Marke durch <code>WAITM</code> gesetzt. Der vorhergehenden Satz wird mit Genauhalt beendet. Die Marke wird nach Synchronisation gelöscht.</p> <p>Gleichzeitig können max. 10 Marken pro Kanal gesetzt werden.</p> <p><Marken-Nr.>: Nummer der Marke Typ: INT</p> <p><Kanal-Nr.>, ...: Aufzählung der Kanalnummern (eigener Kanal muss nicht angegeben werden) Typ: INT</p>
WAITE:	<p>Vordefinierte Prozedur zum Warten auf das Programmende in einem oder mehreren anderen Kanälen</p> <p><Kanal-Nr.>, ...: Aufzählung der Kanalnummern Typ: INT</p>
WAITMC:	<p>Vordefinierte Prozedur zum Warten auf das Erreichen einer Marke in den angegebenen Kanälen</p> <p>Im Ggs. zu <code>WAITM</code> wird Genauhalt nur eingeleitet, wenn die anderen Kanäle die Marke noch nicht erreicht haben.</p> <p>Parameter wie bei <code>WAITM</code>.</p>
SETM:	<p>Vordefinierte Prozedur zum Setzen einer oder mehrerer Marken für die Kanalkoordinierung</p> <p>Die Bearbeitung im eigenen Kanal wird davon nicht beeinflusst. <code>SETM</code> behält Gültigkeit über Reset und NC-Start hinweg.</p> <p><Marken-Nr.>, ...: Aufzählung der Markennummern</p>
CLEARM:	<p>Vordefinierte Prozedur zum Löschen einer oder mehrerer Marken für die Kanalkoordinierung</p> <p>Die Bearbeitung im eigenen Kanal wird davon nicht beeinflusst. Alle Marken im Kanal können mit <code>CLEARM()</code> gelöscht werden. <code>CLEARM(0)</code> löscht die Marke "0". <code>CLEARM</code> behält Gültigkeit über Reset und NC-Start hinweg.</p> <p><Marken-Nr.>, ...: Aufzählung der Markennummern</p>

Hinweis**Kanalnummer**

Kanalnamen müssen über Variablen in Nummern gewandelt werden.

 **VORSICHT****Kanalnummer**

Die Nummernzuordnung ist vor leichtfertiger Änderung zu sichern.

Hinweis**Kanalname**

Anstelle von Kanalnummern können auch die über \$MC_CHAN_NAME definierten Kanalnamen (Bezeichner oder Schlüsselwort) programmiert werden (Typ: STRING).

 **VORSICHT****Kanalname**

Die Namen dürfen nicht bereits in der NC in anderer Bedeutung wie z. B. als Schlüsselwort, Sprachbefehl, Achsname etc. vorhanden sein.

Hinweis

Zum Datenaustausch zwischen den Programmen können die Variablen benutzt werden, über die Kanäle gemeinsam verfügen (NCK-spezifische globale Variablen). Ansonsten wird die Programmerstellung für jeden Kanal getrennt vorgenommen.

Beispiele

Beispiel 1: Kanalnummernvergabe

Kanal mit Namen "MASCHINE" soll Kanalnummer 1 erhalten, Kanal mit Namen "LADER" soll Kanalnummer 2 erhalten. Die Variablen erhalten den gleichen Namen wie die Kanäle:
 DEF INT MASCHINE=1, LADER=2

Damit lautet beispielsweise die Anweisung `START`:
`START (MASCHINE)`

Beispiel 2: Möglichkeiten der Kanalangabe

`$MC_CHAN_NAME[0] = "CHAN_X"` ; Name des 1. Kanals
`$MC_CHAN_NAME[1] = "CHAN_Y"` ; Name des 2. Kanals

Programmierung mit:

- Kanalnummern:

Programmcode	Kommentar
<code>START(1,2)</code>	; Start im 1. und 2. Kanal ausführen.

- Kanalbezeichnern:

Programmcode	Kommentar
<code>START(CHAN_X, CHAN_Y)</code>	; Start im 1. und 2. Kanal ausführen. ; Die Bezeichner Kanal_X und Kanal_Y repräsentieren aufgrund des Maschinendatums \$MC_CHAN_NAME intern die Kanalnummer 1 und 2. Sie führen daher ebenfalls einen Start im 1. und 2. Kanal aus.

- Integer-Variablen:

Programmcode	Kommentar
<code>DEF INT chanNo1, chanNo2</code>	; Variablen definieren.
<code>chanNo1=CHAN_X chanNo2=CHAN_Y</code>	
<code>START(chanNo1, chanNo2)</code>	; Start im 1. und 2. Kanal ausführen.

Beispiel 3: INIT-Befehl mit absoluter Pfadangabe

Programmcode
<code>N10 INIT(2, "/_N_WKS_DIR/_N_WELLE1_WPD/_N_ABSPAN1_MPF")</code>

Beispiel 4: INIT-Befehl mit relativer Pfadangabe

Programmcode	Kommentar
<code>N10 INIT(2, "MYPROG")</code>	; Programm /_N_MPF_DIR/_N_MYPROG_MPF in Kanal 2 anwählen.

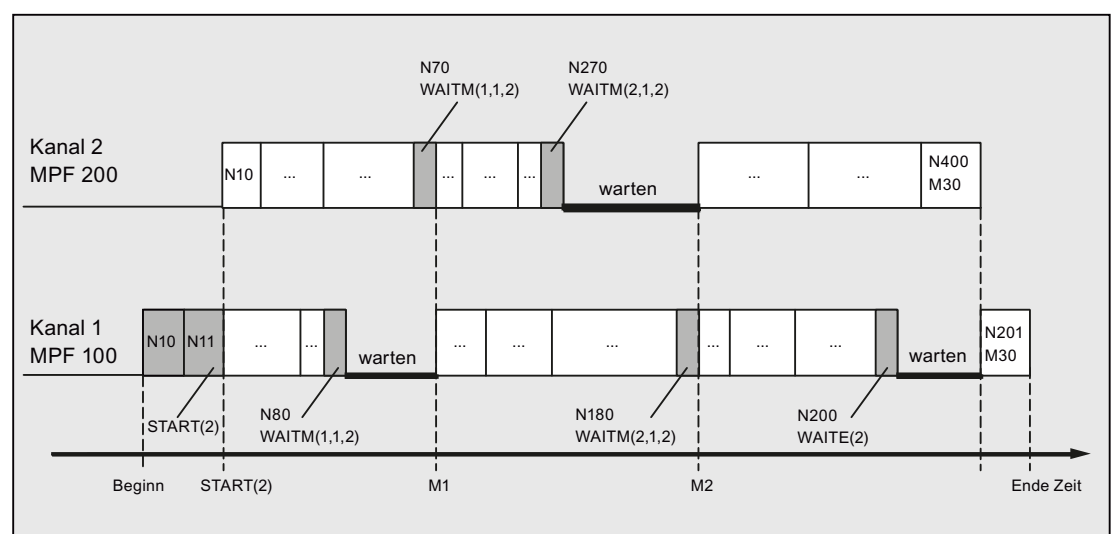
Beispiel 5: Programmkoordination mit WAITM

Kanal 1: Das Programm /_N_MPF_DIR/_N_MPF100_MPF ist angewählt.

Programmcode	Kommentar
N10 INIT(2,"MPF200","N")	
N11 START(2)	
...	; Bearbeiten im Kanal 1
N80 WAITM(1,1,2)	; Warten auf Erreichen der Wartemarke 1 in den Kanälen 1 und 2.
...	; Weiteres Bearbeiten in Kanal 1.
N180 WAITM(2,1,2)	; Warten auf Erreichen der Wartemarke 2 in den Kanälen 1 und 2.
...	; Weiteres Bearbeiten in Kanal 1.
N200 WAITE(2)	; Warten auf Programmende des Kanals 2.
N201 M30	; Programmende Kanal 1, Gesamtende.

Kanal 2: Mit dem INIT-Befehl (siehe N10 in _N_MPF100_MPF) wird das Programm _N_MPF200_MPF zur Abarbeitung im Kanal 2 angewählt.

Programmcode	Kommentar
;\$PATH=/_N_MPF_DIR	
...	; Bearbeiten im Kanal 2
N70 WAITM(1,1,2)	; Warten auf Erreichen der Wartemarke 1 in den Kanälen 1 und 2.
...	; Weiteres Bearbeiten in Kanal 2.
N270 WAITM(2,1,2)	; Warten auf Erreichen der Wartemarke 2 in den Kanälen 1 und 2.
...	; Weiteres Bearbeiten in Kanal 2.
N400 M30	; Programmende des Kanals 2.



1.14 Interruptroutine (ASUP)

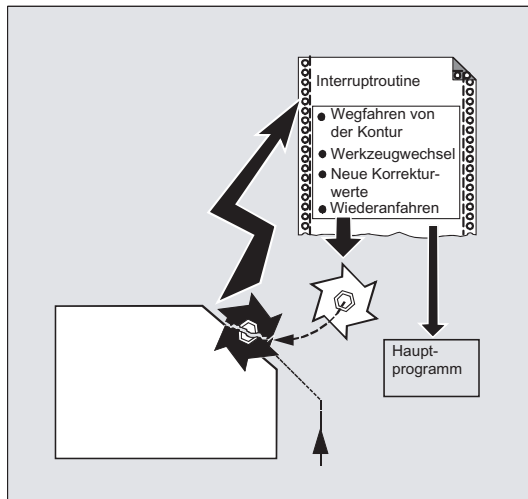
1.14.1 Funktion einer Interruptroutine

Hinweis

Die in der folgenden Beschreibung abwechselnd vorkommenden Begriffe "Asynchrones Unterprogramm (ASUP)" und "Interruptroutine" kennzeichnen die gleiche Funktionalität.

Funktion

Die Funktion einer Interruptroutine soll anhand eines typischen Beispiels verdeutlicht werden:



Während der Bearbeitung bricht das Werkzeug. Hierdurch wird ein Signal ausgelöst, das den laufenden Bearbeitungsablauf stoppt und gleichzeitig ein Unterprogramm – die sogenannte Interruptroutine – startet. In diesem Unterprogramm stehen alle Anweisungen, die in diesem Fall ausgeführt werden sollen.

Ist das Unterprogramm abgearbeitet (und hierdurch die Betriebsbereitschaft hergestellt), springt die Steuerung in das Hauptprogramm zurück und setzt die Bearbeitung – je nach `REPOS`-Befehl – an der Unterbrechungsstelle fort (siehe "Wiederauffahren an Kontur (Seite 495)").

VORSICHT

Wenn im Unterprogramm kein `REPOS`-Befehl programmiert ist, dann wird auf den Endpunkt des Satzes positioniert, der auf den unterbrochenen Satz folgt.

Literatur

Funktionshandbuch Grundfunktionen; BAG, Kanal, Programmbetrieb, Reset-Verhalten (K1), Kapitel: "Asynchrone Unterprogramme (ASUPs), Interruptroutinen"

1.14.2 Interruptroutine erstellen

Interruptroutine als Unterprogramm erstellen

Die Interruptroutine wird bei der Definition wie ein Unterprogramm gekennzeichnet.

Beispiel:

Programmcode	Kommentar
PROC ABHEB_Z	; Programmname "ABHEB_Z"
N10 ...	; Danach folgen die NC-Sätze.
...	
N50 M17	; Zum Schluss Programmende und Rückkehr ins Hauptprogramm.

Modale G-Funktionen sichern (SAVE)

Die Interruptroutine kann bei der Definition mit `SAVE` gekennzeichnet werden.

Das Attribut `SAVE` bewirkt, dass die vor dem Aufruf der Interruptroutine aktiven modalen G-Funktionen gesichert und nach dem Ende der Interruptroutine wieder reaktiviert werden (siehe "Unterprogramme mit SAVE-Mechanismus (SAVE) (Seite 180)").

Dadurch ist es möglich, die Bearbeitung nach Ablauf der Interruptroutine an der Unterbrechungsstelle fortzusetzen.

Beispiel:

Programmcode
PROC ABHEB_Z SAVE
N10 ...
...
N50 M17

Weitere Interruptroutinen zuordnen (SETINT)

Innerhalb der Interruptroutine können `SETINT`-Anweisungen (siehe "Interruptroutine zuordnen und starten (SETINT)" (Seite 132)) programmiert und hierdurch weitere Interruptroutinen scharf geschaltet werden. Das Auslösen erfolgt erst durch den Eingang.

Literatur

Für weitere Informationen zur Erstellung von Unterprogrammen siehe Kapitel "Unterprogrammtechnik, Makrotechnik".

1.14.3 Interruptroutine zuordnen und starten (SETINT, PRIO, BLSYNC)

Funktion

Die Steuerung verfügt über Signale (Eingang 1...8), die eine Unterbrechung des laufenden Programms auslösen und eine entsprechende Interruptroutine starten können.

Die Zuordnung, welcher Eingang welches Programm startet, erfolgt im Teileprogramm mit dem Befehl `SETINT`.

Falls im Teileprogramm mehrere `SETINT`-Anweisungen stehen und dadurch mehrere Signale gleichzeitig eintreffen können, müssen den zugeordneten Interruptroutinen Prioritätswerte zugewiesen werden, die die Reihenfolge bei der Abarbeitung festlegen: `PRIO=<Wert>`

Treffen während der Interruptbearbeitung neue Signale ein, unterbrechen Routinen höherer Priorität die aktuelle Interruptroutine.

Syntax

```
SETINT (<n>) PRIO=<Wert> <NAME>
SETINT (<n>) PRIO=<Wert> <NAME> BLSYNC
SETINT (<n>) PRIO=<Wert> <NAME> LIFTFAST
```

Bedeutung

<code>SETINT (<n>):</code>	Befehl: Eingang <n> einer Interruptroutine zuordnen. Die zugeordnete Interruptroutine startet, wenn Eingang <n> schaltet.
	Hinweis: Wird einem belegten Eingang eine neue Routine zugeordnet, ist die alte Zuordnung automatisch unwirksam.
<code><n>:</code>	Parameter: Nummer des Eingangs Typ: INT Wertebereich: 1 ... 8
<code>PRIO= :</code>	Befehl: Festlegung der Priorität
<code><Wert>:</code>	Prioritätswert Typ: INT Wertebereich: 1 ... 128 Priorität 1 entspricht der höchsten Priorität.
<code><NAME>:</code>	Name des Unterprogramms (Interruptroutine), das abgearbeitet werden soll.
<code>BLSYNC:</code>	Wenn die <code>SETINT</code> -Anweisung zusammen mit <code>BLSYNC</code> programmiert wird, dann wird beim Eintreffen des Interruptsignals der laufende Programmsatz noch abgearbeitet und erst danach die Interruptroutine gestartet.
<code>LIFTFAST:</code>	Wenn die <code>SETINT</code> -Anweisung zusammen mit <code>LIFTFAST</code> programmiert wird, dann wird beim Eintreffen des Interruptsignals vor dem Start der Interruptroutine ein "Schnellabheben des Werkzeugs von der Kontur" durchgeführt (siehe "Schnellabheben von der Kontur (SETINT LIFTFAST, ALF) (Seite 136)").

Beispiele

Beispiel 1: Interruptroutinen zuordnen und Priorität festlegen

Programmcode	Kommentar
...	
N20 SETINT(3) PRIO=1 ABHEB_Z	; Wenn Eingang 3 schaltet, dann soll die Interruptroutine "ABHEB_Z" starten.
N30 SETINT(2) PRIO=2 ABHEB_X	; Wenn Eingang 2 schaltet, dann soll die Interruptroutine "ABHEB_X" starten.
...	

Die Interruptroutinen werden in der Reihenfolge der Prioritätswerte nacheinander abgearbeitet, wenn die Eingänge gleichzeitig anstehen: zuerst "ABHEB_Z", dann "ABHEB_X".

Beispiel 2: Interruptroutine neu zuordnen

Programmcode	Kommentar
...	
N20 SETINT(3) PRIO=2 ABHEB_Z	; Wenn Eingang 3 schaltet, dann soll die Interruptroutine "ABHEB_Z" starten.
...	
N120 SETINT(3) PRIO=1 ABHEB_X	; Eingang 3 wird eine neue Interruptroutine zugeordnet: statt "ABHEB_Z" soll "ABHEB_X" starten, wenn Eingang 3 schaltet.

1.14.4 Zuordnung einer Interruptroutine deaktivieren/reaktivieren (DISABLE, ENABLE)

Funktion

Eine SETINT-Anweisung kann mit `DISABLE` deaktiviert und mit `ENABLE` wieder aktiviert werden, ohne dass die Zuordnung Eingang → Interruptroutine verloren geht.

Syntax

```
DISABLE (<n>)
ENABLE (<n>)
```

Bedeutung

`DISABLE (<n>):` Befehl: **Deaktivieren** der Interruptroutinen-Zuordnung von Eingang <n>
`ENABLE (<n>):` Befehl: **Reaktivieren** der Interruptroutinen-Zuordnung von Eingang <n>
<n>: Parameter: Nummer des Eingangs
Typ: INT
Wertebereich: 1 ... 8

Beispiel

Programmcode	Kommentar
...	
N20 SETINT(3) PRIO=1 ABHEB_Z	; Wenn Eingang 3 schaltet, dann soll die Interruptroutine "ABHEB_Z" starten.
...	
N90 DISABLE(3)	; Die SETINT-Anweisung aus N20 wird deaktiviert.
...	
N130 ENABLE(3)	; Die SETINT-Anweisung aus N20 wird wieder aktiviert.
...	

1.14.5 Zuordnung einer Interruptroutine löschen (CLRINT)

Funktion

Eine mit `SETINT` definierte Zuordnung Eingang → Interruptroutine kann mit `CLRINT` gelöscht werden.

Syntax

`CLRINT (<n>)`

Bedeutung

`CLRINT (<n>):` Befehl: Löschen der Interruptroutinen-Zuordnung von Eingang <n>
<n>: Parameter: Nummer des Eingangs
Typ: INT
Wertebereich: 1 ... 8

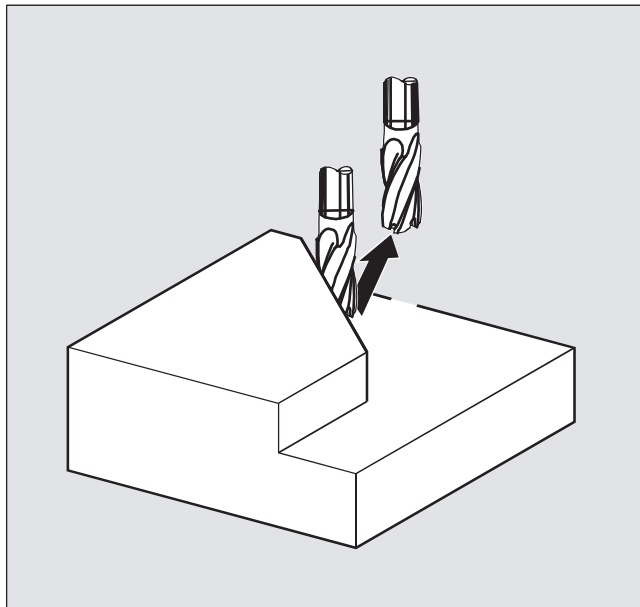
Beispiel

Programmcode	Kommentar
...	
N20 SETINT(3) PRIO=2 ABHEB_Z	;
...	
N50 CLRINT(3)	; Die Zuordnung zwischen Eingang "3" und der Interruptroutine "ABHEB_Z" ist gelöscht.
...	

1.14.6 Schnellabheben von der Kontur (SETINT LIFTFAST, ALF)

Funktion

Bei einer `SETINT`-Anweisung mit `LIFTFAST` wird beim Schalten des Eingangs das Werkzeug durch schnelles Abheben von der Werkstückkontur weggefahren.



Der weitere Ablauf ist davon abhängig, ob die `SETINT`-Anweisung neben `LIFTFAST` eine Interruptroutine enthält:

Mit Interruptroutine:	Nach dem Schnellabheben wird die Interruptroutine ausgeführt.
Ohne Interruptroutine:	Die Bearbeitung wird nach dem Schnellabheben mit Alarm gestoppt.

Syntax

```
SETINT(<n>) PRIO=1 LIFTFAST  
SETINT(<n>) PRIO=1 <NAME> LIFTFAST
```


Bedeutung

SETINT (<n>):	Befehl: Eingang <n> einer Interruptroutine zuordnen. Die zugeordnete Interruptroutine startet, wenn Eingang <n> schaltet.
<n>:	Parameter: Nummer des Eingangs Typ: INT Wertebereich: 1 ... 8
PRIO= :	Festlegung der Priorität
<Wert>:	Prioritätswert Wertebereich: 1 ... 128 Priorität 1 entspricht der höchsten Priorität.
<NAME>:	Name des Unterprogramms (Interruptroutine), das abgearbeitet werden soll.
LIFTFAST:	Befehl: Schnellabheben von der Kontur
ALF=... :	Befehl: Programmierbare Verfahrrichtung (steht im Bewegungssatz) Zu den Programmiermöglichkeiten mit ALF siehe Thema " Verfahrrichtung beim Schnellabheben von der Kontur (Seite 139) ".

Randbedingungen

Verhalten bei aktivem Frame mit Spiegelung

Bei der Bestimmung der Abheberichtung wird geprüft, ob ein Frame mit Spiegelung aktiv ist. In diesem Fall werden bei der Abheberichtung bezogen auf die Tangentenrichtung rechts und links vertauscht. Die Richtungsanteile in Werkzeugrichtung werden nicht gespiegelt. Aktiviert wird dieses Verhalten durch die MD-Einstellung:

MD21202 \$MC_LIFTFAST_WITH_MIRROR = TRUE

Beispiel

Ein abgebrochenes Werkzeug soll automatisch durch ein Schwesterwerkzeug ersetzt werden. Die Bearbeitung wird dann mit dem neuen Werkzeug fortgesetzt.

Hauptprogramm:

Hauptprogramm	Kommentar
N10 SETINT(1) PRIO=1 W_WECHS LIFTFAST	; Wenn Eingang 1 schaltet, wird sofort das Werkzeug mit Schnellabheben (Code Nr. 7 für Werkzeugradiuskorrektur G41) von der Kontur weggefahren. Dann wird die Interruptroutine "W_WECHS" abgearbeitet.
N20 G0 Z100 G17 T1 ALF=7 D1	
N30 G0 X-5 Y-22 Z2 M3 S300	
N40 Z-7	
N50 G41 G1 X16 Y16 F200	
N60 Y35	
N70 X53 Y65	
N90 X71.5 Y16	
N100 X16	
N110 G40 G0 Z100 M30	

Unterprogramm:

Unterprogramm	Kommentar
PROC W_WECHS SAVE	; Unterprogramm mit Speicherung des aktuellen Betriebszustandes
N10 G0 Z100 M5	; Werkzeugwechselposition, Spindelstopp
N20 T11 M6 D1 G41	; Werkzeug wechseln
N30 REPOS L RMB M3	; Kontur wiederanfahren und Rücksprung ins Hauptprogramm (wird in einem Satz programmiert)

1.14.7 Verfahrrichtung beim Schnellabheben von der Kontur

Rückzugsbewegung

Die Ebene der Rückzugsbewegung wird durch folgende G-Codes bestimmt:

- LFTXT

Die Ebene der Rückzugsbewegung wird aus der Bahntangente und der Werkzeugrichtung bestimmt (Standardeinstellung).

- LFWP

Die Ebene der Rückzugsbewegung ist die aktive Arbeitsebene, die mit den G-Codes G_{17} , G_{18} oder G_{19} ausgewählt wird. Die Richtung der Rückzugsbewegung ist unabhängig von der Bahntangente. Damit ist ein achsparalleles Schnellabheben programmierbar.

- LFPOS

Rückzug der mit $POLFMASK$ / $POLFMLIN$ bekannt gemachten Achse auf die mit $POLF$ programmierte absolute Achsposition.

ALF hat keinen Einfluss auf die Abheberichtung für mehrere Achsen sowie für mehrere Achsen im linearen Zusammenhang.

Literatur:

Programmierhandbuch Grundlagen; Kapitel: "Schnellrückzug während Gewindeschneiden"

Programmierbare Verfahrrichtung (ALF=...)

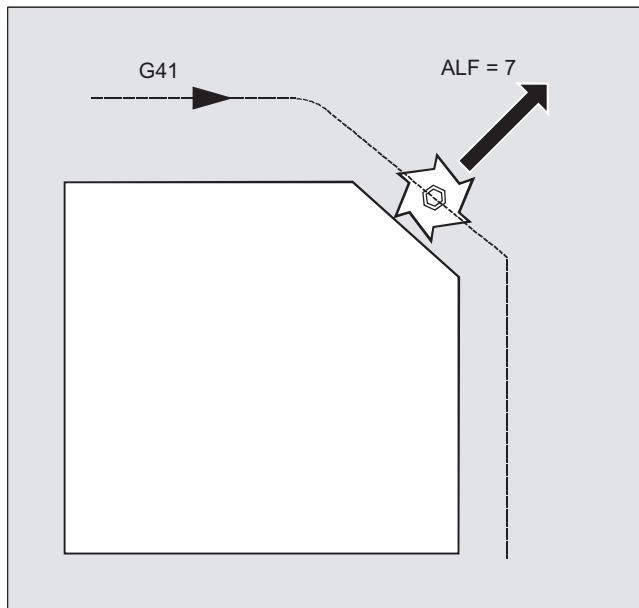
In der Ebene der Rückzugsbewegung wird mit ALF die Richtung in diskreten Schritten von 45 Grad programmiert.

Die möglichen Verfahrrichtungen sind in der Steuerung unter speziellen Code-Nummern gespeichert und unter dieser Nummer abrufbar.

Beispiel:

```
Programmcode
-----
N10 SETINT(2) PRIO=1 ABHEB_Z LIFTFAST
ALF=7
```

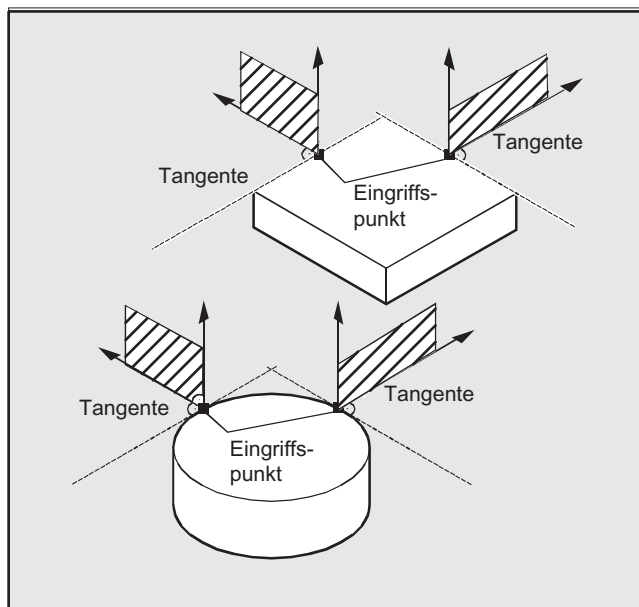
Das Werkzeug fährt bei eingeschaltetem G_{41} (Bearbeitungsrichtung links von der Kontur) senkrecht von der Kontur weg.



Bezugsebene für die Beschreibung der Verfahrrichtungen bei LFTXT

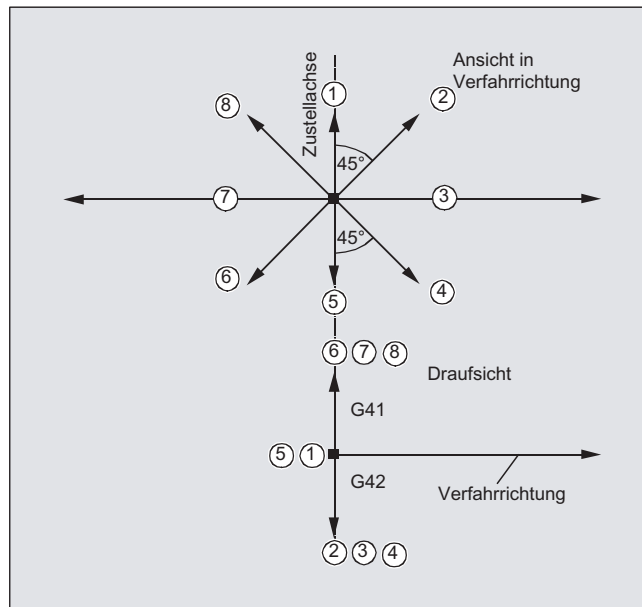
Im Eingriffspunkt des Werkzeugs an der programmierten Kontur wird eine Ebene aufgespannt, die als Bezug für die Angabe der Abhebebewegung mit der entsprechenden Code-Nummer dient.

Die Bezugsebene wird aufgespannt aus der Werkzeuglängsachse (Zustellrichtung) und einem Vektor, der zu dieser und senkrecht zur Tangente im Eingriffspunkt des Werkzeugs an der Kontur steht.



Code-Nummern mit Verfahrrichtungen bei LFTXT

Ausgehend von der Bezugsebene finden Sie in folgender Abbildung die Code-Nummern mit Verfahrrichtungen.



Für $ALF=1$ ist der Rückzug in Werkzeugrichtung festgelegt.

Mit $ALF=0$ ist die Funktion "Schnellabheben" ausgeschaltet.

! VORSICHT

Bei eingeschalteter Werkzeugradiuskorrektur sollten:

- bei $G41$ die Codierungen 2, 3, 4
- bei $G42$ die Codierungen 6, 7, 8

nicht verwendet werden, da in diesen Fällen das Werkzeug zur Kontur hinfahren und mit dem Werkstück kollidieren würde.

Code-Nummern mit Verfahrrichtungen bei LFWP

Bei $LFWP$ ergibt sich die Richtung in der Arbeitsebene nach folgender Zuordnung:

- $G17$: X/Y-Ebene
 $ALF=1$: Rückzug in X-Richtung
 $ALF=3$: Rückzug in Y-Richtung
- $G18$: Z/X-Ebene
 $ALF=1$: Rückzug in Z-Richtung
 $ALF=3$: Rückzug in X-Richtung
- $G19$: Y/Z-Ebene
 $ALF=1$: Rückzug in Y-Richtung
 $ALF=3$: Rückzug in Z-Richtung

1.14.8 Bewegungsablauf bei Interruptroutinen

Interruptroutine ohne LIFTFAST

Die Achsbewegungen werden auf der Bahn bis zum Stillstand abgebremst. Anschließend startet die Interruptroutine.

Die Stillstandsposition wird als Unterbrechungsposition abgespeichert und wird bei `REPOS` mit `RMI` am Ende der Interruptroutine angefahren.

Interruptroutine mit LIFTFAST

Die Achsbewegungen werden auf der Bahn abgebremst. Gleichzeitig wird die `LIFTFAST`-Bewegung als überlagerte Bewegung ausgeführt. Wenn die Bahnbewegung und `LIFTFAST`-Bewegung zum Stillstand gekommen sind, wird die Interruptroutine gestartet.

Als Unterbrechungsposition wird die Position auf der Kontur abgespeichert, bei der die `LIFTFAST`-Bewegung gestartet und dadurch die Bahn verlassen wurde.

Die Interruptroutine verhält sich mit `LIFTFAST` und `ALF=0` identisch wie die Interruptroutine ohne `LIFTFAST`.

Hinweis

Der Betrag, um den die Geometrieachsen beim Schnellabheben von der Kontur wegfahren, ist über ein Maschinendatum einstellbar.

1.15 Achstausch, Spindeltausch (RELEASE, GET, GETD)

Funktion

Eine oder mehrere Achsen bzw. Spindeln können immer nur in einem Kanal interpoliert werden. Muss eine Achse wechselweise in zwei verschiedenen Kanälen arbeiten (z. B. Palettenwechsler), so muss sie zunächst im aktuellen Kanal freigegeben und dann in den anderen Kanal übernommen werden. Die Achse wird zwischen den Kanälen getauscht.

Achstauscherweiterungen

Eine Achse/Spindel kann mit Vorlaufstopp und Synchronisation zwischen Vorlauf und Hauptlauf oder alternativ auch ohne Vorlaufstopp getauscht werden. Außerdem ist ein Achstausch auch möglich über

- Achscontainer-Drehung AXCTSWE bzw. AXCTWED mittels impliziten GET/GETD.
- Frame mit Rotation, wenn diese Achse hierüber mit anderen Achsen verknüpft ist.
- Synchronaktionen, siehe Bewegungssynchronaktionen, "Achstausch RELEASE, GET".

Maschinenhersteller

Bitte beachten Sie die Angaben des Maschinenherstellers. Über projektierbare Maschinendaten muss eine Achse für den Achstausch in allen Kanälen eindeutig definiert sein und das Achstauschverhalten ist auch über Maschinendaten veränderbar einstellbar.

Syntax

RELEASE (Achsname, Achsname, ...) oder RELEASE (S1)

GET (Achsname, Achsname, ...) oder GET (S2)

GETD (Achsname, Achsname, ...) oder GETD (S3)


Mit GETD (GET Directly) wird eine Achse direkt aus einem anderen Kanal geholt. Das bedeutet, dass zu diesem GETD kein passendes RELEASE in einem anderen Kanal programmiert sein muss. Es bedeutet aber auch, dass jetzt eine andere Kanalkommunikation aufgebaut werden muss (z. B. Waitmarken).

Bedeutung

RELEASE (Achsname, Achsname, ...):	Freigeben der Achse(n)
GET (Achsname, Achsname, ...):	Übernehmen der Achse(n)
GETD (Achsname, Achsname, ...):	Direktes Übernehmen der Achse(n)
Achsname:	Achsnummer im System: AX1, AX2, ... oder Angabe der Maschinenachsnamen
RELEASE (S1) :	Freigeben der Spindel S1, S2, ...
GET (S2) :	Übernehmen der Spindel S1, S2, ...
GETD (S3) :	Direktes Übernehmen der Spindel S1, S2, ...

GET-Anforderung ohne Vorlaufstopp

Wird nach einer GET-Anforderung **ohne** Vorlaufstopp die Achse mit `RELEASE (Achse)` oder `WAITP (Achse)` wieder freigegeben, so führt ein nachfolgender `GET` zu einem `GET` **mit** Vorlaufstopp.

 **VORSICHT**

Eine mit GET übernommene Achse bzw. Spindel bleibt auch nach einem Tasten- oder Programm-RESET diesem Kanal zugeordnet.

Bei neuem Programmstart muss die Zuordnung der getauschten Achsen bzw. Spindeln programmtechnisch erfolgen, falls die Achse in ihrem Grundkanal benötigt wird.

Bei POWER ON wird sie dem im Maschinendatum hinterlegten Kanal zugeordnet.

Beispiele

Beispiel 1: Achstausch zwischen zwei Kanälen

Von 6 Achsen werden in Kanal 1 zur Bearbeitung benutzt: 1., 2., 3. und 4. Achse. 5. und 6. Achse werden in Kanal 2 zum Werkstückwechsel benutzt.

Achse 2 soll zwischen beiden Kanälen getauscht werden können und nach POWER ON dem Kanal 1 zugeordnet sein.

Programm "MAIN" in Kanal 1:

Programmcode	Kommentar
INIT (2, "TAUSCH2")	; Programm TAUSCH2 im Kanal 2 anwählen.
N... START (2)	; Programm in Kanal 2 starten.
N... GET (AX2)	; Achse AX2 übernehmen.
...	
N... RELEASE (AX2)	; Achse AX2 freigeben.
N... WAITM (1,1,2)	; Warten auf WAIT-Marke in Kanal 1 und 2 zur Synchronisation in den beiden Kanälen.
...	; Weiterer Ablauf nach Achstausch.
N... M30	

Programm "TAUSCH2" in Kanal 2:

Programmierung	Kommentar
N... RELEASE (AX2)	
N160 WAITM(1,1,2)	; Warten auf WAIT-Marke in Kanal 1 und 2 zur Synchronisation in den beiden Kanälen.
N150 GET (AX2)	; Achse AX2 übernehmen.
...	; Weiterer Ablauf nach Achstausch.
N... M30	

Beispiel 2: Achstausch ohne Synchronisierung

Wenn die Achse nicht synchronisiert werden muss, wird durch GET kein Vorlaufstopp erzeugt.

Programmierung	Kommentar
N01 G0 X0	
N02 RELEASE (AX5)	
N03 G64 X10	
N04 X20	
N05 GET (AX5)	; Wenn keine Synchronisation nötig, wird dies kein ausführbarer Satz.
N06 G01 F5000	; Kein ausführbarer Satz.
N07 X20	; Kein ausführbarer Satz, da X-Position wie in N04.
N08 X30	; Erster ausführbarer Satz nach N05.
...	

Beispiel 3: Aktivierung eines Achstausches ohne Vorlaufstopp

Voraussetzung: Der Achstausch ohne Vorlaufstopp muss über ein Maschinendatum projiziert werden.

Programmierung	Kommentar
N010 M4 S100	
N011 G4 F2	
N020 M5	
N021 SPOS=0	
N022 POS[B]=1	
N023 WAITP(B)	; Achse B wird zur neutralen Achse.
N030 X1 F10	
N031 X100 F500	
N032 X200	
N040 M3 S500	; Achse löst kein Vorlaufstopp/REORG aus.
N041 G4 F2	
N050 M5	
N099 M30	

Wird die Spindel bzw. Achse B unmittelbar nach dem Satz N023 als **PLC-Achse** z. B. auf 180 Grad und zurück auf 1 Grad verfahren, dann wird diese Achse wieder zur neutralen Achse und löst im Satz N40 keinen Vorlaufstopp auf.

Voraussetzung

Voraussetzungen für den Achstausch

- Die Achse muss über Maschinendaten in allen Kanälen definiert sein, die Achse verwenden wollen.
- Über das **achsspezifische** Maschinendatum muss festgelegt sein, welchem Kanal die Achse nach POWER ON zugeordnet werden soll.

Beschreibung

Achse freigeben: RELEASE

Bei der Achsfreigabe ist zu beachten:

1. Die Achse darf an keiner Transformation beteiligt sein.
2. Bei Achskopplungen (Tangentialsteuerung), müssen alle Achsen des Verbands freigegeben werden.
3. Eine konkurrierende Positionierachse kann in diesem Zustand nicht getauscht werden.
4. Bei einer Gantry-Masterachse werden auch alle Folgeachsen getauscht.
5. Bei Achskopplungen (Mitschleppen, Leitwertkopplung, Elektronisches Getriebe) kann nur die Leitachse des Verbandes freigegeben werden.

Achse übernehmen: GET

Mit diesem Befehl wird der eigentliche Achstausch durchgeführt. Die Verantwortung für die Achse liegt vollständig bei dem Kanal, in dem der Befehl programmiert wurde.

Auswirkungen von GET:

Achstausch mit Synchronisierung:

Eine Achse muss immer dann synchronisiert werden, wenn sie zwischenzeitlich in einem anderen Kanal oder der PLC zugeordnet war, und vor dem GET keine Synchronisierung durch "WAITP", G74 oder Restweglöschen stattgefunden hat.

- Ein Vorlaufstopp erfolgt (wie bei STOPRE).
- Die Bearbeitung wird so lange unterbrochen, bis der Tausch vollständig ausgeführt ist.

Automatisches "GET"

Wenn eine Achse prinzipiell im Kanal verfügbar, jedoch derzeit nicht als "Kanal-Achse" vorhanden ist, wird automatisch ein GET ausgeführt. Falls die Achse(n) schon synchronisiert ist (sind), wird kein Vorlaufstopp erzeugt.

Achstauschverhalten veränderbar einstellen

Der Abgabezeitpunkt von Achsen lässt sich über ein Maschinendatum wie folgt einstellen:

- Automatischer Achstausch findet zwischen zwei Kanälen auch dann statt, wenn die Achse durch WAITP in einen neutralen Zustand gebracht wurde (Verhalten wie bisher)
- Bei der Anforderung einer Achs-Containerdrehung werden alle dem ausführenden Kanal zuordenbaren Achsen des Achs-Containers mittels impliziten GET bzw. GETD in den Kanal geholt. Ein anschließender Achstausch ist erst nach dem Abschluss der Achs-Containerdrehung wieder erlaubt.
- Nach einem eingeschobenen Zwischensatz im Hauptlauf wird geprüft, ob ein Reorganisieren erforderlich ist oder nicht. Nur wenn die Achszustände dieses Satzes mit den aktuellen Achszuständen **nicht** übereinstimmen, ist ein Reorganisieren erforderlich.
- Statt eines GET-Satzes mit Vorlaufstopp und Synchronisation zwischen Vorlauf und Hauptlauf kann ein Achstausch auch ohne Vorlaufstopp erfolgen. Es wird dann nur ein Zwischensatz mit der GET-Anforderung erzeugt. Im Hauptlauf wird bei Abarbeitung dieses Satzes überprüft, ob die Zustände der Achse im Satz mit den aktuellen Achszuständen übereinstimmen.

Weitere Informationen zur Funktionalität eines Achs- oder Spindeltausches siehe Funktionshandbuch Erweiterungsfunktionen; BAGs, Kanäle, Achstausch (K5).

1.16 Achse einem anderen Kanal übergeben (AXTOCHAN)

Funktion

Mit dem Sprachbefehl `AXTOCHAN` kann eine Achse angefordert werden, um diese Achse einem anderen Kanal zu übergeben. Die Achse kann sowohl vom NC-Teileprogramm als auch aus einer Synchronaktion heraus in den entsprechenden Kanal gebracht werden.

Syntax

`AXTOCHAN (Achurname, Kanalnummer [, Achurname, Kanalnummer [, ...]])`

Bedeutung

Element	Beschreibung
AXTOCHAN:	Achse für einen bestimmten Kanal anfordern
Achsname:	Achszuordnung im System: X, Y, ... oder Angabe der beteiligten Maschinenachsennamen. Der auszuführende Kanal muss nicht der eigene Kanal sein und es muss auch nicht der Kanal sein, der aktuell das Interpolationsrecht für die Achse besitzt
Kanalnummer:	Nummer des Kanals, dem die Achse zugeordnet werden soll

Hinweis

Konkurrierende Positionierachse und ausschließlich PLC kontrollierte Achse

Eine PLC-Achse kann als konkurrierende Positionierachse den Kanal nicht wechseln. Eine ausschließlich von der PLC kontrollierte Achse kann nicht dem NC-Programm zugeordnet werden.

Literatur:

Funktionshandbuch Erweiterungsfunktionen; Positionierachsen (P2)

Beispiel**AXTOCHAN im NC-Programm**

Die Achsen X und Y sind im 1. Kanal und im 2. Kanal bekannt. Aktuell hat der Kanal 1 das Interpolationsrecht und im Kanal 1 wird folgendes Programm gestartet:

Programmcode	Kommentar
N110 AXTOCHAN(Y,2)	; Y-Achse in den 2. Kanal schieben.
N111 M0	
N120 AXTOCHAN(Y,1)	; Y-Achse wieder zurückholen (neutral).
N121 M0	
N130 AXTOCHAN(Y,2,X,2)	; Y-Achse und X-Achse in den 2. Kanal schieben (Achsen neutral).
N131 M0	
N140 AXTOCHAN(Y,2)	; Y-Achse in den 2. Kanal schieben (NC-Programm).
N141 M0	

Weitere Informationen**AXTOCHAN im NC-Programm**

Dabei wird nur bei einer Anforderung der Achse für das NC-Programm im eigenen Kanal ein `GET` durchgeführt und damit auch auf die tatsächliche Zustandsänderung gewartet. Wird die Achse für einen anderen Kanal angefordert oder soll sie zur neutralen Achse im eigenen Kanal werden, dann nur wird die Anforderung entsprechend abgesetzt.

AXTOCHAN aus einer Synchronaktion

Wird eine Achse für den eigenen Kanal angefordert so wird `AXTOCHAN` aus einer Synchronaktion auf ein `GET` aus einer Synchronaktion abgebildet. In diesem Fall wird die Achse bei der ersten Anforderung für den eigenen Kanal zur neutralen Achse. Bei der zweiten Anforderung wird die Achse dem NC-Programm analog zur `GET`-Anforderung im NC-Programm zugeordnet. Zur `GET`-Anforderung aus einer Synchronaktion siehe Kapitel "Bewegungssynchronaktionen".

1.17 Maschinendaten wirksam setzen (NEWCONF)

Funktion

Mit dem Befehl `NEWCONF` werden alle Maschinendaten der Wirksamkeitsstufe "NEW_CONFIG" wirksam gesetzt. Die Funktion kann auch in der Bedienoberfläche HMI durch Betätigen des Softkeys "MD wirksam setzen" aktiviert werden.

Bei der Ausführung der Funktion "NEWCONF" erfolgt ein impliziter Vorlaufstopp, d. h. die Bahnbewegung wird unterbrochen.

Syntax

`NEWCONF`

Bedeutung

`NEWCONF`: Befehl zum Wirksamsetzen aller Maschinendaten der Wirksamkeitsstufe "NEW_CONFIG"

NEWCONF aus dem Teileprogramm kanalübergreifend ausführen

Werden axiale Maschinendaten aus dem Teileprogramm heraus verändert und anschließend mit `NEWCONF` aktiviert, so setzt der Befehl `NEWCONF` nur die Maschinendaten aktiv, die Änderungen für den Kanal des Teileprogramms bewirken.

Hinweis

Um alle Änderungen sicher wirksam werden zu lassen, muss der Befehl `NEWCONF` in jedem Kanal ausgeführt werden, in dem auch die von den veränderten Maschinendaten betroffenen Achsen oder Funktionen aktuell gerechnet werden.

Bei `NEWCONF` werden keine axialen Maschinendaten wirksam gesetzt.

Für PLC-kontrollierte Achsen muss ein axialer RESET ausgeführt werden.

Beispiel

Fräsbearbeitung: Bohrposition mit verschiedenen Technologien bearbeiten

Programmcode	Kommentar
N10 \$MA_CONTOUR_TOL[AX]=1.0	; Maschinendatum ändern.
N20 NEWCONF	; Maschinendaten wirksam setzen.
...	

1.18 Datei schreiben (WRITE)

Funktion

Mit dem `WRITE`-Befehl können Sätze/Daten aus dem NC-Programm an das Ende einer angegebenen Datei im passiven Filesystem (Protokolldatei) geschrieben werden. Dies kann auch das gerade in Abarbeitung befindliche Programm sein.

Hinweis

Eine per `WRITE`-Befehl zu beschreibende Datei wird neu angelegt, wenn sie nicht in der NC existiert.

Ablageort ist der statische NC-Speicher. Bei SINUMERIK 840D sl ist dies die CompactFlash Card. Gegenüber SINUMERIK 840D erhöht sich dadurch die Laufzeit des `WRITE`-Befehls um ca. 75 ms.

Existiert eine Datei gleichen Namens auf der Festplatte, wird diese nach dem Schließen der Datei (in der NC) überschrieben. Abhilfe: Über die Bedienoberfläche den Dateinamen in der NC ändern.

Darüber hinaus ist es mit dem `WRITE`-Befehl auch möglich, Daten aus einem NC-Programm heraus auf ein externes Gerät / eine externe Datei zu schreiben (siehe auch "Process DataShare - Ausgabe auf ein externes Gerät/Datei (EXTOPEN, WRITE, EXTCLOSE) (Seite 623)").

Voraussetzung

Die aktuell eingestellte Schutzstufe muss gleich oder größer dem `WRITE`-Recht der Datei sein. Ist dies nicht der Fall, wird der Zugriff mit Fehlermeldung (Rückgabewert der Fehlervariablen = 13) abgelehnt.

Syntax

```
DEF INT <Fehler>  
...  
WRITE (<Fehler>, "<Dateiname>"/"<ExtG>", "<Satz/Daten>")
```

Bedeutung

<code>WRITE:</code>	Befehl zum Anfügen eines Satzes bzw. von Daten an das Ende der angegebenen Datei
<code><Fehler>:</code>	Parameter 1: Variable für die Rückgabe des Fehlerwerts
	Typ: INT
	Wert: 0 kein Fehler
	1 Pfad nicht erlaubt

- 2 Pfad nicht gefunden
- 3 Datei nicht gefunden
- 4 falscher Dateityp
- 10 Datei ist voll
- 11 Datei wird benutzt
- 12 keine Ressourcen frei
- 13 keine Zugriffsrechte
- 14 fehlendes oder fehlgeschlagenes EXTOPEN für das Ausgabegerät
- 15 Fehler beim Schreiben auf externes Gerät
- 16 ungültiger externer Pfad programmiert

<Dateiname>:

Parameter 2: Name der Datei im passiven Filesystem, in der der angegebene Satz bzw. die angegebenen Daten angefügt werden sollen

Typ: STRING

Bei der Angabe des Dateinamens sind folgende Punkte zu beachten:

- Der angegebene Dateiname darf keine Leer- oder Steuerzeichen (Zeichen mit ASCII-Code ≤ 32) enthalten, da sonst der `WRITE`-Befehl mit Fehlerkennung 1 "Pfad nicht erlaubt" abgebrochen wird.
- Der Dateiname kann mit Pfadangabe und Datei-Kennung angegeben werden:

- Pfangaben

Pfangaben müssen absolut sein, d. h. sie beginnen mit "/".

Ohne Pfangabe wird die Datei im aktuellen Verzeichnis (= Verzeichnis des angewählten Programms) abgelegt.

- Datei-Kennung

Enthält der Dateiname keine Domain-Kennung ("_N_"), wird er entsprechend ergänzt.

Enthält der Dateinamen als viertletztes Zeichen einen Unterstrich "_", so werden die nachfolgenden drei Zeichen als Datei-Kennung interpretiert. Um bei allen Datei-Befehlen denselben Dateinamen verwenden zu können, z. B. über eine Variable vom Typ STRING, dürfen nur die Datei-Kennungen `_SPF` und `_MPF` verwendet werden.

Ist keine Kennung `"_MPF"` oder `"_SPF"` angegeben, wird automatisch `_MPF` ergänzt.

- Die Länge des Dateinamens darf maximal 32 Bytes, die Länge der Pfangabe maximal 128 Bytes betragen.

Beispiel:

```
"PROFILE"  
"_N_PROFILE"  
"_N_PROFILE_MPF"  
"/_N_MPF_DIR/_N_PROFILE_MPF/"
```


<ExtG>: Sollen Daten auf ein externes Gerät/Datei ausgegeben werden, muss statt des Dateinamens der symbolische Bezeichner für das zu öffnende externe Gerät/Datei angegeben werden.
 Typ: STRING
 Weitere Informationen siehe "Process DataShare - Ausgabe auf ein externes Gerät/Datei (EXTOPEN, WRITE, EXTCLOSE) (Seite 623)".
Hinweis:
 Der Bezeichner muss identisch zu dem im EXTOPEN-Befehl angegebenen Bezeichner sein.

<Satz/Daten>: Satz bzw. Daten, die in der angegebenen Datei angefügt werden sollen.
 Typ: STRING

Hinweis

Beim Schreiben ins passive Dateisystem des NCK fügt der WRITE-Befehl implizit ein "LF"-Zeichen (LINE FEED = neue Zeile) an das Ende des Ausgabestrings an.

Für die Ausgabe auf ein externes Gerät/Datei gilt dieses Verhalten nicht. Soll ein "LF" mit ausgegeben werden, muss das explizit im Ausgabestring mit angegeben werden.

→ Siehe hierzu Beispiel 3: Implizites/explizites "LF"!

Randbedingungen

- **Maximale Dateigröße (→ Maschinenhersteller!)**

Die maximal mögliche Dateigröße von Protokolldateien im passiven Filesystem wird eingestellt mit dem Maschinendatum:

```
MD11420 $MN_LEN_PROTOCOL_FILE
```

Die maximale Dateigröße gilt für alle Dateien, die mit dem WRITE-Befehl im passiven Filesystem angelegt werden. Bei Überschreitung wird eine Fehlermeldung ausgegeben und der Satz bzw. die Daten werden nicht abgespeichert. Sofern der Speicher ausreicht, kann eine neue Datei angelegt werden.

Beispiele

Beispiel 1: WRITE-Befehl ins passive Filesystem ohne absolute Pfadangabe

Programmcode	Kommentar
N10 DEF INT ERROR	; Definition der Fehlervariablen.
N20 WRITE(ERROR, "PROT", "PROTOKOLL VOM 7.2.97")	; Schreibe den Text "PROTOKOLL VOM 7.2.97" in die Datei _N_PROT_MPF.
N30 IF ERROR	; Fehlerauswertung.
N40 MSG ("Fehler bei WRITE-Befehl:" << ERROR)	
N50 M0	
N60 ENDIF	
...	

Beispiel 2: WRITE-Befehl ins passive Filesystem mit absoluter Pfadangabe

Programmcode

```
...  
WRITE(ERROR, "/_N_WKS_DIR/_N_PROT_WPD/_N_PROT_MPF", "PROTOKOLL VOM 7.2.97")  
...
```

Beispiel 3: Implizites/explizites "LF"

a, Schreiben ins passive Dateisystem mit implizit erzeugtem "LF"

Programmcode

```
...  
N110 DEF INT ERROR  
N120 WRITE(ERROR, "/_N_MPF_DIR/_N_MYPROTFILE_MPF", "MY_STRING")  
N130 WRITE(ERROR, "/_N_MPF_DIR/_N_MYPROTFILE_MPF", "MY_STRING")  
N140 M30
```

Ausgabeergebnis:

MY_STRING

MY_STRING

b, Schreiben in externe Datei ohne implizit erzeugtem "LF"

Programmcode

```
...  
N200 DEF STRING[30] DEV_1  
N210 DEF INT ERROR  
N220 DEV_1="LOCAL_DRIVE/myprotfile.mpf"  
N230 EXTOPEN(ERROR,DEV_1)  
N240 WRITE(ERROR,DEV_1,"MY_STRING")  
N250 WRITE(ERROR,DEV_1,"MY_STRING")  
N260 EXTCLOSE(ERROR,DEV_1)  
N270 M30
```

Ausgabeergebnis:

MY_STRINGMY_STRING

c, Schreiben in externe Datei mit explizit programmiertem "LF"

Um dasselbe Ergebnis wie bei a, zu erzielen, muss Folgendes programmiert werden:

Programmcode

```
...  
N200 DEF STRING[30] DEV_1  
N210 DEF INT ERROR  
N220 DEV_1="LOCAL_DRIVE/myprotfile.mpf"  
N230 EXTOPEN(ERROR,DEV_1)  
N240 WRITE(ERROR,DEV_1,"MY_STRING'H0A'")  
N250 WRITE(ERROR,DEV_1,"MY_STRING'H0A'")  
N260 EXTCLOSE(ERROR,DEV_1)  
N270 M30
```

Ausgabeergebnis:

MY_STRING

MY_STRING

Siehe auch

Process DataShare - Ausgabe auf ein externes Gerät/Datei (EXTOPEN, WRITE, EXTCLOSE) (Seite 623)

1.19 Datei löschen (DELETE)

Funktion

Mit dem `DELETE`-Befehl können alle Dateien gelöscht werden, egal, ob diese per `WRITE`-Befehl entstanden sind oder nicht. Auch Dateien, die unter höherer Zugriffsstufe erstellt wurden, können mit `DELETE` gelöscht werden.

Syntax

```
DEF INT <Fehler>  
DELETE (<Fehler>, "<Dateiname>")
```

Bedeutung

<code>DELETE:</code>	Befehl zum Löschen der angegebenen Datei
<code><Fehler>:</code>	Variable für die Rückgabe des Fehlerwerts
Typ.	INT
Wert:	0 kein Fehler
	1 Pfad nicht erlaubt
	2 Pfad nicht gefunden
	3 Datei nicht gefunden
	4 falscher Dateityp
	11 Datei wird benutzt
	12 keine Ressourcen frei
	20 sonstiger Fehler

<Dateiname>:

Name der zu löschenden Datei

Typ: STRING

Bei der Angabe des Dateinamens sind folgende Punkte zu beachten:

- Der angegebene Dateiname darf keine Leer- oder Steuerzeichen (Zeichen mit ASCII-Code ≤ 32) enthalten, da sonst der `DELETE`-Befehl mit Fehlerkennung 1 "Pfad nicht erlaubt" abgebrochen wird.
- Der Dateiname kann mit Pfadangabe und Datei-Kennung angegeben werden:

- Pfadangaben

Pfadangaben müssen absolut sein, d. h. sie beginnen mit `"/`.

Ohne Pfadangabe wird die Datei im aktuellen Verzeichnis (= Verzeichnis des angewählten Programms) gesucht.

- Datei-Kennung

Enthält der Dateiname keine Domain-Kennung (`"_N_"`), wird er entsprechend ergänzt.

Enthält der Dateinamen als viertletztes Zeichen einen Unterstrich `"_"`, so werden die nachfolgenden drei Zeichen als Datei-Kennung interpretiert. Um bei allen Datei-Befehlen denselben Dateinamen verwenden zu können, z. B. über eine Variable vom Typ `STRING`, dürfen nur die Datei-Kennungen `_SPF` und `_MPF` verwendet werden.

Ist keine Kennung `"_MPF"` oder `"_SPF"` angegeben, wird automatisch `_MPF` ergänzt.

- Die Länge des Dateinamens darf maximal 32 Bytes, die Länge der Pfadangabe maximal 128 Bytes betragen.

Beispiel:

```
"PROTFILE"
"_N_PROTFILE"
"_N_PROTFILE_MPF"
"/_N_MPF_DIR/_N_PROTFILE_MPF/"
```

Beispiel

Programmcode	Kommentar
N10 DEF INT ERROR	; Definition der Fehlervariablen.
N15 STOPRE	; Vorlaufstopp.
N20 DELETE (ERROR, "/_N_SPF_DIR/_N_TEST1_SPF")	; Lösche die Datei TEST1 im Unterprogrammverzeichnis.
N30 IF ERROR	; Fehlerauswertung.
N40 MSG("Fehler bei DELETE-Befehl:" <<ERROR)	
N50 M0	
N60 ENDIF	

1.20 Zeilen in Datei lesen (READ)

Funktion

Der `READ`-Befehl liest in der angegebenen Datei eine oder mehrere Zeilen und legt die gelesenen Informationen in einem Feld vom Typ `STRING` ab. Jede gelesene Zeile belegt in diesem Feld ein Feldelement.

Hinweis

Die Datei muss sich im statischen Anwenderspeicher des NCK (Passives Filesystem) befinden.

Voraussetzung

Die aktuell eingestellte Schutzstufe muss gleich oder größer dem `READ`-Recht der Datei sein. Ist dies nicht der Fall, wird der Zugriff mit Fehlermeldung (Rückgabewert der Fehlervariablen = 13) abgelehnt.

Syntax

```
DEF INT <Fehler>  
DEF STRING [<Stringlänge>] <Ergebnis> [<n>, <m>]  
READ (<Fehler>, "<Dateiname>", <Anfangszeile>, <Zeilenanzahl>, <Ergebnis>)
```

Bedeutung

<code>READ:</code>	Befehl zum Lesen von Zeilen der angegebenen Datei und zur Ablage dieser Zeilen in einem Variablenfeld.	
<code><Fehler>:</code>	Variable für die Rückgabe des Fehlerwerts (Call-By-Reference-Parameter)	
Typ.	INT	
Wert:	0	kein Fehler
	1	Pfad nicht erlaubt
	2	Pfad nicht gefunden
	3	Datei nicht gefunden
	4	falscher Dateityp
	13	Zugriffsrechte nicht ausreichend
	21	Zeile nicht vorhanden (Parameter <code><Anfangszeile></code> oder <code><Zeilenanzahl></code> größer als Anzahl der Zeilen in der angegebenen Datei).
	22	Feldlänge der Ergebnisvariablen (<code><Ergebnis></code>) ist zu klein.
	23	Zeilenbereich zu groß (Parameter <code><zeilenanzahl></code> so groß gewählt, dass über das Dateiende hinausgelesen wird).

<Dateiname>:	Name der zu lesenden Datei (Call-By-Value-Parameter) Typ: STRING Bei der Angabe des Dateinamens sind folgende Punkte zu beachten: <ul style="list-style-type: none">• Der angegebene Dateiname darf keine Leer- oder Steuerzeichen (Zeichen mit ASCII-Code ≤ 32) enthalten, da sonst der READ-Befehl mit Fehlerkennung 1 "Pfad nicht erlaubt" abgebrochen wird.• Der Dateiname kann mit Pfadangabe und Datei-Kennung angegeben werden:<ul style="list-style-type: none">– Pfadangaben Pfadangaben müssen absolut sein, d. h. sie beginnen mit "/". Ohne Pfadangabe wird die Datei im aktuellen Verzeichnis (= Verzeichnis des angewählten Programms) gesucht.– Datei-Kennung Enthält der Dateiname keine Domain-Kennung ("_N_"), wird sie entsprechend ergänzt. Enthält der Dateinamen als viertletztes Zeichen einen Unterstrich "_", so werden die nachfolgenden drei Zeichen als Datei-Kennung interpretiert. Um bei allen Datei-Befehlen denselben Dateinamen verwenden zu können, z. B. über eine Variable vom Typ STRING, dürfen nur die Datei-Kennungen _SPF und _MPF verwendet werden. Ist keine Kennung "_MPF" oder "_SPF" angegeben, wird automatisch _MPF ergänzt.• Die Länge des Dateinamens darf maximal 32 Bytes, die Länge der Pfadangabe maximal 128 Bytes betragen.
	Beispiel: "PROTFILE" "_N_PROTFILE" "_N_PROTFILE_MPF" "/_N_MPF_DIR/_N_PROTFILE_MPF/"
<Anfangszeile>:	Anfangszeile des zu lesenden Dateibereichs (Call-By-Value-Parameter) Typ: INT Wert: 0 Es werden die mit dem Parameter <Zeilenanzahl> angegebene Anzahl an Zeilen vor dem Dateiende gelesen. 1 ... n Nummer der ersten zu lesenden Zeile.
<Zeilenanzahl>:	Anzahl der zu lesenden Zeilen (Call-By-Value-Parameter) Typ: INT

<Ergebnis>: Ergebnisvariable (Call-By-Reference-Parameter)
 Variablenfeld, in dem der gelesene Text abgelegt wird.
 Typ: STRING (max. Länge: 255)
 Wenn im Parameter <Zeilenanzahl> weniger Zeilen angegeben sind als die Feldgröße [<n>, <m>] der Ergebnisvariablen beträgt, dann werden die restlichen Feldelemente nicht verändert.
 Der Abschluss einer Zeile durch die Steuerzeichen "LF" (Line Feed) oder "CR LF" (Carriage Return Line Feed) wird **nicht** in der Ergebnisvariablen abgelegt.
 Gelesene Zeilen werden abgeschnitten, wenn die Zeile länger ist als die definierte Stringlänge. Es erfolgt keine Fehlermeldung.

Hinweis

Binäre Files können nicht eingelesen werden. Es wird der Fehler "falscher Dateityp" (Rückgabewert der Fehlervariablen = 4) ausgegeben. Folgenden Dateitypen sind nicht lesbar: _BIN, _EXE, _OBJ, _LIB, _BOT, _TRC, _ACC, _CYC, _NCK.

Beispiel

Programmcode	Kommentar
N10 DEF INT ERROR	; Definition der Fehlervariablen.
N20 DEF STRING[255] RESULT[5]	; Definition der Ergebnisvariablen.
N30 READ(ERROR, "/_N_CST_DIR/_N_TESTFILE_MPF", 1, 5, RESULT)	; Dateiname mit Domain-, Dateikennung und Pfadangabe.
N40 IF ERROR <>0	; Fehlerauswertung.
N50 MSG("FEHLER"<<ERROR<<"BEI READ-BEFEHL")	
N60 M0	
N70 ENDIF	
...	

1.21 Vorhandensein einer Datei prüfen (ISFILE)

Funktion

Mit dem `ISFILE`-Befehl kann geprüft werden, ob eine Datei im statischen Anwenderspeicher des NCK (passives Filesystem) existiert.

Syntax

```
<Ergebnis>=ISFILE("<Dateiname>")
```

Bedeutung

- `ISFILE`: Befehl zur Prüfung, ob die angegebene Datei im passiven Filesystem existiert.
- `<Dateiname>`: Name der Datei, deren Vorhandensein im passiven Filesystem geprüft werden soll.
- Typ: `STRING`
- Bei der Angabe des Dateinamens sind folgende Punkte zu beachten:
- Der angegebene Dateiname darf keine Leer- oder Steuerzeichen (Zeichen mit ASCII-Code ≤ 32) enthalten.
 - Der Dateiname kann mit Pfadangabe und Datei-Kennung angegeben werden:
 - Pfadangaben

Pfadangaben müssen absolut sein, d. h. sie beginnen mit `/`.

Ohne Pfadangabe wird die Datei im aktuellen Verzeichnis (= Verzeichnis des angewählten Programms) gesucht.
 - Datei-Kennung

Enthält der Dateiname keine Domain-Kennung ("`_N_`"), wird er entsprechend ergänzt.

Enthält der Dateinamen als viertletztes Zeichen einen Unterstrich "`_`", so werden die nachfolgenden drei Zeichen als Datei-Kennung interpretiert. Um bei allen Datei-Befehlen denselben Dateinamen verwenden zu können, z. B. über eine Variable vom Typ `STRING`, dürfen nur die Datei-Kennungen `_SPF` und `_MPF` verwendet werden.

Ist keine Kennung "`_MPF`" oder "`_SPF`" angegeben, wird automatisch `_MPF` ergänzt.
 - Die Länge des Dateinamens darf maximal 32 Bytes, die Länge der Pfadangabe maximal 128 Bytes betragen.

Beispiel:

```
"PROFILE"
"_N_PROFILE"
"_N_PROFILE_MPF"
"/_N_MPF_DIR/_N_PROFILE_MPF/"
```

<Ergebnis>: Ergebnisvariable zur Aufnahme des Prüfergebnisses
 Typ: BOOL
 Wert: TRUE Datei vorhanden
 FALSE Datei nicht vorhanden

Beispiel

Programmcode	Kommentar
N10 DEF BOOL RESULT	; Definition der Ergebnisvariablen.
N20 RESULT=ISFILE("TESTFILE")	
N30 IF (RESULT==FALSE)	
N40 MSG("DATEI NICHT VORHANDEN")	
N50 M0	
N60 ENDIF	
...	

oder:

Programmcode	Kommentar
N10 DEF BOOL RESULT	; Definition der Ergebnisvariablen.
N20 RESULT=ISFILE("TESTFILE")	
N30 IF (NOT ISFILE("TESTFILE"))	
N40 MSG("DATEI NICHT VORHANDEN")	
N50 M0	
N60 ENDIF	
...	

1.22 Datei-Informationen auslesen (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO)

Funktion

Über die Befehle `FILEDATE`, `FILETIME`, `FILESIZE`, `FILESTAT` und `FILEINFO` können bestimmte Datei-Informationen wie Datum / Uhrzeit des letzten schreibenden Zugriffs, aktuelle Dateigröße, Datei-Status oder die Summe dieser Informationen ausgelesen werden.

Hinweis

Die Datei muss sich im statischen Anwenderspeicher des NCK (Passives Filesystem) befinden.

Voraussetzung

Die aktuell eingestellte Schutzstufe muss gleich oder größer dem Show-Recht des übergeordneten Verzeichnisses sein. Ist dies nicht der Fall, wird der Zugriff mit Fehlermeldung (Rückgabewert der Fehlervariablen = 13) abgelehnt.

Syntax

```
FILE... (<Fehler>, "<Dateiname>", <Ergebnis>)
```

Bedeutung

<code>FILEDATE:</code>	Liefert das Datum des letzten Schreibzugriffs auf eine Datei
<code>FILETIME:</code>	Liefert die Uhrzeit des letzten Schreibzugriffs auf eine Datei
<code>FILESIZE:</code>	Liefert die aktuelle Größe einer Datei
<code>FILESTAT:</code>	Liefert für den Status einer Datei bezüglich folgender Rechte : <ul style="list-style-type: none"> • Lesen (r: read) • Schreiben (w: write) • Ausführen (x: execute) • Anzeigen (s: show) • Löschen (d: delete)
<code>FILEINFO:</code>	Liefert für eine Datei die Summe der Informationen , die über <code>FILEDATE</code> , <code>FILETIME</code> , <code>FILESIZE</code> und <code>FILESTAT</code> auslesbar sind

<Fehler>: Variable für die Rückgabe des Fehlerwerts (Call-By-Reference-Parameter)

Typ: VAR INT

Wert: 0 kein Fehler
 1 Pfad nicht erlaubt
 2 Pfad nicht gefunden
 3 Datei nicht gefunden
 4 falscher Dateityp
 13 Zugriffsrechte nicht ausreichend
 22 Stringlänge der Ergebnisvariablen (<Ergebnis>) ist zu klein.

<Dateiname>: Name der Datei, von der Datei-Information(en) ausgelesen werden soll(en)

Typ: CHAR[160]

Bei der Angabe des Dateinamens sind folgende Punkte zu beachten:

- Der angegebene Dateiname darf keine Leer- oder Steuerzeichen (Zeichen mit ASCII-Code ≤ 32) enthalten, da sonst der FILE...-Befehl mit Fehlerkennung 1 "Pfad nicht erlaubt" abgebrochen wird.
- Der Dateiname kann mit Pfadangabe und Datei-Kennung angegeben werden:
 - Pfadangaben

Pfadangaben müssen absolut sein, d. h. sie beginnen mit "/".

Ohne Pfadangabe wird die Datei im aktuellen Verzeichnis (= Verzeichnis des angewählten Programms) gesucht.

- Datei-Kennung

Enthält der Dateiname keine Domain-Kennung ("_N_"), wird er entsprechend ergänzt.

Enthält der Dateinamen als viertletztes Zeichen einen Unterstrich "_", so werden die nachfolgenden drei Zeichen als Datei-Kennung interpretiert. Um bei allen Datei-Befehlen denselben Dateinamen verwenden zu können, z. B. über eine Variable vom Typ STRING, dürfen nur die Datei-Kennungen _SPF und _MPF verwendet werden.

Ist keine Kennung "_MPF" oder "_SPF" angegeben, wird automatisch _MPF ergänzt.

- Die Länge des Dateinamens darf maximal 32 Bytes, die Länge der Pfadangabe maximal 128 Bytes betragen.

Beispiel:

"PROTFILE"

"_N_PROTFILE"

"_N_PROTFILE_MPF"

"/_N_MPF_DIR/_N_PROTFILE_MPF/"

<Ergebnis>:	Ergebnisvariable (Call-By-Reference-Parameter)		
	Variable, in der die angeforderte Datei-Information abgelegt wird.		
Typ:	VAR CHAR[8]	bei	FILEDATE Format: "dd.mm.yy"
	VAR CHAR[8]	bei	FILETIME Format: "hh:mm:ss"
	VAR INT	bei	FILESIZE Die Dateigröße wird in Byte ausgegeben.
	VAR CHAR[5]	bei	FILESTAT Format: "rwxsd" (r: read, w: write, x: execute, s: show, d: delete)
	VAR CHAR[32]	bei	FILEINFO Format: "rwxsd nnnnnnnn dd.mm.yy hh:mm:ss"

Beispiel

Programmcode	Kommentar
N10 DEF INT ERROR	; Definition der Fehlervariablen.
N20 STRING[32] RESULT	; Definition der Ergebnisvariablen.
N30 FILEINFO(ERROR, "/_N_MPF_DIR/_N_TESTFILE_MPF", RESULT)	; Dateiname mit Domain-, Dateikennung und Pfadangabe.
N40 IF ERROR <>0	; Fehlerauswertung
N50 MSG("FEHLER"<<ERROR<<"BEI FILEINFO-BEFEHL")	
N60 M0	
N70 ENDIF	
...	

Das Beispiel könnte in der Ergebnisvariablen RESULT z. B. folgendes Ergebnis liefern:

"77777 12345678 26.05.00 13:51:30"

1.23 Checksummenberechnung über ein Feld (CHECKSUM)

Funktion

Mit dem Befehl `CHECKSUM` kann die Checksumme über ein Feld berechnet werden. Durch den Vergleich dieser Checksumme mit dem Ergebnis einer früheren Checksummenberechnung kann festgestellt werden, ob sich die Daten des Feldes verändert haben.

Anwendung

Prüfung, ob sich beim Abspannen die Eingangskontur geändert hat.

Syntax

```
DEF INT <Fehler>
DEF STRING [<Stringlänge>] <Checksumme>
DEF ... <Feld> [<n>, <m>, <o>]
<Fehler>=CHECKSUM(<Checksumme>, "<Feld>" [, <Anfangsspalte>, <Endspalte>])
```

Bedeutung

<code>CHECKSUM:</code>	Befehl zur Berechnung der Checksumme über ein Feld
<code><Fehler>:</code>	Variable für die Rückgabe des Fehlerwerts
	Typ: INT
	Wert: 0 kein Fehler
	1 Symbol nicht gefunden
	2 kein Feld
	3 Index 1 zu groß
	4 Index 2 zu groß
	5 ungültiger Datentyp
	10 Überlauf der Checksumme
<code><Checksumme>:</code>	Ergebnisvariable zur Aufnahme des Ergebnisses der Checksummenberechnung (Call-By-Reference-Parameter)
	Typ: STRING
	Erforderliche Stringlänge: 16
	Die Checksumme wird als Zeichenkette von 16 Hex-Ziffern dargestellt. Es werden aber keine Formatzeichen mit angegeben.
	Beispiel: "A6FC3404E534047C"

<Feld>:	Name des Feldes, über das die Checksumme gebildet werden soll (Call-By-Value-Parameter) Typ: STRING Max. Stringlänge: 32 Zulässige Felder sind 1- bis 3-dimensionale Felder der Typen: BOOL, CHAR, INT, REAL, STRING Hinweis: Felder von Maschinendaten sind nicht zulässig.
<Anfangsspalte>:	Nummer der Anfangsspalte des Feldes für die Berechnung der Checksumme (optionaler Parameter)
<Endspalte>:	Nummer der Endspalte des Feldes für die Berechnung der Checksumme (optionaler Parameter)

Hinweis

Die Parameter <Anfangsspalte> und <Endspalte> sind optional. Werden keine Spaltenindizes angegeben, so wird die Checksumme über das komplette Feld gebildet.

Das Ergebnis der Checksumme ist immer eindeutig. Bei Änderungen eines Feldelements ergibt sich auch ein anderer Ergebnisstring.

Beispiel

Programmcode	Kommentar
N10 DEF INT ERROR	; Definition der Fehlervariablen.
N20 DEF STRING[16] MY_CHECKSUM	; Definition der Ergebnisvariablen.
N30 DEF INT MY_VAR[4,4]	; Felddefinition.
N40 MY_VAR=...	
N50 ERROR=CHECKSUM(MY_CHECKSUM, "MY_VAR", 0, 2)	
...	

Das Beispiel könnte in der Ergebnisvariablen MY_CHECKSUM z. B. folgendes Ergebnis liefern:

"A6FC3404E534047C"

1.24 Aufrunden (ROUNDUP)

Funktion

Mit der Funktion "ROUNDUP" können Eingabewerte vom Typ REAL (gebrochene Zahlen mit Dezimalpunkt) auf die nächste größere ganze Zahl aufgerundet werden.

Syntax

ROUNDUP (<Wert>)

Bedeutung

ROUNDUP: Befehl zum Aufrunden eines Eingabewerts
<Wert>: Eingabewert vom Typ REAL

Hinweis

Eingabewerte vom Typ INTEGER (eine ganze Zahl) werden unverändert zurückgeliefert.

Beispiele

Beispiel 1: Verschiedene Eingabewerte und deren Rundungsergebnisse

Beispiel	Rundungsergebnis
ROUNDUP (3 . 1)	4.0
ROUNDUP (3 . 6)	4.0
ROUNDUP (-3 . 1)	-3.0
ROUNDUP (-3 . 6)	-3.0
ROUNDUP (3 . 0)	3.0
ROUNDUP (3)	3.0

Beispiel 2: ROUNDUP im NC-Programm

Programmcode

```
N10 X=ROUNDUP (3 . 5) Y=ROUNDUP (R2+2)
N15 R2=ROUNDUP ($AA_IM [Y])
N20 WHEN X=100 DO Y=ROUNDUP ($AA_IM [X])
...
```


1.25 Unterprogrammtechnik

1.25.1 Allgemeines

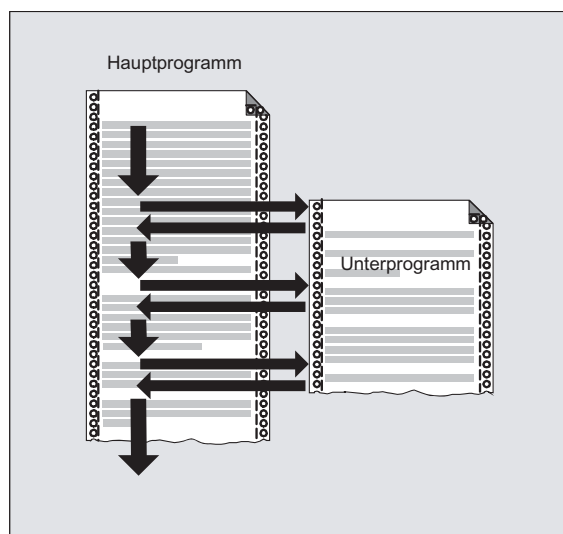
1.25.1.1 Unterprogramm

Funktion

Die Bezeichnung "Unterprogramm" stammt noch aus der Zeit, als Teileprogramme fest in Haupt- und Unterprogramme unterteilt waren. Hauptprogramme waren dabei die Teileprogramme, die an der Steuerung zum Abarbeiten angewählt und dann gestartet wurden. Unterprogramme waren die Teileprogramme, die vom Hauptprogramm aus aufgerufen wurden.

Diese feste Einteilung besteht mit der heutigen SINUMERIK NC-Sprache nicht mehr. Jedes Teileprogramm kann prinzipiell als Hauptprogramm angewählt und gestartet oder als Unterprogramm von einem anderen Teileprogramm aus aufgerufen werden.

Somit wird im weiteren Verlauf mit Unterprogramm ein Teileprogramm bezeichnet, das von einem anderen Teileprogramm aus aufgerufen wird.



Anwendung

Wie in allen höheren Programmiersprachen werden auch in der NC-Sprache Unterprogramme dazu angewandt, um Programmteile, die mehrfach verwendet werden, in eigenständige, in sich abgeschlossene Programme auszulagern.

Unterprogrammen bieten folgende Vorteile:

- Erhöhen die Übersichtlichkeit und Lesbarkeit der Programme
- Erhöhen die Qualität durch Wiederverwendung getesteter Programmteile
- Bieten die Möglichkeit zur Schaffung spezifischer Bearbeitungsbibliotheken
- Sparen Speicherplatz

1.25.1.2 Unterprogrammnamen

Regeln zur Benennung

Bei der Namensgebung von Unterprogrammen sind folgenden Regeln zu beachten:

- Die ersten beiden Zeichen müssen Buchstaben (A - Z, a - z) sein.
- Die folgenden Zeichen können in beliebiger Kombination Buchstaben, Ziffern (0 - 9) und Unterstrich ("_") sein.
- Es dürfen maximal 31 Zeichen verwendet werden.

Hinweis

In der SINUMERIK NC-Sprache wird **nicht** zwischen Groß- und Kleinschreibung unterschieden.

Erweiterungen des Programmnamens

Der bei der Programmerstellung vergebene Programmname wird steuerungsintern mit einem Pre- und Postfix erweitert:

- Prefix: `_N_`
- Postfix:
 - Hauptprogramme: `_MPF`
 - Unterprogramme: `_SPF`

Verwendung des Programmnamens

Bei der Verwendung des Programmnamens, z. B. bei einem Unterprogrammaufruf, sind alle Kombinationen von Prefix, Programmnamen und Postfix möglich.

Beispiel:

Das Unterprogramm mit dem Programmnamen "SUB_PROG" kann über folgende Aufrufe gestartet werden:

1. `SUB_PROG`
2. `_N_SUB_PROG`
3. `SUB_PROG_SPF`
4. `_N_SUB_PROG_SPF`

Hinweis

Namensgleichheit von Haupt- und Unterprogrammen

Existieren Hauptprogramme (.MPF) und Unterprogramme (.SPF) mit gleichem Programmnamen, muss bei Verwendung des Programmnamens im Teileprogramm der jeweilige Postfix angegeben werden, um das Programm eindeutig zu kennzeichnen.

1.25.1.3 Schachtelung von Unterprogrammen

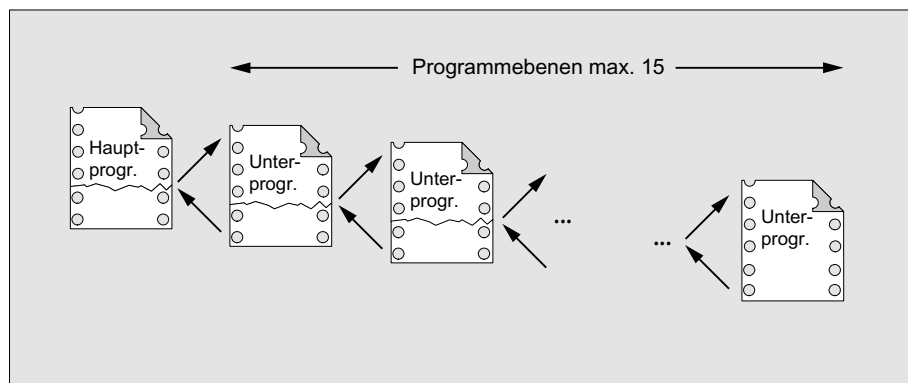
Ein Hauptprogramm kann Unterprogramme aufrufen, die wiederum Unterprogramme aufrufen. Die Abläufe der Programme sind somit ineinander geschachtelt. Jedes Programm läuft dabei in einer eigenen Programmebene.

Schachtelungstiefe

Die NC-Sprache stellt aktuell 16 Programmebenen zur Verfügung. Das Hauptprogramm läuft immer in der obersten Programmebene 0. Ein Unterprogramm läuft immer in der dem Aufruf folgenden nächstniedrigeren Programmebene. Die Programmebene 1 ist somit die erste Unterprogrammebene.

Unterteilung der Programmebenen:

- Programmebene 0: Hauptprogrammebene
- Programmebene 1 - 15: Unterprogrammebene 1 - 15



Interruptroutinen (ASUP)

Wird im Rahmen einer Interruptroutine ein Unterprogramm aufgerufen, wird dieses nicht in der aktuellen im Kanal aktiven Programmebene (n), sondern ebenfalls in der nächstniedrigeren Programmebene (n+1) abgearbeitet. Damit dies auch in der untersten Programmebene noch möglich ist, stehen im Zusammenhang mit Interruptroutinen 2 zusätzliche Programmebenen (16 und 17) zur Verfügung.

Werden mehr als 2 Programmebenen benötigt, muss dies explizit in der Strukturierung des im Kanal abgearbeiteten Teileprogramms berücksichtigt werden. D. h. es darf dann maximal nur so viele Programmebenen beanspruchen, dass noch ausreichend Programmebenen für die Interruptbearbeitung zur Verfügung stehen.

Benötigt die Interruptbearbeitung z. B. 4 Programmebenen, muss das Teileprogramm so strukturiert werden, dass es maximal Programmebene 13 belegt. Erfolgt dann ein Interrupt, stehen diesem die benötigten 4 Programmebenen (14 bis 17) zur Verfügung.

Siemens-Zyklen

Siemens-Zyklen benötigen 3 Programmebenen. Der Aufruf eines Siemens-Zyklus muss daher spätestens erfolgen in:

- Teileprogrammabarbeitung: Programmebene 12
- Interruptroutine: Programmebene 14

1.25.1.4 Suchpfad

Beim Aufruf eines Unterprogramms ohne Pfadangabe sucht die Steuerung in der angegebenen Reihenfolge in folgenden Verzeichnissen:

Reihenfolge	Verzeichnis	Beschreibung
1.	Aktuelles Verzeichnis	Verzeichnis des aufrufenden Programms
2.	/_N_SPF_DIR /	Globales Unterprogrammverzeichnis
3.	/_N_CUS_DIR /	Anwender-Zyklen
4.	/_N_CMA_DIR /	Hersteller-Zyklen
5.	/_N_CST_DIR /	Standard-Zyklen

1.25.1.5 Formal- und Aktualparameter

Von Formal- und Aktualparameter spricht man im Zusammenhang mit der Definition und dem Aufruf von Unterprogrammen mit Parameterübergabe.

Formalparameter

Bei der Definition eines Unterprogramms müssen die dem Unterprogramm zu übergebenden Parameter, die sogenannten Formalparameter, mit Typ und Parameternamen definiert werden.

Die Formalparameter definieren somit die Schnittstelle des Unterprogramms.

Beispiel:

Programmcode	Kommentar
PROC KONTUR (REAL X, REAL Y)	; Formalparameter: X und Y beide vom Typ REAL
N20 X1=X Y1=Y	; Verfahren der Achse X1 auf Position X und der Achse Y1 auf Position Y
...	
N100 RET	

Aktualparameter

Beim Aufruf eines Unterprogramms müssen dem Unterprogramm absolute Werte oder Variablen, die sogenannten Aktualparameter, übergeben werden.

Die Aktualparameter befüllen somit beim Aufruf die Schnittstelle des Unterprogramms mit aktuellen Werten.

Beispiel:

Programmcode	Kommentar
N10 DEF REAL BREITE	; Variablendefinition
N20 BREITE=20.0	; Variablenzuweisung
N30 KONTUR(5.5, BREITE)	; Unterprogrammaufruf mit Aktualparametern: 5.5 und BREITE
...	
N100 M30	

1.25.1.6 Parameterübergabe

Definition eines Unterprogramms mit Parameterübergabe

Die Definition eines Unterprogramms mit Parameterübergabe erfolgt mit dem Schlüsselwort `PROC` und einer vollständigen Auflistung aller vom Unterprogramm erwarteten Parameter.

Unvollständige Parameterübergabe

Beim Aufruf des Unterprogramms müssen nicht immer alle in der Unterprogrammchnittstelle definierten Parameter explizit übergeben werden. Wird ein Parameter weggelassen, wird für diesen Parameter der Standardwert "0" übergeben.

Zur eindeutigen Kennzeichnung der Reihenfolge der Parameter müssen allerdings die Kommas als Trennzeichen der Parameter immer mit angegeben werden. Eine Ausnahme bildet der letzte Parameter. Wird dieser beim Aufruf weggelassen, kann auch das letzte Komma entfallen.

Beispiel:

Unterprogramm:

Programmcode	Kommentar
<code>PROC SUB_PROG (REAL X, REAL Y, REAL Z)</code>	<code>; Formalparameter: X, Y und Z</code>
<code>...</code>	
<code>N100 RET</code>	

Hauptprogramm:

Programmcode	Kommentar
<code>PROC MAIN_PROG</code>	
<code>...</code>	
<code>N30 SUB_PROG(1.0,2.0,3.0)</code>	<code>; Unterprogrammaufruf mit vollständiger Parameterübergabe: X=1.0, Y=2.0, Z=3.0</code>
<code>...</code>	
<code>N100 M30</code>	

Beispiele für den Unterprogrammaufruf in `N30` mit unvollständiger Parameterübergabe:

```
N30 SUB_PROG( ,2.0,3.0) ; X=0.0, Y=2.0, Z=3.0
N30 SUB_PROG(1.0, ,3.0) ; X=1.0, Y=0.0, Z=3.0
N30 SUB_PROG(1.0,2.0) ; X=1.0, Y=2.0, Z=0.0
N30 SUB_PROG( , ,3.0) ; X=0.0, Y=0.0, Z=3.0
N30 SUB_PROG( , , ) ; X=0.0, Y=0.0, Z=0.0
```

VORSICHT
Parameterübergabe Call-by-Reference
Parameter, die über Call-by-Reference übergeben werden, dürfen beim Unterprogramm-Aufruf nicht weggelassen werden.
VORSICHT
Datentyp AXIS
Parameter vom Datentyp AXIS dürfen beim Unterprogramm-Aufruf nicht weggelassen werden.

Überprüfung der Übergabeparameter

Über die Systemvariable \$P_SUBPAR [n] mit n = 1, 2, ... kann im Unterprogramm überprüft werden, ob ein Parameter explizit übergeben oder weggelassen wurde. Der Index n bezieht sich auf die Reihenfolge der Formalparameter. Index n = 1 bezieht sich auf den 1. Formalparameter, Index n = 2 auf den 2. Formalparameter usw.

Der folgende Programmausschnitt zeigt beispielhaft für den 1. Formalparameter, wie eine Überprüfung realisiert werden kann:

Programmierung	Kommentar
PROC SUB_PROG (REAL X, REAL Y, REAL Z)	; Formalparameter: X, Y und Z
N20 IF \$P_SUBPAR[1]==TRUE	; Überprüfung des 1.Formalparameters X.
...	; Diese Aktionen werden ausgeführt, wenn der Formalparameter X explizit übergeben wurde.
N40 ELSE	
...	; Diese Aktionen werden ausgeführt, wenn der Formalparameter X nicht übergeben wurde.
N60 ENDIF	
...	; Allgemeine Aktionen
N100 RET	

1.25.2 Definition eines Unterprogramms

1.25.2.1 Unterprogramm ohne Parameterübergabe

Funktion

Bei der Definition von Unterprogrammen ohne Parameterübergabe kann die Definitionszeile am Programmanfang entfallen.

Syntax

```
[PROC <Programmname>]  
...
```

Bedeutung

PROC:	Definitionsanweisung am Anfang eines Programms
<Programmname>:	Name des Programms

Beispiel

Beispiel 1: Unterprogramm mit PROC-Anweisung

Programmcode	Kommentar
PROC SUB_PROG	; Definitionszeile
N10 G01 G90 G64 F1000	
N20 X10 Y20	
...	
N100 RET	; Unterprogrammrücksprung

Beispiel 2: Unterprogramm ohne PROC-Anweisung

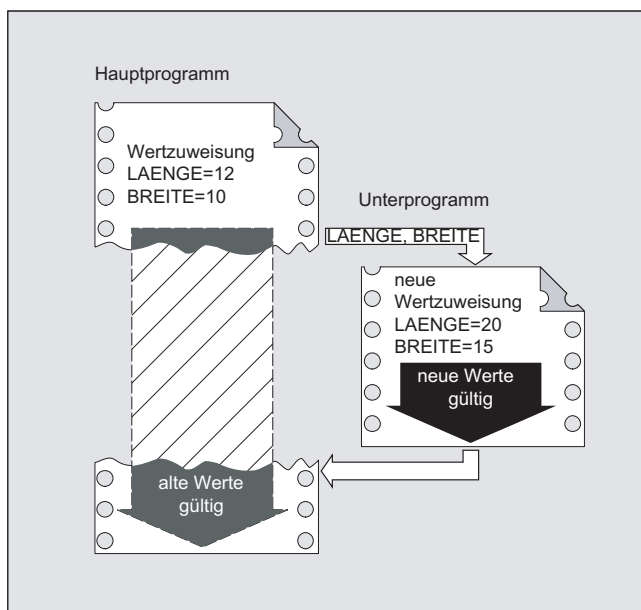
Programmcode	Kommentar
N10 G01 G90 G64 F1000	
N20 X10 Y20	
...	
N100 RET	; Unterprogrammrücksprung

1.25.2.2 Unterprogramm mit Parameterübergabe Call-by-Value (PROC)

Funktion

Die Definition eines Unterprogramms mit Parameterübergabe Call-by-Value erfolgt mit dem Schlüsselwort `PROC`, gefolgt vom Programmnamen und einer vollständigen Auflistung aller vom Unterprogramm erwarteten Parameter mit Typ und Namen. Die Definitionsanweisung muss in der ersten Programmzeile stehen.

Die Parameterübergabe Call-by-Value hat keine Rückwirkungen auf das aufrufende Programm. Das aufrufende Programm übergibt dem Unterprogramm nur die Werte der Aktualparameter.



Hinweis

Es können maximal 127 Parameter übergeben werden.

Syntax

```
PROC <Prog_Name> (<Par_Typ> <Par_1>[=<Init_Wert_1>] {, Par_2 [=<Init_Wert_1>] })
```


Bedeutung

PROC:	Definitionsanweisung am Anfang eines Programms
<Prog_Name>:	Name des Programms
<Par_Typ>:	Datentyp des Parameters (z. B. REAL, INT, BOOL)
<Par_n>:	Name des Parameters
<Init_Wert>:	Optionaler Wert zur Initialisierung des Parameters Wird beim Aufruf des Unterprogramms kein Parameter angegeben, erhält der Parameter den Initialisierungswert zugewiesen.

Beispiel

Definition eines Unterprogramms SUB_PROG mit drei Parametern vom Typ REAL mit Defaultwerten:

Programmcode

```
PROC SUB_PROG (REAL LENGTH=10.0, REAL WIDTH=20.0, REAL HIGHT=30.0)
...
N100 RET
```

Verschieden Aufrufvarianten:

Programmcode

```
PROC MAIN_PROG
REAL PAR_1 = 100
REAL PAR_2 = 200
REAL PAR_3 = 300
; Aufrufvarianten
SUB_PROG
SUB_PROG (PAR_1, PAR_2, PAR_3)
SUB_PROG (PAR_1)
SUB_PROG (PAR_1, , PAR_3)
SUB_PROG ( , , PAR_3)
N100 RET
```

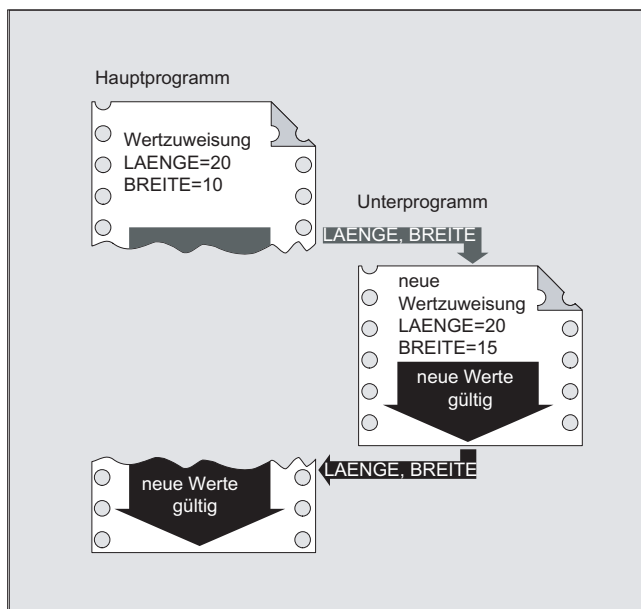
1.25.2.3 Unterprogramm mit Parameterübergabe Call-by-Reference (PROC, VAR)

Funktion

Die Definition eines Unterprogramms mit Parameterübergabe Call-by-Reference erfolgt mit dem Schlüsselwort `PROC`, gefolgt vom Programmnamen und einer vollständigen Auflistung aller vom Unterprogramm erwarteten Parameter mit Schlüsselwort `VAR`, Typ und Namen. Die Definitionsanweisung muss in der ersten Programmzeile stehen.

Bei der Parameterübergabe Call-by-Reference können auch Referenzen auf Felder übergeben werden.

Die Parameterübergabe Call-by-Reference hat Rückwirkungen auf das aufrufende Programm. Das aufrufende Programm übergibt dem Unterprogramm eine Referenz auf den Aktualparameter und ermöglicht dem Unterprogramm somit einen direkten Zugriff auf die entsprechende Variable.



Hinweis

Es können maximal 127 Parameter übergeben werden.

Hinweis

Eine Parameterübergabe Call-by-Reference ist nur dann erforderlich, wenn die übergebene Variable im aufrufenden Programm definiert wurde (LUD). Kanal-globale oder NC-globale Variablen müssen nicht übergeben werden, da auf diese auch direkt vom Unterprogramm aus zugegriffen werden kann.

Syntax

```
PROC <Programmname> (VAR <Parametertyp> <Parametername>, ...)  
PROC <Programmname> (VAR <Feldtyp> <Feldname> [<m>,<n>,<o>], ...)
```

Bedeutung

PROC:	Definitionsanweisung am Anfang eines Programms
VAR:	Schlüsselwort für die Parameterübergabe per Referenz
<Programmname>:	Name des Programms
<Parametertyp>:	Datentyp des Parameters (z. B. REAL, INT, BOOL)
<Parametername>:	Name des Parameters
<Feldtyp>:	Datentyp der Feldelemente (z. B. REAL, INT, BOOL)
<Feldname>:	Name des Feldes
<m>,<n>,<o>:	Feldgröße
	Aktuell sind maximal 3-dimensionale Felder möglich:
<m>:	Feldgröße für 1. Dimension
<n>:	Feldgröße für 2. Dimension
<o>:	Feldgröße für 3. Dimension

ACHTUNG

Der nach dem Schlüsselwort `PROC` angegebene Programmname muss mit dem an der Bedienoberfläche vergebenen Programmnamen übereinstimmen.

Hinweis

Mit Feldern unbestimmter Feldlänge als Formalparameter können Unterprogramme Felder variabler Länge bearbeiten. Dazu wird bei der Definition z. B. eines zweidimensionalen Feldes als Formalparameter die Länge der 1. Dimension nicht angegeben. Das Komma aber muss geschrieben werden.

Beispiel: `PROC <Programmname> (VAR REAL FELD[,5])`

Beispiel

Definition eines Unterprogramms mit 2 Parameter als Referenz auf Typ REAL:

Programmcode	Kommentar
PROC SUB_PROG (VAR REAL LAENGE, VAR REAL BREITE)	; Parameter 1: Referenz auf Typ: REAL, Name: LAENGE
...	Parameter 2: Referenz auf Typ: REAL, Name: BREITE
N100 RET	

1.25.2.4 Modale G-Funktionen sichern (SAVE)

Funktion

Das Attribut `SAVE` bewirkt, dass die vor dem Unterprogrammaufruf aktiven modalen G-Funktionen gesichert und nach dem Unterprogrammende wieder reaktiviert werden.

VORSICHT
Unterbrechung des Bahnsteuerbetriebs
Wird bei aktivem Bahnsteuerbetrieb ein Unterprogramme mit Attribut <code>SAVE</code> aufgerufen, wird der Bahnsteuerbetrieb am Ende des Unterprogramms (Rücksprung) unterbrochen.

Syntax

`PROC <Unterprogrammname> SAVE`

Bedeutung

`SAVE:` Sichern der modalen G-Funktionen vor dem Unterprogrammaufruf und Wiederherstellen nach Unterprogrammende

Beispiel

Im Unterprogramm KONTUR wirkt die modale G-Funktion `G91` (Kettenmaß). Im Hauptprogramm wirkt die modale G-Funktion `G90` (Absolutmaß). Durch die Unterprogrammdefinition mit `SAVE` wirkt nach dem Unterprogrammende im Hauptprogramm wieder `G90`.

Unterprogramm-Definition:

Programmcode	Kommentar
<code>PROC KONTUR (REAL WERT1) SAVE</code>	<code>; Unterprogramm-Definition mit Parameter SAVE</code>
<code>N10 G91 ...</code>	<code>; Modale G-Funktion G91: Kettenmaß</code>
<code>N100 M17</code>	<code>; Unterprogrammende</code>

Hauptprogramm:

Programmcode	Kommentar
<code>N10 G0 X... Y... G90</code>	<code>; Modale G-Funktion G90: Absolutmaß</code>
<code>N20 ...</code>	
<code>...</code>	
<code>N50 KONTUR (12.4)</code>	<code>; Unterprogrammaufruf</code>
<code>N60 X... Y...</code>	<code>; Modale G-Funktion G90 durch SAVE reaktiviert</code>

Randbedingungen

Frames

Das Verhalten von Frames bezüglich Unterprogrammen mit dem Attribut `SAVE` ist abhängig vom Typ des Frames und kann über Maschinendaten eingestellt werden.

Literatur

Funktionshandbuch Grundfunktionen; Achsen, Koordinatensysteme, Frames (K2), Kapitel: "Unterprogrammrückprung mit SAVE"

1.25.2.5 Einzelsatzbearbeitung unterdrücken (SBLOF, SBLON)

Funktion

Einzelsatzunterdrückung für das gesamte Programm

Mit `SBLOF` gekennzeichnete Programme werden bei aktiver Einzelsatzbearbeitung wie ein Satz komplett abgearbeitet, d. h. für das gesamte Programm wird die Einzelsatzbearbeitung unterdrückt.

`SBLOF` steht in der `PROC`-Zeile und gilt bis zum Ende oder Abbruch des Unterprogramms. Mit dem Rückprung-Befehl wird entschieden, ob am Ende des Unterprogramms angehalten wird oder nicht:

Rückprung mit <code>M17</code> :	Stopp am Ende des Unterprogramms
Rückprung mit <code>RET</code> :	Kein Stopp am Ende des Unterprogramms

Einzelsatzunterdrückung innerhalb des Programms

`SBLOF` muss allein im Satz stehen. Ab diesem Satz wird Einzelsatz ausgeschaltet bis:

- zum nächsten `SBLON`
oder
- zum Ende der aktiven Unterprogrammebene

Syntax

Einzelsatzunterdrückung für das gesamte Programm:

```
PROC ... SBLOF
```

Einzelsatzunterdrückung innerhalb des Programms:

```
| SBLOF  
| ...  
| SBLON
```

Bedeutung

PROC:	Erste Anweisung eines Programms
SBLOF:	Befehl zum Ausschalten der Einzelsatzbearbeitung SBLOF kann in einem PROC-Satz oder allein im Satz stehen.
SBLON:	Befehl zum Einschalten der Einzelsatzbearbeitung SBLON muss in einem eigenen Satz stehen.

Randbedingungen

- **Einzelsatzunterdrückung und Satzanzeige**

Die aktuelle Satzanzeige kann in Zyklen/Unterprogrammen mit `DISPLOF` unterdrückt werden. Wird `DISPLOF` zusammen mit `SBLOF` programmiert, so wird bei Einzelsatz-Stops innerhalb des Zyklus/Unterprogramms nach wie vor der Aufruf des Zyklus/Unterprogramms angezeigt.

- **Einzelsatzunterdrückung im System-ASUP oder Anwender-ASUP**

Wenn der Einzelsatz-Stopp im System- oder Anwender-ASUP über die Einstellungen im Maschinendatum MD10702 `$MN_IGNORE_SINGLEBLOCK_MASK` unterdrückt wird (Bit0 = 1 bzw. Bit1 = 1), dann kann der Einzelsatz-Stopp durch Programmierung von `SBLON` im ASUP wieder aktiviert werden.

Wird der Einzelsatz-Stopp im Anwender-ASUP über die Einstellung im Maschinendatum MD20117 `$MC_IGNORE_SINGLEBLOCK_ASUP` unterdrückt, dann kann der Einzelsatz-Stopp durch Programmierung von `SBLON` im ASUP **nicht** wieder aktiviert werden.

- **Besonderheiten der Einzelsatzunterdrückung bei den verschiedenen Einzelsatzbearbeitungstypen**

Bei aktiver Einzelsatzbearbeitung SBL2 (Stopp nach jedem Teileprogrammsatz) wird im `SBLON`-Satz **nicht** angehalten, wenn im MD10702 `$MN_IGNORE_SINGLEBLOCK_MASK` (Einzelsatzstopp verhindern) Bit 12 auf "1" gesetzt ist.

Bei aktiver Einzelsatzbearbeitung SBL3 (Stopp nach jedem Teileprogrammsatz auch im Zyklus) wird der Befehl `SBLOF` unterdrückt.

Beispiele

Beispiel 1: Einzelsatzunterdrückung innerhalb eines Programms

Programmcode	Kommentar
N10 G1 X100 F1000	
N20 SBLOF	; Einzelsatz ausschalten
N30 Y20	
N40 M100	
N50 R10=90	
N60 SBLON	; Einzelsatz wieder einschalten
N70 M110	
N80 ...	

Der Bereich zwischen `N20` und `N60` wird im Einzelsatzbetrieb als ein Schritt bearbeitet.

Beispiel 2: Zyklus soll für den Anwender wie ein Befehl wirken

Hauptprogramm:

Programmcode
N10 G1 X10 G90 F200
N20 X-4 Y6
N30 CYCLE1
N40 G1 X0
N50 M30

Zyklus CYCLE1:

Programmcode	Kommentar
N100 PROC CYCLE1 DISPLOF SBLOF	; Einzelsatz unterdrücken
N110 R10=3*SIN(R20)+5	
N120 IF (R11 <= 0)	
N130 SETAL(61000)	
N140 ENDIF	
N150 G1 G91 Z=R10 F=R11	
N160 M17	

Der Zyklus CYCLE1 wird bei aktiver Einzelsatzbearbeitung abgearbeitet, d. h. es muss für die Bearbeitung von CYCLE1 einmal die Start-Taste gedrückt werden.

Beispiel 3:

Ein von der PLC gestartetes ASUP zum Aktivieren von geänderten Nullpunktverschiebung und Werkzeugkorrekturen soll nicht sichtbar sein.

```
Programmcode
N100 PROC NV SBLOF DISPLOF
N110 CASE $P_UIFRNUM OF      0 GOTOF _G500
                             1 GOTOF _G54
                             2 GOTOF _G55
                             3 GOTOF _G56
                             4 GOTOF _G57
                             DEFAULT GOTOF END
N120 _G54: G54 D=$P_TOOL T=$P_TOOLNO
N130 RET
N140 _G54: G55 D=$P_TOOL T=$P_TOOLNO
N150 RET
N160 _G56: G56 D=$P_TOOL T=$P_TOOLNO
N170 RET
N180 _G57: G57 D=$P_TOOL T=$P_TOOLNO
N190 RET
N200 END: D=$P_TOOL T=$P_TOOLNO
N210 RET
```

Beispiel 4: Mit MD10702 Bit 12 = 1 wird nicht angehalten

Ausgangssituation:

- Einzelsatzbearbeitung ist aktiv.
- MD10702 \$MN_IGNORE_SINGLEBLOCK_MASK Bit12 = 1

Hauptprogramm:

Programmcode	Kommentar
N10 G0 X0	; In dieser Teileprogrammzeile stoppen.
N20 X10	; In dieser Teileprogrammzeile stoppen.
N30 CYCLE	; Vom Zyklus generierter Verfahrssatz.
N50 G90 X20	; In dieser Teileprogrammzeile stoppen.
M30	

Zyklus CYCLE:

Programmcode	Kommentar
PROC CYCLE SBLOF	; Einzelsatz-Stopp unterdrücken
N100 R0 = 1	
N110 SBLON	; Wegen MD10702 Bit12=1 wird in dieser Teileprogrammzeile nicht gestoppt.
N120 X1	; In dieser Teileprogrammzeile wird gestoppt.
N140 SBLOF	
N150 R0 = 2	
RET	

Beispiel 5: Einzelsatzunterdrückung bei Programmschachtelung

Ausgangssituation:

Einzelsatzbearbeitung ist aktiv.

Programmverschachtelung:

Programmcode	Kommentar
N10 X0 F1000	; In diesem Satz wird gestoppt.
N20 UP1(0)	
PROC UP1(INT _NR) SBLOF	; Einzelsatz-Stopp unterdrücken.
N100 X10	
N110 UP2(0)	
PROC UP2(INT _NR)	
N200 X20	
N210 SBLON	; Einzelsatz-Stopp einschalten.
N220 X22	; In diesem Satz wird gestoppt.
N230 UP3(0)	
PROC UP3(INT _NR)	
N300 SBLOF	; Einzelsatz-Stopp unterdrücken.
N305 X30	
N310 SBLON	; Einzelsatz-Stopp einschalten.
N320 X32	; In diesem Satz wird gestoppt.
N330 SBLOF	; Einzelsatz-Stopp unterdrücken.
N340 X34	
N350 M17	; SBLOF ist aktiv.
N240 X24	; In diesem Satz wird gestoppt. SBLON ist aktiv.
N250 M17	; In diesem Satz wird gestoppt. SBLON ist aktiv.
N120 X12	
N130 M17	; In diesem Rücksprungssatz wird gestoppt. SBLOF der PROC-Anweisung ist aktiv.
N30 X0	; In diesem Satz wird gestoppt.
N40 M30	; In diesem Satz wird gestoppt.

Weitere Informationen

Einzelsatzsperrung für asynchrone Unterprogramme

Um ein ASUP im Einzelsatz in einem Schritt abzuarbeiten, muss im ASUP eine `PROC`-Anweisung mit `SBLOF` programmiert werden. Dies gilt auch für die Funktion "Editierbares System-ASUP" (MD11610 `$MN_ASUP_EDITABLE`).

Beispiel für ein editierbares System-ASUP:

Programmcode	Kommentar
N10 PROC ASUP1 SBLOF DISPLOF	
N20 IF \$AC_ASUP=='H200'	
N30 RET	; Kein REPOS bei BA-Wechsel.
N40 ELSE	
N50 REPOSA	; REPOS in allen übrigen Fällen.
N60 ENDIF	

Programmbeeinflussungen im Einzelsatz

In der Einzelsatzbearbeitung kann der Anwender das Teileprogramm satzweise abarbeiten. Es existieren folgende Einstellungsarten:

- SBL1: IPO-Einzelsatz mit Stopp nach jedem Maschinenfunktionssatz.
- SBL2: Einzelsatz mit Stopp nach jedem Satz.
- SBL3: Halt im Zyklus (durch die Anwahl von SBL3 wird der `SBLOF`-Befehl unterdrückt).

Einzelsatzunterdrückung bei Programmschachtelung

Wurde in einem Unterprogramm `SBLOF` in der `PROC`-Anweisung programmiert, so wird auf den Unterprogrammrückprung mit `M17` angehalten. Damit wird verhindert, dass im aufrufenden Programm bereits der nächste Satz ausgeführt wird. Wird in einem Unterprogramm mit `SBLOF`, ohne `SBLOF` in der `PROC`-Anweisung, eine Einzelsatzunterdrückung aktiviert, wird erst nach dem nächsten Maschinenfunktionssatz des aufrufenden Programms angehalten. Ist dies nicht erwünscht, muss im Unterprogramm noch vor dem Rückprung (`M17`) wieder `SBLON` programmiert werden. Bei einem Rückprung mit `RET` in ein übergeordnetes Programm wird nicht angehalten.

1.25.2.6 Aktuelle Satzanzeige unterdrücken (DISPLOF, DISPLON, ACTBLOCNO)

Funktion

In der Satzanzeige wird standardmäßig der aktuelle Programmsatz angezeigt. In Zyklen bzw. Unterprogrammen kann die Anzeige des aktuellen Satzes mit dem Befehl `DISPLOF` unterdrückt werden. Anstelle des aktuellen Satzes wird dann der Aufruf des Zyklus bzw. Unterprogramms angezeigt. Mit dem Befehl `DISPLON` kann die Unterdrückung der Satzanzeige wieder aufgehoben werden.

`DISPLOF` bzw. `DISPLON` wird in der Programmzeile mit der `PROC`-Anweisung programmiert und wirkt für das gesamte Unterprogramm und implizit für alle von diesem Unterprogramm aufgerufenen Unterprogramme, die keinen `DISPLON`- bzw. `DISPLOF`-Befehl enthalten. Dieses Verhalten gilt auch für ASUPs.

Syntax

```
PROC ... DISPLOF
PROC ... DISPLOF ACTBLOCNO
PROC ... DISPLON
```

Bedeutung

<code>DISPLOF:</code>	<p>Befehl zum Unterdrücken der aktuellen Satzanzeige.</p> <p>Platzierung: Am Ende der Programmzeile mit der <code>PROC</code>-Anweisung</p> <p>Wirksamkeit: Bis zum Rücksprung aus dem Unterprogramm oder Programmende.</p> <p>Hinweis: Wenn aus dem Unterprogramm mit dem <code>DISPLOF</code>-Befehl weitere Unterprogramme aufgerufen werden, dann wird auch in diesen Unterprogrammen die aktuelle Satzanzeige unterdrückt, sofern in diesen nicht explizit <code>DISPLON</code> programmiert ist.</p>
<code>DISPLON:</code>	<p>Befehl zum Aufheben der Unterdrückung der aktuellen Satzanzeige</p> <p>Platzierung: Am Ende der Programmzeile mit der <code>PROC</code>-Anweisung</p> <p>Wirksamkeit: Bis zum Rücksprung aus dem Unterprogramm oder Programmende.</p> <p>Hinweis: Wenn aus dem Unterprogramm mit dem <code>DISPLON</code>-Befehl weitere Unterprogramme aufgerufen werden, dann wird auch in diesen Unterprogrammen der aktuelle Programmsatz angezeigt, sofern in diesen nicht explizit <code>DISPLOF</code> programmiert ist.</p>
<code>ACTBLOCNO:</code>	<p><code>DISPLOF</code> zusammen mit dem Attribut <code>ACTBLOCNO</code> bewirkt, dass im Falle eines Alarms die Nummer des aktuellen Satzes ausgegeben wird, in dem der Alarm aufgetreten ist. Dies gilt auch dann, wenn in einer niedrigeren Programmebene nur <code>DISPLOF</code> programmiert ist.</p> <p>Bei <code>DISPLOF</code> ohne <code>ACTBLOCNO</code> wird dagegen die Satznummer des Zyklus- bzw. Unterprogrammaufrufs aus der letzten nicht mit <code>DISPLOF</code> gekennzeichneten Programmebene angezeigt.</p>

Beispiele

Beispiel 1: Aktuelle Satzanzeige im Zyklus unterdrücken

Programmcode	Kommentar
PROC CYCLE(AXIS TOMOV, REAL POSITION) SAVE DISPLOF	; Aktuelle Satzanzeige unterdrücken. Stattdessen soll der Zyklus-Aufruf angezeigt werden, z. B.: CYCLE(X,100.0)
DEF REAL DIFF	; Zyklen-Inhalt
G01 ...	
...	
RET	; Unterprogramm-Rücksprung. In der Satzanzeige wird der auf den Zyklus-Aufruf folgende Satz angezeigt.

Beispiel 2: Satzanzeige bei der Alarmausgabe

Unterprogramm SUBPROG1 (mit ACTBLOCNO):

Programmcode	Kommentar
PROC SUBPROG1 DISPLOF ACTBLOCNO	
N8000 R10 = R33 + R44	
...	
N9040 R10 = 66 X100	; Alarm 12080 auslösen
...	
N10000 M17	

Unterprogramm SUBPROG2 (ohne ACTBLOCNO):

Programmcode	Kommentar
PROC SUBPROG2 DISPLOF	
N5000 R10 = R33 + R44	
...	
N6040 R10 = 66 X100	; Alarm 12080 auslösen
...	
N7000 M17	

Hauptprogramm:

Programmcode	Kommentar
N1000 G0 X0 Y0 Z0	
N1010 ...	
...	
N2050 SUBPROG1	; Alarmausgabe = "12080 Kanal K1 Satz N9040 Syntaxfehler bei Text R10="
N2060 ...	
N2350 SUBPROG2	; Alarmausgabe = "12080 Kanal K1 Satz N2350 Syntaxfehler bei Text R10="
...	
N3000 M30	

Beispiel 3: Unterdrückung der aktuellen Satzanzeige aufheben

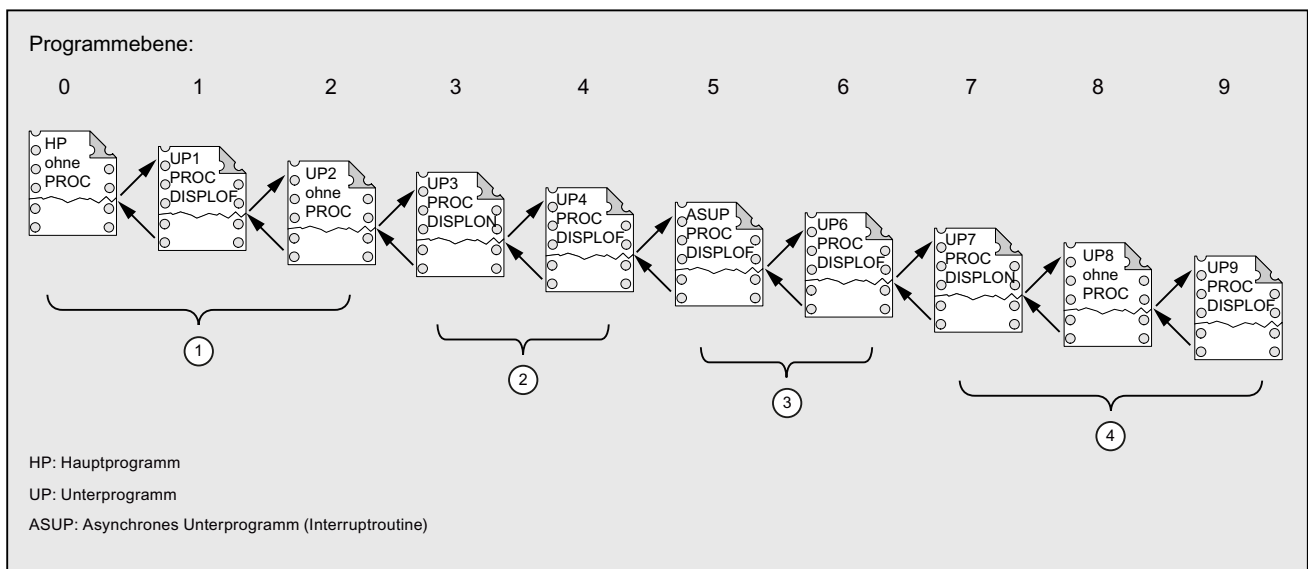
Unterprogramm SUB1 mit Unterdrückung:

Programmcode	Kommentar
PROC SUB1 DISPLOF	; Aktuelle Satzanzeige im Unterprogramm SUB1 unterdrücken. Stattdessen soll der Satz mit dem SUB1-Aufruf angezeigt werden.
...	
N300 SUB2	; Unterprogramm SUB2 aufrufen.
...	
N500 M17	

Unterprogramm SUB2 ohne Unterdrückung:

Programmcode	Kommentar
PROC SUB2 DISPLON	; Unterdrückung der aktuellen Satzanzeige im Unterprogramm SUB2 aufheben.
...	
N200 M17	; Rücksprung ins Unterprogramm SUB1. In SUB1 wird die aktuelle Satzanzeige wieder unterdrückt.

Beispiel 4: Anzeigeverhalten bei unterschiedlichen DISPLON/DISPLOF-Kombinationen



- ① In der aktuellen Satzanzeige werden die Teileprogrammzeilen aus Programmebene 0 angezeigt.
- ② In der aktuellen Satzanzeige werden die Teileprogrammzeilen aus Programmebene 3 angezeigt.
- ③ In der aktuellen Satzanzeige werden die Teileprogrammzeilen aus Programmebene 3 angezeigt.
- ④ In der aktuellen Satzanzeige werden die Teileprogrammzeilen aus Programmebene 7/8 angezeigt.

1.25.2.7 Unterprogramme mit Vorbereitung kennzeichnen (PREPRO)

Funktion

Mit dem Schlüsselwort `PREPRO` können im Hochlauf am Ende der `PROC`-Anweisungszeile alle Dateien gekennzeichnet werden.

Hinweis

Diese Art der Programmvorbereitung ist vom entsprechend eingestellten Maschinendatum abhängig. Bitte Angaben des Maschinenherstellers beachten.

Literatur:

Funktionshandbuch Sonderfunktionen; Vorverarbeitung (V2)

Syntax

```
PROC ... PREPRO
```

Bedeutung

`PREPRO`: Schlüsselwort für Kennzeichnung aller im Hochlauf vorbereiteten Dateien, der in Zyklenverzeichnissen abgelegten NC-Programme

Unterprogramme mit Vorbereitung einlesen und Unterprogrammaufruf

Sowohl im Hochlauf vorbereiteter Unterprogramme mit Parametern als auch beim Unterprogrammaufruf werden die Zyklenverzeichnissen in der gleichen Reihenfolge behandelt:

1. `_N_CUS_DIR` Anwenderzyklen
2. `_N_CMA_DIR` Herstellerzyklen
3. `_N_CST_DIR` Standardzyklen

Im Falle gleichnamiger NC-Programme mit unterschiedlicher Ausprägung wird die zuerst gefundene `PROC`-Anweisung aktiviert und die andere `PROC`-Anweisung wird ohne Alarmmeldung überlesen.

1.25.2.8 Unterprogrammrücksprung M17

Funktion

Am Ende eines Unterprogramms steht der Rücksprung-Befehl `M17` (bzw. der Teileprogrammende-Befehl `M30`). Er bewirkt den Rücksprung in das aufrufende Programm auf den Teileprogrammsatz nach dem Unterprogrammaufruf.

Hinweis

`M17` und `M30` werden in der NC-Sprache gleichwertig behandelt.

Syntax

```
PROC <Programmname>
...
M17/M30
```

Randbedingungen

Auswirkung des Unterprogrammrücksprungs auf den Bahnsteuerbetrieb

Steht `M17` (bzw. `M30`) alleine im Teilprogrammsatz, wird dadurch ein im Kanal aktiver Bahnsteuerbetrieb unterbrochen.

Um zu vermeiden, dass der Bahnsteuerbetrieb unterbrochen wird, ist `M17` (bzw. `M30`) mit in den letzten Verfahrssatz zu schreiben. Zusätzlich muss folgendes Maschinendatum auf "0" gesetzt sein:

MD20800 \$MC_SPF_END_TO_VDI = 0 (keine M30/M17-Ausgabe an die NC/PLC-Nahtstelle)

Beispiel

1. Unterprogramm mit `M17` im eigenen Satz

Programmcode	Kommentar
N10 G64 F2000 G91 X10 Y10	
N20 X10 Z10	
N30 M17	; Rücksprung mit Unterbrechung des Bahnsteuerbetriebs.

2. Unterprogramm mit `M17` im letzten Verfahrssatz

Programmcode	Kommentar
N10 G64 F2000 G91 X10 Y10	
N20 X10 Z10 M17	; Rücksprung ohne Unterbrechung des Bahnsteuerbetriebs.

1.25.2.9 Unterprogrammrückprung RET

Funktion

Als Ersatz für den Rücksprungsbehehl M17 kann im Unterprogramm auch der Befehl RET verwendet werden. RET muss in einem eigenen Teileprogrammssatz programmiert werden. Wie M17 bewirkt RET den Rücksprung in das aufrufende Programm auf den Teileprogrammssatz nach dem Unterprogrammaufruf.

Hinweis

Durch die Programmierung von Parametern kann das Rücksprungsverhalten von RET geändert werden (siehe "Parametrierbarer Unterprogrammrückprung (RET ...) (Seite 193)").

Anwendung

Die RET-Anweisung ist dann zu benutzen, wenn ein G64-Bahnsteuerbetrieb (G641 ... G645) durch den Rücksprung nicht unterbrochen werden soll.

Voraussetzung

Der Befehl RET kann nur in Unterprogrammen verwendet werden, die nicht mit dem Attribut SAVE definiert wurden.

Syntax

```
PROC <Programmname>  
...  
RET
```

Beispiel

Hauptprogramm:

Programmcode	Kommentar
PROC MAIN_PROGRAM	; Programmanfang
...	
N50 SUB_PROG	; Unterprogrammaufruf: SUB_PROG
N60 ...	
...	
N100 M30	; Programmende

Unterprogramm:

Programmcode	Kommentar
PROC SUB_PROG	
...	
N100 RET	; Rücksprung erfolgt auf Satz N60 im Hauptprogramm.

1.25.2.10 Parametrierbarer Unterprogrammrücksprung (RET ...)

Funktion

Im Allgemeinen wird aus einem Unterprogramm mit einem Unterprogrammende `RET` oder `M17` in das Programm zurückgesprungen, aus dem das Unterprogramm aufgerufen wurde, und die Bearbeitung wird mit der auf den Unterprogrammaufruf folgenden Programmzeile fortgesetzt.

Daneben gibt es jedoch auch Anwendungsfälle, wo die Programmbearbeitung an einer anderen Stelle fortgesetzt werden soll, z. B.:

- Fortsetzung der Programmbearbeitung nach Aufruf der Abspanzyklen im ISO-Dialekt-Modus (nach der Konturbeschreibung).
- Rücksprung ins Hauptprogramm aus einer beliebigen Unterprogrammebene (auch nach ASUP) beim Fehlerhandling.
- Rücksprung über mehrere Programmebenen für spezielle Anwendungen in Compilazyklen und im ISO-Dialekt-Modus.

In solchen Fällen wird der Befehl `RET` zusammen mit "Rücksprungsparametern" programmiert.

Syntax

```
RET("<Zielsatz>")  
RET("<Zielsatz>",<Satz nach Zielsatz>)  
RET("<Zielsatz>",<Satz nach Zielsatz>,<Anzahl der Rücksprungebenen>)  
RET("<Zielsatz>",<Anzahl der Rücksprungebenen>)  
RET("<Zielsatz>",<Satz nach Zielsatz>,<Anzahl der Rücksprungebenen>,<Rücksprung auf Programmanfang>)  
RET( , ,<Anzahl der Rücksprungebenen>,<Rücksprung auf Programmanfang>)
```

Bedeutung

<code>RET:</code>	Unterprogrammende (Verwendung statt <code>M17</code>)
<code><Zielsatz>:</code>	Rücksprungsparameter 1 Nennt als Sprungziel den Satz, an dem die Programmbearbeitung fortgesetzt werden soll. Wenn Rücksprungsparameter 3 nicht programmiert ist, dann befindet sich das Sprungziel in dem Programm, aus dem das aktuelle Unterprogramm aufgerufen wurde. Mögliche Angaben sind: " <code><Satznummer></code> " Nummer des Zielsatzes " <code><Sprungmarke></code> " Sprungmarke, die im Zielsatz gesetzt sein muss.

"<Zeichenkette>" Zeichenkette, die im Programm bekannt sein muss (z. B. Programm- oder Variablenname).
 Für die Programmierung der Zeichenkette im Zielsatz gelten folgende Regeln:

- **Leerzeichen am Ende** (im Unterschied zur Sprungmarke, die durch einen ":" am Ende gekennzeichnet ist).
- **Vor der Zeichenkette** dürfen nur eine Satznummer und/oder eine Sprungmarke gesetzt sein, **keine Programmbefehle**.

<Satz nach Zielsatz>:

Rücksprungsparameter 2

Bezieht sich auf den Rücksprungsparameter 1.

Typ: INT

Wert: 0 Rücksprung erfolgt auf den Satz, der mit dem Rücksprungsparameter 1 angegeben wurde.
 > 0 Rücksprung erfolgt auf den Satz, der auf den mit dem Rücksprungsparameter 1 angegeben Satz folgt.

<Anzahl der Rücksprungebenen>:

Rücksprungsparameter 3

Nennt die Anzahl an Ebenen, die zurückgesprungen werden soll, um zu der Programmebene zu gelangen, in der die Programmbearbeitung fortgesetzt werden soll.

Typ: INT

Wert: 1 Das Programm wird in der "aktuellen Programmebene - 1" fortgesetzt (wie RET ohne Parameter).
 2 Das Programm wird in der "aktuellen Programmebene - 2" fortgesetzt, d. h. es wird eine Ebene übersprungen.
 3 Das Programm wird in der "aktuellen Programmebene - 3" fortgesetzt, d. h. es werden zwei Ebenen übersprungen.

...

Wertebereich: 1 ... 15

<Rücksprung auf Programmanfang>:

Rücksprungsparameter 4

Typ: BOOL

Wert: 1 Wenn der Rücksprung ins Hauptprogramm erfolgt und dort ein **ISO-Dialekt-Modus** aktiv ist, wird auf den Programmanfang verzweigt.

Hinweis


Bei einem Unterprogrammrückprung mit einer Zeichenkette als Angabe für die Zielsatzsuche wird im aufrufenden Programm immer zuerst nach einer Sprungmarke gesucht.

Wenn ein Sprungziel durch eine Zeichenkette eindeutig definiert sein soll, darf die Zeichenkette daher nicht mit dem Namen einer Sprungmarke übereinstimmen, da sonst der Unterprogrammrückprung immer auf die Sprungmarke und nicht auf die Zeichenkette ausgeführt wird (siehe Beispiel 2).

Randbedingungen

Beim Rückprung über mehrere Programmebenen werden die `SAVE`-Anweisungen der einzelnen Programmebenen ausgewertet.

Ist bei einem Rückprung über mehrere Programmebenen ein modales Unterprogramm aktiv und ist in einem der übersprungenen Unterprogramme der Abwahlbefehl `MCALL` für das modale Unterprogramm programmiert, bleibt das modale Unterprogramm weiterhin aktiv.

	VORSICHT
Der Programmierer muss darauf achten, dass beim Rückprung über mehrere Programmebenen mit den richtigen modalen Einstellungen fortgesetzt wird. Dies wird z. B. durch Programmierung eines entsprechenden Hauptsatzes erreicht.	

Beispiele

Beispiel 1: Wiederaufsetzen im Hauptprogramm nach ASUP-Bearbeitung

Programmierung	Kommentar
N10010 CALL "UP1"	; Programmebene 0 (Hauptprogramm)
N11000 PROC UP1	; Programmebene 1
N11010 CALL "UP2"	
N12000 PROC UP2	; Programmebene 2
...	
N19000 PROC ASUP	; Programmebene 3 (ASUP-Bearbeitung)
...	
N19100 RET("N10900", , \$P_STACK)	; Unterprogrammrückprung
N10900	; Wiederaufsetzen im Hauptprogramm.
N10910 MCALL	; Modales Unterprogramm ausschalten.
N10920 G0 G60 G40 M5	; Weitere modale Einstellungen korrigieren.

Beispiel 2: Zeichenkette (<String>) als Angabe für die Zielsatzsuche

Hauptprogramm:

Programmcode	Kommentar
PROC MAIN_PROGRAM	
N1000 DEF INT iVar1=1, iVar2=4	
N1010 ...	
N1200 subProg1	; Aufruf von Unterprogramm "subProg1"
N1210 M2 S1000 X10 F1000	
N1220	
N1400 subProg2	; Aufruf von Unterprogramm "subProg2"
N1410 M3 S500 Y20	
N1420 ..	
N1500 lab1: iVar1=R10*44	
N1510 F500 X5	
N1520 ...	
N1550 subprog1: G1 X30	; "subProg1" ist hier als Sprungmarke definiert.
N1560 ...	
N1600 subProg3	Aufruf von Unterprogramm "subProg3"
N1610 ...	
N1900 M30	

Unterprogramm subProg1:

Programmcode	Kommentar
PROC subProg1	
N2000 R10=R20+100	
N2010 ...	
N2200 RET("subProg2")	; Rücksprung ins Hauptprogramm auf den Satz N1400

Unterprogramm subProg2:

Programmcode	Kommentar
PROC subProg2	
N2000 R10=R20+100	
N2010 ...	
N2200 RET("iVar1")	; Rücksprung ins Hauptprogramm auf den Satz N1500

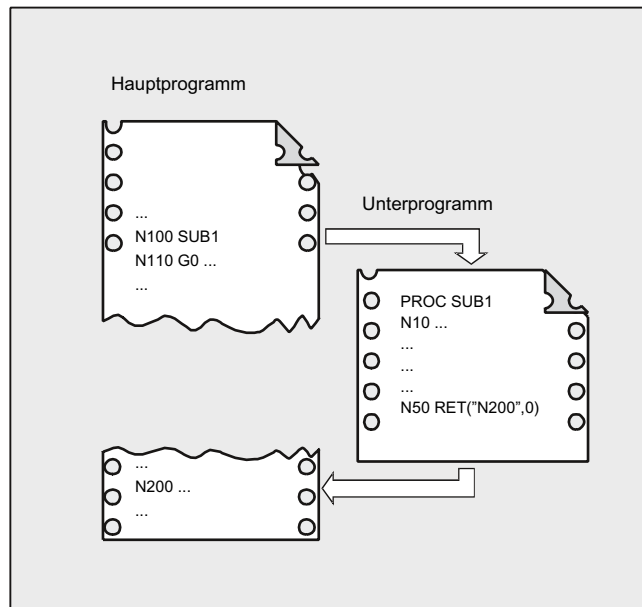
Unterprogramm subProg3:

Programmcode	Kommentar
PROC subProg3	
N2000 R10=R20+100	
N2010 ...	
N2200 RET("subProg1")	; Rücksprung ins Hauptprogramm auf den Satz N1550

Weitere Informationen

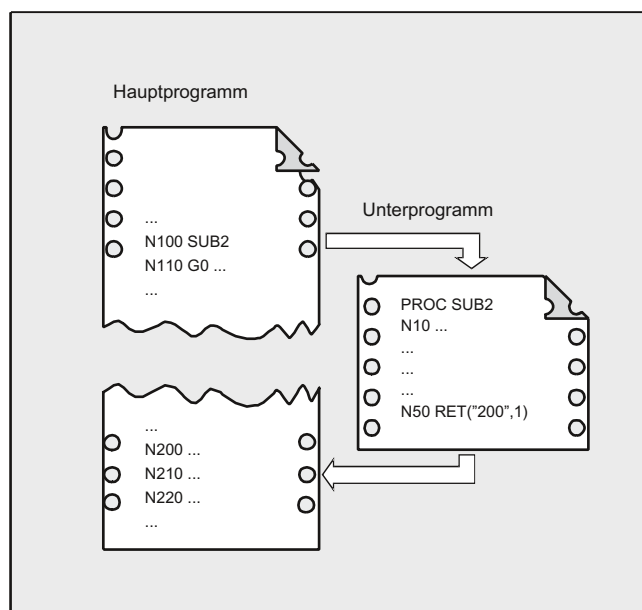
Die folgenden Grafiken sollen die unterschiedlichen Wirkungen der Rücksprungsparameter 1 bis 3 veranschaulichen.

1. Rücksprungsparameter 1 = "N200", Rücksprungsparameter 2 = 0



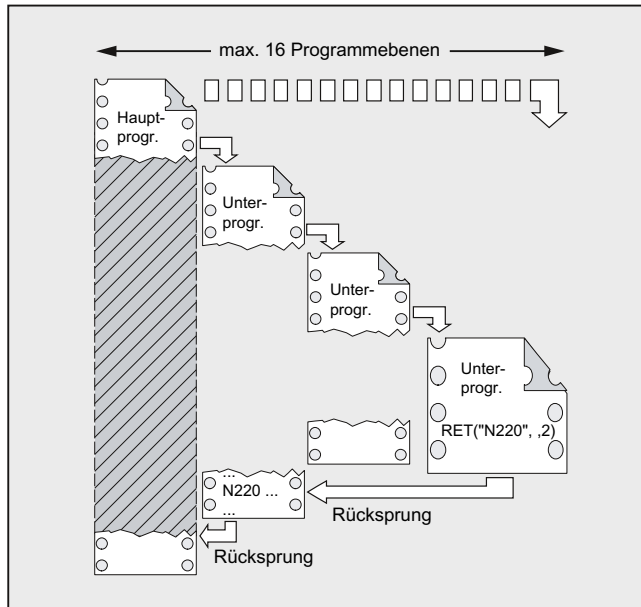
Nach dem RET-Befehl wird die Programmbearbeitung mit dem Satz N200 im Hauptprogramm fortgesetzt.

2. Rücksprungsparameter 1 = "N200", Rücksprungsparameter 2 = 1



Nach dem RET-Befehl wird die Programmbearbeitung mit dem Satz (N210) fortgesetzt, der auf den Satz N200 im Hauptprogramm folgt.

3. Rücksprungsparameter 1 = "N220", Rücksprungsparameter 3 = 2



Nach dem `RET`-Befehl wird zwei Programmebenen zurückgesprungen und die Programmbearbeitung wird mit dem Satz `N220` fortgesetzt.

1.25.3 Aufruf eines Unterprogramms

1.25.3.1 Unterprogrammaufruf ohne Parameterübergabe

Funktion

Der Aufruf eines Unterprogramms erfolgt entweder mit Adresse L und Unterprogrammnummer oder durch Angabe des Programmnamens.

Auch ein Hauptprogramm kann als Unterprogramm aufgerufen werden. Das im Hauptprogramm gesetzte Programmende M_2 oder M_{30} wird in diesem Fall wie M_{17} (Programmende mit Rücksprung ins aufrufende Programm) gewertet.

Hinweis

Entsprechend kann ein Unterprogramm auch als Hauptprogramm gestartet werden.

Suchstrategie der Steuerung:

Gibt es *_MPF?

Gibt es *_SPF?

Daraus folgt: Falls der Name des aufzurufenden Unterprogramms mit dem Namen des Hauptprogramms identisch ist, dann wird das aufrufende Hauptprogramm wieder aufgerufen. Dieser in der Regel nicht gewünschte Effekt muss durch eindeutige Namenswahl über Unterprogramme und Hauptprogramme vermieden werden.

Hinweis

Unterprogramme, die keine Parameterübergabe erfordern, können auch aus einer Initialisierungsdatei aufgerufen werden.

Syntax

L<Nummer>/<Programmname>

Hinweis

Der Aufruf eines Unterprogramms muss immer im eigenen NC-Satz programmiert werden.

Bedeutung

L: Adresse für den Unterprogrammaufruf

<Nummer>: Nummer des Unterprogramms

Typ: INT

Wert: Maximal 7 Dezimalstellen

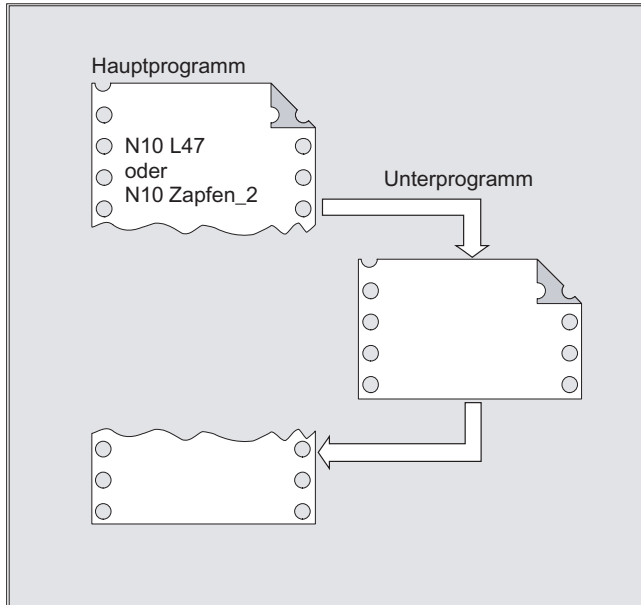
Achtung:

Führende Nullen sind bei der Namensgebung von Bedeutung (\Rightarrow L123, L0123 und L00123 sind drei verschiedene Unterprogramme).

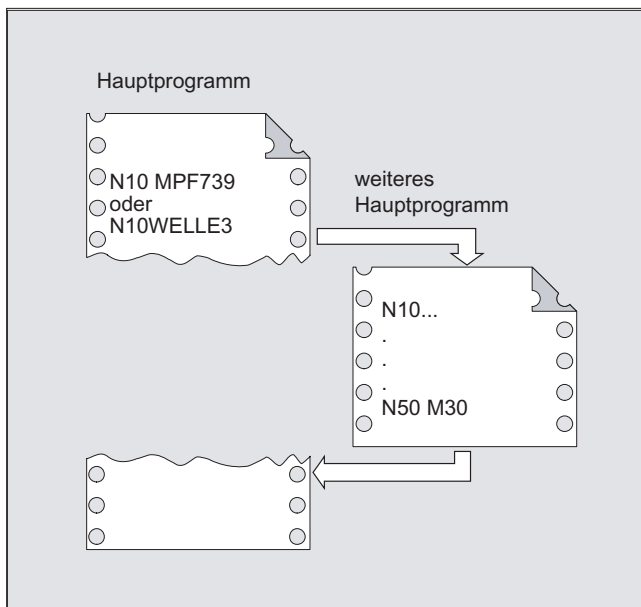
<Programmname>: Name des Unterprogramms (oder Hauptprogramms)

Beispiele

Beispiel 1: Aufruf eines Unterprogramms ohne Parameterübergabe



Beispiel 2: Aufruf eines Hauptprogramms als Unterprogramm



1.25.3.2 Unterprogrammaufruf mit Parameterübergabe (EXTERN)

Funktion

Beim Unterprogrammaufruf mit Parameterübergabe können Variablen oder Werte direkt übergeben werden (nicht bei VAR-Parametern).

Unterprogramme mit Parameterübergabe müssen vor dem Aufruf im Hauptprogramm mit `EXTERN` bekannt gemacht werden (z. B. am Programmanfang). Angegeben werden dabei der Name des Unterprogramms und die Variablentypen in der Reihenfolge der Übergabe.

VORSICHT

Sowohl die Variablentypen als auch die Reihenfolge der Übergabe muss mit den Definitionen, die im Unterprogramm unter `PROC` vereinbart wurden, übereinstimmen. Die Parameternamen können in Haupt- und Unterprogramm unterschiedlich sein.

Syntax

```
EXTERN <Programmname> (<Typ_Par1>, <Typ_Par2>, <Typ_Par3>)  
...  
<Programmname> (<Wert_Par1>, <Wert_Par2>, <Wert_Par3>)
```

VORSICHT

Der Unterprogrammaufruf muss immer im eigenen NC-Satz programmiert werden.

Bedeutung

<Programmname>:

EXTERN:

<Typ_Par1>, <Typ_Par2>, <Typ_Par3>:

<Wert_Par1>, <Wert_Par2>, <Wert_Par3>:

Name des Unterprogramms

Schlüsselwort für die Bekanntmachung eines Unterprogramms mit Parameterübergabe

Hinweis:

`EXTERN` muss nur dann angegeben werden, wenn das Unterprogramm im Werkstück- oder im globalen Unterprogrammverzeichnis steht. Zyklen müssen nicht als `EXTERN` erklärt werden.

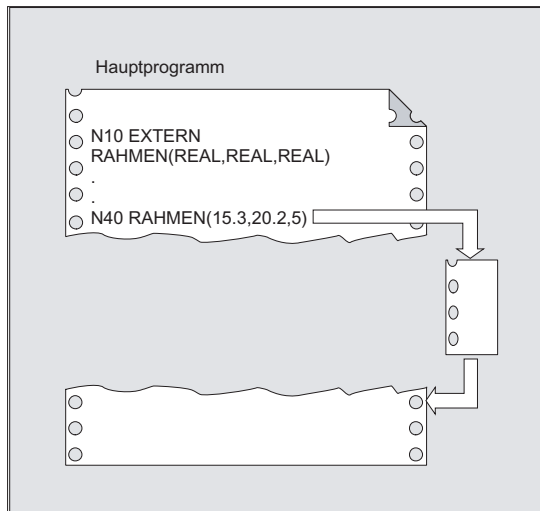
Variablentypen der zu übergebenden Parameter in der Reihenfolge der Übergabe

Variablenwerte für die zu übergebenden Parameter

Beispiele

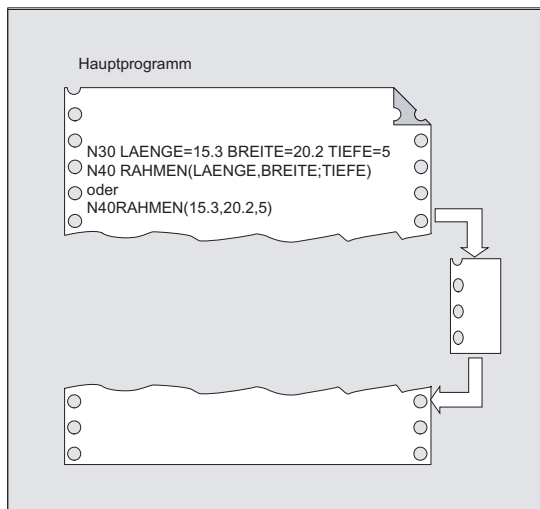
Beispiel 1: Unterprogrammaufruf mit vorhergehender Bekanntmachung

Programmcode	Kommentar
N10 EXTERN RAHMEN (REAL, REAL, REAL)	; Angabe des Unterprogramms.
...	
N40 RAHMEN (15.3, 20.2, 5)	; Aufruf des Unterprogramms mit Parameterübergabe.



Beispiel 2: Unterprogrammaufruf ohne Bekanntmachung


Programmcode	Kommentar
N10 DEF REAL LAENGE, BREITE, TIEFE	
N20 ...	
N30 LAENGE=15.3 BREITE=20.2 TIEFE=5	
N40 RAHMEN (LAENGE, BREITE, TIEFE)	; oder: N40 RAHMEN (15.3, 20.2, 5)



1.25.3.3 Anzahl der Programmwiederholungen (P)

Funktion

Soll ein Unterprogramm mehrfach hintereinander abgearbeitet werden, kann im Satz mit dem Unterprogrammaufruf unter der Adresse P die gewünschte Anzahl der Programmwiederholungen programmiert werden.

 VORSICHT
Unterprogrammaufruf mit Programmwiederholung und Parameterübergabe
Parameter werden nur beim Programmaufruf bzw. ersten Durchlauf übergeben. Für die weiteren Wiederholungen bleiben die Parameter unverändert. Falls Sie bei Programmwiederholungen die Parameter verändern wollen, müssen Sie im Unterprogramm entsprechende Vereinbarungen festlegen.

Syntax

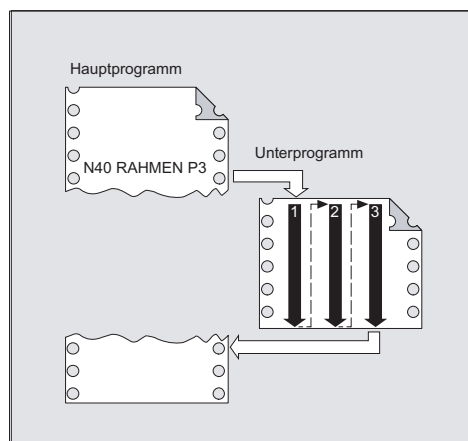
<Programmname> P<Wert>

Bedeutung

<Programmname>:	Unterprogrammaufruf
P:	Adresse für die Programmierung von Programmwiederholungen
<Wert>:	Anzahl der Programmwiederholungen
Typ:	INT
Wertebereich:	1 ... 9999 (ohne Vorzeichen)

Beispiel

Programmcode	Kommentar
...	
N40 RAHMEN P3	; Das Unterprogramm RAHMEN soll dreimal hintereinander abgearbeitet werden.
...	



1.25.3.4 Modaler Unterprogrammaufruf (MCALL)

Funktion

Bei einem modalen Unterprogrammaufruf mit `MCALL` wird das Unterprogramm nach jedem Satz mit Bahnbewegung automatisch aufgerufen und abgearbeitet. Hierdurch lässt sich der Aufruf von Unterprogrammen automatisieren, die an unterschiedlichen Werkstückpositionen abgearbeitet werden sollen (zum Beispiel für die Herstellung von Bohrbildern).

Das Ausschalten der Funktion erfolgt mit `MCALL` ohne Unterprogrammaufruf oder durch Programmierung eines neuen modalen Unterprogrammaufrufs für ein neues Unterprogramm.

VORSICHT

In einem Programmablauf kann gleichzeitig nur ein `MCALL`-Aufruf wirken. Parameter werden nur einmal beim `MCALL`-Aufruf übergeben.

Das modale Unterprogramm wird in folgenden Situationen auch ohne Programmierung einer Bewegung aufgerufen:

- Bei Programmierung der Adressen `S` und `F` wenn `G0` oder `G1` aktiv ist.
- Wenn `G0/G1` allein im Satz oder mit weiteren `G`-Codes programmiert wurde.

ACHTUNG

In ASUPs, durch die die Bearbeitung eines Teileprogramms unterbrochen wird (siehe "Interruptroutine (ASUP) (Seite 130)"), werden **keine** modalen Unterprogrammaufrufe ausgeführt.

ASUPs, die aus dem Reset-Zustand gestartet werden, verhalten sich bezüglich der modalen Unterprogrammaufrufe dagegen wie normale Teileprogramme.

Syntax

`MCALL <Programmname>`

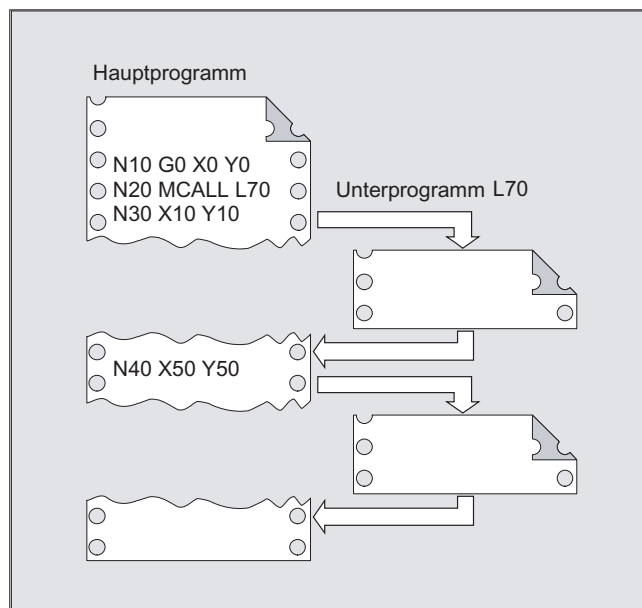
Bedeutung

<code>MCALL:</code>	Befehl für den modalen Unterprogrammaufruf
<code><Programmname>:</code>	Name des Unterprogramms

Beispiele

Beispiel 1:

Programmcode	Kommentar
N10 G0 X0 Y0	
N20 MCALL L70	; Modaler Unterprogrammaufruf.
N30 X10 Y10	; Die programmierte Position wird angefahren und anschließend das Unterprogramm L70 abgearbeitet.
N40 X50 Y50	; Die programmierte Position wird angefahren und anschließend das Unterprogramm L70 abgearbeitet.



Beispiel 2:


Programmcode
N10 G0 X0 Y0
N20 MCALL L70
N30 L80

In diesem Beispiel stehen die nachfolgenden NC-Sätze mit programmierten Bahnachsen in Unterprogramm L80. L70 wird durch L80 aufgerufen.

1.25.3.5 Indirekter Unterprogrammaufruf (CALL)

Funktion

In Abhängigkeit von den gegebenen Bedingungen können an einer Stelle unterschiedliche Unterprogramme aufgerufen werden. Hierzu wird der Name des Unterprogramms in einer Variablen vom Typ STRING hinterlegt. Der Unterprogrammaufruf erfolgt mit `CALL` und dem Variablennamen.

 **VORSICHT**

Der indirekte Unterprogrammaufruf ist nur für Unterprogramme ohne Parameterübergabe möglich. Für den direkten Aufruf eines Unterprogramms hinterlegen Sie den Namen in einer STRING-Konstanten.

Syntax

`CALL <Programmname>`

Bedeutung

`CALL:` Befehl für den indirekten Unterprogrammaufruf
`<Programmname>:` Name des Unterprogramms (Variable oder Konstante)
 Typ: STRING

Beispiel

Direkter Aufruf mit STRING-Konstante:

Programmcode	Kommentar
...	
<code>CALL "/_N_WKS_DIR/_N_SUBPROG_WPD/_N_TEIL1_SPF"</code>	<code>; Unterprogramm TEIL1 mit CALL direkt aufrufen.</code>
...	

Indirekter Aufruf über Variable:

Programmcode	Kommentar
...	
<code>DEF STRING[100] PROGNAME</code>	<code>; Variable definieren.</code>
<code>PROGNAME="/_N_WKS_DIR/_N_SUBPROG_WPD/_N_TEIL1_SPF"</code>	<code>; Unterprogramm TEIL1 der Variablen PROGNAME zuordnen.</code>
<code>CALL PROGNAME</code>	<code>; Unterprogramm TEIL1 über CALL und die Variable PROGNAME indirekt aufrufen.</code>
...	

1.25.3.6 Indirekter Unterprogrammaufruf mit Angabe des auszuführenden Programmteils (CALL BLOCK ... TO ...)

Funktion

Mit `CALL` und der Schlüsselwortkombination `BLOCK ... TO` wird ein Unterprogramm indirekt aufgerufen und der mit Start- und Endmarke gekennzeichnete Programmteil ausgeführt.

Syntax

```
CALL <Programmname> BLOCK <Startmarke> TO <Endmarke>  
CALL BLOCK <Startmarke> TO <Endmarke>
```

Bedeutung

<code>CALL:</code>	Befehl für den indirekten Unterprogrammaufruf
<code><Programmname>:</code>	Name des Unterprogramms (Variable oder Konstante), das den zu bearbeitenden Programmteil enthält (Angabe optional). Typ: <code>STRING</code>
	Hinweis: Ist kein <code><Programmname></code> programmiert, wird der mit <code><Startmarke></code> und <code><Endmarke></code> gekennzeichnete Programmteil im aktuellen Programm gesucht und ausgeführt.
<code>BLOCK ... TO ... :</code>	Schlüsselwortkombination für indirekte Programmteilausführung
<code><Startmarke>:</code>	Variable, die auf den Beginn des zu bearbeitenden Programmteils verweist. Typ: <code>STRING</code>
<code><Endmarke>:</code>	Variable, die auf das Ende des zu bearbeitenden Programmteils verweist. Typ: <code>STRING</code>

Beispiel

Hauptprogramm:

Programmcode	Kommentar
...	
DEF STRING[20] STARTLABEL, ENDLABEL	; Variablendefinition für die Start- und Endmarke.
STARTLABEL="LABEL_1"	
ENDLABEL="LABEL_2"	
...	
CALL "CONTUR_1" BLOCK STARTLABEL TO ENDLABEL	; Indirekter Unterprogrammaufruf und Kennzeichnung des auszuführenden Programmteils.
...	

Unterprogramm:

Programmcode	Kommentar
PROC CONTUR_1 ...	
LABEL_1	; Startmarke: Beginn der Programmteilausführung
N1000 G1 ...	
...	
LABEL_2	; Endmarke: Ende der Programmteilausführung
...	

1.25.3.7 Indirekter Aufruf eines in ISO-Sprache programmierten Programms (ISOCALL)

Funktion

Mit dem indirekten Programmaufruf `ISOCALL` kann ein in einer ISO-Sprache programmiertes Programm aufgerufen werden. Dabei wird der in den Maschinendaten eingestellte ISO-Modus aktiviert. Am Programmende wird wieder der ursprüngliche Bearbeitungsmodus wirksam. Ist in den Maschinendaten kein ISO-Modus eingestellt, erfolgt der Aufruf des Unterprogramms im Siemens-Modus.

Weitere Informationen zum ISO-Modus siehe:

Literatur:

Funktionsbeschreibung ISO-Dialekte

Syntax

`ISOCALL <Programmname>`

Bedeutung

<code>ISOCALL:</code>	Schlüsselwort für indirekten Unterprogrammaufruf, mit dem der in den Maschinendaten eingestellte ISO-Modus aktiviert wird
<code><Programmname>:</code>	Name des in einer ISO-Sprache programmierten Programms (Variable oder Konstante vom Typ STRING)

Beispiel: Kontur mit Zyklenprogrammierung aus dem ISO-Modus heraus aufrufen

Programmcode	Kommentar
0122_SPF	; Konturbeschreibung im ISO-Modus
N1010 G1 X10 Z20	
N1020 X30 R5	
N1030 Z50 C10	
N1040 X50	
N1050 M99	
N0010 DEF STRING[5] PROGNAME = "0122"	; Siemens-Teileprogramm (-Zyklus)
...	
N2000 R11 = \$AA_IW[X]	
N2010 ISOCALL PROGNAME	
N2020 R10 = R10+1	; Programm 0122.spf im ISO-Modus bearbeiten
...	
N2400 M30	

1.25.3.8 Unterprogramm mit Pfadangabe und Parametern aufrufen (PCALL)

Funktion

Mit `PCALL` können Unterprogramme mit absoluter Pfadangabe und Parameterübergabe aufgerufen werden.

Syntax

```
PCALL <Pfad/Programmname> (<Parameter 1>, ..., <Parameter n>)
```

Bedeutung

<code>PCALL:</code>	Schlüsselwort für Unterprogrammaufruf mit absoluter Pfadangabe.
<code><Pfad/Programmname>:</code>	Absolute Pfadangabe beginnend mit "/", einschließlich Unterprogrammnamen. Wurde kein absoluter Pfad angegeben, verhält sich <code>PCALL</code> wie ein Standard-Unterprogrammaufruf mit Programmbezeichner. Der Programmbezeichner wird ohne Vorspann <code>_N_</code> und ohne Erweiterung angegeben. Soll der Programmname mit Vorspann und Erweiterung programmiert werden, so muss er explizit mit Vorspann und Erweiterung mit dem Befehl <code>EXTERN</code> erklärt werden.
<code><Parameter 1>, ...:</code>	Aktual-Parameter entsprechend der <code>PROC</code> -Anweisung des Unterprogramms.

Beispiel

Programmcode

```
PCALL/_N_WKS_DIR/_N_WELLE_WPD/WELLE(parameter1,parameter2,...)
```

1.25.3.9 Suchpfad bei Unterprogrammaufrufen erweitern (CALLPATH)

Funktion

Mit dem Befehl `CALLPATH` kann der Suchpfad für Unterprogrammaufrufe erweitert werden.

Damit können auch Unterprogramme aus einem nicht ausgewählten Werkstückverzeichnis aufgerufen werden, ohne den vollständigen, absoluten Pfadnamen des Unterprogramms anzugeben.

Die Suchpfaderweiterung erfolgt vor dem Eintrag für Anwenderzyklen (`_N_CUS_DIR`).

Durch folgende Ereignisse wird die Suchpfaderweiterung wieder abgewählt:

- `CALLPATH` mit Leerzeichen
- `CALLPATH` ohne Parameter
- Teileprogrammende
- Reset

Syntax

```
CALLPATH ("<Pfadname>")
```

Bedeutung

<code>CALLPATH:</code>	Schlüsselwort für die programmierbare Suchpfaderweiterung. Wird in einer eigenen Teileprogrammzeile programmiert.
<code><Pfadname>:</code>	Konstante oder Variable vom Typ <code>STRING</code> . Enthält die absolute Pfadangabe eines Verzeichnisses, um das der Suchpfad erweitert werden soll. Die Pfadangabe beginnt mit <code>"/</code> . Der Pfad muss vollständig mit Präfixen und Suffixen angegeben werden. Die maximale Pfadlänge beträgt 128 Bytes. Enthält der <code><Pfadname></code> ein Leerzeichen oder wird <code>CALLPATH</code> ohne Parameter aufgerufen, wird die Suchpfadanweisung wieder zurückgesetzt.

Beispiel

Programmcode

<pre>CALLPATH ("/_N_WKS_DIR/_N_MYWPD_WPD")</pre>
--

Damit wird folgender Suchpfad eingestellt (Position 5. ist neu):

1. Aktuelles Directory/unterprogrammbezeichner
2. Aktuelles Directory/unterprogrammbezeichner_SPF
3. Aktuelles Directory/unterprogrammbezeichner_MPF
4. /_N_SPF_DIR/unterprogrammbezeichner_SPF
5. **/_N_WKS_DIR/_N_MYWPD/unterprogrammbezeichner_SPF**
6. /N_CUS_DIR/_N_MYWPD/unterprogrammbezeichner_SPF
7. /_N_CMA_DIR/unterprogrammbezeichner_SPF
8. /_N_CST_DIR/unterprogrammbezeichner_SPF

Randbedingungen

- `CALLPATH` prüft, ob der programmierte Pfadname tatsächlich vorhanden ist. Im Fehlerfall wird die Teileprogrammbearbeitung mit Korrektursatz-Alarm 14009 abgebrochen.
- `CALLPATH` kann auch in INI-Dateien programmiert werden. Er wirkt dann für die Bearbeitungsdauer der INI-Datei (WPD-INI-Datei oder Initialisierungsprogramm für NC-aktive Daten, z. B. Frames im 1. Kanal `_N_CH1_UFR_INI`). Danach wird der Suchpfad wieder zurückgesetzt.

1.25.3.10 Externes Unterprogramm abarbeiten (840D sl) (EXTCALL)

Funktion

Mit dem Befehl `EXTCALL` kann ein Teileprogramm von einem externen Speicher nachgeladen und abgearbeitet werden.

Als externer Speicher stehen zur Verfügung:

- Lokales Laufwerk
- Netzlaufwerk
- USB-Laufwerk

Hinweis

USB-Laufwerk

Als Schnittstelle zum Abarbeiten eines auf einem USB-Laufwerk befindlichen externen Programms dürfen nur die USB-Schnittstellen an der Bedientafelfront bzw. TCU verwendet werden.

ACHTUNG

Kein USB-FlashDrive verwenden

Es wird empfohlen zum Abarbeiten eines externen Unterprogramms kein USB-FlashDrive zu verwenden. Ein Kommunikationsabbruch zum USB-FlashDrive während der Abarbearbeitung des Teileprogramms durch Kontaktschwierigkeiten, Herausfallen, Abbrechen durch Anstoßen oder versehentliches Abziehen, führt zum sofortigen Stop der Bearbeitung. Werkzeug und/oder Werkstück könnten dabei beschädigt werden.

Voreinstellung des externen Programmpfades

Der Pfad zum externen Programmverzeichnis kann voreingestellt mit dem Settingdatum:

SD42700 \$SC_EXT_PROG_PATH

Zusammen mit dem beim `EXTCALL`-Aufruf angegebenen Programmpfad und -bezeichner ergibt sich daraus der Gesamtpfad des aufzurufenden Teileprogramms.

Hinweis

Sprungziel

Bei externen Programmen die Spunganweisungen enthalten (`GOTOF`, `GOTOB`, `CASE`, `FOR`, `LOOP`, `WHILE`, `REPEAT`, `IF`, `ELSE`, `ENDIF` etc.) müssen die Sprungziele innerhalb des Nachladespeichers liegen. Die Größe des Nachladespeichers wird eingestellt über:

MD18360 MM_EXT_PROG_BUFFER_SIZE

Parameter

Beim Aufruf eines externen Programms können diesem keine Parameter übergeben werden.

Syntax

```
EXTCALL ("<Pfad>:<Programmname>")
```

Bedeutung

EXTCALL: Befehl zum Aufrufen eines externen Unterprogramms

"<Pfad><Programmname>": Konstante/Variable vom Typ STRING

<Pfad>: Absolute oder relative Pfadangabe (**optional**)

<Programmname>: Der Programmname wird ohne Präfix "_N_" angegeben.

Die Dateierweiterung ("MPF", "SPF") kann mit dem Zeichen "_" oder "." am Programmnamen angefügt werden (**optional**).

Beispiel:

```
"WELLE"  
"WELLE_SPF"  
"WELLE.SPF"
```

Pfadangabe: Kurzbezeichnungen

Folgende Kurzbezeichnungen können bei der Pfadangabe verwendet werden:

- Lokales Laufwerk: "LOCAL_DRIVE:"
- CF-Karte: "CF_CARD:"
- USB-Laufwerk (Bedientafelfront): "USB:"

Die Kurzbezeichnungen "CF_CARD:" und "LOCAL_DRIVE:" sind alternativ verwendbar.

Beispiel

Abarbeiten von lokalem Laufwerk

Das Hauptprogramm "MAIN.MPF" befindet sich im NC-Speicher und ist zur Abarbeitung ausgewählt.

Unterprogramm "SP_1"

Das externe Unterprogramm "SP_1.SPF" bzw. "SP_1.MPF" befindet sich auf dem lokalen Laufwerk in dem Verzeichnis "/user/sinumerik/data/prog/WKS.DIR/WST1.WPD".

Der Pfad zum externen Programmverzeichnis ist einzustellen mit:

```
SD42700 $SC_EXT_PROG_PATH = LOCAL_DRIVE:WKS.DIR/WST1.WPD
```

Hinweis

Pfadangabe für den Aufruf des externen Unterprogramms:

- Ohne Verwendung der Voreinstellung: "LOCAL_DRIVE:WKS.DIR/WST1.WPD/SP_1"
 - Mit Verwendung der Voreinstellung: "SP_1"
-

Unterprogramm "SP_2"

Das externe Unterprogramm "SP_2.SPF" bzw. "SP_2.MPF" befindet sich im Verzeichnis WKS.DIR/WST1.WPD des USB-Laufwerks. Die Voreinstellung des Pfads zum externen Programmverzeichnis wird für den Pfad des Unterprogramms "SP_1" verwendet und wird im Hauptprogramm auch nicht umgeschrieben. Daher muss beim Aufruf des Unterprogramms "SP_2" der vollständige Pfad angegeben werden.

Hauptprogramm "MAIN"

Programmcode
N010 PROC MAIN
N020 ...
N030 EXTCALL("SP_1")
N030 EXTCALL("USB:WKS.DIR/WST1.WPD/SP_2")
N050 ...
N060 M30

Weitere Informationen

EXTCALL-Aufruf mit absoluter Pfadangabe

Ist das Unterprogramm unter dem angegebenen Pfad vorhanden, wird es mit dem `EXTCALL`-Aufruf ausgeführt. Ist das Unterprogramm unter dem angegebenen Pfad nicht vorhanden, wird die Programmausführung mit dem `EXTCALL`-Aufruf abgebrochen.

EXTCALL-Aufruf mit relativer Pfadangabe / ohne Pfadangabe

Bei einem `EXTCALL`-Aufruf mit relativer Pfadangabe bzw. ohne Pfadangabe werden die vorhandenen Programmspeicher nach folgendem Muster durchsucht:

1. Ist in SD42700 `$SC_EXT_PROG_PATH` eine Pfadangabe voreingestellt, wird zuerst ausgehend von diesem Pfad nach der Angabe im `EXTCALL`-Aufruf (Programmname ggf. mit relativer Pfadangabe) gesucht. Der absolute Pfad ergibt sich dann durch Zeichenverkettung aus:
 - Voreingestellter Pfadangabe im SD42700 `$SC_EXT_PROG_PATH`
 - Trennzeichen "/"
 - Pfadangabe und Unterprogrammname im Befehl `EXTCALL`
2. Wurde das Unterprogramm unter 1. nicht gefunden, werden die Verzeichnisse des Anwenderspeichers durchsucht.

Die Suche endet, wenn das Unterprogramm erstmalig gefunden wurde. Wird das Unterprogramm nicht gefunden, wird die Programmausführung mit dem `EXTCALL`-Aufruf abgebrochen.

Einstellbarer Nachladespeicher (FIFO-Puffer)

Für das Abarbeiten eines externen Unterprogramms wird ein Nachladespeicher benötigt. Die Größe des Nachladespeichers ist mit 30 kByte voreingestellt und kann nur vom Maschinenhersteller verändert werden.

Für parallel abgearbeitete externe Unterprogramme wird jeweils ein eigener Nachladespeicher benötigt.

Reset / Programmende / POWER ON

Durch Reset und POWER ON werden externe Unterprogrammaufrufe abgebrochen und die jeweiligen Nachladespeicher gelöscht.

Ein für "Abarbeiten von Extern" angewähltes Unterprogramm bleibt über Reset / Programmende hinaus weiter für "Abarbeiten von Extern" angewählt. Durch POWER ON geht die Anwahl verloren.

Literatur

Weitere Informationen zu "Abarbeiten von Extern" finden sich in:

Funktionshandbuch Grundfunktionen; BAG, Kanal, Programmbetrieb, Reset-Verhalten (K1)

1.25.3.11 Externes Unterprogramm abarbeiten (828D) (EXTCALL)

Funktion

Mit dem Befehl `EXTCALL` kann ein Teileprogramm von einem externen Speicher nachgeladen und abgearbeitet werden.

Als externer Speicher stehen zur Verfügung:

- Anwender CF-Karte
- Netzlaufwerk
- USB-Laufwerk

Hinweis

USB-Laufwerk

Als Schnittstelle zum Abarbeiten eines auf einem USB-Laufwerk befindlichen externen Programms darf nur die USB-Schnittstelle der Bedientafelfront (PPU) verwendet werden.

ACHTUNG

Kein USB-FlashDrive verwenden

Es wird empfohlen zum Abarbeiten eines externen Unterprogramms kein USB-FlashDrive zu verwenden. Ein Kommunikationsabbruch zum USB-FlashDrive während der Abarbeitung des Teileprogramms durch Kontaktschwierigkeiten, Herausfallen, Abbrechen durch Anstoßen oder versehentliches Abziehen, führt zum sofortigen Stop der Bearbeitung. Werkzeug und/oder Werkstück könnten dabei beschädigt werden.

Voreinstellung des externen Programmpfades

Der Pfad zum externen Programmverzeichnis kann voreingestellt mit dem Settingdatum:

SD42700 \$SC_EXT_PROG_PATH

Zusammen mit dem beim `EXTCALL`-Aufruf angegebenen Programmpfad und -bezeichner ergibt sich daraus der Gesamtpfad des aufzurufenden Teileprogramms.

Hinweis

Sprungziel

Bei externen Programmen, die Sprunganweisungen enthalten (`GOTO`, `GOTOB`, `CASE`, `FOR`, `LOOP`, `WHILE`, `REPEAT`, `IF`, `ELSE`, `ENDIF` etc.), müssen die Sprungziele innerhalb des Nachladespeichers liegen.

Parameter

Beim Aufruf eines externen Programms können diesem keine Parameter übergeben werden.

Syntax

```
EXTCALL ("<Pfad>:<Programmname>")
```

Bedeutung

EXTCALL:	Befehl zum Aufrufen eines externen Unterprogramms
"<Pfad><Programmname>":	Konstante / Variable vom Typ STRING
<Pfad>:	Absolute oder relative Pfadangabe (optional)
<Programmname>:	Der Programmname wird ohne Präfix "_N_" angegeben. Die Dateierweiterung ("MPF", "SPF") kann mit dem Zeichen "_" oder "." am Programmnamen angefügt werden (optional). Beispiel: "WELLE" "WELLE.SPF" "WELLE.SP_1"

Pfadangabe: Kurzbezeichnungen

Folgende Kurzbezeichnungen können bei der Pfadangabe verwendet werden:

- Anwender CF-Karte: "CF_CARD:"
- USB-Laufwerk (Bedientafelfront): "USB:"

Beispiel

Das Hauptprogramm "MAIN.MPF" befindet sich im NC-Speicher und ist zur Abarbeitung angewählt.

Unterprogramm "SP_1"

Das externe Unterprogramm "SP_1.SPF" bzw. "SP_1.MPF" befindet sich auf der Anwender CF-Karte im Verzeichnis "WKS.DIR/WST1.WPD".

Der Pfad zum externen Programmverzeichnis ist einzustellen mit:

```
SD42700 $SC_EXT_PROG_PATH = CF_CARD:WKS.DIR/WST1.WPD
```

Hinweis

Pfadangabe für den Aufruf des externen Unterprogramms:

- Ohne Verwendung der Voreinstellung: "CF_CARD:WKS.DIR/WST1.WPD/SP_1"
 - Mit Verwendung der Voreinstellung: "SP_1"
-

Unterprogramm "SP_2"

Das externe Unterprogramm "SP_2.SPF" bzw. "SP_2.MPF" befindet sich im Verzeichnis WKS.DIR/WST1.WPD des USB-Laufwerks. Die Voreinstellung des Pfads zum externen Programmverzeichnis wird für den Pfad des Unterprogramms "SP_1" verwendet und wird im Hauptprogramm auch nicht umgeschrieben. Daher muss beim Aufruf des Unterprogramms "SP_2" der vollständige Pfad angegeben werden.

Hauptprogramm "MAIN"

Programmcode
N010 PROC MAIN
N020 ...
N030 EXTCALL ("SP_1")
N030 EXTCALL ("USB:WKS.DIR/WST1.WPD/SP_2")
N050 ...
N060 M30

Weitere Informationen

EXTCALL-Aufruf mit absoluter Pfadangabe

Ist das Unterprogramm unter dem angegebenen Pfad vorhanden, wird es mit dem `EXTCALL`-Aufruf ausgeführt. Ist das Unterprogramm unter dem angegebenen Pfad nicht vorhanden, wird die Programmausführung mit dem `EXTCALL`-Aufruf abgebrochen.

EXTCALL-Aufruf mit relativer Pfadangabe / ohne Pfadangabe

Bei einem `EXTCALL`-Aufruf mit relativer Pfadangabe bzw. ohne Pfadangabe werden die vorhandenen Programmspeicher nach folgendem Muster durchsucht:

1. Ist in SD42700 `$SC_EXT_PROG_PATH` eine Pfadangabe voreingestellt, wird zuerst ausgehend von diesem Pfad nach der Angabe im `EXTCALL`-Aufruf (Programmname ggf. mit relativer Pfadangabe) gesucht. Der absolute Pfad ergibt sich dann durch Zeichenverkettung aus:
 - Voreingestellter Pfadangabe im SD42700 `$SC_EXT_PROG_PATH`
 - Trennzeichen "/"
 - Pfadangabe und Unterprogrammname im Befehl `EXTCALL`
2. Wurde das Unterprogramm unter 1. nicht gefunden, werden die Verzeichnisse des Anwenderspeichers durchsucht.

Die Suche endet, wenn das Unterprogramm erstmalig gefunden wurde. Wird das Unterprogramm nicht gefunden, wird die Programmausführung mit dem `EXTCALL`-Aufruf abgebrochen.

Einstellbarer Nachladespeicher (FIFO-Puffer)

Für das Abarbeiten eines externen Unterprogramms wird ein Nachladespeicher benötigt. Die Größe des Nachladespeichers ist mit 30 kByte voreingestellt und kann nur vom Maschinenhersteller verändert werden.

Für parallel abgearbeitete externe Unterprogramme wird jeweils ein eigener Nachladespeicher benötigt.

Reset / Programmende / POWER ON

Durch Reset und POWER ON werden externe Unterprogrammaufrufe abgebrochen und die jeweiligen Nachladespeicher gelöscht.

Ein für "Abarbeiten von Extern" angewähltes Unterprogramm bleibt über Reset / Programmende hinaus weiter für "Abarbeiten von Extern" angewählt. Durch POWER ON geht die Anwahl verloren.

Literatur

Weitere Informationen zu "Abarbeiten von Extern" finden sich in:

Funktionshandbuch Grundfunktionen; BAG, Kanal, Programmbetrieb, Reset-Verhalten (K1)

1.26 Makrotechnik (DEFINE ... AS)

 **VORSICHT**

Mit Makrotechnik kann die Programmiersprache der Steuerung stark verändert werden!
Setzen Sie deshalb die Makrotechnik mit großer Sorgfalt ein!

Funktion

Als Makro bezeichnet man die Zusammenfassung von einzelnen Anweisungen zu einer neuen Gesamtanweisung mit eigenem Namen. Auch G-, M- und H-Funktionen oder L-Unterprogrammnamen können als Makros angelegt werden. Bei Aufruf des Makros im Programmablauf werden die unter dem Makronamen programmierten Anweisungen nacheinander abgearbeitet.

Anwendung

Anweisungsfolgen, die sich wiederholen, programmiert man nur einmal als Makro in einem eigenen Makrobaustein (Makrodatei) oder einmal am Programmanfang. Das Makro kann dann in jedem beliebigen Haupt- oder Unterprogramm aufgerufen und abgearbeitet werden.

Aktivierung

Um die Makros einer Makrodatei im NC-Programm verwenden zu können, muss die Makrodatei in die NC geladen werden.

Syntax

Makro-Definition:

```
DEFINE <Makroname> AS <Anweisung 1> <Anweisung 2> ...
```

Aufruf im NC-Programm:


```
<Makroname>
```

Bedeutung

DEFINE ... AS :	Schlüsselwort-Kombination zur Definition eines Makros
<Makroname>:	Name des Makros Als Makronamen sind nur Bezeichner zulässig. Mit dem Makronamen wird das Makro aus dem NC-Programm heraus aufgerufen.
<Anweisung>:	Programmieranweisung, die im Makro enthalten sein soll.

Regeln zur Makro-Definition

- Im Makro können beliebige Bezeichner, G-, M-, H-Funktionen und L-Programmnamen definiert werden.
- Makros können auch im NC-Programm definiert werden.
- G-Funktions-Makros können nur steuerungsglobal im Makrobaustein definiert werden.
- H- und L-Funktionen sind 2-stellig programmierbar.
- M- und G-Funktionen können 3-stellig programmiert werden.

 VORSICHT
Schlüsselworte und reservierte Namen dürfen nicht mit Makros überdefiniert werden.

Randbedingungen

Eine Schachtelung von Makros ist nicht möglich.

Beispiele

Beispiel 1: Makrodefinition am Programmanfang

Programmcode	Kommentar
DEFINE LINIE AS G1 G94 F300	; Makro-Definition
...	
...	
N70 LINIE X10 Y20	; Makro-Aufruf
...	

Beispiel 2: Makrodefinitionen in einer Makrodatei

Programmcode	Kommentar
DEFINE M6 AS L6	; Beim Werkzeugwechsel wird ein Unterprogramm aufgerufen, das den nötigen Datentransfer übernimmt. Im Unterprogramm wird die eigentliche Werkzeugwechsel-M-Funktion ausgegeben (z. B. M106).
DEFINE G81 AS DRILL(81)	; Nachbildung der DIN-G-Funktion.
DEFINE G33 AS M333 G333	; Beim Gewindeschneiden wird Synchronisation mit der PLC angefordert. Die ursprüngliche G-Funktion G33 wurde per MD in G333 umbenannt, die Programmierung bleibt für den Anwender gleich.

Beispiel 3: Externe Makrodatei

Nach dem Einlesen der externen Makrodatei in die Steuerung muss die Makrodatei in die NC geladen werden. Erst dann können die Makros im NC-Programm verwendet werden.

Programmcode	Kommentar
%_N_UMAC_DEF	
;\$PATH=/_N_DEF_DIR	; Kundenspezifische Makros
DEFINE PI AS 3.14	
DEFINE TC1 AS M3 S1000	
DEFINE M13 AS M3 M7	; Spindel rechts, Kühlmittel ein
DEFINE M14 AS M4 M7	; Spindel links, Kühlmittel ein
DEFINE M15 AS M5 M9	; Spindel Halt, Kühlmittel aus
DEFINE M6 AS L6	; Aufruf des Werkzeugwechselprogramms
DEFINE G80 AS MCALL	; Abwahl Bohrzyklus
M30	

Datei- und Programmverwaltung

2.1 Programmspeicher

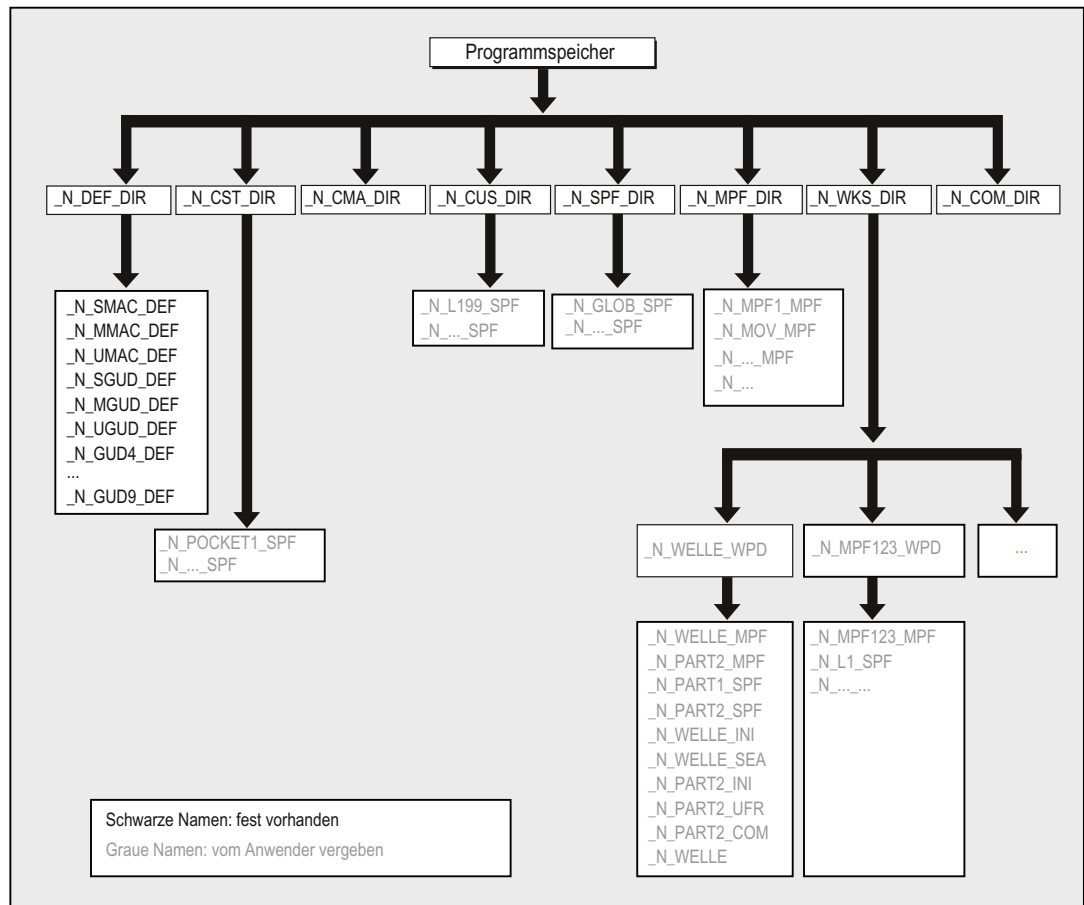
Funktion

Im Programmspeicher werden Dateien und Programme (z. B. Haupt- und Unterprogramme, Makro-Definitionen) persistent gespeichert (→ Passives Filesystem).

Literatur:

Funktionshandbuch Erweiterungsfunktionen; Speicherkonfiguration (S7)

Daneben gibt es eine Anzahl von Dateitypen, die hier zwischengespeichert werden können und bei Bedarf (z. B. bei Bearbeitung eines bestimmten Werkstückes) in den Arbeitsspeicher zu übertragen sind (z. B. für Initialisierungszwecke).



Standard-Verzeichnisse

Folgende Verzeichnisse sind standardmäßig vorhanden:

Verzeichnis	Inhalt
_N_DEF_DIR	Datenbausteine und Makrobausteine
_N_CST_DIR	Standard-Zyklen
_N_CMA_DIR	Hersteller-Zyklen
_N_CUS_DIR	Anwender-Zyklen
_N_WKS_DIR	Werkstücke
_N_SPF_DIR	Globale Unterprogramme
_N_MPF_DIR	Hauptprogramme
_N_COM_DIR	Kommentare

Dateitypen

Im Programmspeicher können folgende Dateitypen eingebracht werden:

Dateityp	Beschreibung
name_MPF	Hauptprogramm
name_SPF	Unterprogramm
name_TEA	Maschinendaten
name_SEA	Settingdaten
name_TOA	Werkzeugkorrekturen
name_UFR	Nullpunktverschiebungen/Frame
name_INI	Initialisierungsdatei
name_GUD	Globale Anwenderdaten
name_RPA	R-Parameter
name_COM	Kommentar
name_DEF	Definitionen für globale Anwenderdaten und Makros

Werkstück-Hauptverzeichnis (_N_WKS_DIR)

Das Werkstück-Hauptverzeichnis ist standardmäßig unter der Bezeichnung `_N_WKS_DIR` im Programmspeicher eingerichtet. Das Werkstück-Hauptverzeichnis enthält für alle Werkstücke, die Sie programmiert haben, die entsprechenden Werkstückverzeichnisse.

Werkstückverzeichnisse (..._WPD)

Für eine flexiblere Handhabung von Daten und Programmen können bestimmte Daten und Programme gebündelt oder in einzelnen Werkstückverzeichnissen abgelegt werden.

Ein Werkstückverzeichnis enthält alle Dateien, die zum Bearbeiten eines Werkstückes notwendig sind. Dies können Hauptprogramme, Unterprogramme, beliebige Initialisierungsprogramme und Kommentar-Dateien sein.

Initialisierungsprogramme werden nach der Programmanwahl mit dem ersten Teileprogrammstart einmalig ausgeführt (entsprechend Maschinendatum MD11280 \$MN_WPD_INI_MODE).

Beispiel:

Das Werkstückverzeichnis `_N_WELLE_WPD`, das für das Werkstück WELLE angelegt wurde, enthält folgende Dateien:

Datei	Beschreibung
<code>_N_WELLE_MPF</code>	Hauptprogramm
<code>_N_PART2_MPF</code>	Hauptprogramm
<code>_N_PART1_SPF</code>	Unterprogramm
<code>_N_PART2_SPF</code>	Unterprogramm
<code>_N_WELLE_INI</code>	Allgemeines Initialisierungsprogramm der Daten für das Werkstück
<code>_N_WELLE_SEA</code>	Initialisierungsprogramm Settingdaten
<code>_N_PART2_INI</code>	Allgemeines Initialisierungsprogramm der Daten für Programm Part 2
<code>_N_PART2_UFR</code>	Initialisierungsprogramm für Frame-Daten für Programm Part 2
<code>_N_WELLE_COM</code>	Kommentardatei

Werkstückverzeichnisse am externen PC anlegen

Die nachstehend beschriebene Vorgehensweise wird an einer externen Datenstation durchgeführt. Für die Datei- und Programmverwaltung (vom PC zur Steuerung), direkt an der Steuerung, finden Sie die Informationen in Ihrer Bedienungsanleitung.

Werkstückverzeichnis anlegen mit Pfadangabe (\$PATH=...)

In der zweiten Zeile einer Datei wird der Zielpfad mit `$PATH=...` angegeben. Die Datei wird dann unter dem angegebenen Pfad abgelegt.

Beispiel:

Programmcode
<code>%_N_WELLE_MPF</code>
<code>;\$PATH=/_N_WKS_DIR/_N_WELLE_WPD</code>
<code>N10 G0 X... Z...</code>
<code>...</code>
<code>M2</code>

Die Datei `_N_WELLE_MPF` wird im Verzeichnis `/_N_WKS_DIR/_N_WELLE_WPD` abgelegt.

Werkstückverzeichnis anlegen ohne Pfadangabe

Fehlt die Pfadangabe, so werden Dateien mit der Endung _SPF im Verzeichnis /_N_SPF_DIR, Dateien mit der Endung _INI im Arbeitsspeicher und alle übrigen Dateien im Verzeichnis /_N_MPF_DIR abgelegt.

Beispiel:

```
Programmcode
-----
%_N_WELLE_SPF
...
M17
```

Die Datei _N_WELLE_SPF wird im Verzeichnis /_N_SPF_DIR abgelegt.

Werkstück für die Bearbeitung anwählen

Ein Werkstückverzeichnis kann für die Abarbeitung in einem Kanal angewählt werden. Befindet sich in diesem Verzeichnis ein Hauptprogramm **gleichen Namens** oder nur ein einziges Hauptprogramm (_MPF), so wird dieses automatisch für die Abarbeitung angewählt.

Literatur:

Bedienhandbuch

Suchpfade beim Unterprogrammaufruf

Wird der Aufruf-Pfad nicht explizit im Teileprogramm beim Aufruf eines Unterprogramms (oder auch Initialisierungsdatei) angegeben, so wird das aufgerufene Programm nach einem festen Suchpfad ermittelt.

Unterprogrammaufruf mit absoluter Pfadangabe

Beispiel:

```
Programmcode
-----
...
CALL"/_N_CST_DIR/_N_CYCLE1_SPF"
...
```

Unterprogrammaufruf ohne absoluter Pfadangabe

In der Regel werden die Programme ohne Pfadangabe aufgerufen.

Beispiel:

```

Programmcode
...
CYCLE1
...

```

Die Verzeichnisse werden nach dem aufgerufenen Programm in der folgenden Reihenfolge durchsucht:

Nr.	Verzeichnis	Beschreibung
1	aktuelles Directory / <i>name</i>	Werkstück-Hauptverzeichnis oder Standard-Verzeichnis <i>_N_MPF_DIR</i>
2	aktuelles Directory / <i>name_SPF</i>	
3	aktuelles Directory / <i>name_MPF</i>	
4	<i>/_N_SPF_DIR / name_SPF</i>	Globale Unterprogramme
5	<i>/_N_CUS_DIR / name_SPF</i>	Anwender-Zyklen
6	<i>/_N_CMA_DIR / name_SPF</i>	Hersteller-Zyklen
7	<i>/_N_CST_DIR / name_SPF</i>	Standard-Zyklen

Suchpfade beim Unterprogrammaufruf programmieren (CALLPATH)

Der Suchpfad beim Unterprogrammaufruf kann mit dem Teileprogrammbehehl `CALLPATH` erweitert werden.

Beispiel:

```

Programmcode
CALLPATH ("/_N_WKS_DIR/_N_MYWPD_WPD")
...

```

Der Suchpfad wird vor Position 5 (Anwender-Zyklus) entsprechend der angegebenen Programmierung abgelegt.

Weitere Informationen zum programmierbaren Suchpfad bei Unterprogrammaufrufen mit `CALLPATH` siehe "Suchpfad bei Unterprogrammaufrufen erweitern (CALLPATH) (Seite 211)".

2.2 Arbeitsspeicher (CHANDATA, COMPLETE, INITIAL)

Funktion

Der Arbeitsspeicher enthält die aktuellen System- und Anwenderdaten, mit denen die Steuerung betrieben wird (aktives Filesystem), z. B.:

- Aktive Maschinendaten
- Werkzeugkorrekturdaten
- Nullpunktverschiebungen
- ...

Initialisierungsprogramme

Hierbei handelt es sich um Programme, mit denen die Daten des Arbeitsspeichers vorbesetzt (initialisiert) werden. Hierfür können folgende Dateitypen verwendet werden:

Dateityp	Beschreibung
name_TEA	Maschinendaten
name_SEA	Settingdaten
name_TOA	Werkzeugkorrekturen
name_UFR	Nullpunktverschiebungen/Frame
name_INI	Initialisierungsdatei
name_GUD	Globale Anwenderdaten
name_RPA	R-Parameter

Informationen zu allen Dateitypen finden Sie im Bedienhandbuch zur Bedienoberfläche.

Datenbereiche

Die Daten können in unterschiedliche Bereiche eingegliedert werden, in denen sie gelten sollen. Beispielsweise kann eine Steuerung über mehrere Kanäle verfügen oder gewöhnlich auch über mehrere Achsen.

Es gibt:

Kennung	Datenbereiche
NCK	NCK-spezifische Daten
CH<n>	Kanalspezifische Daten (<n> gibt die Kanalnummer an)
AX<n>	Achsspezifische Daten (<n> gibt die Nummer der Maschinenachse an)
TO	Werkzeugdaten
COMPLETE	Alle Daten

Initialisierungsprogramm am externen PC erzeugen

Mit Hilfe von Datenbereichskennung und Datentypenkennung können die Bereiche bestimmt werden, die bei der Datensicherung als Einheit betrachtet werden:

_N_AX5_TEA_INI	Maschinendaten für Achse 5
_N_CH2_UFR_INI	Frames des Kanals 2
_N_COMPLETE_TEA_INI	Alle Maschinendaten

Nach Inbetriebnahme der Steuerung ist ein Datensatz im Arbeitsspeicher vorhanden, der den ordnungsgemäßen Betrieb der Steuerung gewährleistet.

Vorgehensweise bei mehrkanaligen Steuerungen (CHANDATA)

CHANDATA (<Kanalnummer>) für mehrere Kanäle ist nur in der Datei _N_INITIAL_INI zulässig. Das ist die Inbetriebnahmedatei, mit der alle Daten der Steuerung initialisiert werden.

Programmcode	Kommentar
%_N_INITIAL_INI	
CHANDATA (1)	
	; Maschinenachszuordnung Kanal 1:
\$MC_AXCONF_MACHAX_USED [0] =1	
\$MC_AXCONF_MACHAX_USED [1] =2	
\$MC_AXCONF_MACHAX_USED [2] =3	
CHANDATA (2)	
	; Maschinenachszuordnung Kanal 2:
\$MC_AXCONF_MACHAX_USED [0] =4	
\$MC_AXCONF_MACHAX_USED [1] =5	
CHANDATA (1)	
	; Axiale Maschinendaten:
	; Genauhaltfenster grob:
\$MA_STOP_LIMIT_COARSE [AX1] =0.2	; Achse 1
\$MA_STOP_LIMIT_COARSE [AX2] =0.2	; Achse 2
	; Genauhaltfenster fein:
\$MA_STOP_LIMIT_FINE [AX1] =0.01	; Achse 1
\$MA_STOP_LIMIT_FINE [AX1] =0.01	; Achse 2

 **VORSICHT**

CHANDATA-Anweisung

Im Teileprogramm darf die CHANDATA-Anweisung nur für den Kanal gesetzt werden, auf dem das NC-Programm abgearbeitet wird, d. h. die Anweisung kann dazu benutzt werden, NC-Programme davor zu schützen, dass sie auf einem nicht vorgesehenen Kanal abgearbeitet werden.

Im Fehlerfall wird die Programmabarbeitung abgebrochen.

Hinweis

INI-Dateien in Joblisten enthalten keine CHANDATA-Anweisungen.

Initialisierungsprogramme sichern (COMPLETE, INITIAL)

Die Dateien des Arbeitsspeichers können auf einem externen PC gesichert und von dort wieder eingelesen werden.

- Die Dateien werden mit COMPLETE gesichert.
- Mit INITIAL wird über alle Bereiche eine INI-Datei (_N_INITIAL_INI) erzeugt.

Initialisierungsprogramme einlesen

ACHTUNG

Wird die Datei mit dem Namen "INITIAL_INI" eingelesen, so werden alle Daten, die in der Datei nicht versorgt werden, mit Standarddaten initialisiert. Ausgenommen davon sind nur die Maschinendaten. Es werden also **Settingdaten, Werkzeugdaten, NPV, GUD-Werte, ...** mit Standarddaten (normalerweise "NULL") versorgt.

Zum Einlesen von einzelnen Maschinendaten eignet sich z. B. die Datei COMPLETE_TEA_INI. In dieser Datei erwartet die Steuerung nur Maschinendaten. Damit bleiben die anderen Datenbereiche in diesem Fall unberührt.

Initialisierungsprogramme laden

Die INI-Programme können auch als Teileprogramme angewählt und aufgerufen werden, wenn sie nur Daten eines Kanals verwenden. So ist es auch möglich, programmgesteuerte Daten zu initialisieren.

Schutzbereiche

3.1 Festlegung der Schutzbereiche (CPROTDEF, NPROTDEF)

Funktion

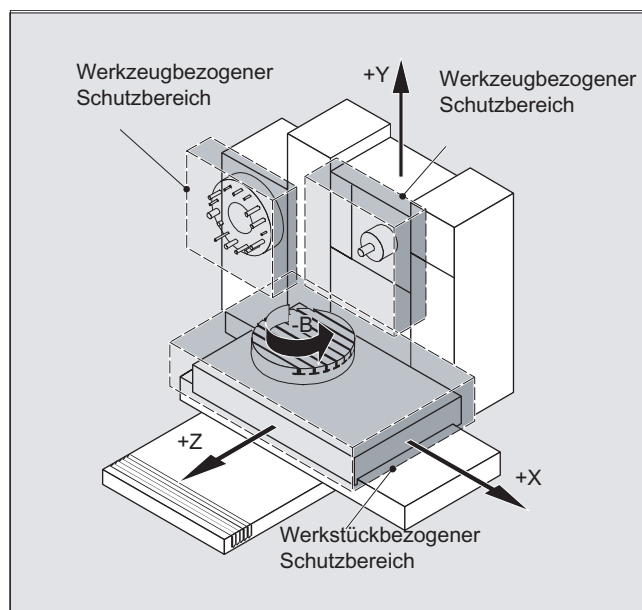
Mit Hilfe von Schutzbereichen lassen sich verschiedene Elemente an der Maschine, die Ausrüstung sowie das Werkstück vor falschen Bewegungen schützen.

Werkzeugbezogene Schutzbereiche:

Für Teile, die zum Werkzeug gehören (z. B. Werkzeug, Werkzeugträger).

Werkstückbezogene Schutzbereiche:

Für Teile, die zum Werkstück gehören (z. B. Teile des Werkstücks, Aufspanntisch, Spannpratzen, Spindelfutter, Reitstock).



Syntax

```
DEF INT NOT_USED
G17/G18/G19
CPROTDEF/NPROTDEF (<n>, <t>, <applim>, <applus>, <appminus>)
G0/G1/... X/Y/Z...
...
EXECUTE (NOT_USED)
```

Bedeutung

DEF INT NOT_USED:	Lokale Variable vom Datentyp INTEGER definieren
G17/G18/G19:	Die gewünschte Ebene wird vor CPROTDEF bzw. NPROTDEF mit G17/G18/G19 angewählt und darf vor EXECUTE nicht geändert werden. Eine Programmierung der Applikate zwischen CPROTDEF bzw. NPROTDEF und EXECUTE ist nicht zulässig.
CPROTDEF:	Kanalspezifische Schutzbereiche definieren
NPROTDEF:	Maschinenspezifische Schutzbereiche definieren
G0/G1/... X/Y/Z... :	Die Kontur der Schutzbereiche wird mit maximal 11 Verfahrbewegungen in der angewählten Ebene angegeben. Dabei ist die erste Verfahrbewegung die Bewegung an die Kontur. Als Schutzbereich gilt dabei der Bereich links von der Kontur.
	Hinweis: Die zwischen CPROTDEF bzw. NPROTDEF und EXECUTE stehenden Verfahrbewegungen werden nicht ausgeführt, sondern definieren den Schutzbereich.
EXECUTE:	Definition beenden
<n>:	Nummer des definierten Schutzbereichs
<t>:	Typ des Schutzbereichs
	TRUE: Werkzeugbezogener Schutzbereich
	FALSE: Werkstückbezogener Schutzbereich
<aplim>:	Art der Begrenzung in der 3. Dimension
	0: keine Begrenzung
	1: Begrenzung in Plus-Richtung
	2: Begrenzung in Minus-Richtung
	3: Begrenzung in Plus- und Minus-Richtung
<applus>:	Wert der Begrenzung in Plus-Richtung der 3. Dimension
<appminus>:	Wert der Begrenzung in Minus-Richtung der 3. Dimension
NOT_USED:	Die Fehlervariable ist bei Schutzbereichen mit EXECUTE wirkungslos

Randbedingungen

Während der Definition der Schutzbereiche darf:

- keine Fräserradius- bzw. Schneidenradiuskorrektur aktiv sein.
- keine Transformation aktiv sein.
- kein Frame aktiv sein.

Es darf auch nicht Referenzpunktfahren (G74), Festpunktfahren (G75), Satzvorlauf-Stopp oder Programmende programmiert sein.

Weitere Informationen

Definition von Schutzbereichen

Zur Definition von Schutzbereichen gehören:

- CPROTDEF für kanalspezifische Schutzbereiche
- NPROTDEF für maschinenspezifische Schutzbereiche
- Konturbeschreibung des Schutzbereichs
- Abschluss der Definition mit EXECUTE

Bei Aktivierung des Schutzbereichs im NC-Teilprogramm können Sie den Bezugspunkt des Schutzbereichs relativ verschieben.

Bezugspunkt der Konturbeschreibung

Die werkstückbezogenen Schutzbereiche werden im Basiskoordinatensystem definiert.

Die werkzeugbezogenen Schutzbereiche werden bezogen auf den Werkzeugträgerbezugspunkt F angegeben.

Zulässige Konturelemente

Für die Konturbeschreibung des Schutzbereichs sind zulässig:

- G0, G1 für gerade Konturelemente
- G2 für Kreisabschnitte im Uhrzeigersinn (nur für werkstückbezogene Schutzbereiche)
- G3 für Kreisabschnitte gegen den Uhrzeigersinn

Hinweis

Soll ein Vollkreis den Schutzbereich beschreiben, so ist er in zwei Teilkreise aufzuteilen. Die Folge G2, G3 bzw. G3, G2 ist nicht zulässig. Hier ist ggf. ein kurzer G1-Satz einzuschieben.

Der letzte Punkt der Konturbeschreibung muss mit dem ersten Punkt zusammenfallen.

Außenschutzbereiche

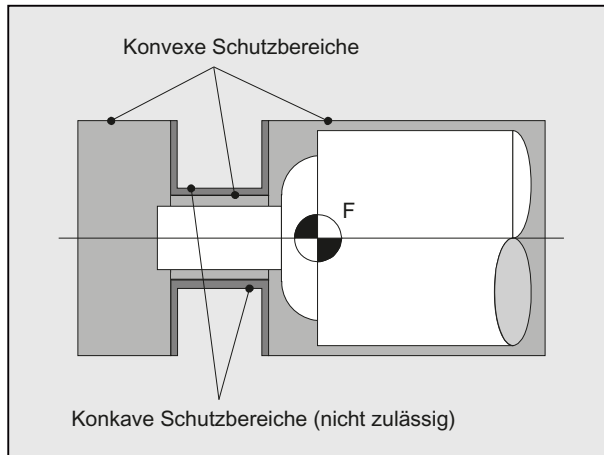
Außenschutzbereiche (nur bei werkstückbezogenen Schutzbereichen möglich) sind im Uhrzeigersinn zu definieren.

Rotationssymmetrische Schutzbereiche

Bei rotationssymmetrischen Schutzbereichen (z. B. Spindelfutter) muss die Gesamtkontur beschrieben werden (nicht nur bis zur Drehmitte!).

Werkzeugbezogene Schutzbereiche

Werkzeugbezogene Schutzbereiche müssen immer konvex sein. Falls ein konkaver Schutzbereich gewünscht ist, ist dieser in mehrere konvexe Schutzbereiche zu zerlegen.



3.2 Schutzbereiche aktivieren/deaktivieren (CPROT, NPROT)

Funktion

Vorher definierte Schutzbereiche zur Kollisionsüberwachung aktivieren, voraktivieren oder aktive Schutzbereiche deaktivieren.

Die maximale Anzahl der gleichzeitig in einem Kanal aktiven Schutzbereiche wird über Maschinendatum festgelegt.

Ist kein werkzeugbezogener Schutzbereich aktiv, so wird die Werkzeugbahn gegen die werkstückbezogenen Schutzbereiche geprüft.

Hinweis

Ist kein werkstückbezogener Schutzbereich aktiv, so findet keine Schutzbereichsüberwachung statt.

Syntax

```
CPROT (<n>, <state>, <xMov>, <yMov>, <zMov>)
NPROT (<n>, <state>, <xMov>, <yMov>, <zMov>)
```

Bedeutung

CPROT:	Aufruf kanalspezifischer Schutzbereich
NPROT:	Aufruf maschinenspezifischer Schutzbereich
<n>:	Nummer des Schutzbereichs
<state>:	Statusangabe
	0: Schutzbereich deaktivieren
	1: Schutzbereich voraktivieren
	2: Schutzbereich aktivieren
	3: Schutzbereich voraktivieren mit bedingtem Stopp
<xMov>, <yMov>, <zMov>:	Bereits definierten Schutzbereich in den Geometrieachsen verschieben

Randbedingungen

Schutzbereichsüberwachung bei aktiver Werkzeugradiuskorrektur

Bei aktiver Werkzeugradiuskorrektur ist eine funktionsfähige Schutzbereichsüberwachung nur möglich, wenn die Ebene der Werkzeugradiuskorrektur identisch ist mit der Ebene der Schutzbereichsdefinitionen.

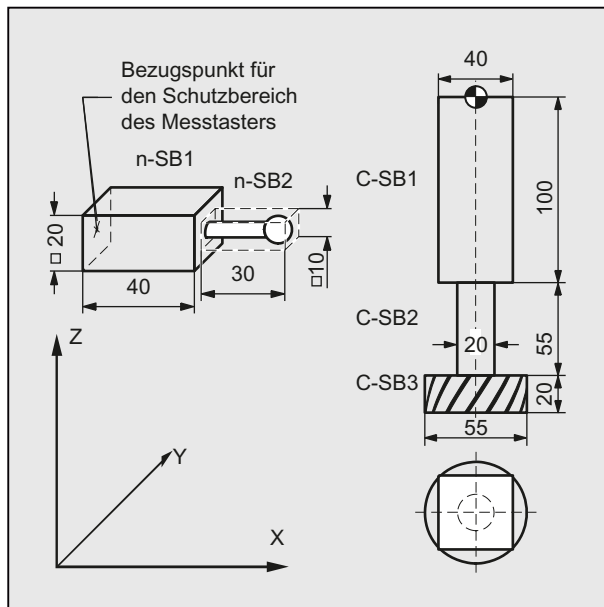
Beispiel

Für eine Fräsmaschine soll eine mögliche Kollision des Fräasers mit dem Messtaster überwacht werden. Die Lage des Messtasters soll bei der Aktivierung durch eine Verschiebung angegeben werden. Es werden dafür folgende Schutzbereiche definiert:

- Jeweils ein maschinenspezifischer und werkstückbezogener Schutzbereich für den Messtasterhalter (n-SB1) und für den Messtaster selbst (n-SB2).
- Jeweils ein kanalspezifischer und werkzeugbezogener Schutzbereich für den Fräserhalter (c-SB1), den Fräserschaft (c-SB2) und für den Fräser selbst (c-SB3).

Die Orientierung aller Schutzbereiche liegt in Z-Richtung.

Die Lage des Bezugspunkts des Messtasters bei der Aktivierung soll bei X = -120, Y = 60 und Z = 80 liegen.



Programmcode	Kommentar
DEF INT SCHUTZB	; Definition einer Hilfsvariablen
Definition der SchutzbereicheG17	; Orientierung einstellen
NPROTDEF (1, FALSE, 3, 10, -10)G01 X0 Y-10	; Schutzbereich n-SB1
X40	
Y10	
X0	
Y-10	
EXECUTE (SCHUTZB)	

3.2 Schutzbereiche aktivieren/deaktivieren (CPROT, NPROT)

Programmcode	Kommentar
NPROTDEF(2, FALSE, 3, 5, -5)	; Schutzbereich n-SB2
G01 X40 Y-5	
X70	
Y5	
X40	
Y-5	
EXECUTE(SCHUTZB)	
CPROTDEF(1, TRUE, 3, 0, -100)	; Schutzbereich c-SB1
G01 X-20 Y-20	
X20	
Y20	
X-20	
Y-20	
EXECUTE(SCHUTZB)	
CPROTDEF(2, TRUE, 3, -100, -150)	; Schutzbereich c-SB2
G01 X0 Y-10	
G03 X0 Y10 J10	
X0 Y-10 J-10	
EXECUTE(SCHUTZB)	
CPROTDEF(3, TRUE, 3, -150, -170)	; Schutzbereich c-SB3
G01 X0 Y-27,5	
G03 X0 Y27,5 J27,5	
X0 Y27,5 J-27,5	
EXECUTE(SCHUTZB)	
Aktivierung der Schutzbereiche:	
NPROT(1, 2, -120, 60, 80)	; Schutzbereich n-SB1 m. Versch. aktivieren
NPROT(2, 2, -120, 60, 80)	; Schutzbereich n-SB2 m. Versch. aktivieren
CPROT(1, 2, 0, 0, 0)	; Schutzbereich c-SB1 m. Versch. aktivieren
CPROT(2, 2, 0, 0, 0)	; Schutzbereich c-SB2 m. Versch. aktivieren
CPROT(3, 2, 0, 0, 0)	; Schutzbereich c-SB3 m. Versch. aktivieren

Weitere Informationen

Aktivierungsstatus (<state>)

- **<state>=2**

Ein Schutzbereich wird im Allgemeinen im Teileprogramm mit Status = 2 aktiviert.

Der Status ist immer kanalspezifisch, auch bei maschinenbezogenen Schutzbereichen.

- **<state>=1**

Wenn durch das PLC-Anwenderprogramm vorgesehen ist, dass ein Schutzbereich durch das PLC-Anwenderprogramm wirksam gesetzt werden kann, so erfolgt die dafür erforderliche Voraktivierung durch den Status = 1.

- **<state>=3**

Bei der Voraktivierung mit bedingtem Stopp wird nicht grundsätzlich vor einem verletzten, voraktivierten Schutzbereich angehalten. Der Stopp erfolgt nur dann, wenn der Schutzbereich wirksam gesetzt worden ist. Dies ermöglicht eine unterbrechungsfreie Bearbeitung, wenn die Schutzbereiche nur in besonderen Fällen wirksam gesetzt werden. Zu beachten ist, dass infolge der Bremsrampe ggf. in einen Schutzbereich gefahren wird, falls der Schutzbereich erst unmittelbar vor dem Einfahren wirksam gesetzt worden ist.

Die Voraktivierung mit bedingtem Stopp erfolgt durch den Status = 3.

- **<state>=0**

Die Deaktivierung und damit das Ausschalten der Schutzbereiche erfolgt durch den Status = 0. Es ist dabei keine Verschiebung notwendig.

Verschiebung von Schutzbereichen beim (Vor-)Aktivieren

Die Verschiebung kann in 1, 2 oder 3 Dimensionen erfolgen. Die Angabe der Verschiebung bezieht sich auf:

- den Maschinennullpunkt bei werkstückspezifischen Schutzbereichen.
- den Werkzeugträgerbezugspunkt F bei werkzeugspezifischen Schutzbereichen.

Status nach dem Hochlaufen

Schutzbereiche können bereits nach dem Hochlaufen und anschließendem Referenzpunktanfahren aktiviert sein. Es muss dafür die Systemvariable \$SN_PA_ACTIV_IMMED[<n>] bzw. \$SC_PA_ACTIV_IMMED[<n>] auf TRUE gesetzt sein. Sie werden immer mit dem Status = 2 aktiviert und haben keine Verschiebung.

Mehrfache Aktivierung von Schutzbereichen

Ein Schutzbereich kann gleichzeitig auch in mehreren Kanälen wirksam sein (z. B. Pinole bei zwei gegenüberliegenden Schlitten). Die Überwachung der Schutzbereiche erfolgt nur, wenn alle Geometrieachsen referiert sind.

Dabei gilt:

- Der Schutzbereich ist in einem Kanal nicht gleichzeitig mehrfach mit verschiedenen Verschiebungen aktivierbar.
- Maschinenbezogene Schutzbereiche müssen in beiden Kanälen die gleiche Orientierung aufweisen.

3.3 Überprüfung auf Schutzbereichsverletzung, Arbeitsfeldbegrenzung und Softwarelimits (CALCPOSI)

Funktion

Die Funktion CALCPOSI dient dazu zu überprüfen, ob ausgehend von einem gegebenen Startpunkt die Geometrieachsen einen vorgegebenen Weg verfahren können, ohne die Achsgrenzen (Softwarelimits), Arbeitsfeldbegrenzungen oder Schutzbereiche zu verletzen.

Für den Fall, dass der vorgegebene Weg nicht gefahren werden kann, wird der maximal zulässige Wert zurückgegeben.

Die Funktion CALCPOSI ist ein vordefiniertes Unterprogramm. Sie muss alleine in einem Satz stehen.

Syntax

```
Status=CALCPOSI(_STARTPOS, _MOVDIST, _DLIMIT, _MAXDIST, _BASE_SYS, _TESTLIM)
```

Bedeutung

Status

0: Funktion o. k.,
der vorgegebene Weg kann vollständig abgefahren werden.

–: In **_DLIMIT** ist mindestens eine Komponente negativ

–: In einer Transformationsberechnung ist ein Fehler aufgetreten

Kann der vorgegebene Weg nicht vollständig abgefahren werden, wird ein positiver, dezimal codierter Wert zurückgegeben:

Einerstelle (Art der verletzten Grenze):

1: Softwarelimits begrenzen den Verfahrenweg.

2: Arbeitsfeldbegrenzung begrenzt den Verfahrenweg.

3: Schutzbereiche begrenzen den Verfahrenweg.

Sind gleichzeitig mehrere Grenzen verletzt (z. B. Softwarelimits und Schutzbereiche), wird in der Einerstelle die Grenze gemeldet, die zur stärksten Einschränkung des vorgegebenen Verfahrenweges führt.

Zehnerstelle

10:
Der Anfangswert verletzt die Grenze

20:
Die vorgegebene Gerade verletzt die Grenze. Dieser Wert wird auch dann zurückgegeben, wenn der Endpunkt selbst keine Grenze verletzt, auf dem Weg vom Start–zum Endpunkt aber eine Verletzung eines Grenzwertes auftreten würde (z. B. Durchfahren eines Schutzbereiches, gekrümmte Softwarelimits im WKS bei nichtlinearen Transformationen, z. B. Transmit).

Hunderterstelle

100:

Der positive Grenzwert ist verletzt (nur, wenn die Einerstelle 1 oder 2 ist, d. h. bei Softwarelimits und Arbeitsfeldbegrenzung)

100:

Es ist ein NCKSchutzbereich verletzt (nur, wenn die Einerstelle 3 ist).

200:

Der negative Grenzwert ist verletzt (nur, wenn die Einerstelle 1 oder 2 ist, d. h. bei Softwarelimits und Arbeitsfeldbegrenzung)

200:

Es ist ein kanalspezifischer Schutzbereich verletzt (nur, wenn die Einerstelle 3 ist).

Tausenderstelle

1000:

Faktor, mit dem die Nummer der Achse multipliziert wird, die die Grenze verletzt (nur, wenn die Einerstelle 1 oder 2 ist, d. h. bei Softwarelimits und Arbeitsfeldbegrenzung).

Die Zählung der Achsen beginnt bei 1 und bezieht sich bei verletzten Softwarelimits (Einerstelle = 1) auf die Maschinenachsen und bei verletzter Arbeitsfeldbegrenzung (Einerstelle =2) auf die Geometrieachsen.

1000:

Faktor, mit dem die Nummer des verletzten Schutzbereiches multipliziert wird (nur, wenn die Einerstelle 3 ist).

Sind mehrere Schutzbereiche verletzt, wird in den Hunderter- und Tausenderstellen der Schutzbereich gemeldet, der zur stärksten Einschränkung des vorgegebenen Verfahrenweges führt.

`_STARTPOS`

Anfangswert für Abszisse [0], Ordinate [1] und Applikate [2] im (WKS)

`_MOVEDIST`

Wegvorgabe inkrementell für Abszisse [0], Ordinate [1] und Applikate [2]

`_DLIMIT`

[0] - [2]: Mindestabstände die den Geometrieachsen zugeordnet sind.

[3]: Mindestabstand, der einer linearen Maschinenachse zugeordnet wird bei einer nicht linearen Transformation, wenn keine Geometrieachse eindeutig zugeordnet werden kann.

[4]: Mindestabstand, der einer rotatorischen Maschinenachse zugeordnet wird bei einer nicht linearen Transformation, wenn keine Geometrieachse eindeutig zugeordnet werden kann. Nur bei speziellen Transformationen, wenn SW-Limits überwacht werden sollen.

`_MAXDIST`

Feld [0] - [2] für Rückgabewert. Inkrementeller Weg in allen drei Geometrieachsen, ohne dass der vorgegebene Mindestabstand von einer Achsgrenze in den beteiligten Maschinenachsen unterschritten wird.

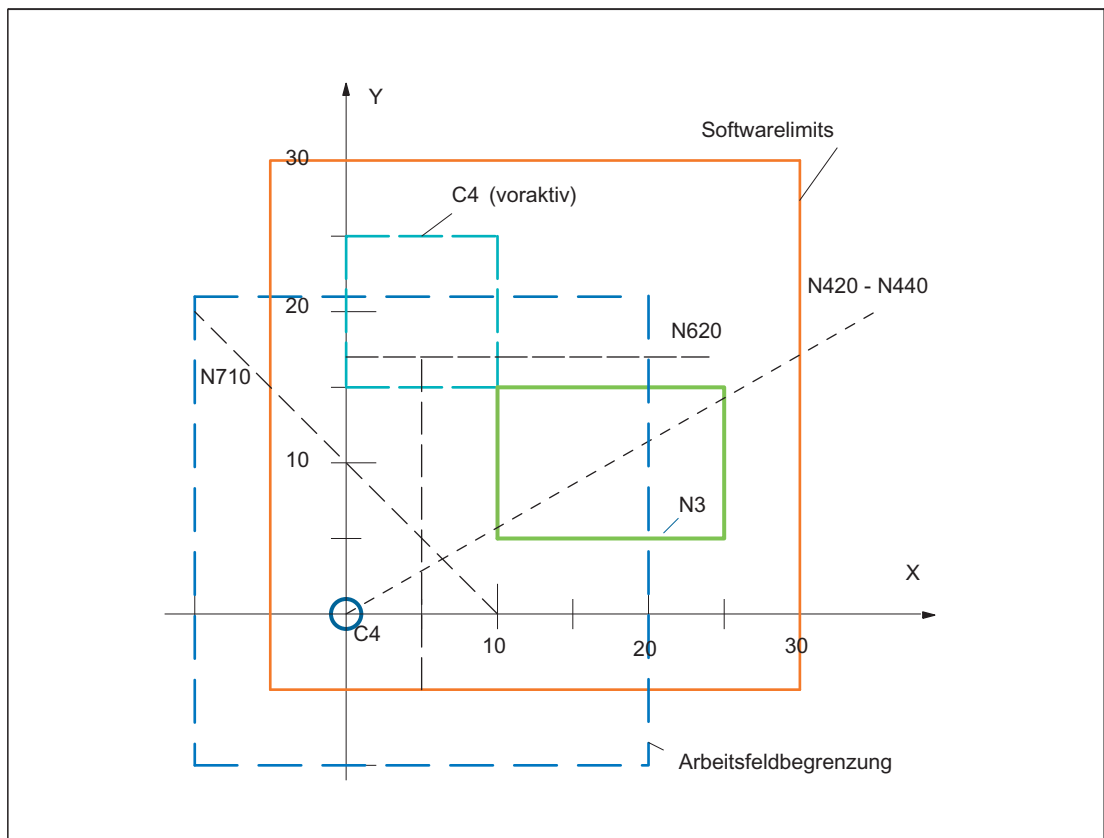
Ist der Verfahrenweg nicht eingeschränkt, ist der Inhalt dieses Rückgabeparameters gleich dem Inhalt von `_MOVEDIST`.

3.3 Überprüfung auf Schutzbereichsverletzung, Arbeitsfeldbegrenzung und Softwarelimits (CALCPOSI)

<code>_BASE_SYS</code>	<p>FALSE oder Parameter nicht angegeben:</p> <p>Bei der Bewertung der Positions- und Längenangaben wird der G-Code der Gruppe 13 (G70, G71, G700, G710; inch/metrisch) ausgewertet. Bei aktivem G70 und metrischem Grundsystem (bzw. aktivem G71 und inch) werden die WKSSystemvariablen \$AA_IW[X] und \$AA_MW[X] im Grundsystem geliefert und müssen gegebenenfalls zur Verwendung durch die Funktion CALCPOSI umgerechnet werden.</p> <p>TRUE:</p> <p>Bei der Bewertung der Positions- und Längenangaben wird stets das Grundsystem der Steuerung unabhängig vom Wert des aktiven Gder Gruppe 13 verwendet.</p>
<code>_TESTLIM</code>	<p>Zu überprüfende Begrenzungen (binär codiert):</p> <ol style="list-style-type: none">1: Software Limits überwachen2: Arbeitsfeldbegrenzungen überwachen3: Aktivierte Schutzbereiche überwachen4: Voraktivierte Schutzbereiche überwachen <p>Kombinationen durch addieren der Werte. Default: 15; alle prüfen.</p>

Beispiel

Im Beispiel (siehe Bild) sind in der XSoftwarelimits und Arbeitsfeldbegrenzungen eingezeichnet. Zusätzlich sind drei Schutzbereiche definiert, die beiden kanalspezifischen Schutzbereiche C2 und C4 sowie der NCKSchutzbereich N3. C2 ist ein kreisförmiger aktiver, werkzeugbezogener Schutzbereich mit 2 mm Radius. C4 ist ein quadratischer, voraktivierter und werkstückbezogener Schutzbereich mit 10 mm Seitenlänge und N3 ist ein rechteckiger aktiver Schutzbereich mit 10 mm bzw. 15 mm Seitenlänge. Im folgenden NC werden zunächst die Schutzbereiche und Arbeitsfeldbegrenzungen wie skizziert definiert, und anschließend wird die Funktion CALCPOSI mit verschiedenen Parametrierungen aufgerufen. Die Ergebnisse der einzelnen Aufrufe von CALCPOSI sind in der Tabelle am Beispielenende zusammengefasst.



3.3 Überprüfung auf Schutzbereichsverletzung, Arbeitsfeldbegrenzung und Softwarelimits (CALCPOSI)

Programmcode	Kommentar
N10 def real _STARTPOS[3]	
N20 def real _MOVDIST[3]	
N30 def real _DLIMIT[5]	
N40 def real _MAXDIST[3]	
N50 def int _SB	
N60 def int _STATUS	
N70 cprotdef(2, true, 0)	; werkzeugbezogener Schutzbereich
N80 g17 g1 x-y0	
N90 g3 i2 x2	
N100 i-x-	
N110 execute(_SB)	
N120 cprotdef(4, false, 0)	; werkstückbezogener Schutzbereich
N130 g17 g1 x0 y15	
N140 x10	
N150 y25	
N160 x0	
N170 y15	
N180 execute(_SB)	
N190 nprotdef(3, false, 0)	; maschinenbezogener Schutzbereich
N200 g17 g1 x10 y5	
N210 x25	
N220 y15	
N230 x10	
N240 y5	
N250 execute(_SB)	
N260 cprot(2,2,0, 0, 0)	; Schutzbereiche aktivieren bzw. voraktivieren
N270 cprot(4,1,0, 0, 0)	
N280 nprot(3,2,0, 0, 0)	
N290 g25 XX=-YY=-	; Arbeitsfeldbegrenzungen definieren
N300 g26 xx= 20 yy= 21	
N310 _STARTPOS[0] = 0.	
N320 _STARTPOS[1] = 0.	
N330 _STARTPOS[2] = 0.	
N340 _MOVDIST[0] = 35.	
N350 _MOVDIST[1] = 20.	
N360 _MOVDIST[2] = 0.	
N370 _DLIMIT[0] = 0.	
N380 _DLIMIT[1] = 0.	
N390 _DLIMIT[2] = 0.	
N400 _DLIMIT[3] = 0.	
N410 _DLIMIT[4] = 0.	

3.3 Überprüfung auf Schutzbereichsverletzung, Arbeitsfeldbegrenzung und Softwarelimits (CALCPOS)

Programmcode	Kommentar
;Verschiede Funktionsaufrufe	; Anderer Startpunkt
N420 _STATUS = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST)	
N430 _STATUS = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST,,3)	
N440 _STATUS = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST,,1)	
N450 _STARTPOS[0] = 5.	; Anderes Ziel
N460 _STARTPOS[1] = 17.	
N470 _STARTPOS[2] = 0.	
N480 _MOVDIST[0] = 0.	
N490 _MOVDIST[1] =--.	
N500 _MOVDIST[2] = 0.	
;Verschiede Funktionsaufrufe	
N510 _STATUS = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST,,14)	
N520 _STATUS = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST,, 6)	
N530 _DLIMIT[1] = 2.	
N540 _STATUS = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST,, 6)	
N550 _STARTPOS[0] = 27.	
N560 _STARTPOS[1] = 17.1	
N570 _STARTPOS[2] = 0.	
N580 _MOVDIST[0] =--.	
N590 _MOVDIST[1] = 0.	
N600 _MOVDIST[2] = 0.	
N610 _DLIMIT[3] = 2.	
N620 _STATUS = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST,, 12)	
N630 _STARTPOS[0] = 0.	
N640 _STARTPOS[1] = 0.	
N650 _STARTPOS[2] = 0.	
N660 _MOVDIST[0] = 0.	
N670 _MOVDIST[1] = 30.	
N680 _MOVDIST[2] = 0.	
N690 trans x10	
N700 arot z45	
N710 _STATUS = calcposi(_STARTPOS,_MOVDIST, _DLIMIT, _MAXDIST)	
N720 M30	

Ergebnisse der Prüfungen im Beispiel:

Satznr. N...	_STATUS	_MAXDIST [0] (= X)	_MAXDIST [1] (= Y)	Bemerkungen
420	3123	8.040	4.594	Schutzbereich SB N3 wird verletzt.
430	1122	20.000	11.429	Keine SB-Überwachung,-Arbeitsfeldbegrenzung wird verletzt.
440	1121	30.000	17.143	Nur noch Überwachung der Softwarelimits aktiv.
510	4213	0.000	0.000	Startpunkt verletzt SB C4
520	0000	0.000	-0.000	Voraktivierter SB C4 wird nicht überwacht. Vorgegebener Weg kann vollständig verfahren werden.
540	2222	0.000	-0.000	Wegen _DLIMIT[1]=2 wird der Verfahrweg durch Arbeitsfeldbegrenzung eingeschränkt.
620	4223	-0.000	0.000	Abstand zu C4 wegen C2 und _DLIMIT[3] insgesamt 4 mm. Abstand C2-N3 von 0.1 mm führt nicht zur Beschränkung des Verfahrwegs.
710	1221	0.000	21.213	Frame mit Translation und Rotation aktiv. Der zulässige Verfahrweg in _MOVDIST gilt im verschobenen und gedrehten Koordinatensystem (WKS).

Sonderfälle und weitere Details

Alle Wegangaben sind immer im Radiusmaß auch bei einer Planachse mit aktivem G"DIAMON". Kann der Weg einer der beteiligten Achsen nicht vollständig verfahren werden, werden im Rückgabewert _MAXDIST auch die Wege der anderen Achsen entsprechend reduziert, so dass der resultierende Endpunkt auf der vorgegebenen Bahn liegt.

Es ist zulässig, dass für eine oder mehrere der beteiligten Achsen keine Softwarelimits bzw. Arbeitsfeldbegrenzungen oder Schutzbereiche definiert sind. Sämtliche Grenzen werden nur überwacht, wenn die beteiligten Achsen referiert sind. Eventuell beteiligte Rundachsen werden nur überwacht, wenn sie keine Moduloachsen sind.

Die Überwachung der Softwarelimits und der Arbeitsfeldbegrenzungen ist wie im normalen Verfahrenbetrieb abhängig von aktiven Einstellungen (Interfacesignale zur Auswahl der Softwarelimits 1 bzw. Softwarelimits 2, GWALIMON/WALIMOF, Settingdaten zur individuellen Aktivierung der Arbeitsfelddgrenzen und zur Festlegung, ob bei der Überwachung der Arbeitsfeldbegrenzungen der Radius des aktiven Werkzeuges berücksichtigt werden soll oder nicht).

Bei bestimmten kinematischen Transformationen (z.B. TRANSMIT) kann die Position der Maschinenachsen aus den Positionen im Werkstückkoordinatensystem (WKS) nicht eindeutig bestimmt werden (Mehrdeutigkeit). Im normalen Verfahrenbetrieb ergibt sich die Eindeutigkeit in der Regel aus der Vorgeschichte und der Bedingung, dass einer kontinuierlichen Bewegung im WKS eine kontinuierliche Bewegung der Maschinenachsen entsprechen muss. Bei der Überwachung der Softwarelimits mit Hilfe der Funktion CALCPOSI wird deshalb in derartigen Fällen die gegenwärtige Maschinenposition zur Auflösung der Mehrdeutigkeit herangezogen. Gegebenenfalls muss deshalb vor CALCPOSI ein **STOPRE** programmiert werden, um die Funktion mit gültigen Maschinenachspositionen versorgen zu können.

Es ist nicht sichergestellt, dass zu den Schutzbereichen bei einer Bewegung auf dem vorgegebenen Verfahrenweg der in _DLIMIT[3] spezifizierte Abstand überall eingehalten wird. Dafür kann bei Verlängerung des in _MOVDIST zurückgelieferten Endpunktes um diese Distanz kein Schutzbereich verletzt werden. Die Gerade kann in ihrem Verlauf aber beliebig dicht an einem Schutzbereich vorbei führen.

Hinweis

Details zu Arbeitsfeldbegrenzungen finden Sie im
/PG/ Programmierhandbuch Grundlagen,

zu den Softwarelimits in
/FB1/ Funktionshandbuch Grundfunktionen; Achsüberwachungen, Schutzbereiche (A3).

Spezielle Wegbefehle

4.1 Codierte Positionen anfahren (CAC, CIC, CDC, CACP, CACN)

Funktion

Über die folgenden Befehle können Sie Linear- und Rundachsen über Positionsnummern auf in Maschinendaten-Tabellen hinterlegte feste Achspositionen verfahren. Diese Art der Programmierung wird als "Anfahren von codierten Positionen" bezeichnet.

Syntax

```
CAC (<n>)
CIC (<n>)
CACP (<n>)
CACN (<n>)
```

Bedeutung

CAC (<n>)	Codierte Position von Positionsnummer n anfahren
CIC (<n>)	Codierte Position, ausgehend von der aktuellen Positionsnummer, n-Positionsplätze vor (+n) oder zurück (-n) anfahren
CDC (<n>)	Codierte Position von Positionsnummer n auf kürzestem Weg anfahren (nur für Rundachsen)
CACP (<n>)	Codierte Position von Positionsnummer n in positiver Richtung anfahren (nur für Rundachsen)
CACN (<n>)	Codierte Position von Positionsnummer n in negativer Richtung anfahren (nur für Rundachsen)
<n>	Positionsnummer innerhalb der Maschinendaten-Tabelle Wertebereich: 0, 1, ... (max. Anzahl Tabellenplätze - 1)

Beispiel: Anfahren von codierten Positionen einer Positionierachse

Programmiercode	Kommentar
N10 FA[B]=300	; Vorschub für Positionierachse B
N20 POS[B]=CAC(10)	; Codierte Position von Positionsnummer 10 anfahren
N30 POS[B]=CIC(-4)	; Codierte Position von "aktuelle Positionsnummer" - 4 anfahren

Literatur

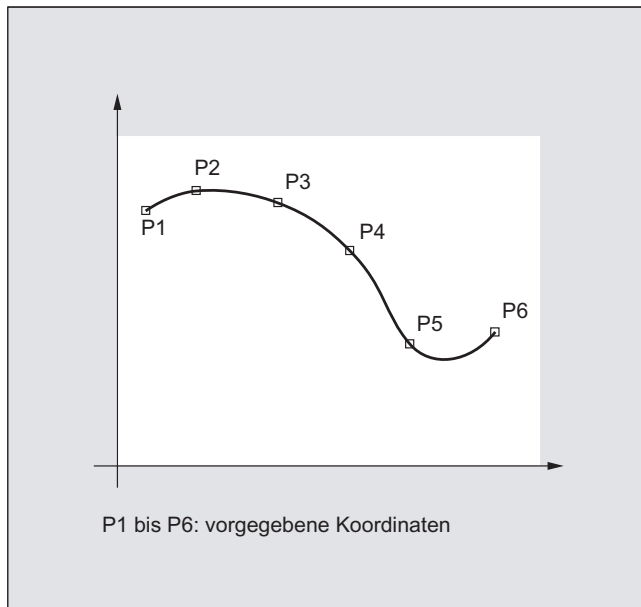
- Funktionshandbuch Erweiterungsfunktionen; Teilungsachsen (T1)
- Funktionshandbuch Synchronaktionen

4.2 Spline-Interpolation (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL)

Funktion

Beliebig gekrümmte Konturen an Werkstücken können nicht analytisch exakt beschrieben werden. Derartige Konturen werden daher durch eine begrenzte Anzahl von Stützpunkten, z. B. beim Digitalisieren von Oberflächen, angenähert. Zur Erzeugung der digitalisierten Oberfläche an einem Werkstück müssen die Stützpunkte zu einer Konturbeschreibung verbunden werden. Dies ermöglicht die Spline-Interpolation.

Ein Spline definiert eine Kurve, die aus Polynomen 2. oder 3. Grades zusammengesetzt wird. Die Eigenschaften an den Stützpunkten eines Splines sind **abhängig vom verwendeten Spline-Typ** definierbar.



Folgende Spline-Typen stehen bei SINUMERIK solution line zur Verfügung:

- A-Spline
- B-Spline
- C-Spline

Syntax

Allgemein:

```
ASPLINE X... Y... Z... A... B... C...  
BSPLINE X... Y... Z... A... B... C...  
CSPLINE X... Y... Z... A... B... C...
```

4.2 Spline-Interpolation (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL)

Bei B-Spline zusätzlich programmierbar:

PW=<n>
SD=2
PL=<Wert>

Bei A- und C-Spline zusätzlich programmierbar:

BAUTO / BNAT / BTAN
EAUTO / ENAT / ETAN

Bedeutung

Spline-Interpolationstyp:

ASPLINE	Befehl zum Einschalten der A-Spline-Interpolation
BSPLINE	Befehl zum Einschalten der B-Spline-Interpolation
CSPLINE	Befehl zum Einschalten der C-Spline-Interpolation

Die Befehle ASPLINE, BSPLINE und CSPLINE sind modal wirksam und gehören zur Gruppe der Wegbefehle.

Stützpunkte bzw. Kontrollpunkte:

X... Y... Z...	Positionen in kartesischen Koordinaten
A... B... C...	

Punktgewicht (nur B-Spline):

PW

Mit dem Befehl PW ist für jeden Stützpunkt die Programmierung eines sogenannten "Punktgewichts" möglich.

<n>

"Punktgewicht"

Wertebereich: $0 \leq n \leq 3$

Schrittweite: 0.0001

Wirkung: $n > 1$ Die Kurve wird vom Kontrollpunkt stärker angezogen.

$n < 1$ Die Kurve wird vom Kontrollpunkt weniger stark angezogen.

Spline-Grad (nur B-Spline):

SD

Standardmäßig wird ein Polygon 3. Grades verwendet. Durch Programmierung von SD=2 kann aber auch ein Polygon 2. Grades verwendet werden.

Knotenabstand (nur B-Spline):

PL

Die Knotenabstände werden intern geeignet berechnet. Die Steuerung kann aber auch vorgegebene Knotenabstände verarbeiten, die mit dem Befehl PL als sog. Parameter-Intervall-Länge angegeben werden.

<Wert>

Parameter-Intervall-Länge

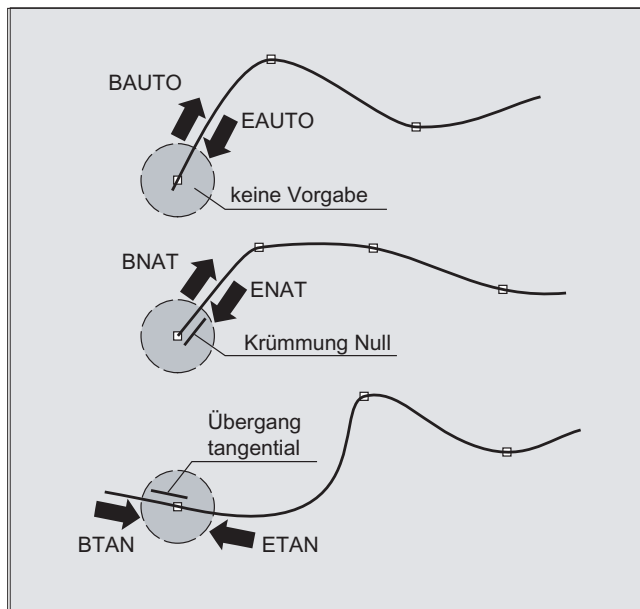
Wertebereich: wie Wegmaß

Übergangsverhalten am Beginn der Spline-Kurve (nur A- oder C-Spline):

BAUTO	Keine Vorgabe für das Übergangsverhalten. Der Anfang ergibt sich aus der Lage des ersten Punkts.
BNAT	Krümmung Null
BTAN	Tangentialer Übergang zum Satz vorher (Löschstellung)

Übergangsverhalten am Ende der Spline-Kurve (nur A- oder C-Spline):

EAUTO	Keine Vorgabe für das Übergangsverhalten. Das Ende ergibt sich aus der Lage des letzten Punkts.
ENAT	Krümmung Null
ETAN	Tangentialer Übergang zum Satz vorher (Löschstellung)



Hinweis

Das programmierbare Übergangsverhalten hat keinen Einfluss auf den B-Spline. Der B-Spline ist in Start- und Endpunkt immer tangential zum Kontrollpolygon.

Randbedingungen

- Die Werkzeugradiuskorrektur ist einsetzbar.
- Kollisionsüberwachung erfolgt in der Projektion auf die Ebene.

Beispiele

Beispiel 1: B-Spline

Programmcode 1 (alle Gewichte 1)

```
N10 G1 X0 Y0 F300 G64
N20 BSPLINE
N30 X10 Y20
N40 X20 Y40
N50 X30 Y30
N60 X40 Y45
N70 X50 Y0
```

Programmcode 2 (unterschiedliche Gewichte)

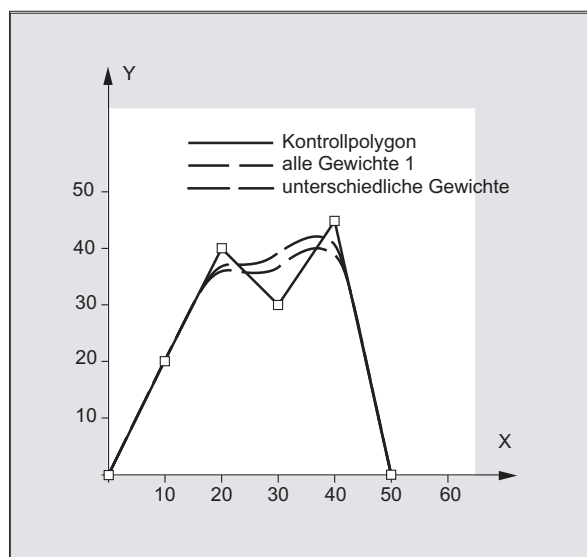
```
N10 G1 X0 Y0 F300 G64
N20 BSPLINE
N30 X10 Y20 PW=2
N40 X20 Y40
N50 X30 Y30 PW=0.5
N60 X40 Y45
N70 X50 Y0
```

Programmcode 3 (Kontrollpolygon)

Kommentar

```
N10 G1 X0 Y0 F300 G64
N20
N30 X10 Y20
N40 X20 Y40
N50 X30 Y30
N60 X40 Y45
N70 X50 Y0
```

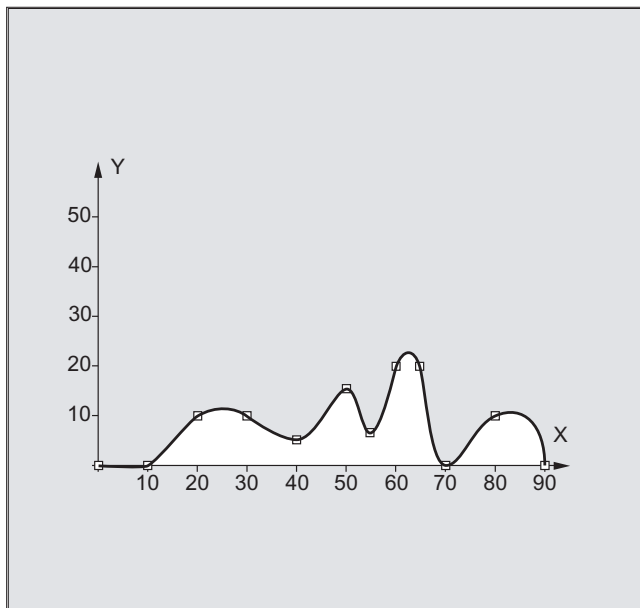
; entfällt



Beispiel 2: C-Spline, am Anfang und am Ende Krümmung Null

Programmcode

```
N10 G1 X0 Y0 F300  
N15 X10  
N20 BNAT ENAT  
N30 CSPLINE X20 Y10  
N40 X30  
N50 X40 Y5  
N60 X50 Y15  
N70 X55 Y7  
N80 X60 Y20  
N90 X65 Y20  
N100 X70 Y0  
N110 X80 Y10  
N120 X90 Y0  
N130 M30
```



Beispiel 3: Spline-Interpolation (A-Spline) und Koordinatentransformation (ROT)

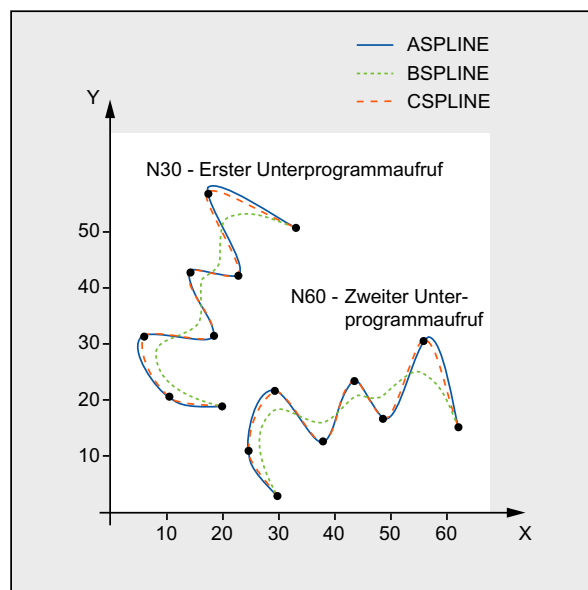
Hauptprogramm:

Programmcode	Kommentar
N10 G00 X20 Y18 F300 G64	; Startpunkt anfahren.
N20 ASPLINE	; Interpolationstyp A-Spline aktivieren.
N30 KONTUR	; Erster Aufruf des Unterprogramms.
N40 ROT Z-45	; Koordinatentransformation: Drehung des WKS um -45° um die Z-Achse.
N50 G00 X20 Y18	; Konturstartpunkt anfahren.
N60 KONTUR	; Zweiter Aufruf des Unterprogramms.
N70 M30	; Programmende

Unterprogramm "Kontur" (enthält die Stützpunkt-Koordinaten):

Programmcode
N10 X20 Y18
N20 X10 Y21
N30 X6 Y31
N40 X18 Y31
N50 X13 Y43
N60 X22 Y42
N70 X16 Y58
N80 X33 Y51
N90 M1

In der folgenden Abbildung sind neben der Spline-Kurve, die aus dem Programmbeispiel resultiert (ASPLINE), auch die Spline-Kurven enthalten, die sich bei Aktivierung einer B- oder C-Spline-Interpolation ergeben hätten (BSPLINE, CSPLINE):



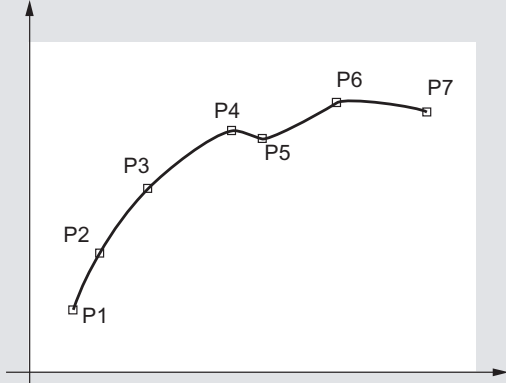
Weitere Informationen

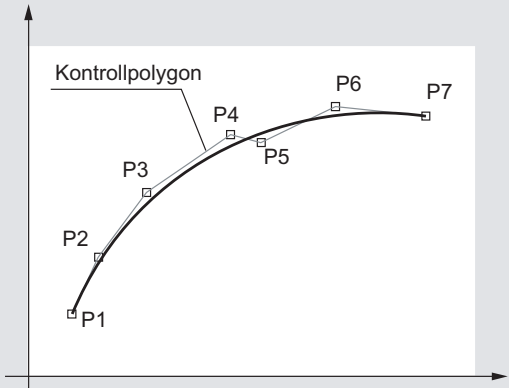
Vorteile der Spline-Interpolation

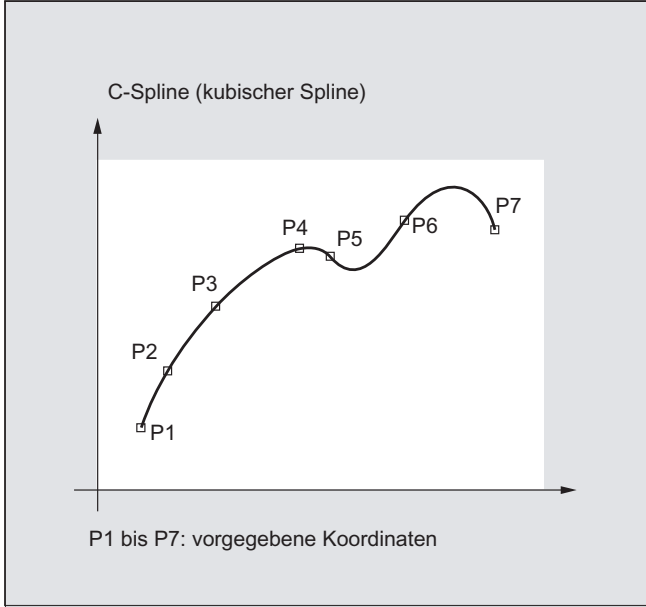
Durch Verwendung der Spline-Interpolation lassen sich, im Gegensatz zur Verwendung von Geradensätzen G01, folgende Vorteile erzielen:

- Reduzierung der Anzahl von benötigten Teileprogrammssätzen zur Beschreibung der Kontur
- Weicher, mechanischschonender Kurvenverlauf beim Übergange zwischen den Teileprogrammssätzen

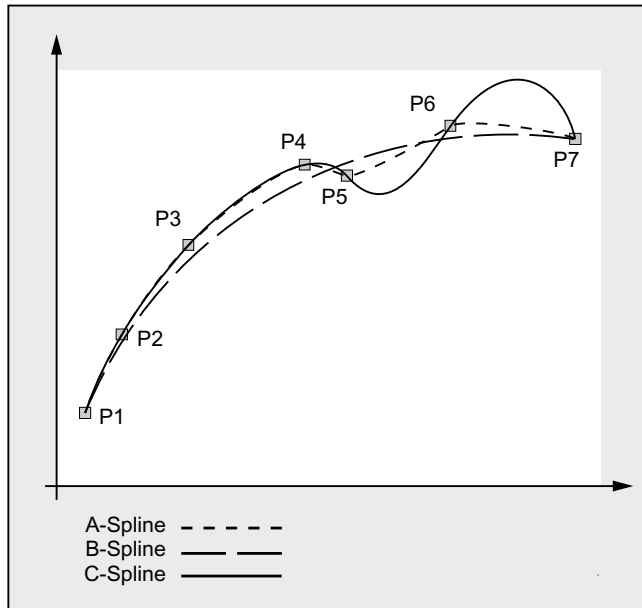
Eigenschaften und Anwendung der verschiedenen Spline-Typen

Spline-Typ	Eigenschaften und Anwendung
<p>A-Spline</p>	<div data-bbox="572 891 1219 1496" style="border: 1px solid black; padding: 10px; margin-bottom: 10px;"> <p style="text-align: center;">A-Spline (Akima-Spline)</p>  <p style="text-align: center;">P1 bis P7: vorgegebene Koordinaten</p> </div> <p>Eigenschaften:</p> <ul style="list-style-type: none"> • Verläuft exakt durch die vorgegebenen Stützpunkte. • Der Kurvenverlauf ist tangential- aber nicht krümmungsstetig. • Erzeugt kaum ungewollte Schwingungen. • Der Einflussbereich von Stützpunktänderungen ist lokal, d. h. Veränderung eines Stützpunkts wirkt sich nur auf bis zu max. 6 benachbarte Stützpunkte aus. <p>Anwendung:</p> <p>Der A-Spline eignet sich vor allem für die Interpolation von Kurvenverläufen mit großen Steigungsänderungen (z. B. treppenförmige Kurvenverläufe).</p>

Spline-Typ	Eigenschaften und Anwendung
<p>B-Spline</p>	<div data-bbox="611 398 1257 1003" style="border: 1px solid black; padding: 10px; margin-bottom: 10px;"> <p style="text-align: center;">B-Spline</p>  <p style="text-align: center;">P1 bis P7: vorgegebene Koordinaten</p> </div> <p>Eigenschaften:</p> <ul style="list-style-type: none"> • Verläuft nicht durch die vorgegebenen Stützpunkte, sondern nur in deren Nähe. Die Kurve wird durch die Stützpunkte angezogen. Durch Gewichtung der Stützpunkte mit einem Faktor, kann der Kurvenverlauf zusätzlich beeinflusst werden. • Der Kurvenverlauf ist tangential- und krümmungsstetig. • Erzeugt keine ungewollten Schwingungen. • Der Einflussbereich von Stützpunktänderungen ist lokal, d. h. Veränderung eines Stützpunkts wirkt sich nur auf bis zu max. 6 benachbarte Stützpunkte aus. <p>Anwendung:</p> <p>Der B-Spline ist primär als Schnittstelle zu CAD-Systemen gedacht.</p>

Spline-Typ	Eigenschaften und Anwendung
<p>C-Spline</p>	<div style="text-align: center;">  </div> <p>Eigenschaften:</p> <ul style="list-style-type: none"> • Verläuft exakt durch die vorgegebenen Stützpunkte. • Der Kurvenverlauf ist tangential- und krümmungsstetig. • Erzeugt häufig ungewollten Schwingungen, besonders an Stellen mit großen Steigungsänderungen. • Der Einflussbereich von Stützpunktänderungen ist global, d. h. Veränderung eines Stützpunkts wirkt sich auf den gesamten Kurvenverlauf aus. <p>Anwendung:</p> <p>Der C-Spline kann dann gut eingesetzt werden, wenn die Stützpunkte auf einer analytisch bekannten Kurve liegen (Kreis, Parabel, Hyperbel)</p>

Gegenüberstellung der drei Spline-Typen bei gleichen Stützpunkten



Mindestanzahl an Spline-Sätzen

Die G-Codes `ASPLINE`, `BSPLINE` und `CSPLINE` verbinden Satzendpunkte mit Splines. Dazu müssen im Vorlauf eine Reihe von Sätzen (Endpunkte) gleichzeitig berechnet werden. Die Größe des Puffers für die Berechnung beträgt standardmäßig 10 Sätze. Nicht jede Satzinformation ist ein Spline-Endpunkt. Die Steuerung benötigt jedoch von 10 Sätzen eine bestimmte Anzahl an Spline-Endpunkt-Sätzen:

Spline-Typ	Mindestanzahl an Spline-Sätzen
A-Spline:	Von je 10 Sätzen müssen mindestens 4 Spline-Sätze sein. Kommentarsätze und Parameterrechnungen zählen hierbei nicht.
B-Spline:	Von je 10 Sätzen müssen mindestens 6 Spline-Sätze sein. Kommentarsätze und Parameterrechnungen zählen hierbei nicht.
C-Spline:	Die benötigte Mindestanzahl an Spline-Sätzen ergibt sich aus folgender Summe: Wert aus MD20160 \$MC_CUBIC_SPLINE_BLOCKS + 1 Im MD20160 wird die Anzahl der Punkte eingetragen, über die der Spline-Abschnitt berechnet wird. Die Standardeinstellung beträgt 8. Von je 10 Sätzen müssen daher im Standardfall mindestens 9 Spline-Sätze sein.

Hinweis

Bei Unterschreitung des tolerierbaren Werts wird ein Alarm ausgegeben, ebenso, wenn eine am Spline beteiligte Achse als Positionierachse programmiert wird.

Zusammenfassung kurzer Spline-Sätze

Bei der Spline-Interpolation können kurze Spline-Sätze entstehen, die zu einer unnötigen Reduzierung der Bahngeschwindigkeit führen. Mit der Funktion "Zusammenfassung kurzer Spline-Sätze" können diese Sätze so zusammengefasst werden, dass die resultierende Satzlänge ausreichend groß ist und nicht zu einer Verringerung der Bahngeschwindigkeit führt.

Die Funktion wird aktiviert über das kanalspezifische Maschinendatum:

MD20488 \$MC_SPLINE_MODE (Einstellung für Spline-Interpolation)

Literatur:

Funktionshandbuch Grundfunktionen; Bahnsteuerbetrieb, Genauhalt, LookAhead (B1),
Kapitel: Zusammenfassung kurzer Spline-Sätze

4.3 Spline-Verbund (SPLINEPATH)

Funktion

Die im Spline-Verbund zu interpolierenden Achsen werden mit dem Befehl `SPLINEPATH` ausgewählt. Bis zu acht Bahnachsen sind bei der Spline-Interpolation möglich.

Hinweis

Wird `SPLINEPATH` nicht explizit programmiert, so werden die ersten drei Achsen des Kanals als Spline-Verbund verfahren.

Syntax

Die Festlegung des Spline-Verbundes erfolgt in einem gesonderten Satz:

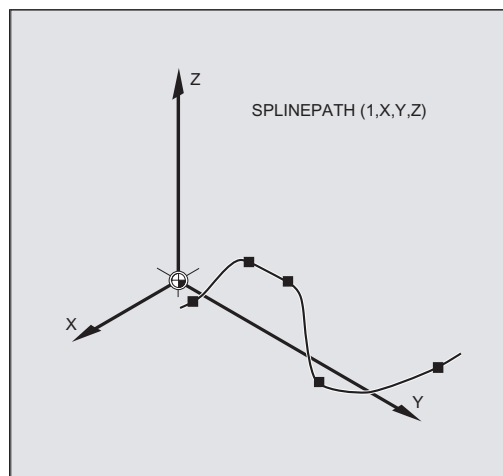
```
SPLINEPATH (n, X, Y, Z, ...)
```

Bedeutung

<code>SPLINEPATH</code>	Befehl zur Festlegung eines Spline-Verbundes
<code>n</code>	=1 (fester Wert)
<code>X, Y, Z, ...</code>	Bezeichner der im Spline-Verbund zu interpolierenden Bahnachsen

Beispiel: Spline-Verbund mit drei Bahnachsen

Programmcode	Kommentar
N10 G1 X10 Y20 Z30 A40 B50 F350	
N11 SPLINEPATH (1, X, Y, Z)	; Spline-Verbund
N13 CSPLINE BAUTO EAUTO X20 Y30 Z40 A50 B60	; C-Spline
N14 X30 Y40 Z50 A60 B70	; Stützpunkte
...	
N100 G1 X... Y...	; Abwahl Spline-Interpolation



4.4 NC-Satz-Kompression (COMPON, COMPCURV, COMPCAD, COMPOF)

Funktion

CAD/CAM-Systeme liefern in der Regel Linearsätze, welche die parametrisierte Genauigkeit einhalten. Dies führt bei komplexen Konturen zu einer erheblichen Datenmenge und zu eventuell kurzen Bahnabschnitten. Diese kurzen Bahnabschnitte begrenzen die Abarbeitungsgeschwindigkeit.

Durch die Anwendung einer Kompressor-Funktion erfolgt eine Annäherung an die durch Linear-Sätze vorgegebene Kontur durch Polynom-Sätze. Dadurch ergeben sich folgende Vorteile:

- Reduzierung der Anzahl von benötigten Teileprogrammsätzen zur Beschreibung der Werkstückkontur
- Stetige Satzübergänge
- Erhöhung der maximal möglichen Bahngeschwindigkeiten

Folgende Kompressor-Funktionen stehen zur Verfügung:

- COMPON

Die Satzübergänge sind nur stetig in der **Geschwindigkeit**, während die Beschleunigung der beteiligten Achsen an den Satzübergängen Sprünge machen kann.

- COMPCURV

Die Satzübergänge sind **beschleunigungsstetig**. Damit ist sowohl ein glatter Verlauf der Geschwindigkeit als auch der Beschleunigung aller Achsen an den Satzübergängen gewährleistet.

- COMPCAD

Rechenzeit- und speicherplatzintensive Kompression, die bezüglich Oberflächengüte und Geschwindigkeit optimiert. COMPCAD sollte nur eingesetzt werden, wenn Maßnahmen zur Oberflächenverbesserung vom CAD/CAM-Programm nicht vorab geleistet werden können.

Beendet wird die Kompressor-Funktion mit `COMPOF`.

Syntax

```
COMPON  
COMPCURV  
COMPCAD  
COMPOF
```

Bedeutung

COMPON:	Befehl zum Einschalten der Kompressor-Funktion COMPON. Wirksamkeit: modal
COMPCURV:	Befehl zum Einschalten der Kompressor-Funktion COMPCURV. Wirksamkeit: modal
COMPCAD:	Befehl zum Einschalten der Kompressor-Funktion COMPCAD. Wirksamkeit: modal
COMPOF:	Befehl zum Ausschalten der aktuell aktiven Kompressor-Funktion.

Hinweis

Zur zusätzlichen Verbesserung der Oberflächengüte kann die Überschleiffunktion `G642` und die Ruckbegrenzung `SOFT` verwendet werden. Diese Befehle sind am Programmanfang zu schreiben.

Randbedingungen

- Die NC-Satz-Kompression wird i. d. R. für Linearsätze (G_1) durchgeführt.
- Es werden nur Sätze komprimiert, die einer einfachen Syntax genügen:

```
N... G1X... Y... Z... F... ;Kommentar
```

Alle anderen Sätze werden unverändert abgearbeitet (ohne Kompression).
- Bewegungssätze mit erweiterten Adressen wie $C=100$ oder $A=AC(100)$ werden auch komprimiert.
- Positionswerte müssen nicht direkt programmiert werden, sondern können auch indirekt über Parameterzuweisungen angegeben werden, z. B. $X=R1*(R2+R3)$.
- Wenn die Option "Orientierungstransformation" zur Verfügung steht, dann können auch NC-Sätze komprimiert werden, in denen die Werkzeugorientierung (und ggf. auch die Werkzeugdrehung) mittels Richtungsvektoren programmiert ist (siehe "Komprimierung der Orientierung (Seite 368)").
- Der Kompressionsvorgang wird unterbrochen durch jede andere NC-Anweisung, z. B. eine Hilfsfunktionsausgabe.

Beispiele

Beispiel 1: COMPON

Programmcode	Kommentar
N10 COMPON	; Kompressor-Funktion COMPON ein.
N11 G1 X0.37 Y2.9 F600	; G1 vor Endpunkt und Vorschub.
N12 X16.87 Y-.698	
N13 X16.865 Y-.72	
N14 X16.91 Y-.799	
...	
N1037 COMPOF	; Kompressor-Funktion aus.
...	

Beispiel 2: COMPCAD

Programmcode	Kommentar
G00 X30 Y6 Z40	
G1 F10000 G642	; Überschleiffunktion G642 ein.
SOFT	; Ruckbegrenzung SOFT ein.
COMPCAD	; Kompressor-Funktion COMPCAD ein.
STOPFIFO	
N24050 Z32.499	
N24051 X41.365 Z32.500	
N24052 X43.115 Z32.497	
N24053 X43.365 Z32.477	
N24054 X43.556 Z32.449	
N24055 X43.818 Z32.387	
N24056 X44.076 Z32.300	
...	
COMPOF	; Kompressor-Funktion aus.
G00 Z50	
M30	

Literatur

Funktionshandbuch Grundfunktionen; Bahnsteuerbetrieb, Genauhalt, LookAhead (B1), Kapitel: "NC-Satz-Kompression"

4.5 Polynom-Interpolation (POLY, POLYPATH, PO, PL)

Funktion

Im eigentlichen Sinn handelt es sich bei der Polynom-Interpolation (POLY) nicht um eine Spline-Interpolationsart. Sie ist in erster Linie als Schnittstelle für die Programmierung extern erzeugter Spline-Kurven gedacht. Hierbei können die Spline-Abschnitte direkt programmiert werden.

Diese Interpolationsart entlastet die NC von der Berechnung der Polynom-Koeffizienten. Sie ist dann optimal einsetzbar, wenn die Koeffizienten direkt von einem CAD-System oder Post-Prozessor kommen.

Syntax

Polynom 3. Grades:

```
POLY PO [X] = (xe, a2, a3) PO [Y] = (ye, b2, b3) PO [Z] = (ze, c2, c3) PL=n
```

Polynome 5. Grades und neue Polynomsyntax:

```
POLY X=PO(xe, a2, a3, a4, a5) Y=PO(ye, b2, b3, b4, b5) Z=PO(ze, c2, c3, c4, c5)
PL=n
POLYPATH ("AXES", "VECT")
```

Hinweis

Die Summe der in einem NC-Satz programmierten Polynom-Koeffizienten und Achsen darf die maximal erlaubte Achszahl pro Satz nicht überschreiten.

Bedeutung

<small>POLY</small> :	Einschalten der Polynom-Interpolation mit einem Satz mit <small>POLY</small> .
<small>POLYPATH</small> :	Polynom-Interpolation selektierbar für die beiden Achsgruppen <small>AXIS</small> oder <small>VECT</small>
<small>PO</small> [Achsbezeichner/Variable] :	Endpunkte und Polynom-Koeffizienten
<small>X</small> , <small>Y</small> , <small>Z</small> :	Achsbezeichner
<small>xe</small> , <small>ye</small> , <small>ze</small> :	Angabe der Endposition für die jeweilige Achse; Wertebereich wie Wegmaß
<small>a2</small> , <small>a3</small> , <small>a4</small> , <small>a5</small> :	Die Koeffizienten <small>a2</small> , <small>a3</small> , <small>a4</small> , und <small>a5</small> werden mit ihrem Wert geschrieben; Wertebereich wie Wegmaß. Der jeweils letzte Koeffizient kann entfallen, wenn er den Wert Null hat.
<small>PL</small> :	Länge des Parameterintervalls, auf dem die Polynome definiert sind (Definitionsbereich der Funktion $f(p)$). Das Intervall beginnt immer bei 0, <small>p</small> kann Werte von 0 bis <small>PL</small> annehmen. Theoretischer Wertebereich für <small>PL</small> : 0,0001 ... 99 999,9999
	Hinweis: Der <small>PL</small> -Wert gilt für den Satz, in dem er steht. Ist kein <small>PL</small> programmiert, wirkt <small>PL</small> =1.

Ein-/Ausschalten der Polynom-Interpolation

Die Polynom-Interpolation wird im Teileprogramm durch den G-Befehl POLY eingeschaltet.

Der G-Befehl POLY gehört zusammen mit G0, G1, G2, G3, ASPLINE, BSPLINE und CSPLINE zur 1. G-Gruppe.

Achsen, die nur mit Namen und Endpunkt programmiert sind (z.B. X10), werden linear verfahren. Sind alle Achsen eines NC-Satzes so programmiert, verhält sich die Steuerung wie bei G1.

Die Polynom-Interpolation wird durch die Programmierung eines anderen Befehls der 1. G-Gruppe (z. B. G0, G1) implizit wieder ausgeschaltet.

Polynomkoeffizient

Der PO-Wert (PO[]=) bzw. ...=PO(...) gibt alle Polynom-Koeffizienten für eine Achse an. Entsprechend dem Grad des Polynoms werden mehrere Werte durch Kommata getrennt angegeben. Innerhalb eines Satzes sind unterschiedliche Polynomgrade für verschiedene Achsen möglich.

Unterprogramm POLYPATH

Mit POLYPATH(...) kann die Polynom-Interpolation selektiv für bestimmte Achsgruppen freigegeben werden:

Nur Bahnachsen und Zusatzachsen:	POLYPATH ("AXES")
Nur Orientierungsachsen: (beim Verfahren mit Orientierungs-Transformation)	POLYPATH ("VECT")

Die jeweils nicht freigegebenen Achsen werden linear verfahren.

Standardmäßig ist die Polynom-Interpolation für beide Achsgruppen freigegeben.

Durch Programmierung ohne Parameter POLYPATH() wird die Polynom-Interpolation für alle Achsen deaktiviert.

Beispiel

Programmcode	Kommentar
N10 G1 X... Y... Z... F600	
N11 POLY PO[X]=(1,2.5,0.7) PO[Y]=(0.3,1,3.2) PL=1.5	; Polynom-Interpolation ein
N12 PO[X]=(0,2.5,1.7) PO[Y]=(2.3,1.7) PL=3	
...	
N20 M8 H126 ...	
N25 X70 PO[Y]=(9.3,1,7.67) PL=5	; gemischte Angaben für die Achsen
N27 PO[X]=(10,2.5) PO[Y]=(2.3)	; kein PL programmiert; es wirkt PL=1
N30 G1 X... Y... Z.	; Polynom-Interpolation aus
...	

Beispiel: Neue Polynomsyntax

Weiterhin gültige Polynomsyntax	Neue Polynomsyntax
PO[Achsbezeichner]=(.. , ..)	Achsbezeichner=PO(.. , ..)
PO[PHI]=(.. , ..)	PHI=PO(.. , ..)
PO[PSI]=(.. , ..)	PSI=PO(.. , ..)
PO[THT]=(.. , ..)	THT=PO(.. , ..)
PO[]=(.. , ..)	PO(.. , ..)
PO[variable]=IC(.. , ..)	variable=PO IC(.. , ..)

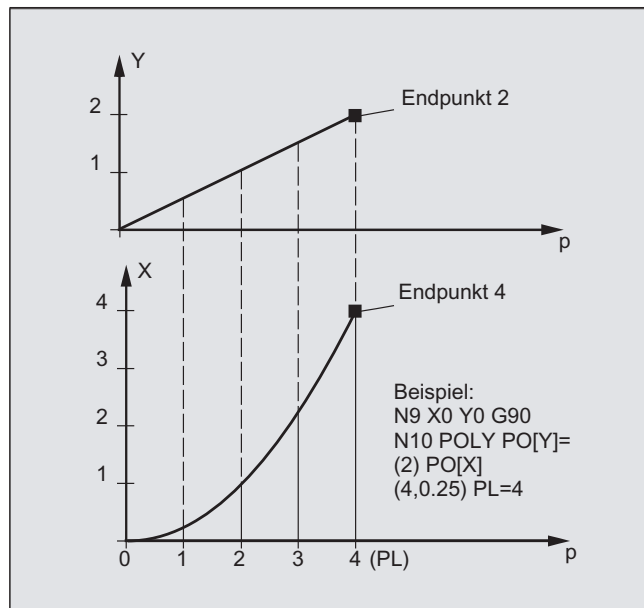
Beispiel: Kurve in der X/Y-Ebene

Programmierung

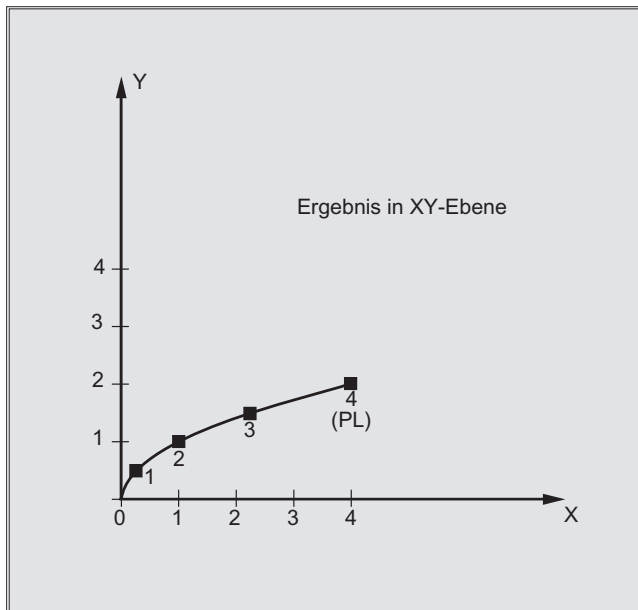
```

Programmcode
N9 X0 Y0 G90 F100
N10 POLY PO[Y]=(2) PO[X]=(4,0.25) PL=4
    
```

Verlauf der Kurven X(p) und Y(p)



Verlauf der Kurve in der XY-Ebene



Beschreibung

Die allgemeine Form der Polynom-Funktion lautet:

$$f(p) = a_0 + a_1p + a_2p^2 + \dots + a_np^n$$

mit: a_n : konstante Koeffizienten
 p : Parameter

In der Steuerung können maximal Polynome 5. Grades programmiert werden:

$$f(p) = a_0 + a_1p + a_2p^2 + a_3p^3 + a_4p^4 + a_5p^5$$

Durch Belegen der Koeffizienten mit konkreten Werten sind verschiedenen Kurvenverläufe, wie Geraden, Parabeln und Potenzfunktionen, erzeugbar.

Eine Gerade wird erzeugt durch $a_2 = a_3 = a_4 = a_5 = 0$:

$$f(p) = a_0 + a_1p$$

Weiter gilt:

a_0 : Achsposition am Ende des vorangehenden Satzes
 $p = PL$
 $a_1 = (x_E - a_0 - a_2 \cdot p^2 - a_3 \cdot p^3) / p$

Es ist möglich Polynome zu programmieren, **ohne** dass die Polynom-Interpolation durch den G-Befehl `POLY` aktiviert wurde. In diesem Fall werden nicht die programmierten Polynome interpoliert, sondern die programmierten Endpunkte der Achsen linear angefahren (`G1`). Erst nach expliziter Aktivierung der Polynom-Interpolation im Teileprogramm (`POLY`) werden die programmierten Polynome auch als solche verfahren.

Besonderheit: Nenner-Polynom

Für die Geometrieachsen kann mit `PO [] = (...)` ohne Angabe eines Achsnamens auch ein gemeinsames Nenner-Polynom programmiert werden, d. h. die Bewegung der Geometrieachsen wird als Quotient zweier Polynome interpoliert.

Damit lassen sich z. B. Kegelschnitte (Kreis, Ellipse, Parabel, Hyperbel) exakt darstellen.

Beispiel:

Programmcode	Kommentar
<code>POLY G90 X10 Y0 F100</code>	; Geometrieachsen verfahren linear auf die Position X10 Y0.
<code>PO[X] = (0, -10) PO[Y] = (10) PO[] = (2, 1)</code>	; Geometrieachsen verfahren im Viertelkreis auf X0 Y10.

Der konstante Koeffizient (a_0) des Nenner-Polynoms wird stets mit 1 angenommen. Der programmierte Endpunkt ist unabhängig von $G90 / G91$.

Aus den programmierten Werten berechnen sich $X(p)$ und $Y(p)$ zu:

$$X(p) = (10 - 10 * p^2) / (1 + p^2)$$

$$Y(p) = 20 * p / (1 + p^2)$$

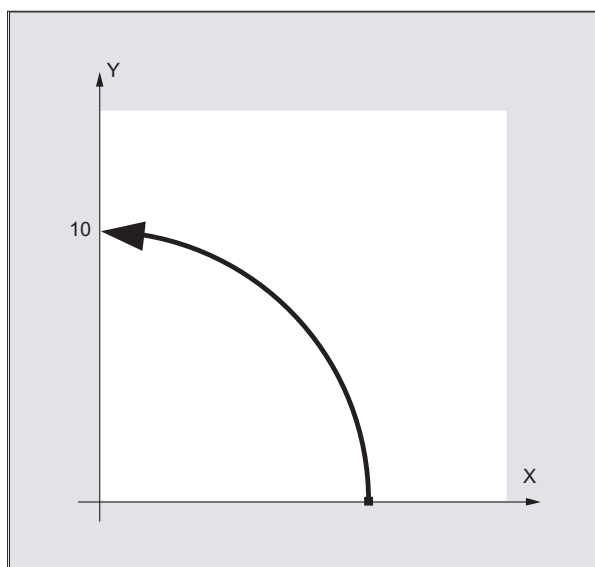
$$\text{mit } 0 \leq p \leq 1$$

Aufgrund der programmierten Anfangspunkte, Endpunkte, Koeffizient a_2 und $PL=1$ ergeben sich folgende Zwischenergebnisse:

$$\text{Zähler (X)} = 10 + 0 * p - 10 * p^2$$

$$\text{Zähler (Y)} = 0 + 20 * p + 0 * p^2$$

$$\text{Nenner} = 1 + p^2$$



Bei eingeschalteter Polynom-Interpolation wird die Programmierung eines Nenner-Polynoms mit Nullstellen innerhalb des Intervalls $[0, PL]$ mit einem Alarm abgelehnt. Auf die Bewegung von Zusatzachsen hat das Nenner-Polynom keinen Einfluss.

Hinweis

Eine Werkzeugradiuskorrektur ist bei der Polynom-Interpolation mit $G41$, $G42$ einschaltbar und wie für Geraden- oder Kreisinterpolation verwendbar.

4.6 Einstellbarer Bahnbezug (SPATH, UPATH)

Funktion

Während Polynominterpolation können vom Anwender zwei unterschiedliche Beziehungen zwischen den geschwindigkeitsbestimmenden FGROUP-Achsen und den übrigen Bahnachsen gewünscht sein: Letztere sollen entweder synchron zum Bahnweg S oder synchron zum Kurvenparameter U der FGROUP-Achsen geführt werden.

Beide Arten der Bahninterpolation werden in unterschiedlichen Applikationen gebraucht und können durch die beiden in der 45. G-Code-Gruppe enthaltenen modal wirksamen Sprachbefehle `SPATH` und `UPATH` eingestellt/umgeschaltet werden.

Syntax

`SPATH`
`UPATH`

Bedeutung

`SPATH`: Bahnbezug für FGROUP-Achsen ist Bogenlänge
`UPATH`: Bahnbezug für FGROUP-Achsen ist Kurvenparameter

Hinweis

`UPATH` und `SPATH` bestimmen auch den Zusammenhang des F-Wort-Polynoms (`FPOLY`, `FCUB`, `FLIN`) mit der Bahnbewegung.

Randbedingungen

Der eingestellte Bahnbezug hat keine Bedeutung:

- bei Linear- und Kreisinterpolation
- in Gewindesätzen
- wenn alle Bahnachsen in FGROUP enthalten sind.

Beispiele

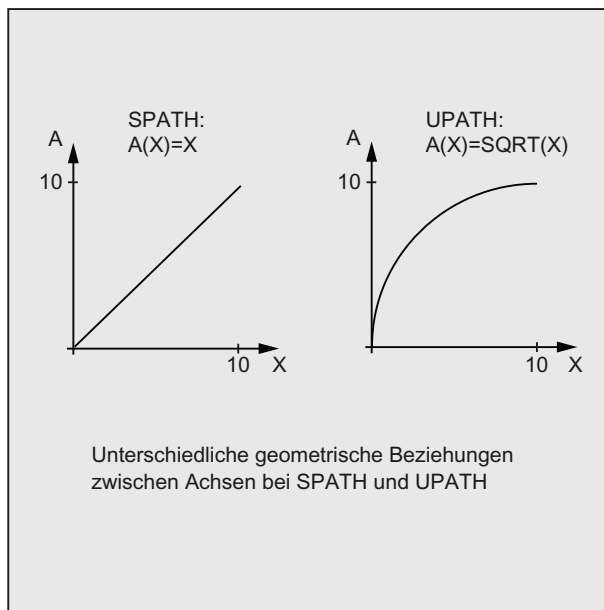
Beispiel 1:

Im nachfolgenden Beispiel wird ein Quadrat mit 20 mm Kantenlänge mit G643 überschleifen. Die maximalen Abweichungen von der exakten Kontur werden dabei durch das achsspezifische Maschinendatum MD33100 \$MA_COMPRESS_POS_TOL[<n>] für jede Achse festgelegt.

Programmcode	Kommentar
N10 G1 X... Y... Z... F500	
N20 G643	; Satzinternes Überschleifen mit G643
N30 XO YO	
N40 X20 Y0	; Kantenlänge (mm) für die Achsen
N50 X20 Y20	
N60 X0 Y20	
N70 X0 YO	
N100 M30	

Beispiel 2:

Das folgende Beispiel illustriert den Unterschied zwischen den beiden Arten der Bewegungsführung. Beide Male sei die Voreinstellung FGROUP(X,Y,Z) aktiv.



Programmcode

```
N10 G1 X0 A0 F1000 SPATH  
N20 POLY PO[X]=(10,10) A10
```

Bzw.:

Programmcode

```
N10 G1 X0 F1000 UPATH  
N20 POLY PO[X]=(10,10) A10
```

Im Satz N20 hängt der Weg S der FGROU- Achsen vom Quadrat des Kurvenparameters U ab. Daher ergeben sich entlang des Wegs von X unterschiedliche Positionen der Synchronachse A, je nachdem, ob SPATH oder UPATH aktiv ist.

Weitere Informationen

Während Polynominterpolation - und damit seien immer die Polynominterpolation im engeren Sinne (POLY), alle Spline-Interpolationsarten (ASPLINE, BSPLINE, CSPLINE) und Linearinterpolation mit Kompressorfunktion (COMPON, COMPCURV) verstanden - sind die Positionen aller Bahnachsen i durch Polynome $\pi_i(U)$ vorgegeben. Der Kurvenparameter U bewegt sich dabei innerhalb eines NC-Satzes von 0 bis 1, ist also normiert.

Durch den Sprachbefehl FGROU können innerhalb der Bahnachsen diejenigen Achsen ausgewählt werden, auf die sich der programmierte Bahnvorschub beziehen soll. Eine Interpolation mit konstanter Geschwindigkeit auf dem Weg S dieser Achsen bedeutet während Polynominterpolation jedoch in der Regel eine nicht konstante Änderung des Kurvenparameters U.

Steuerungsverhalten bei Reset und Maschinen-/Optionsdaten

Nach Reset ist der durch MD20150 \$MC_GCODE_RESET_VALUES[44] bestimmte G-Code wirksam (45. G-Code-Gruppe). Um kompatibel zu bestehenden Anlagen zu bleiben, wird hier als Standardwert SPATH voreingestellt.

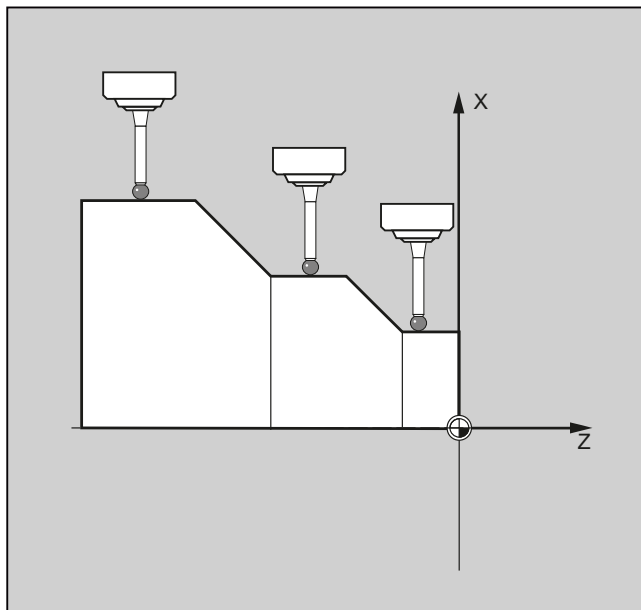
Der Grundstellungswert für die Art des Überschleifens wird mit MD20150 \$MC_GCODE_RESET_VALUES[9] festgelegt (10. G-Code-Gruppe).

Das achsspezifische Maschinendatum MD33100 \$MA_COMPRESS_POS_TOL[<n>] hat eine erweiterte Bedeutung: es enthält die Toleranzen für die Kompressorfunktion und für das Überschleifen mit G642.

4.7 Messen mit schaltendem Taster (MEAS, MEAW)

Funktion

Mit der Funktion "Messen mit schaltendem Taster" werden Istpositionen am Werkstück angefahren und bei der Schaltflanke des Messtasters werden für alle im Messatz programmierten Achsen die Positionen gemessen und für jede Achse in die entsprechende Speicherzelle geschrieben.



Für die Programmierung der Funktion stehen die beiden folgenden festen Adressen zur Verfügung:

- MEAS

Mit MEAS wird der Restweg zwischen Ist- und Sollposition gelöscht.

- MEAW

MEAW wird für Messaufgaben eingesetzt, bei denen in jedem Fall die programmierte Position angefahren werden soll.

MEAS und MEAW sind satzweise wirksam und werden zusammen mit Bewegungsanweisungen programmiert. Vorschub und Interpolationsart (G0, G1, ...), ebenso wie die Anzahl der Achsen, müssen dabei dem jeweiligen Messproblem angepasst sein.

Syntax

MEAS=<TE> G... X... Y... Z...

MEAW=<TE> G... X... Y... Z...

Bedeutung

MEAS:	Befehl: Messen mit Restweglöschen Wirksamkeit: satzweise
MEAW:	Befehl: Messen ohne Restweglöschen Wirksamkeit: satzweise
<TE>:	Trigger-Ereignis zur Auslösung der Messung Typ: INT Wertebereich: -2, -1, 1, 2 Bedeutung: <ul style="list-style-type: none"> (+)1 steigende Flanke von Messtaster 1 (auf Messeingang 1) -1 fallende Flanke von Messtaster 1 (auf Messeingang 1) (+)2 steigende Flanke von Messtaster 2 (auf Messeingang 2) -2 fallende Flanke von Messtaster 2 (auf Messeingang 2) Hinweis: Es existieren maximal 2 Messtaster (je nach Ausbaustufe).
G...:	Interpolationsart, z. B. G0, G1, G2 oder G3
X... Y... Z...:	Endpunkte in kartesischen Koordinaten

Beispiel

Programmcode	Kommentar
N10 MEAS=1 G1 F1000 X100 Y730 Z40	; Messsatz mit Messtaster des ersten Messeingangs und Geradeninterpolation. Vorlaufstopp wird automatisch erzeugt.
...	

Weitere Informationen

Messauftragsstatus

Ist im Programm eine Auswertung erforderlich, ob der Messtaster geschaltet hat oder nicht, kann die Zustandsvariable \$AC_MEA[<n>] (<n> = Nummer des Messtasters) abgefragt werden:

Wert	Bedeutung
0	Messauftrag nicht erfüllt
1	Messauftrag erfolgreich beendet (Messtaster hat geschaltet)

Hinweis

Wird der Messtaster im Programm ausgelenkt, wird die Variable auf 1 gesetzt. Beim Start eines Messsatzes wird die Variable automatisch auf den Anfangszustand des Tasters gesetzt.

Messwerteaufnahme

Es werden die Positionen aller verfahrenen Bahn- und Positionierachsen des Satzes (maximale Anzahl an Achsen je nach Steuerungskonfiguration) erfasst. Bei MEAS wird die Bewegung nach dem Schalten des Messtasters definiert abgebremst.

Hinweis

Ist in einem Messsatz eine Geometrieachse programmiert, werden die Messwerte für alle aktuellen Geometrieachsen abgelegt.

Ist in einem Messsatz eine an einer Transformation beteiligte Achse programmiert, werden die Messwerte aller an dieser Transformation beteiligten Achsen abgelegt.

Messergebnisse lesen

Die Messergebnisse für die mit Messtaster erfassten Achsen können über die folgenden Systemvariablen gelesen werden:

- \$AA_MM[<Achse>]
Messergebnisse im Maschinenkoordinatensystem
- \$AA_MW[<Achse>]
Messergebnisse im Werkstückkoordinatensystem

Beim Lesen dieser Variablen wird intern kein Vorlaufstopp erzeugt.

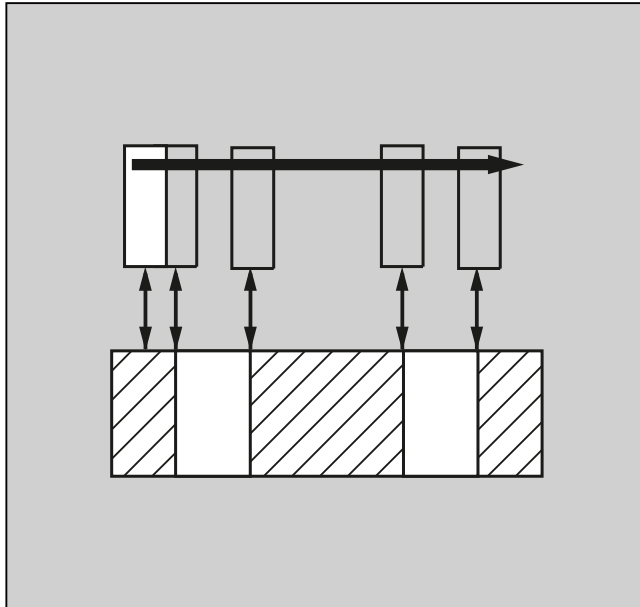
Hinweis

Mit STOPRE muss im NC-Programm an geeigneter Stelle ein Vorlaufstopp programmiert werden. Ansonsten werden falsche Werte gelesen.

4.8 Axiales Messen (MEASA, MEAWA, MEAC) (Option)

Funktion

Beim axialen Messen können mehrere Messtaster und mehrere Messsysteme benutzt werden.



Für die Programmierung der Funktion stehen die Schlüsselwörter `MEASA`, `MEAWA` und `MEAC` zur Verfügung.

Mit `MEASA` bzw. `MEAWA` werden für die jeweils programmierte Achse bis zu vier Messwerte pro Messung erfasst und passend zum Trigger-Ereignis in Systemvariablen abgelegt.

Kontinuierliche Messaufträge können mit `MEAC` durchgeführt werden. In diesem Fall werden die Messergebnisse in FIFO-Variablen abgelegt.

Syntax

```
MEASA [<Achse>] = (<Modus>, <TE1>, ..., <TE4>)
MEAWA [<Achse>] = (<Modus>, <TE1>, ..., <TE4>)
MEAC [<Achse>] = (<Modus>, <Messspeicher>, <TE1>, ..., <TE4>)
```

Hinweis

`MEASA` und `MEAWA` sind satzweise wirksam und können zusammen in einem Satz programmiert werden. Wird dagegen `MEASA/MEAWA` zusammen mit `MEAS/MEAW` in einem Satz programmiert, kommt es zu einer Fehlermeldung.

Bedeutung

MEASA:	Schlüsselwort: Axiales Messen mit Restweglöschen Wirksamkeit: satzweise
MEAWA:	Schlüsselwort: Axiales Messen ohne Restweglöschen Wirksamkeit: satzweise
MEAC:	Schlüsselwort: Axiales kontinuierliches Messen ohne Restweglöschen Wirksamkeit: satzweise
<Achse>:	Name der zur Messung verwendeten Kanalachse
<Modus>:	Zweistellige Ziffer zur Angabe des Betriebsmodus (Messmodus und Messsystem) Einerdekade (Messmodus): 0 Messauftrag abbrechen. 1 Bis zu 4 verschiedene gleichzeitig aktivierbare Trigger-Ereignisse. 2 Bis zu 4 nacheinander aktivierbare Trigger-Ereignisse. 3 Bis zu 4 nacheinander aktivierbare Trigger-Ereignisse, jedoch keine Überwachung von Trigger-Ereignis 1 beim Start (Alarme 21700/21703 werden unterdrückt). Hinweis: Dieser Modus ist bei MEAC nicht möglich. Zehnerdekade (Messsystem): 0 (oder keine Angabe) aktives Messsystem 1 Messsystem 1 2 Messsystem 2 3 beide Messsysteme
<TE>:	Trigger-Ereignis zur Auslösung der Messung Typ: INT Wertebereich: -2, -1, 1, 2 Bedeutung: (+)1 steigende Flanke von Messtaster 1 -1 fallende Flanke von Messtaster 1 (+)2 steigende Flanke von Messtaster 2 -2 fallende Flanke von Messtaster 2
<Messspeicher>:	Nummer des FIFO (Umlaufspeichers)

Beispiele

Beispiel 1: Axiales Messen mit Restweglöschen im Modus 1 (Auswertung in zeitlicher Reihenfolge)**a) mit 1 Messsystem**

Programmcode	Kommentar
...	
N100 MEASA[X]=(1,1,-1) G01 X100 F100	; Messen im Modus 1 mit aktivem Messsystem. Warten auf Messsignal mit steigender/fallender Flanke von Messtaster 1 auf dem Verfahrweg nach X=100.
N110 STOPRE	; Vorlaufstopp
N120 IF \$AC_MEA[1]==FALSE GOTOF ENDE	; Erfolg der Messung kontrollieren.
N130 R10=\$AA_MM1 [X]	; Zum ersten programmierten Trigger-Ereignis (steigende Flanke) gehörigen Messwert speichern.
N140 R11=\$AA_MM2 [X]	; Zum zweiten programmierten Trigger-Ereignis (fallende Flanke) gehörigen Messwert speichern.
N150 ENDE:	

b) mit 2 Messsystemen

Programmcode	Kommentar
...	
N200 MEASA[X]=(31,1,-1) G01 X100 F100	; Messen im Modus 1 mit beiden Messsystemen. Warten auf Messsignal mit steigender/fallender Flanke von Messtaster 1 auf dem Verfahrweg nach X=100.
N210 STOPRE	; Vorlaufstopp
N220 IF \$AC_MEA[1]==FALSE GOTOF ENDE	; Erfolg der Messung kontrollieren.
N230 R10=\$AA_MM1 [X]	; Messwert des Messsystems 1 bei steigender Flanke speichern.
N240 R11=\$AA_MM2 [X]	; Messwert des Messsystems 2 bei steigender Flanke speichern.
N250 R12=\$AA_MM3 [X]	; Messwert des Messsystems 1 bei fallender Flanke speichern.
N260 R13=\$AA_MM4 [X]	; Messwert des Messsystems 2 bei fallender Flanke speichern.
N270 ENDE:	

Beispiel 2: Axiales Messen mit Restweglößen im Modus 2 (Auswertung in programmierter Reihenfolge)

Programmcode	Kommentar
...	
N100 MEASA[X]=(2,1,-1,2,-2) G01 X100 F100	; Messen im Modus 2 mit aktivem Messsystem. Warten auf Messsignal in der Reihenfolge steigende Flanke von Messtaster 1, fallende Flanke Messtaster 1, steigende Flanke von Messtaster 2, fallende Flanke Messtaster 2 auf dem Verfahrensweg nach X=100.
N110 STOPRE	; Vorlaufstopp
N120 IF \$AC_MEA[1]==FALSE GOTOF MESSTASTER2	; Erfolg der Messung mit Messtaster 1 kontrollieren.
N130 R10=\$AA_MM1[X]	; Zum ersten programmierten Trigger-Ereignis (steigende Flanke Messtaster 1) gehörigen Messwert speichern.
N140 R11=\$AA_MM2[X]	; Zum zweiten programmierten Trigger-Ereignis (steigende Flanke Messtaster 1) gehörigen Messwert speichern.
N150 MESSTASTER2:	
N160 IF \$AC_MEA[2]==FALSE GOTOF ENDE	; Erfolg der Messung mit Messtaster 2 kontrollieren.
N170 R12=\$AA_MM3[X]	; Zum dritten programmierten Trigger-Ereignis (steigende Flanke Messtaster 2) gehörigen Messwert speichern.
N180 R13=\$AA_MM4[X]	; Zum vierten programmierten Trigger-Ereignis (steigende Flanke Messtaster 2) gehörigen Messwert speichern.
N190 ENDE:	

Beispiel 3: Axiales kontinuierliches Messen im Modus 1 (Auswertung in zeitlicher Reihenfolge)

a) Messen von bis zu 100 Messwerten

Programmcode	Kommentar
...	
N110 DEF REAL MESSWERT[100]	
N120 DEF INT Schleife=0	
N130 MEAC[X]=(1,1,-1) G01 X1000 F100	; Messen im Modus 1 mit aktivem Messsystem, Speichern der Messwerte unter \$AC_FIFO1, Warten auf Messsignal mit fallender Flanke von Messtaster 1 auf dem Verfahrensweg nach X=1000.
N135 STOPRE	
N140 MEAC[X]=(0)	; Messung nach Erreichen der Achsposition abbrechen.
N150 R1=\$AC_FIFO1[4]	; Anzahl aufgelaufener Messwerte in Parameter R1 speichern.
N160 FOR Schleife=0 TO R1-1	
N170 MESSWERT[Schleife]=\$AC_FIFO1[0]	; Messwerte aus dem \$AC_FIFO1 auslesen und abspeichern.
N180 ENDFOR	

b) Messen mit Restweglöschen nach 10 Messwerten

Programmcode	Kommentar
...	
N10 WHEN \$AC_FIFO1[4] >=10 DO MEAC[x]=(0) DELDTG(x)	; Restweg löschen.
N20 MEAC[x]=(1,1,1,-1) G01 X100 F500	
N30 MEAC[X]=(0)	
N40 R1=\$AC_FIFO1[4]	; Anzahl Messwerte.
...	

c) Messen einer fallenden/steigenden Zahnflanke mit 2 Messtastern

Programmcode	Kommentar
...	
N110 DEF REAL MESSWERT[16]	
N120 DEF INT Schleife=0	
N130 MEAC[X]=(1,1,-1,2) G01 X100 F100	; Messen im Modus 1 mit aktivem Messsystem, Speichern der Messwerte unter \$AC_FIFO1, Warten auf Messsignal in der Reihenfolge fallende Flanke von Messtaster 1, steigende Flanke Messtaster 2, auf dem Verfahrensweg nach X=100.
N140 STOPRE	; Vorlaufstopp
N150 MEAC[X]=(0)	; Messung nach Erreichen der Achsposition abbrechen.
N160 R1=\$AC_FIFO1[4]	; Anzahl aufgelaufener Messwerte in Parameter R1 speichern.
N170 FOR Schleife=0 TO R1-1	
N180 MESSWERT[Schleife]=\$AC_FIFO1[0]	; Messwerte aus dem \$AC_FIFO1 auslesen und abspeichern.
N190 ENDFOR	

Weitere Informationen**Messauftrag**

Die Programmierung eines Messauftrags kann im Teileprogramm oder aus einer Synchronaktion (siehe Kapitel "Synchronaktionen (Seite 569)") heraus erfolgen. Pro Achse kann dabei zu ein- und demselben Zeitpunkt nur ein Messauftrag aktiv sein.

Hinweis

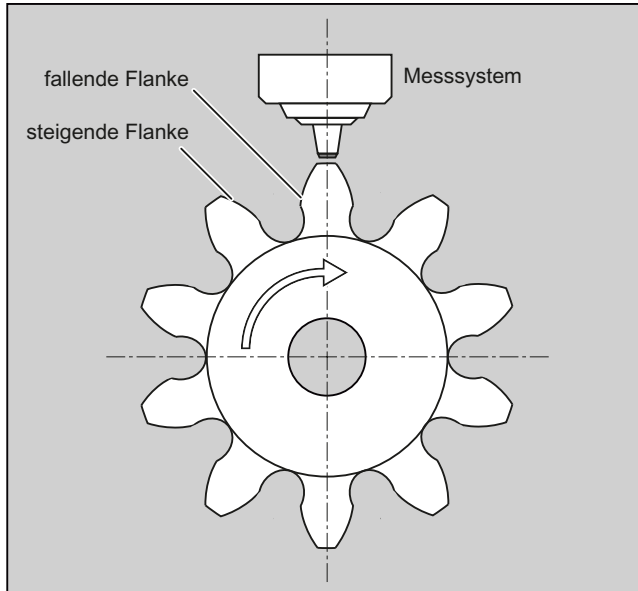
Der Vorschub ist dem jeweiligen Messproblem anzupassen.

Bei MEASA und MEAWA können korrekte Ergebnisse nur bei Vorschüben gewährleistet werden, bei denen nicht mehr als ein gleiches und nicht mehr als 4 verschiedene Trigger-Ereignisse pro Lageregler-Takt eintreffen.

Beim kontinuierlichen Messen mit MEAC darf das Verhältnis zwischen Interpolationstakt und Lageregler-Takt nicht größer als 1:8 werden.

Trigger-Ereignis

Ein Trigger-Ereignis setzt sich zusammen aus der Nummer des Messtasters und dem Auslösekriterium (steigende oder fallende Flanke) des Messsignals.



Für jede Messung können jeweils bis zu 4 Trigger-Ereignisse der angesprochenen Messtaster verarbeitet werden, also bis zu zwei Messtaster mit je zwei Messflanken. Die Reihenfolge der Verarbeitung sowie die maximale Anzahl der Trigger-Ereignisse sind dabei abhängig vom gewählten Modus.

Hinweis

Für Messmodus 1 gilt: Ein gleiches Trigger-Ereignis darf nur einmal in einem Messauftrag programmiert werden!

Bei MEAC kann die Anzahl der Messwerte pro Lageregler-Takt auf 8 von steigender Flanke und 8 von fallender Flanke je Messtaster erhöht werden. Damit lassen sich höhere Vorschübe und Drehzahlen realisieren.

Literatur:

Funktionshandbuch Erweiterungsfunktionen; Messen (M5), Kapitel: Axiales Messen

Betriebsmodus

Mit der ersten Ziffer (Zehnerdekade) des Betriebsmodus wird das gewünschte Messsystem angewählt. Ist nur ein Messsystem vorhanden, jedoch das zweite programmiert, wird automatisch das vorhandene eingesetzt.

Mit der zweiten Ziffer (Einerdekade) wird der gewünschte Messmodus angewählt. Damit wird der Messvorgang an die Möglichkeiten der jeweiligen Steuerung angepasst:

- **Modus 1**

Die Auswertung der Trigger-Ereignisse erfolgt in der zeitlichen Reihenfolge ihres Auftretens. In diesem Modus ist bei Einsatz von Sechssachsbaugruppen nur ein Trigger-Ereignis programmierbar bzw. wird bei Angabe mehrerer Trigger-Ereignisse automatisch in Modus 2 umgesetzt (ohne Meldung).

- **Modus 2**

Die Auswertung der Trigger-Ereignisse erfolgt in der programmierten Reihenfolge.

- **Modus 3**

Die Auswertung der Trigger-Ereignisse erfolgt in der programmierten Reihenfolge, jedoch keine Überwachung von Trigger-Ereignis 1 beim START.

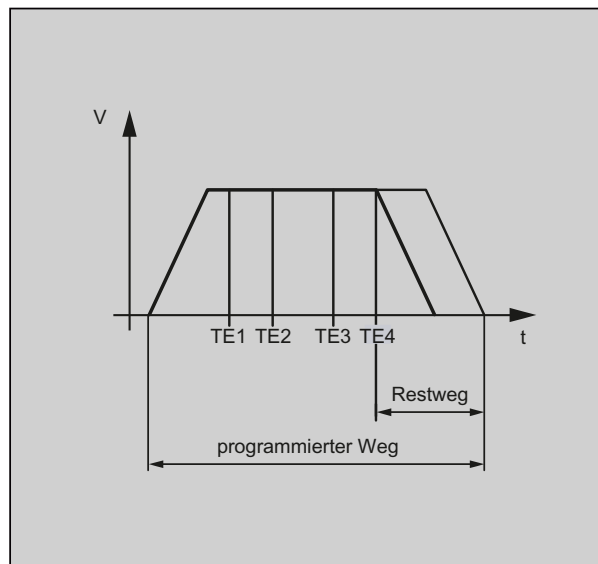
Hinweis

Bei Einsatz von 2 Messsystemen sind nur zwei Trigger-Ereignisse programmierbar.

Messen mit und ohne Restweglöschen

Bei der Programmierung von MEASA wird Restweglöschen erst nach der Erfassung aller geforderten Messwerte durchgeführt.

Für spezielle Messaufgaben, bei denen in jedem Fall die programmierte Position angefahren werden soll, wird MEAWA eingesetzt.



Hinweis

MEASA ist nicht in Synchronaktionen programmierbar. Ersatzweise kann MEAWA plus Restweglöschen als Synchronaktion programmiert werden.

Wird der Messauftrag mit MEAWA aus den Synchronaktionen gestartet, sind die Messwerte nur im Maschinen-Koordinatensystem verfügbar.

Messergebnisse für MEASA, MEAWA

Die Messergebnisse stehen unter folgenden Systemvariablen zur Verfügung:

- im Maschinen-Koordinatensystem:

\$AA_MM1 [<Achse>] Messwert des programmierten Messsystems bei Trigger-Ereignis 1
... ...
\$AA_MM4 [<Achse>] Messwert des programmierten Messsystems bei Trigger-Ereignis 4

- im Werkstück-Koordinatensystem:

\$AA_WM1 [<Achse>] Messwert des programmierten Messsystems bei Trigger-Ereignis 1
... ...
\$AA_WM4 [<Achse>] Messwert des programmierten Messsystems bei Trigger-Ereignis 4

Hinweis

Beim Lesen dieser Variablen wird intern kein Vorlaufstopp erzeugt. Mit `STOPRE` muss an geeigneter Stelle ein Vorlaufstopp programmiert werden. Ansonsten werden falsche Werte eingelesen.

Geometrieachsen / Transformationen

Soll das axiale Messen für eine Geometrieachse gestartet werden, muss der gleiche Messauftrag explizit für alle restlichen Geometrieachsen programmiert werden. Das Gleiche gilt für Achsen, die an einer Transformation beteiligt sind.

Beispiele:

N10 MEASA [Z] = (1, 1) MEASA [Y] = (1, 1) MEASA [X] = (1, 1) G0 Z100

oder

N10 MEASA [Z] = (1, 1) POS [Z] = 100

Messauftrag mit 2 Messsystemen

Wird ein Messauftrag mit zwei Messsystemen durchgeführt, wird jedes der beiden möglichen Trigger-Ereignisse von beiden Messsystemen der jeweiligen Achse erfasst. Die Belegung der reservierten Variablen ist damit vorgegeben:

\$AA_MM1 [<Achse>]	bzw.	\$AA_MW1 [<Achse>]	Messwert von Messsystem 1 bei Trigger-Ereignis 1
\$AA_MM2 [<Achse>]	bzw.	\$AA_MW2 [<Achse>]	Messwert von Messsystem 2 bei Trigger-Ereignis 1
\$AA_MM3 [<Achse>]	bzw.	\$AA_MW3 [<Achse>]	Messwert von Messsystem 1 bei Trigger-Ereignis 2
\$AA_MM4 [<Achse>]	bzw.	\$AA_MW4 [<Achse>]	Messwert von Messsystem 2 bei Trigger-Ereignis 2

Systemvariablen

Der Messtasterstatus steht unter der folgenden Systemvariablen zur Verfügung:

\$A_PROBE[<n>]

Wert	Bedeutung
1	Messtaster ausgelenkt
0	Messtaster nicht ausgelenkt

Die Messtasterbegrenzung steht unter der folgenden Systemvariablen zur Verfügung:

\$A_PROBE_LIMITED[<n>]

Wert	Bedeutung
1	Messtasterbegrenzung aktiv
0	Messtasterbegrenzung inaktiv

<n> = Messtaster

Literatur:

Listenhandbuch Systemvariablen

Messauftragsstatus bei MEASA, MEAWA

Ist im Programm eine Auswertung erforderlich, so kann der Messauftragsstatus über \$AC_MEA[<n>], mit <n> = Nummer des Messtasters, abgefragt werden. Sobald alle in einem Satz programmierten Trigger-Ereignisse der Messtaster <n> erfolgt sind, liefert diese Variable den Wert 1. Anderenfalls ist der Wert 0.

Hinweis

Wird Messen aus Synchronaktionen gestartet, wird \$AC_MEA nicht mehr aktualisiert. In diesem Fall sind das neue PLC-Nahtstellensignal DB31, ... DBX62.3 bzw. die gleichwertige Variable \$AA_MEA[<Achse>] abzufragen.

Bedeutung:

\$AA_MEA==1: Messen aktiv

\$AA_MEA==0: Messen nicht aktiv

Kontinuierliches Messen (MEAC)

Die Messwerte liegen bei MEAC im Maschinenkoordinatensystem vor und werden im angegebenen FIFO[n]-Speicher (Umlaufspeicher) abgelegt. Sind für die Messung zwei Messtaster projiziert, werden die Messwerte des zweiten Messtasters getrennt im zusätzlich dafür projizierten (über MD einstellbar) FIFO[n+1]-Speicher abgelegt.

Der FIFO-Speicher ist ein Umlaufspeicher, in den Messwerte im Umlaufprinzip in \$AC_FIFO-Variablen eingetragen werden (siehe Kapitel "Synchronaktionen (Seite 569)").

Hinweis

Der FIFO-Inhalt kann nur einmal aus dem Umlaufspeicher ausgelesen werden. Zur Mehrfachverwendung der Messdaten müssen diese in den Anwenderdaten zwischengespeichert werden.

Überschreitet die Anzahl der Messwerte für den FIFO-Speicher die im Maschinendatum festgelegte Höchstzahl, so wird die Messung automatisch beendet.

Endloses Messen lässt sich durch zyklisches Auslesen von Messwerten realisieren. Das Auslesen muss dabei mindestens in der gleichen Häufigkeit wie der Eingang von neuen Messwerten erfolgen.

Literatur:

- Funktionshandbuch Synchronaktionen; Ausführliche Beschreibung, Kapitel: Parameter (\$AC_FIFO)
- Funktionshandbuch Erweiterungsfunktionen; Messen (M5), Kapitel: Axiales Messen

Schutz vor Fehlerprogrammierungen

Folgende Fehlprogrammierungen werden erkannt und mit einem Fehler angezeigt:

- MEASA/MEAWA zusammen mit MEAS/MEAW in einem Satz programmiert

Beispiel:

```
N01 MEAS=1 MEASA[X]=(1,1) G01 F100 POS[X]=100
```

- MEASA/MEAWA mit Parameteranzahl <2 oder >5

Beispiel:

```
N01 MEAWA[X]=(1) G01 F100 POS[X]=100
```

- MEASA/MEAWA mit Trigger-Ereignis ungleich 1/ -1/ 2/ -2

Beispiel:

```
N01 MEASA[B]=(1,1,3) B100
```

- MEASA/MEAWA mit falschem Modus

Beispiel:

```
N01 MEAWA[B]=(4,1) B100
```

- MEASA/MEAWA mit doppelt programmiertem Trigger-Ereignis

Beispiel:

```
N01 MEASA[B]=(1,1,-1,2,-1) B100
```

- MEASA/MEAWA und fehlende Geometrieachse

Beispiel:

```
N01 MEASA[X]=(1,1) MEASA[Y]=(1,1) G01 X50 Y50 Z50 F100 ;GEO-Achse X/Y/Z
```

- Uneinheitlicher Messauftrag bei Geometrieachsen

Beispiel:

```
N01 MEASA[X]=(1,1) MEASA[Y]=(1,1) MEASA[Z]=(1,1,2) G01 X50 Y50 Z50 F100
```

4.9 Spezielle Funktionen für den OEM-Anwender (OMA1 ... OMA5, OEMIPO1, OEMIPO2, G810 ... G829)

OEM-Adressen

Die Bedeutung der OEM-Adressen bestimmt der OEM-Anwender. Die Funktionalität wird über Compile-Zyklen eingebracht. 5 OEM-Adressen sind reserviert (OMA1 ... OMA5). Die Adressbezeichner sind einstellbar. OEM-Adressen sind in jedem Satz zulässig.

Reservierte G-Funktionsaufrufe

Für den OEM-Anwender sind folgende G-Funktionsaufrufe reserviert:

- OEMIPO1, OEMIPO2 (aus G-Funktionsgruppe 1)
- G810 ... G819 (G-Funktionsgruppe 31)
- G820 ... G829 (G-Funktionsgruppe 32)

Die Funktionalität wird über Compile-Zyklen eingebracht.

Funktionen und Unterprogramme

Zusätzlich können OEM-Anwender auch vordefinierte Funktionen und Unterprogramme mit Parameterübergabe anlegen.

Hinweis

Werkstücksimulation

Bis SW 4.4 werden keine, ab SW 4.4 nur ausgesuchte Compile-Zyklen (CC) bei der Werkstücksimulation unterstützt.

Sprachbefehle im Teileprogramm von nicht unterstützten Compile-Zyklen (OMA1 ... OMA5, OEMIPO1/2, G810 ... G829, eigene Prozeduren und Funktionen) führen daher ohne individuelle Behandlung zur Alarmmeldung und zum Abbruch der Simulation.

Lösung: Die fehlenden CC-spezifischen Sprachelemente im Teileprogramm individuell behandeln (\$P_SIM-Abfrage).

Beispiel:

```
N1 G01 X200 F500

IF (1== $P_SIM)

N5 X300 ;bei Simulation CC nicht aktiv

ELSE

N5 X300 OMA1=10

ENDIF
```

4.10 Vorschubreduzierung mit Eckenverzögerung (FENDNORM, G62, G621)

Funktion

Bei der automatischen Eckenverzögerung wird der Vorschub glockenförmig kurz vor der betreffenden Ecke abgesenkt. Außerdem kann das Ausmaß des für die Bearbeitung relevanten Werkzeugverhaltens über Settingdaten parametrierbar werden. Dies sind:

- Beginn und Ende der Vorschubreduzierung
- Override, mit dem der Vorschub reduziert wird
- Erkennung der relevanten Ecke

Als relevante Ecken werden diejenigen Ecken berücksichtigt, deren Innenwinkel kleiner als die über Settingdatum parametrisierte Ecke ist.

Mit dem Defaultwert `FENDNORM` wird die Funktion des automatischen Eckenoverride ausgeschaltet.

Literatur:

/FBFA/ Funktionsbeschreibung ISO-Dialekte

Syntax

`FENDNORM`

`G62 G41`

`G621`

Bedeutung

`FENDNORM` Automatischer Eckenverzögerung aus

`G62` Eckenverzögerung an Innenecken bei aktiver Werkzeugradiuskorrektur

`G621` Eckenverzögerung an allen Ecken bei aktiver Werkzeugradiuskorrektur

G62 wirkt nur an den Innenecken mit

- aktiver Werkzeugradiuskorrektur `G41`, `G42` und
- aktiven Bahnsteuerbetrieb `G64`, `G641`

Die entsprechende Ecke wird mit dem abgesenkten Vorschub angefahren, der sich ergibt aus:

$F * (\text{Override zur Vorschubreduzierung}) * \text{Vorschuboverride}$

Die maximal mögliche Vorschubabsenkung wird genau dann erreicht, wenn das Werkzeug, bezogen auf die Mittelpunktswegbahn, den Richtungswechsel an der betreffenden Ecke vornehmen soll.

G621 wirkt analog zu G62 an jeder Ecke, der durch `FGROUP` festgelegten Achsen.

4.11 Programmierbares Bewegungsendekriterium (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA)

Funktion

Ähnlich dem Satzwechselkriterium bei Bahninterpolation (G601, G602 und G603) kann das Bewegungsendekriterium bei Einzelachsinterpolation in einem Teileprogramm bzw. in Synchronaktionen für Kommando-/PLC-Achsen programmiert werden.

Je nachdem, welches Bewegungsendekriterium eingestellt ist, werden Teileprogrammsätze bzw. Technologiezyklussätze mit Einzelachsbewegungen unterschiedlich schnell beendet. Gleiches gilt für PLC über FC15/16/18.

Syntax

```

FINEA [<Achse>]
COARSEA [<Achse>]
IPOENDA [<Achse>]
IPOBRKA (<Achse> [, <Zeitpunkt>])
ADISPOSA (<Achse> [, <Modus>, <Fenstergröße>])
    
```

Bedeutung

FINEA:	Bewegungsendekriterium: "Genauhalt fein"
	Wirksamkeit: modal
COARSEA:	Bewegungsendekriterium: "Genauhalt grob"
	Wirksamkeit: modal
IPOENDA:	Bewegungsendekriterium: "Interpolator-Stopp"
	Wirksamkeit: modal
IPOBRKA:	Satzwechselkriterium: Bremsrampe
	Wirksamkeit: modal
ADISPOSA:	Toleranzfenster zum Bewegungsendekriterium
	Wirksamkeit: modal
<Achse>:	Kanalachsname (X, Y,)
<Zeitpunkt>:	Zeitpunkt des Satzwechsels, bezogen auf die Bremsrampe in %:
	<ul style="list-style-type: none"> • 100% = Beginn der Bremsrampe • 0% = Ende der Bremsrampe, gleichbedeutend mit IPOENDA
	Typ: REAL
<Modus>:	Bezug des Toleranzfensters
	Wertebereich: 0 Toleranzfenster nicht aktiv
	1 Toleranzfenster bezüglich Sollposition
	2 Toleranzfenster bezüglich Istposition
	Typ: INT
<Fenstergröße>:	Größe des Toleranzfensters
	Typ: REAL

Beispiele

Beispiel 1: Bewegungsendekriterium: "Interpolator-Stopp"

Programmcode

```
; Fahre Positionierachse X auf 100, Geschwindigkeit 200 m/min, Beschleunigung 90%,  
; Bewegungsendekriterium: Interpolator-Stopp  
N110 G01 POS[X]=100 FA[X]=200 ACC[X]=90 IPOENDA[X]  
  
; Synchronaktion:  
; IMMER WENN: Eingang 1 gesetzt ist  
; DANN Fahre Positionierachse X auf 50, Geschwindigkeit 200 m/min, Beschleunigung 140%,  
; Bewegungsendekriterium: Interpolator-Stopp  
N120 EVERY $A_IN[1] DO POS[X]=50 FA[X]=200 ACC[X]=140 IPOENDA[X]
```

Beispiel 2: Satzwechselkriterium: "Bremsrampe"

Programmcode

Kommentar

N40 POS[X]=100	; Defaulteinstellung wirksam ; Positionierbewegung von X auf Position 100 ; Satzwechselkriterium: Genauhalt fein
N20 IPOBRKA(X,100)	; Satzwechselkriterium: "Bremsrampe", ; 100% = Beginn der Bremsrampe
N30 POS[X]=200	; Satzwechsel erfolgt, sobald Achse X zu bremsen beginnt
N40 POS[X]=250	; Achse X brems nicht weiter auf Position 200, sondern fährt ; weiter auf Position 250. ; Sobald die Achse zu bremsen beginnt erfolgt der Satzwechsel
N50 POS[X]=0	; Achse X brems und fährt auf Position 0 zurück ; Satzwechsel erfolgt bei Position 0 und "Genauhalt fein"
N60 X10 F100	; Achse X fährt als Bahnachse auf Position 10

Weitere Informationen

Systemvariable für Bewegungsendekriterium

Das wirksame Bewegungsendekriterium kann über die Systemvariable \$AA_MOTEND gelesen werden.

Literatur: /LIS2sl/ Listenhandbuch, Buch 2

Satzwechselkriterium: "Bremsrampe" (IPOBRKA)

Ist bei der Aktivierung des Satzwechselkriteriums "Bremsrampe" für den optionalen Satzwechselzeitpunkt ein Wert programmiert, wird dieser für die nächste Positionierbewegung wirksam und hauptlaufsynchon in das Settingdatum geschrieben. Ist für den Satzwechselzeitpunkt kein Wert angegeben, wird der aktuelle Wert des Settingdatums wirksam.

SD43600 \$SA_IPOBRAKE_BLOCK_EXCHANGE

Mit der nächsten Programmierung eines axialen Bewegungsendekriteriums (FINEA, COARSEA, IPOENDA) wird IPOBRKA für die entsprechende Achse deaktiviert.

Zusätzliches Satzwechselkriterium: "Toleranzfenster" (ADISPOSA)

Mit ADISPOSA kann als zusätzliches Satzwechselkriterium ein Toleranzfenster um den Satzzeitpunkt (wahlweise Ist- oder Sollposition) definiert werden. Für den Satzwechsel müssen dann beide Bedingungen erfüllt sein:

- Satzwechselkriterium: "Bremsrampe"
- Satzwechselkriterium: "Toleranzfenster"

Literatur

Weitere Informationen zum Satzwechselkriterium von Positionierachsen siehe:

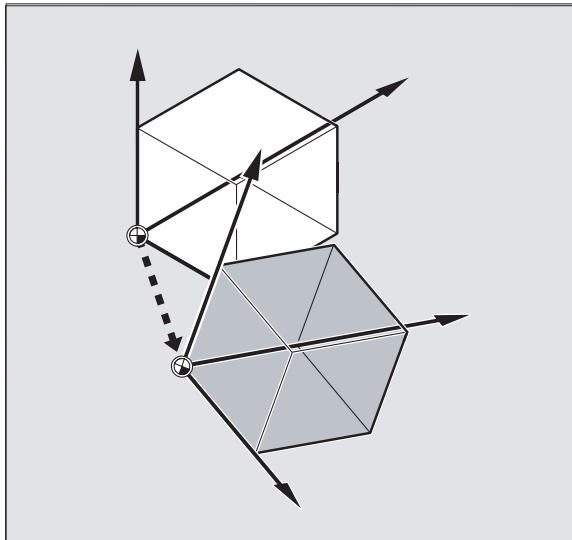
- Funktionshandbuch Erweiterungsfunktionen; Positionierachsen (P2)
- Programmierhandbuch Grundlagen; Kapitel "Vorschubregelung"

Koordinatentransformationen (Frames)

5.1 Koordinatentransformation über Framevariable

Funktion

Neben den im Programmierhandbuch "Grundlagen" bereits beschriebenen Programmiermöglichkeiten können Sie Koordinatensysteme auch mit vordefinierten Framevariablen festlegen.



Folgende Koordinatensysteme sind definiert:

MKS: Maschinen-Koordinatensystem

BKS: Basis-Koordinatensystem

BNS: Basisnullpunkt-Koordinatensystem

ENS: Einstellbares Nullpunkt-Koordinatensystem

WKS: Werkstück-Koordinatensystem

Was ist eine vordefinierte Framevariable?

Vordefinierte Framevariablen sind Schlüsselwörter, die im Sprachgebrauch der Steuerung mit entsprechender Wirkung bereits festgelegt sind und im NC-Programm verarbeitet werden können.

Mögliche Framevariable:

- Basisframe (Basisverschiebung)
- einstellbare Frames
- programmierbarer Frame

Wertzuweisungen und Istwerte auslesen

Zusammenhang Framevariable/Frame

Eine Koordinatentransformation kann durch Wertzuweisung eines Frames an eine Framevariable aktiviert werden.

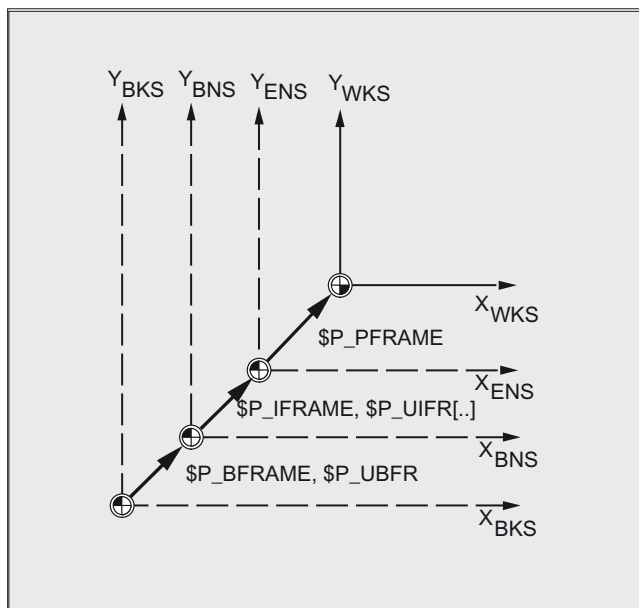
Beispiel: `$P_PFRAME=CTRANS (X, 10)`

Framevariable:

`$P_PFRAME` bedeutet: aktueller programmierbarer Frame.

Frame:

`CTRANS (X, 10)` bedeutet: programmierbare Nullpunktverschiebung der X-Achse um 10 mm.



Istwerte auslesen

Über vordefinierte Variable im Teileprogramm können die aktuellen Istwerte der Koordinatensysteme ausgelesen werden:

`$AA_IM[Achse]`: Lesen Istwert im MKS

`$AA_IB[Achse]`: Lesen Istwert im BKS

`$AA_IBN[Achse:]` Lesen Istwert im BNS

`$AA_IEN[Achse]`: Lesen Istwert im ENS

`$AA_IW[Achse]`: Lesen Istwert im WKS

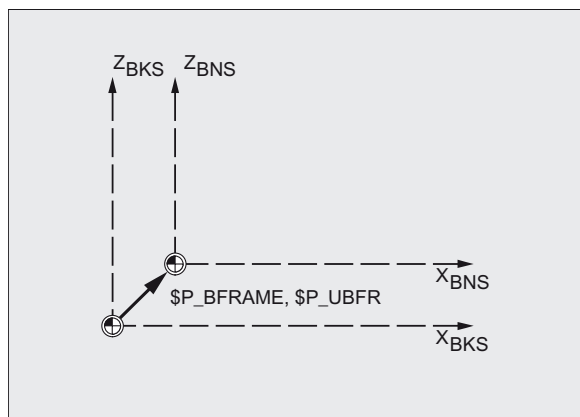
5.1.1 Vordefinierte Framevariable (\$P_BFRAME, \$P_IFRAME, \$P_PFRAME, \$P_ACTFRAME)

\$P_BFRAME

Aktuelle Basisframevariable, die den Bezug zwischen Basiskoordinatensystem (BKS) und Basis-Nullpunktsystem (BNS) herstellt.

Soll der durch \$P_UBFR beschriebene Basisframe sofort im Programm wirksam werden, muss entweder

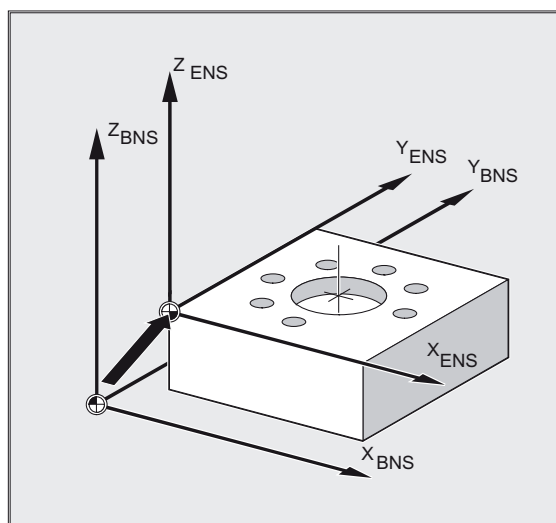
- ein G500, G54...G599 programmiert werden oder
- \$P_BFRAME mit \$ \$P_UBFR beschrieben werden.



\$P_IFRAME

Aktuelle, einstellbare Framevariable, die den Bezug zwischen Basis-Nullpunktsystem (BNS) und Einstellbarem Nullpunktsystem (ENS) herstellt.

- \$P_IFRAME entspricht \$P_UIFR[\$P_IFRNUM]
- \$P_IFRAME enthält nach Programmierung von z. B. G54 die durch G54 definierte Translation, Rotation, Skalierung und Spiegelung.

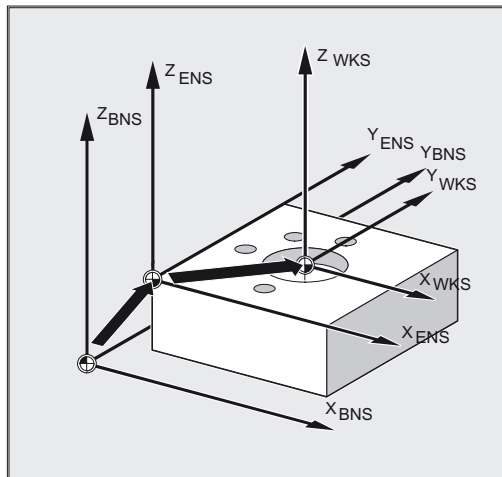


\$P_PFRAME

Aktuelle, programmierbare Framevariable, die den Bezug zwischen dem Einstellbaren Nullpunktsystem (ENS) und dem Werkstückkoordinatensystem (WKS) herstellt.

\$P_PFRAME enthält den resultierenden Frame, der sich

- aus der Programmierung von TRANS/ATRANS, ROT/AROT, SCALE/ASCALE, MIRROR/AMIRROR bzw.
- aus der Zuweisung von CTRANS, CROT, CMIRROR, CSCALE an den programmierbaren FRAME ergibt

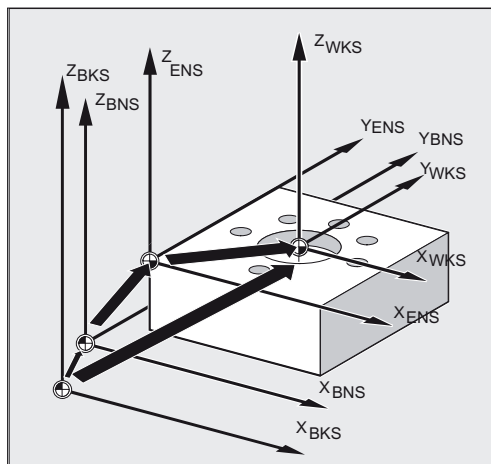


\$P_ACTFRAME

Aktueller, resultierender Gesamtframe, der sich durch Verkettung aus

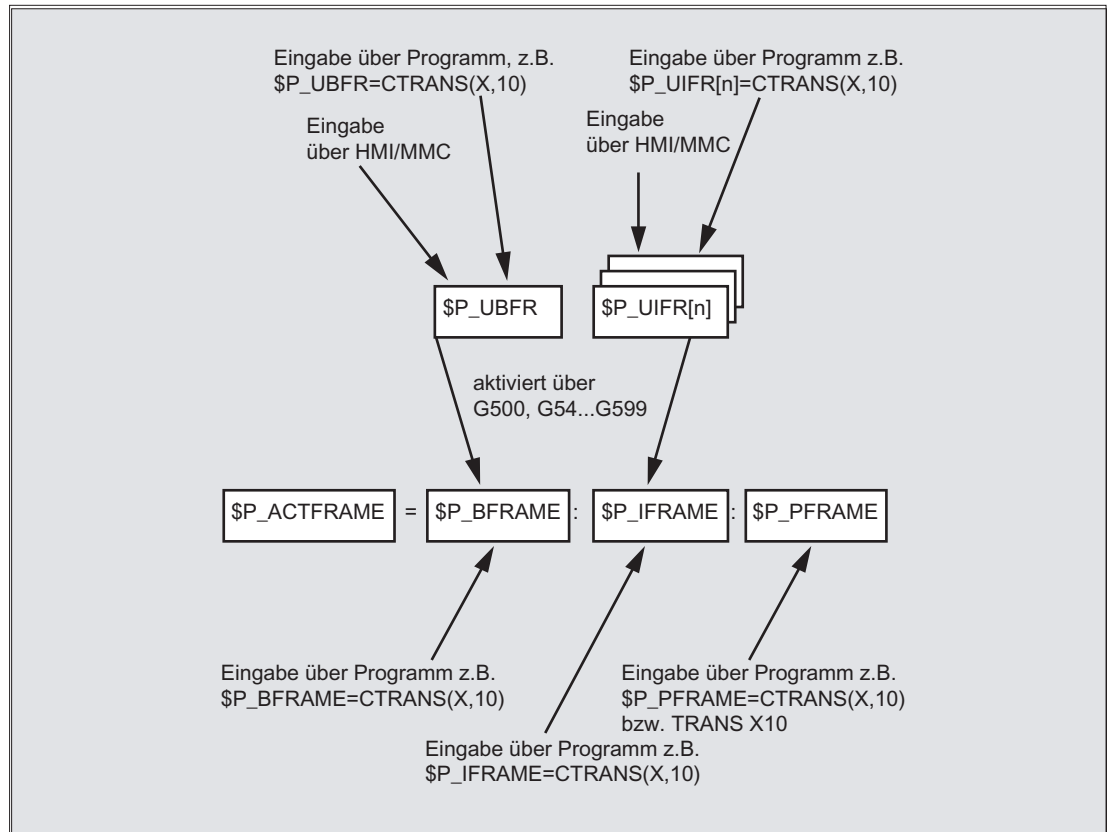
- der aktuellen Basisframevariablen \$P_BFRAME,
- der aktuellen einstellbaren Framevariablen \$P_IFRAME mit Systemframes und
- der aktuellen programmierbaren Framevariablen \$P_PFRAME mit Systemframes ergibt. Systemframes, siehe Kapitel "Im Kanal wirksame Frames"

\$P_ACTFRAME beschreibt den aktuell gültigen Werkstücknullpunkt.



Falls $\$P_BFRAME$, $\$P_IFRAME$ oder $\$P_PFRAME$ verändert werden, wird $\$P_ACTFRAME$ neu berechnet.

$\$P_ACTFRAME$ entspricht $\$P_BFRAME:\$P_IFRAME:\$P_PFRAME$



Basisframe und einstellbarer Frame wirken nach Reset, wenn das MD 20110 RESET_MODE_MASK folgendermaßen eingestellt ist:

Bit0=1, Bit14=1 --> $\$P_UBFR$ (Basisframe) wirkt

Bit0=1, Bit5=1 --> $\$P_UIFR[\$P_UIFRNUM]$ (Einst. Frame) wirkt

Vordefinierte einstellbare Frames $\$P_UBFR$

Mit $\$P_UBFR$ wird der Basisframe programmiert, er wird aber nicht gleichzeitig im Teilprogramm aktiv. Der mit $\$P_UBFR$ geschriebene Basisframe wird eingerechnet, wenn

- Reset geschaltet wurde und die Bits 0 und 14 des MD RESET_MODE_MASK gesetzt sind,
- die Anweisungen G500, G54...G599 ausgeführt wurden.

Vordefinierte einstellbare Frames \$P_UIFR[n]

Durch die vordefinierte Framevariable \$P_UIFR[n] können die einstellbaren Nullpunktverschiebungen G54 bis G599 vom Teileprogramm aus gelesen oder geschrieben werden.

Diese Variablen stellen im Aufbau ein eindimensionales Feld vom Typ FRAME mit dem Namen \$P_UIFR[n] dar.

Zuordnung zu den G-Befehlen

Standardmäßig sind 5 einstellbare Frames \$P_UIFR[0]... \$P_UIFR[4] bzw. 5 gleichbedeutende G-Befehle – G500 und G54 bis G57, unter deren Adressen Werte abgespeichert werden können.

\$P_IFRAME=\$P_UIFR[0] entspricht G500

\$P_IFRAME=\$P_UIFR[1] entspricht G54

\$P_IFRAME=\$P_UIFR[2] entspricht G55

\$P_IFRAME=\$P_UIFR[3] entspricht G56

\$P_IFRAME=\$P_UIFR[4] entspricht G57

Über Maschinendatum können Sie die Anzahl der Frames verändern:

\$P_IFRAME=\$P_UIFR[5] entspricht G505

... ..

\$P_IFRAME=\$P_UIFR[99] entspricht G599

Hinweis

Hierdurch lassen sich insgesamt 100 Koordinatensysteme erzeugen, die z. B. als Nullpunkt für verschiedene Vorrichtungen programmübergreifend aufgerufen werden können.



Die Programmierung von Framevariablen und Frames erfordert im NC-Programm einen eigenen NC-Satz. **Ausnahme:** Programmierung eines einstellbaren Frames mit G54, G55, ...

5.2 Framevariablen/Frames Werte zuweisen

5.2.1 Direkte Werte zuweisen (Achswert, Winkel, Maßstab)

Funktion

Im NC-Programm können Sie direkt Frames oder Framevariablen mit Werten belegen.

Syntax

```
$P_PFRAME=CTRANS (X, Achswert, Y, Achswert, Z, Achswert, ...)  
$P_PFRAME=CROT (X, Winkel, Y, Winkel, Z, Winkel, ...)  
$P_UIFR[.]=CROT (X, Winkel, Y, Winkel, Z, Winkel, ...)  
$P_PFRAME=CSCALE (X, Maßstab, Y, Maßstab, Z, Maßstab, ...)  
$P_PFRAME=CMIRROR (X, Y, Z)
```

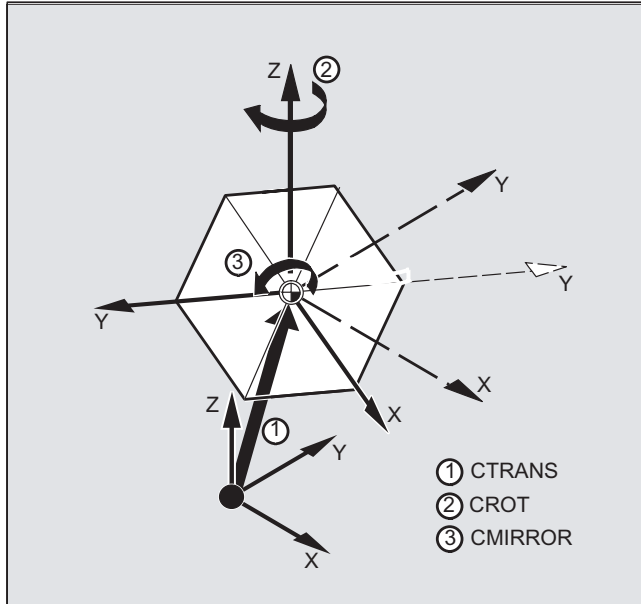
Die Programmierung von \$P_BFRAME erfolgt analog wie \$P_PFRAME.

Bedeutung

CTRANS	Verschiebung in den angegebenen Achsen
CROT	Drehung um die angegebenen Achsen
CSCALE	Maßstabsveränderung in den angegebenen Achsen
CMIRROR	Richtungsumkehr der angegebenen Achse
X Y Z	Verschiebewert in Richtung der angegebenen Geometrieachse
Achswert	Achswert der Verschiebung zuweisen
Winkel	Drehwinkel um die angegebenen Achsen zuweisen
Maßstab	Maßstab verändern

Beispiel

Durch Wertzuweisung an dem aktuellen programmierbaren Frame werden Translation, Drehung und Spiegelung aktiviert.



```
N10 $P_PFRAME=CTRANS (X, 10, Y, 20, Z, 5) : CROT (Z, 45) : CMIRROR (Y)
```

Frame-Rot-Komponenten mit anderen Werten vorbelegen

Mit CROT alle drei Komponenten von UIFR mit Werten vorbelegen

```
Programmcode  
$P_UIFR[5]=CROT (X, 0, Y, 0, Z, 0)  
N100 $P_UIFR[5, y, rt]=0  
  
N100 $P_UIFR[5, x, rt]=0  
  
N100 $P_UIFR[5, z, rt]=0
```

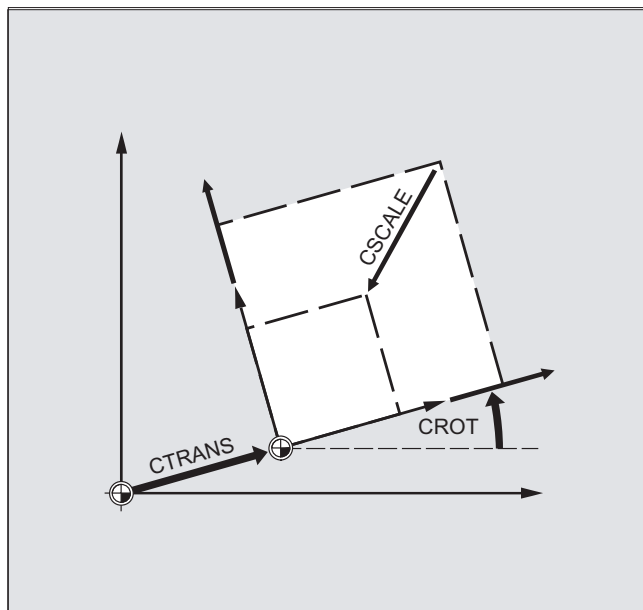
Beschreibung

Sie können mehrere Rechenvorschriften nacheinander programmieren.

Beispiel:

```
$P_PFRAME=CTRANS(...):CROT(...):CSCALE...
```

Beachten Sie, dass die Befehle durch den Kettungsoperator Doppelpunkt (:)(...) miteinander verbunden werden müssen. Dadurch werden die Befehle erstens miteinander verknüpft und zweitens in der programmierten Reihenfolge additiv ausgeführt.



Hinweis

Die mit den genannten Befehlen programmierten Werte werden den Frames zugewiesen und abgespeichert.

Aktiv werden die Werte erst, wenn sie dem Frame einer aktiven Framevariablen `$P_BFRAME` bzw. `$P_PFRAME` zugewiesen werden.

5.2.2 Framekomponenten lesen und verändern (TR, FI, RT, SC, MI)

Funktion

Sie haben die Möglichkeit, auf **einzelne** Daten eines Frames, z. B. auf einen bestimmten Verschiebewert oder Drehwinkel zuzugreifen. Diese Werte können Sie verändern oder einer anderen Variablen zuweisen.

Syntax

<code>R10=\$P_UIFR[\$P_UIFNUM,X,RT]</code>	Der Drehwinkel RT um die X-Achse aus der aktuell gültigen einstellbaren Nullpunktverschiebung \$P_UIFRNUM soll der Variablen R10 zugewiesen werden.
<code>R12=\$P_UIFR[25,Z,TR]</code>	Der Verschiebewert TR in Z aus dem Datensatz des eingestellten Frames Nr. 25 soll der Variablen R12 zugewiesen werden.
<code>R15=\$P_PFRAME[Y,TR]</code>	Der Verschiebewert TR in Y des aktuellen programmierbaren Frames soll der Variablen R15 zugewiesen werden.
<code>\$P_PFRAME[X,TR]=25</code>	Der Verschiebewert TR in X des aktuellen programmierbaren Frames soll verändert werden. Ab sofort gilt X25.

Bedeutung

<code>\$P_UIFRNUM</code>	Mit dieser Variablen wird automatisch der Bezug zur aktuell gültigen einstellbaren Nullpunktverschiebung hergestellt.
<code>P_UIFR[n,...,...]</code>	Durch Angabe der Framenummer n greifen Sie auf den einstellbaren Frame Nr. n zu.
	Angabe der Komponente, die gelesen oder verändert werden soll:
<code>TR</code>	TR Translation
<code>FI</code>	FI Translation Fine
<code>RT</code>	RT Rotation
<code>SC</code>	SC Scale Maßstabsveränderung
<code>MI</code>	MI Spiegelung
<code>X Y Z</code>	Zusätzlich (siehe Beispiele) wird die entsprechende Achse X, Y, Z angegeben.

Wertebereich für Drehung RT

- Drehung um 1. Geometrieachse: -180° bis +180°
- Drehung um 2. Geometrieachse: -90° bis +90°
- Drehung um 3. Geometrieachse: -180° bis +180°

Beschreibung

Frame aufrufen

Durch Angabe der Systemvariablen $\$P_UIFRNUM$ können Sie direkt auf die mit $\$P_UIFR$ bzw. G54, G55, ... aktuell eingestellte Nullpunktverschiebung zugreifen ($\$P_UIFRNUM$ enthält die Nummer des aktuell eingestellten Frames).

Alle anderen gespeicherten einstellbaren Frames $\$P_UIFR$ rufen Sie durch Angabe der entsprechenden Nummer $\$P_UIFR[n]$ auf.

Für vordefinierte Framevariable und eigendefinierte Frames geben Sie den Namen an, z. B. $\$P_IFRAME$.

Daten aufrufen

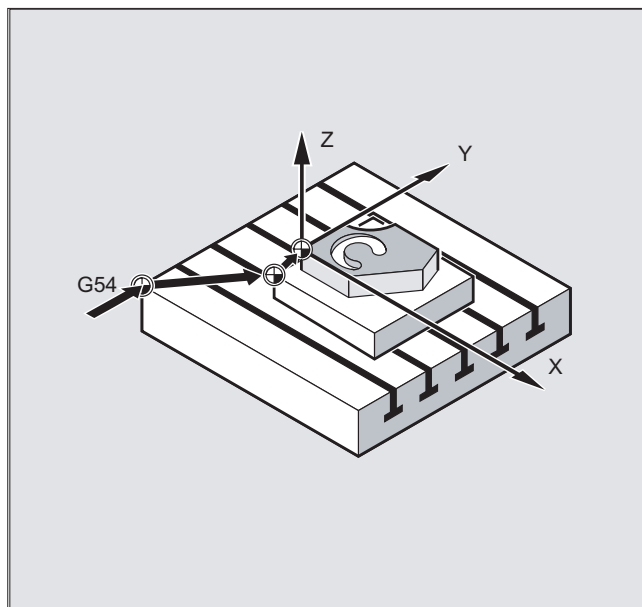
In den eckigen Klammern stehen Achsname und Framekomponente des Wertes, auf den Sie zugreifen oder den Sie verändern wollen, z. B. [X, RT] oder [Z, MI].

5.2.3 Verknüpfung von kompletten Frames

Funktion

Im NC-Programm kann ein kompletter Frame einem anderen Frame zugewiesen oder Frames miteinander verkettet werden.

Framekettungen eignen sich z. B. für die Beschreibung mehrerer Werkstücke, die auf einer Palette angeordnet sind und in einem Fertigungsablauf bearbeitet werden sollen.



Für die Beschreibung von Palettenaufgaben könnten die Framekomponenten z. B. nur bestimmte Teilwerte enthalten, durch deren Verkettung verschiedene Werkstücknullpunkte generiert werden.

Syntax

Frames zuweisen

```
DEF FRAME EINSTELLUNG1  
EINSTELLUNG1=CTRANS(X,10)  
$P_PFRAME=EINSTELLUNG1
```

Dem aktuellen programmierbaren Frame werden die Werte des selbst definierten Frames EINSTELLUNG1 zugewiesen.

```
DEF FRAME EINSTELLUNG4  
EINSTELLUNG4=$P_PFRAME  
$P_PFRAME=EINSTELLUNG4
```

Der aktuelle programmierbare Frame wird zwischengespeichert und dann bei Bedarf wieder zurückgespeichert.

Frameketten

Die Frames werden in der programmierten Reihenfolge miteinander verkettet, die Framekomponenten wie z. B. Verschiebungen, Drehungen usw. werden nacheinander additiv ausgeführt.

```
$P_IFRAME=$P_UIFR[15]:$P_UIFR[16]
```

\$P_UIFR[15] enthält z. B. Daten für Nullpunktverschiebungen. Anschließend werden –darauf aufbauend –die Daten von \$P_UIFR[16] z. B. Daten für Rotationen verarbeitet.

```
$P_UIFR[3]=$P_UIFR[4]:$P_UIFR[5]
```

Der einstellbare Frame 3 wird durch Verkettung der einstellbaren Frames 4 und 5 erzeugt.

Hinweis

Beachten Sie, dass die Frames durch den Kettungsoperator Doppelpunkt : miteinander verbunden werden müssen.

5.2.4 Definition neuer Frames (DEF FRAME)

Funktion

Neben den bisher beschriebenen vordefinierten, einstellbaren Frames haben Sie auch die Möglichkeit, neue Frames zu erzeugen. Dabei handelt es sich um Variable vom Typ FRAME, die Sie mit freier Namensgebung definieren.

Mit den Funktionen CTRANS, CROT, CSCALE, CMIRROR können Sie Ihre Frames im NC-Programm mit Werten belegen.

Syntax

```
DEF FRAME PALETTE1  
  
PALETTE1=CTRANS (...) : CROT (...) ...
```

Bedeutung

DEF FRAME	Neue Frames erzeugen.
PALETTE1	Name des neuen Frames
=CTRANS (...) :	Den möglichen Funktionen Werte zuweisen
CROT (...) ...	

5.3 Grob- und Feinverschiebung (CFINE, CTRANS)

Funktion

Feinverschiebung

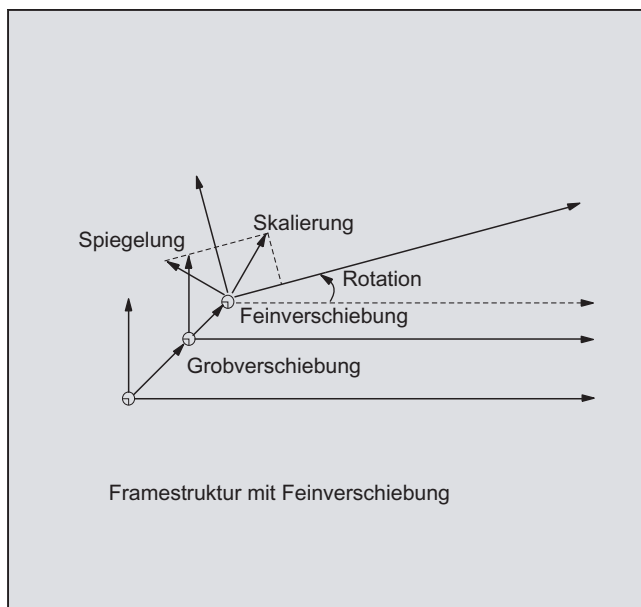
Die Feinverschiebung eines Frames wird mit dem Befehl `CFINE (...)` programmiert.

Hinweis

Freigabe der Feinverschiebung über MD18600 `$MN_MM_FRAME_FINE_TRANS = 1`

Grobverschiebung

Die Grobverschiebung eines Frames wird mit dem Befehl `CTrans (...)` programmiert.



Grob- und Feinverschiebung addieren sich zur Gesamtverschiebung.

Syntax

Feinverschiebung

Am Beispiel des Datenhaltungsframes \$P_UIFR:

- Gesamtframe

- \$P_UIFR [<n>] = CFINE (<K1>, <Wert>)
- \$P_UIFR [<n>] = CFINE (<K1>, <Wert>, <K2>, <Wert>)
- \$P_UIFR [<n>] = CFINE (<K1>, <Wert>, <K2>, <Wert>, <K3>, <Wert>)

- Frame-Komponente

- \$P_UIFR [<n>, <Koordinate>, FI] = <Wert>

Grobverschiebung

Am Beispiel des Datenhaltungsframes \$P_UIFR:

- Gesamtframe

- \$P_UIFR [<n>] = CTRANS (<K1>, <Wert>)
- \$P_UIFR [<n>] = CTRANS (<K1>, <Wert>, <K2>, <Wert>)
- \$P_UIFR [<n>] = CTRANS (<K1>, <Wert>, <K2>, <Wert>, <K3>, <Wert>)

- Frame-Komponente

- \$P_UIFR [<n>, <Koordinate>, TR] = <Wert>

Programmierbarer Frame \$P_PFRAME auch:

- TRANS <K1> <Wert>
- TRANS <K1> <Wert> <K2> <Wert>
- TRANS <K1> <Wert> <K2> <Wert> <K3> <Wert>

Bedeutung

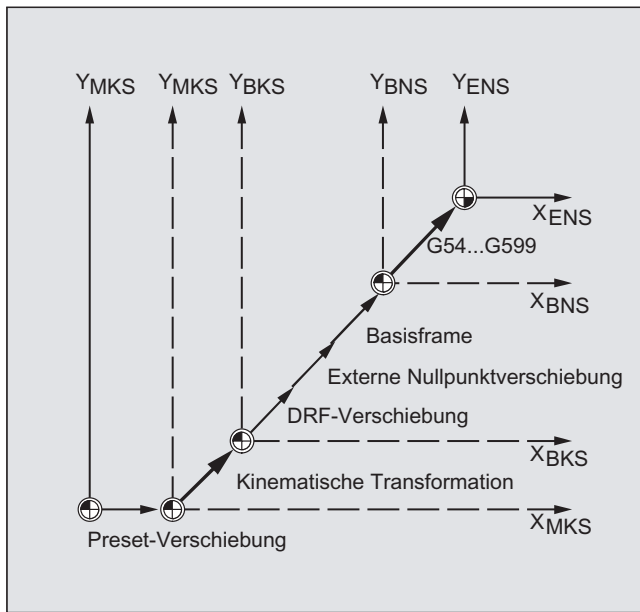
CFINE:	Feinverschiebung, additive Verschiebung (Translation).
CTRANS:	Grobverschiebung, absolute Verschiebung (Translation).
TRANS:	Programmierbarer Frame: Grobverschiebung, absolute Verschiebung (Translation).
X, Y, Z:	Koordinaten
<Kn>:	Koordinate X, Y oder Z
<Wert>:	Verschiebungswert

5.4 Externe Nullpunktverschiebung

Funktion

Hierdurch haben Sie eine weitere Möglichkeit, den Nullpunkt zwischen Basis- und Werkstückkoordinatensystem zu verschieben.

Bei der externen Nullpunktverschiebung können nur lineare Verschiebungen programmiert werden.



Programmierung

Die Programmierung der Verschiebewerte, $\$AA_ETRANS$ erfolgt über die Belegung der achsspezifischen Systemvariablen.

Verschiebewert zuweisen

$\$AA_ETRANS [Achse] = RI$

RI ist die Rechenvariable vom Typ REAL, die den neuen Wert enthält.

Die externe Verschiebung wird in der Regel nicht im Teileprogramm angegeben, sondern von der PLC gesetzt.

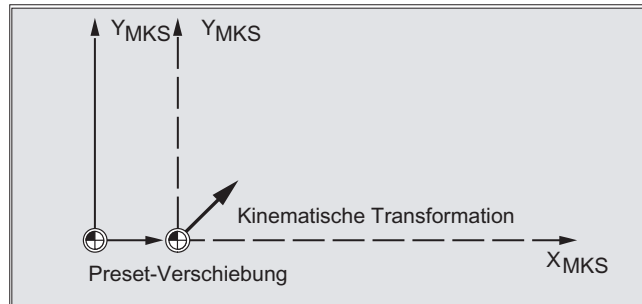
Hinweis

Der im Teileprogramm geschriebene Wert wird erst wirksam, wenn an der VDI-Schnittstelle (NCU-PLC-Schnittstelle) das entsprechende Signal gesetzt ist.

5.5 Preset-Verschiebung mit PRESETON

Funktion

Für spezielle Anwendungen kann es erforderlich werden, einer bereits referenzierten Maschinenachse mit `PRESETON` einen neuen Istwert zuzuweisen. Dies entspricht einer Nullpunktverschiebung im Maschinenkoordinatensystem.



VORSICHT

Nach `PRESETON` ist die Maschinenachse im Status "nicht referenziert". Es wird deshalb empfohlen, die Funktion nur für Maschinenachsen ohne Referenzpunktspflicht zu verwenden. Zum Wiederherstellen des ursprünglichen Maschinenkoordinatensystems muss die Maschinenachse z. B. mit `G74` (Referenzpunktfahren) erneut referenziert werden.

Literatur:
Programmierhandbuch Grundlagen

Syntax

```
PRESETON(<Achse_1>,<Wert_1>,<Achse_2>,<Wert_2>,...)
```

Hinweis

Mit `PRESETON` können Preset-Verschiebungen für bis zu 8 Achsen programmiert werden.

Bedeutung

<code>PRESETON:</code>	vordefinierte Prozedur zum Istwertsetzen
<code><Achse_...>:</code>	Bezeichner der Maschinenachse, deren Steuerungsnullpunkt verändert werden soll
	Wertebereich: alle im Kanal definierten Achsbezeichner
<code><Wert_...>:</code>	neuer Istwert der Maschinenachse im Maschinenkoordinatensystem

Beispiel

Geometrieachse: A

Zugehörige Maschinenachse: X1

Programmcode	Kommentar
N10 G0 A100	; Achse A fährt auf Position 100.
N20 PRESETON(X1,50)	; Maschinenachse X1 erhält an Position 100 den neuen Istwert 50 => neue Istwertanzeige: - Achse X1, MKS: 50 - Achse A, WKS: 50
N30 A100	; Achse A fährt 50 mm auf Position 100.

Literatur

Zur Programmierung der Preset-Verschiebung in Synchronaktionen siehe:

Funktionshandbuch Synchronaktionen; Kapitel: "Aktionen in Synchronaktionen" >
"Istwertsetzen (PRESETON)"

5.6 Frame-Berechnung aus 3 Messpunkten im Raum (MEAFRAME)

Funktion

Die Funktion MEAFRAME wird zur Unterstützung der Messzyklen eingesetzt. Sie berechnet den Frame aus drei idealen und den korrespondierenden gemessenen Punkten.

Wird ein Werkstück für die Bearbeitung positioniert, ist seine Position relativ zum kartesischen Maschinenkoordinatensystem bezüglich seiner Idealposition i. A. sowohl verschoben als auch gedreht. Für exakte Bearbeitung oder Messung ist entweder eine kostspielige physikalische Justierung oder Änderung der Bewegungen im Teileprogramm nötig.

Ein Frame kann durch Abtasten dreier Punkte im Raum festgelegt werden, deren Idealpositionen bekannt sind. Abgetastet wird mit einem Berührungs- oder optischen Sensor, der spezielle, auf der Trägerplatte präzise fixierte Löcher oder Messkugeln berührt.

Syntax

MEAFRAME (<Ideal-Punkte>, <Messpunkte>, <Qualität>)

Bedeutung

MEAFRAME:	Funktionsaufruf	
<Ideal-Punkte>:	2-dim. REAL-Feld, das die drei Koordinaten der Ideal-Punkte enthält	
<Messpunkte>:	2-dim. REAL-Feld, das die drei Koordinaten der gemessenen Punkte enthält	
<Qualität>:	Variable, mit der Informationen zur Qualität der FRAME-Berechnung zurückgegeben werden	
Typ:	VAR REAL	
Wert:	-1	Die idealen Punkte liegen nahezu auf einer Geraden: Der Frame konnte nicht berechnet werden. Die zurückgegebene FRAME-Variable enthält einen neutralen Frame.
	-2	Die Messpunkte liegen nahezu auf einer Geraden: Der Frame konnte nicht berechnet werden. Die zurückgegebene FRAME-Variable enthält einen neutralen Frame.
	-4	Die Berechnung der Rotationsmatrix schlägt aus einem anderen Grund fehl.
	≥ 0.0	Summe der Verzerrungen (Abstände zwischen den Punkten), die zur Überführung des gemessenen Dreiecks in ein zum idealen Dreieck kongruentes benötigt wird.

Hinweis

Qualität der Messung

Damit die gemessenen den idealen Koordinaten mit einer kombinierten Rotation/Translation zugeordnet werden können, muss das von den Messpunkten aufgespannte Dreieck kongruent zum idealen Dreieck sein. Dies wird bewerkstelligt von einem Kompensationsalgorithmus, der die Summe der Quadrate der Abweichungen minimiert, die das gemessene in das ideale Dreieck überführen.

Die effektiv benötigte Verzerrung der Messpunkte kann als Indikator für die Qualität der Messung dienen und wird deshalb als zusätzliche Variable von MEAFRAME ausgegeben.

Hinweis

Das von MEAFRAME erzeugte Frame kann durch die Funktion ADDFRAME in ein anderes Frame in der Frame-Kette transformiert werden (siehe Beispiel "Verkettung mit ADDFRAME").

Beispiele

Beispiel 1:

Teileprogramm 1:

Programmcode
...
DEF FRAME CORR_FRAME

Setzen von Messpunkten:

Programmcode	Kommentar
DEF REAL IDEAL_POINT[3,3]=SET(10.0,0.0,0.0,0.0,10.0,0.0,0.0,0.0,10.0)	
DEF REAL MEAS_POINT[3,3]=SET(10.1,0.2,-0.2,-0.2,10.2,0.1,-0.2,0.2,9.8)	; Für Test.
DEF REAL FIT_QUALITY=0	
DEF REAL ROT_FRAME_LIMIT=5	; Erlaubt max. 5 Grad-Verdrehung der Teileposition.
DEF REAL FIT_QUALITY_LIMIT=3	; Erlaubt max. 3 mm-Verschiebung zwischen dem idealen und dem gemessenen Dreieck.
DEF REAL SHOW_MCS_POS1[3]	
DEF REAL SHOW_MCS_POS2[3]	
DEF REAL SHOW_MCS_POS3[3]	

Programmcode	Kommentar
N100 G01 G90 F5000	
N110 X0 Y0 Z0	
N200 CORR_FRAME=MEAFRAME (IDEAL_POINT, MEAS_POINT, FIT_QUALITY)	
N230 IF FIT_QUALITY < 0	
SETAL (65000)	
GOTOF NO_FRAME	
ENDIF	
N240 IF FIT_QUALITY > FIT_QUALITY_LIMIT	
SETAL (65010)	
GOTOF NO_FRAME	
ENDIF	
N250 IF CORR_FRAME[X,RT] > ROT_FRAME_LIMIT	; Begrenzung des 1. RPY- Winkels
SETAL (65020)	
GOTOF NO_FRAME	
ENDIF	
N260 IF CORR_FRAME[Y,RT] > ROT_FRAME_LIMIT	; Begrenzung des 2. RPY-Winkels.
SETAL (65021)	
GOTOF NO_FRAME	
ENDIF	
N270 IF CORR_FRAME[Z,RT] > ROT_FRAME_LIMIT	; Begrenzung des 3. RPY- Winkels.
SETAL (65022)	
GOTOF NO_FRAME	
ENDIF	
N300 \$P_IFRAME=CORR_FRAME	; Abtast-Frame mit einem setzbaren Frame aktivieren.
	; Frame prüfen durch Positionieren der Geometrieachsen auf die idealen Punkte.
N400 X=IDEAL_POINT[0,0] Y=IDEAL_POINT[0,1] Z=IDEAL_POINT[0,2]	
N410 SHOW_MCS_POS1[0]=\$AA_IM[X]	
N420 SHOW_MCS_POS1[1]=\$AA_IM[Y]	
N430 SHOW_MCS_POS1[2]=\$AA_IM[Z]	
N500 X=IDEAL_POINT[1,0] Y=IDEAL_POINT[1,1] Z=IDEAL_POINT[1,2]	
N510 SHOW_MCS_POS2[0]=\$AA_IM[X]	
N520 SHOW_MCS_POS2[1]=\$AA_IM[Y]	
N530 SHOW_MCS_POS2[2]=\$AA_IM[Z]	
N600 X=IDEAL_POINT[2,0] Y=IDEAL_POINT[2,1] Z=IDEAL_POINT[2,2]	
N610 SHOW_MCS_POS3[0]=\$AA_IM[X]	
N620 SHOW_MCS_POS3[1]=\$AA_IM[Y]	
N630 SHOW_MCS_POS3[2]=\$AA_IM[Z]	
N700 G500	; Setzbaren Frame deaktivieren, da mit Nullframe (kein Wert eingetragen vorbesetzt).

Programmcode	Kommentar
No_FRAME	; Setzbaren Frame deaktivieren, da mit Nullframe (kein Wert eingetragen) vorbesetzt.
M0	
M30	

Beispiel 2: Verkettung von Frames

Verkettung von MEAFRAME für Korrekturen

Die Funktion MEAFRAME liefert ein Korrektur-Frame. Wird dieser Korrektur-Frame mit dem einstellbaren Frame \$P_UIFR[1] verkettet, der bei Aufruf der Funktion aktiv war (z. B. G54), so erhält man einen einstellbaren Frame für weitere Umrechnungen zum Verfahren oder Bearbeiten.

Verkettung mit ADDFRAME

Soll dieser Korrektur-Frame in der Frame-Kette an einer anderen Stelle wirken oder sind vor dem einstellbaren Frame noch andere Frames aktiv, dann kann die Funktion ADDFRAME zum Einketten in einem der Kanal-Basisframes oder einem Systemframe genutzt werden.

In den Frames darf hierbei nicht aktiv sein:

- Spiegelung mit MIRROR
- Skalierung mit SCALE

Die Eingangsparameter für Soll- und Istwerte sind die Werkstückkoordinaten. Im Grundsystem der Steuerung sind diese Koordinaten stets metrisch oder in Inch (G71/G70) und als radiusbezogenes (DIAMOF) Maß anzugeben.

Literatur:

Weitere Informationen zu ADDFRAME siehe:

Funktionshandbuch Grundfunktionen; K2: Achsen, Koordinatensysteme, Frames

5.7 NCU-globale Frames

Funktion

NCU-globale Frames gibt es pro NCU nur einmal für alle Kanäle. NCU-globale Frames können von allen Kanälen aus geschrieben und gelesen werden. Die Aktivierung der NCU-globalen Frames erfolgt im jeweiligen Kanal.

Durch globale Frames können **Kanalachsen und Maschinenachsen** mit Verschiebungen, skaliert und gespiegelt werden.

Geometrische Zusammenhänge und Frameketten

Bei globalen Frames existiert kein geometrischer Zusammenhang zwischen den Achsen. Deshalb können keine Drehungen und keine Programmierung von Geometrie-Achsbezeichnern ausgeführt werden.

- Auf globale Frames lassen sich keine Rotationen anwenden. Die Programmierung einer Rotation wird mit dem Alarm: "18310 Kanal %1 Satz %2 Frame: Rotation unzulässig", abgelehnt.
- Die Verkettung von globalen Frames und kanalspezifischen Frames ist möglich. Der resultierende Frame enthält alle Frameanteile inklusive der Rotationen für alle Achsen. Die Zuweisung eines Frames mit Rotationsanteilen an einen globalen Frame wird mit dem Alarm "Frame: Rotation unzulässig" abgelehnt.

NCU-globale Frames

NCU-globale Basisframes \$P_NCBFR[n]

Es können bis zu 8 NCU-globale Basisframes projiziert werden:

Gleichzeitig können kanalspezifische Basisframes vorhanden sein.

Globale Frames können von allen Kanälen einer NCU geschrieben und gelesen werden. Beim Schreiben von globalen Frames ist vom Anwender für eine Kanalkoordinierung Sorge zu tragen. Dies kann z. B. durch Wait-Marken (WAITMC) realisiert werden.

Maschinenhersteller

Die Anzahl von globalen Basisframes wird über Maschinendaten projiziert.

Literatur:

Funktionshandbuch Grundfunktionen; Achsen, Koordinatensysteme, Frames (K2)

NCU-globale einstellbare Frames \$P_UIFR[n]

Alle einstellbaren Frames G500, G54...G599 können entweder NCU-global oder kanalspezifisch projiziert werden.

Maschinenhersteller

Alle einstellbaren Frames können mit Hilfe des Maschinendatums MD18601 \$MN_MM_NUM_GLOBAL_USER_FRAMES zu globalen Frames umprojiziert werden.

Als Achsbezeichner bei den Frame-Programmbefehlen können Kanalachsbezeichner und Maschinenachsbezeichner verwendet werden. Die Programmierung von Geometrieachsbezeichnern wird mit einem Alarm abgelehnt.

5.7.1 Kanalspezifische Frames (\$P_CHBFR, \$P_UBFR)

Funktion

Einstellbare Frames oder Basisframes können über das Teileprogramm und über BTSS von der Bedienung und von der PLC geschrieben und gelesen werden.

Die Feinverschiebung ist auch für die globalen Frames möglich. Die Unterdrückung von globalen Frames erfolgt ebenso, wie bei kanalspezifischen Frames über G53, G153, SUPA und G500.

Maschinenhersteller

Über das Maschinendatum MD28081 \$MC_MM_NUM_BASE_FRAMES kann die Anzahl der Basisframes im Kanal projektiert werden. Die Standardkonfiguration ist so ausgelegt, dass es mindestens ein Basisframe pro Kanal gibt. Maximal sind 8 Basisframes pro Kanal möglich. Zusätzlich zu den 8 Basisframes im Kanal kann es noch 8 NCU-globale Basisframes geben.

Kanalspezifische Frames

\$P_CHBFR[n]

Über die Systemvariable \$P_CHBFR[n] können die Basisframes gelesen und geschrieben werden. Beim Schreiben eines Basisframes wird der verkettete Gesamt-Basisframe nicht aktiviert, sondern die Aktivierung erfolgt erst mit der Ausführung einer G500, G54 ... G599-Anweisung. Die Variable dient vorwiegend als Speicher für Schreibvorgänge auf das Basisframe von HMI oder PLC. Diese Frame-Variablen werden über die Datensicherung gesichert.

Erster Basisframe im Kanal

Ein Schreiben auf die vordefinierte Variable \$P_UBFR aktiviert den Basisframe mit dem Feldindex 0 nicht gleichzeitig, sondern die Aktivierung erfolgt erst mit der Ausführung einer G500, G54 ... G599-Anweisung. Die Variable kann auch im Programm geschrieben und gelesen werden.

\$P_UBFR

\$P_UBFR ist identisch mit \$P_CHBFR[0]. Standardmäßig gibt es immer einen Basisframe im Kanal, so dass die Systemvariable kompatibel zu älteren Ständen ist. Gibt es kein kanalspezifisches Basisframe, wird beim Schreiben oder Lesen der Alarm "Frame: Anweisung unzulässig" ausgegeben.

5.7.2 Im Kanal wirksame Frames

Funktion

Im Kanal wirksame Frames werden vom Teileprogramm über die betreffenden Systemvariablen dieser Frames eingegeben. Hierzu gehören auch Systemframes. Über diese Systemvariablen kann im Teileprogramm das aktuelle Systemframe gelesen und geschrieben werden.

Aktuelle im Kanal wirksame Frames

Übersicht

Aktuelle Systemframes

\$P_PARTFRAME

\$P_SETFRAME

\$P_EXTFRAME

\$P_NCBFRAME[n]

\$P_CHBFRAME[n]

\$P_BFRAME

\$P_ACTBFRAME

\$P_CHBFRMASK und \$P_NCBFRMASK

\$P_IFFRAME

Aktuelle Systemframes

\$P_TOOLFRAME

\$P_WPFRAME

\$P_TRAFRAME

\$P_PFRAME

Aktuelles Systemframe

\$P_CYCFRAME

P_ACTFRAME

FRAME-Kettung

für:

TCARR und PAROT

Istwertsetzen und Ankratzen

Externe Nullpunktverschiebung

Aktuelle NCU-globale Basisframes

Aktuelle Kanal-Basisframes

Aktueller 1. Basisframe im Kanal

Gesamt-Basisframe

Gesamt-Basisframe

Aktueller einstellbarer Frame

für:

TOROT und TOFRAME

Werkstückbezugspunkte

Transformationen

Aktueller programmierbarer Frame

für:

Zyklen

Aktueller Gesamtframe

Aktuelles Frame setzt sich aus dem Gesamt-Basisframe zusammen

\$P_NCBFRAME [n] Aktuelle NCU-globale Basisframes

Über die Systemvariable \$P_NCBFRAME[n] können die aktuellen globalen Basisframe-Feld Elemente gelesen und geschrieben werden. Das resultierende Gesamt-Basisframe wird durch den Schreibvorgang im Kanal eingerechnet.

Der geänderte Frame wird nur in dem Kanal, in dem der Frame programmiert wurde, aktiv. Soll der Frame für alle Kanäle einer NCU geändert werden, muss gleichzeitig \$P_NCBFR[n] und \$P_NCBFRAME[n] beschrieben werden. Die anderen Kanäle müssen dann noch den Frame mit z. B. G54 aktivieren. Beim Schreiben eines Basisframes wird der Gesamt-Basisframe neu berechnet.

\$P_CHBFRAME[n] Aktuelle Kanal-Basisframes

Über die Systemvariable \$P_CHBFRAME[n] können die aktuellen Kanal-Basisframe-Feldelemente gelesen und geschrieben werden. Der resultierende Gesamt-Basisframe wird durch den Schreibvorgang im Kanal eingerechnet. Beim Schreiben eines Basisframes wird der Gesamt-Basisframe neu berechnet.

\$P_BFRAME Aktueller 1. Basisframe im Kanal

Über die vordefinierte Framevariable \$P_BFRAME kann der aktuelle Basisframe mit dem Feldindex 0, der im Kanal gültig ist, im Teileprogramm gelesen und geschrieben werden. Der geschriebene Basisframe wird sofort eingerechnet.

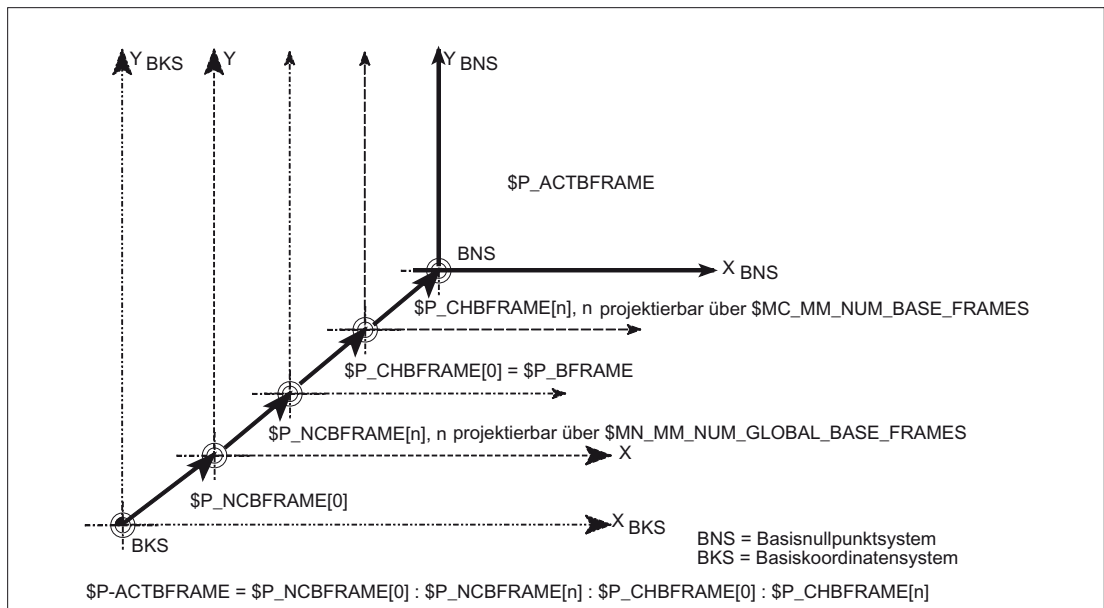
\$P_BFRAME ist identisch mit \$P_CHBFRAME[0]. Die Systemvariable hat standardmäßig immer einen gültigen Wert. Gibt es kein kanalspezifisches Basisframe, wird beim Schreiben oder Lesen der Alarm "Frame: Anweisung unzulässig" ausgegeben.

\$P_ACTBFRAME Gesamt-Basisframe

Die Variable \$P_ACTBFRAME ermittelt das verkettete Gesamt-Basisframe. Die Variable ist nur lesbar.

\$P_ACTBFRAME entspricht:

\$P_NCBFRAME[0] : ... : \$P_NCBFRAME[n] : \$P_CHBFRAME[0] : ... : \$P_CHBFRAME[n].



\$P_CHBFRMASK und \$P_NCBFRMASK Gesamt-Basisframe

Über die Systemvariable \$P_CHBFRMASK und \$P_NCBFRMASK kann der Anwender auswählen, welche Basisframes er in die Berechnung des "Gesamt"-Basisframes mit einbeziehen möchte. Die Variablen können nur im Programm programmiert werden und über BTSS gelesen werden. Der Wert der Variablen wird als Bitmaske interpretiert und gibt an, welches Basisframe-Feldelement von \$P_ACTFRAME in die Berechnung einfließt.

Mit \$P_CHBFRMASK kann vorgegeben werden, welche kanalspezifischen Basisframes, und mit \$P_NCBFRMASK, welche NCU-globalen Basisframes eingerechnet werden.

Mit der Programmierung der Variablen werden der Gesamt-Basisframe und der Gesamt-Frame neu berechnet. Nach Reset und in der Grundeinstellung ist der Wert von \$P_CHBFRMASK und \$P_NCBFRMASK wie folgt:

```
$P_CHBFRMASK = $MC_CHBFRAME_RESET_MASK
```

```
$P_NCBFRMASK = $MC_CHBFRAME_RESET_MASK
```

Beispiel:

```
$P_NCBFRMASK = 'H81' ;$P_NCBFRAME[0] : $P_NCBFRAME[7]
```

```
$P_CHBFRMASK = 'H11' ;$P_CHBFRAME[0] : $P_CHBFRAME[4]
```

\$P_IFRAME Aktueller einstellbarer Frame

Über die vordefinierte Framevariable \$P_IFRAME kann der aktuelle einstellbare Frame, welcher im Kanal gültig ist, im Teileprogramm gelesen und geschrieben werden. Der geschriebene einstellbare Frame wird sofort eingerechnet.

Bei NCU-globalen einstellbaren Frames wirkt der geänderte Frame nur in dem Kanal, in dem der Frame programmiert wurde. Soll der Frame für alle Kanäle einer NCU geändert werden, muss gleichzeitig \$P_UIFR[n] und \$P_IFRAME beschrieben werden. Die anderen Kanäle müssen dann noch den entsprechenden Frame mit z. B. G54 aktivieren.

\$P_PFRAME Aktueller programmierbarer Frame

\$P_PFRAME ist der programmierbare Frame, der sich aus der Programmierung von TRANS/ATRANS, G58/G59, ROT/AROT, SCALE/ASCALE, MIRROR/AMIRROR bzw. aus der Zuweisung von CTRANS, CROT, CMIRROR, CSCALE an den programmierbaren FRAME ergibt.

Aktuelle, programmierbare Framevariable, die den Bezug zwischen dem einstellbaren Nullpunktsystem (ENS) und dem Werkstückkoordinatensystem (WKS) herstellt.

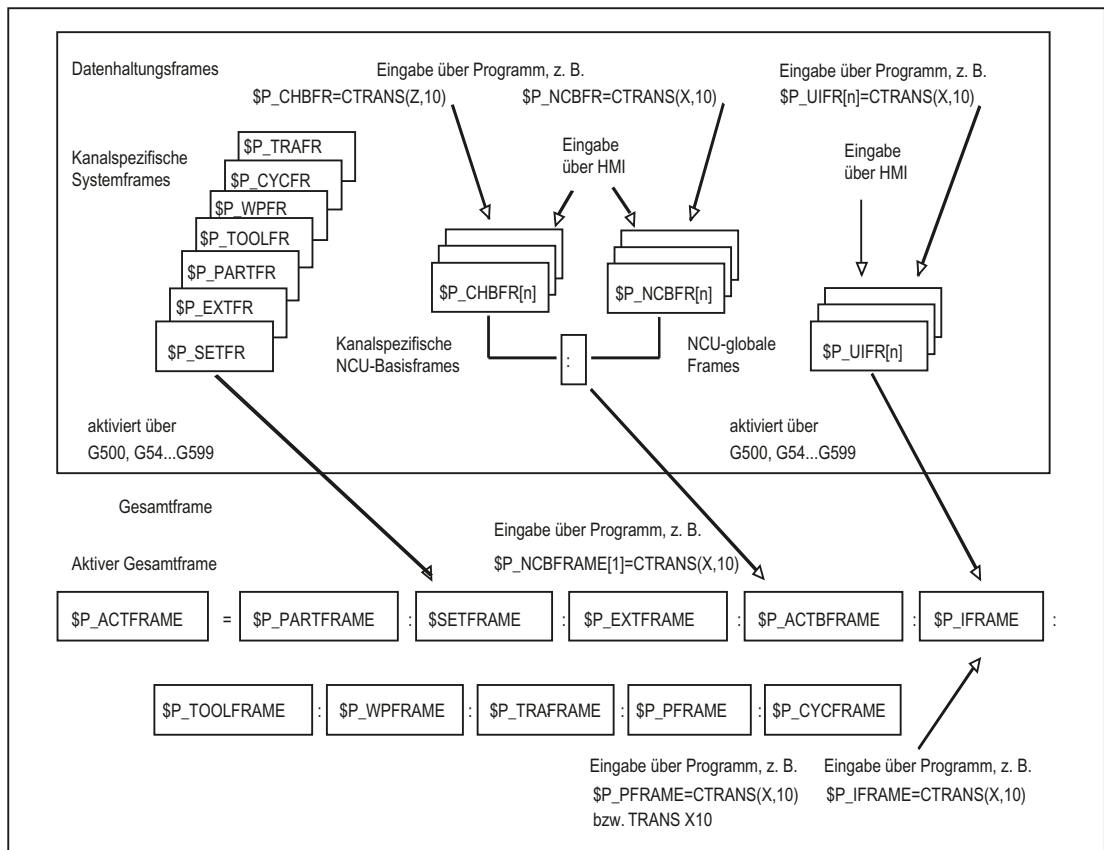
P_ACTFRAME Aktueller Gesamtframe

Der aktuelle resultierende Gesamtframe \$P_ACTFRAME ergibt sich nun als Verkettung aller Basisframes, dem aktuellen einstellbaren Frame und dem programmierbaren Frame. Der aktuelle Frame wird immer dann aktualisiert, wenn sich ein Frameanteil ändert.

\$P_ACTFRAME entspricht:

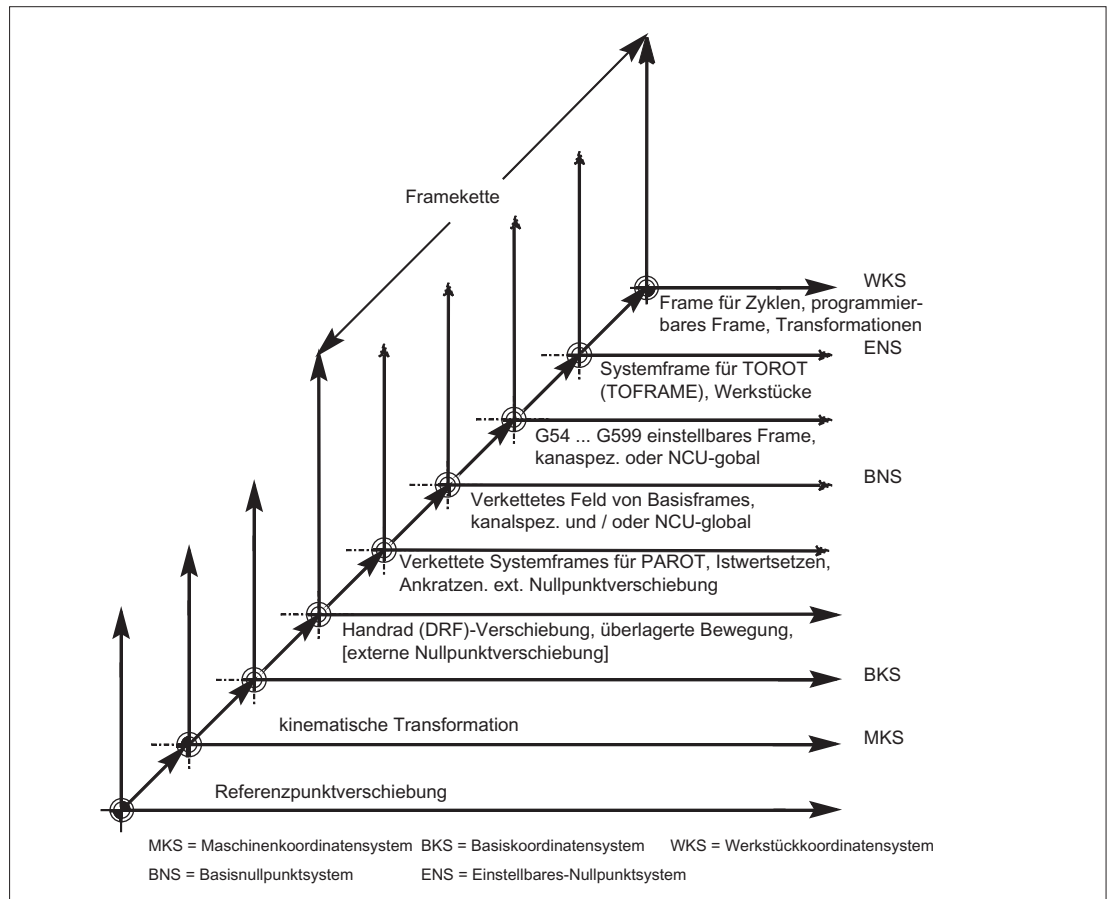
\$P_PARTFRAME : \$P_SETFRAME : \$P_EXTFRAME : \$P_ACTBFRAME : \$P_IFRAME :

\$P_TOOLFRAME : \$P_WPFRAME : \$P_TRAFRAME : \$P_PFRAME : \$P_CYCFRAME



Frame-Kettung

Der aktuelle Frame setzt sich aus dem Gesamt-Basisframe, dem einstellbaren Frame, dem Systemframe und dem programmierbaren Frame gemäß oben angegebenen aktuellen Gesamtframe zusammen.



Transformationen

6.1 Allgemeine Programmierung der Transformationsarten

Allgemeine Funktion

Zur Anpassung der Steuerung an verschiedene Maschinenkinematiken besteht die Auswahl Transformationsarten mit geeigneten Parametern zu programmieren. Über diese Parameter kann für die ausgewählte Transformation sowohl die Orientierung des Werkzeugs im Raum als auch die Orientierungsbewegungen der Rundachsen entsprechend vereinbart werden.

Bei den Drei-, Vier- und Fünf-Achs-Transformationen beziehen sich die programmierten, Positionsangaben immer auf die Spitze des Werkzeugs, welches orthogonal zur im Raum befindlichen Bearbeitungsfläche nachgeführt wird. Die kartesischen Koordinaten werden vom Basiskoordinatensystem ins Maschinenkoordinatensystem umgerechnet und beziehen sich auf die Geometrieachsen. Diese beschreiben den Arbeitspunkt. Virtuelle Rundachsen beschreiben die Orientierungen des Werkzeugs im Raum und werden mit TRAORI programmiert.

Bei der kinematischen Transformation können Positionen im kartesischen Koordinatensystem programmiert werden. Die Steuerung transformiert die mit TRANSMIT, TRACYL und TRAANG programmierten Verfahrbewegungen des kartesischen Koordinatensystems auf die Verfahrbewegungen der realen Maschinenachsen.

Programmierung

Drei-, Vier- und Fünf-Achs-Transformationen TRAORI

Die vereinbarte Orientierungstransformation wird mit dem Befehl TRAORI und den drei möglichen Parametern für Trafonummer, Orientierungsvektor und Rundachsoffsets aktiviert.

`TRAORI (Trafonummer, Orientierungsvektor, Rundachsoffsets)`

Kinematische Transformationen

Zu den Kinematischen Transformationen gehören die vereinbarten Transformationen

`TRANSMIT (Trafonummer)`

`TRACYL (Arbeitsdurchmesser, Trafonummer)`

`TRAANG (Winkel der schräg stehenden Achse, Trafonummer)`

Aktive Transformation ausschalten

Mit `TRAFOOF` kann die gerade aktive Transformation ausgeschaltet werden.

Orientierungstransformation

Drei-, Vier- und Fünf- Achs-Transformationen TRAORI

Zur optimalen Bearbeitung räumlich geformter Flächen im Arbeitsraum der Maschine, benötigen Werkzeugmaschinen außer den drei Linearachsen X, Y und Z noch zusätzliche Achsen. Die zusätzlichen Achsen beschreiben die Orientierung im Raum und werden nachfolgend Orientierungsachsen genannt. Sie stehen als Drehachsen bei vier Maschinentypen mit verschiedener Kinematik zur Verfügung.

1. Zweiachsen-Schwenkkopf, z. B. Kardanischer Werkzeugkopf mit einer Rundachse parallel zu einer Linearachse bei festem Werkzeuggestisch.
2. Zweiachsen-Drehtisch, z. B. fester Schwenkkopf mit drehbarem Werkzeuggestisch um zwei Achsen.
3. Einachs-Schwenkkopf und Einachs-Drehtisch, z. B. ein drehbarer Schwenkkopf mit gedrehtem Werkzeug bei drehbarem Werkzeuggestisch um eine Achse.
4. Zweiachsen-Schwenkkopf und Einachs-Drehtisch, z. B. bei drehbarem Werkzeuggestisch um eine Achse und ein drehbarer Schwenkkopf mit drehbarem Werkzeug um sich selbst.

Die **3- und 4-Achs-Transformationen** sind Sonderformen der 5-Achs-Transformation und werden analog zu den 5-Achs-Transformationen programmiert.

Die "**Generische 3-/4-/5-/6-Achs-Transformation**" deckt mit ihrem Funktionsumfang für rechtwinklig angeordnete Rundachsen sowie die Transformationen für den Kardanischen Fräskopf ab und kann wie jede andere Orientierungstransformation auch für diese vier Maschinentypen mit TRAORI aktiviert werden. Bei der generischen 5/6-Achs-Transformation hat die Werkzeugorientierung einen weiteren dritten Freiheitsgrad, bei dem zur Werkzeugrichtung beliebig im Raum, das Werkzeug um die eigene Achse gedreht werden kann.

Literatur: /FB3/ Funktionshandbuch Sonderfunktionen; 3- bis 5-Achstransformation (F2)

Kinematikunabhängige Grundstellung der Werkzeugorientierung

ORIRESET

Ist mit TRAORI eine Orientierungstransformation aktiv, dann können mit ORIRESET die Grundstellungen von bis zu 3 Orientierungsachsen mit optionalen Parametern A, B, C angegeben werden. Die Zuordnung der Reihenfolge der programmierten Parameter zu den Rundachsen erfolgt gemäß der durch die Transformation festgelegten Reihenfolge der Orientierungsachsen. Die Programmierung von ORIRESET(A, B, C) bewirkt, dass die Orientierungsachsen linear und synchron von ihrer momentanen Position zu der angegebenen Grundstellungsposition fahren.

Kinematische Transformationen

TRANSMIT und TRACYL

Bei Fräsbearbeitungen an Drehmaschinen kann für die vereinbarte Transformation entweder

1. eine stirnseitige Bearbeitung in der Drehaufspannung mit TRANSMIT oder
2. eine Bearbeitung von beliebig verlaufenden Nuten an zylindrischen Körpern mit TRACYL programmiert werden.

TRAANG

Soll die Zustellachse z. B. für die Technologie Schleifen auch schräg zustellbar sein, so kann mit TRAANG für die vereinbarte Transformation ein parametrierbarer Winkel programmiert werden.

Kartesisches PTP-Fahren

Zur kinematischen Transformation gehört auch das "Kartesische PTP-Fahren" bei dem bis zu 8 unterschiedliche Gelenkstellungen STAT= programmiert werden können. Die Positionen werden im kartesischen Koordinatensystem programmiert, wobei die Bewegung der Maschine in Maschinenkoordinaten erfolgt.

Literatur:

/FB2/ Funktionshandbuch Erweiterungsfunktionen; Kinematische Transformation (M1)

Verkettete Transformationen

Es können jeweils zwei Transformationen hintereinander geschaltet werden. Bei der hierdurch verketteten zweiten Transformation werden die Bewegungsanteile der Achsen aus der ersten Transformation übernommen.

Als erste Transformation sind möglich:

- Orientierungstransformation TRAORI
- Polartransformation TRANSMIT
- Zylindertransformation TRACYL
- Transformation Schräge Achse TRAANG

Die zweite Transformation muss Schräge Achse TRAANG sein

6.1.1 Orientierungsbewegungen bei den Transformationen

Verfahrbewegungen und Orientierungsbewegungen

Die Verfahrbewegungen der programmierbaren Orientierungen hängen primär vom Maschinentyp ab. Bei der Drei-, Vier- und Fünf-Achs-Transformation mit TRAORI beschreiben die rotatorischen Achsen oder die schwenkbaren Linearachsen die Orientierungsbewegungen des Werkzeugs.

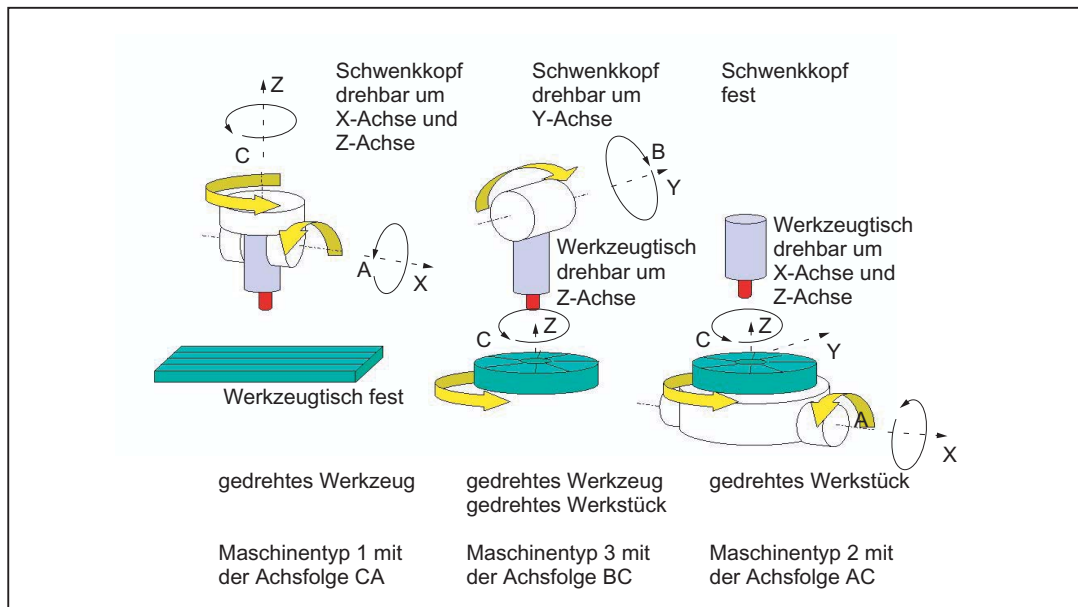
Änderungen der Positionen der an der Orientierungstransformation beteiligten Rundachsen führen zu Ausgleichsbewegungen der übrigen Maschinenachsen. Die Position der Werkzeugspitze bleibt dabei unverändert.

Orientierungsbewegungen des Werkzeugs können über die Rundachsbezeichner A..., B..., C... der virtuellen Achsen je nach Anwendung entweder durch Angabe von Euler- bzw. RPY-Winkeln oder Richtungs- bzw. Flächennormalenvektoren, Normierte Vektoren für die Drehachse eines Kegels oder für die Zwischenorientierung auf einer Kegelmantelfläche programmiert werden.

Bei der Kinematischen Transformation mit TRANSMIT, TRACYL und TRAANG transformiert die Steuerung die programmierten Verfahrbewegungen des kartesischen Koordinatensystems auf die Verfahrbewegungen der realen Maschinenachsen.

Maschinenkinematik bei Drei-, Vier- und Fünf-Achs-Transformation TRAORI

Es kann entweder das Werkzeug oder der Werkzeugschwenkopf mit bis zu zwei Rundachsen drehbar sein. Eine Kombination von jeweils einachsiger Schwenk- und Drehtisch ist auch möglich.



Maschinentyp	Programmierung der Orientierung
Drei-Achs-Transformation Maschinentypen 1 und 2	Programmierung der Werkzeugorientierung nur in der Ebene, die senkrecht zu der rotatorischen Achse ist. Es existieren zwei translatorischen Achsen (Linearachsen) und einer rotatorischen Achse (Rundachse).
Vier-Achs-Transformation Maschinentypen 1 und 2	Programmierung der Werkzeugorientierung nur in der Ebene, die senkrecht zu der rotatorischen Achse ist. Es existieren drei translatorischen Achsen (Linearachsen) und einer rotatorischen Achse (Rundachse).
Fünf-Achs-Transformation Maschinentypen 3 Einachs-Schwenkkopf und Einachs-Drehtisch	Programmierung der Orientierungstransformation. Kinematik mit drei Linearachsen und zwei orthogonalen Rundachsen. Die Rundachsen sind parallel zu zwei der drei Linearachsen. Die erste Rundachse wird von zwei kartesischen Linearachsen bewegt. Sie dreht die dritte Linearachse mit dem Werkzeug. Die zweite Rundachse dreht das Werkstück.

Generische 5/6-Achs Transformationen

Maschinentyp	Programmierung der Orientierungstransformation
Generische Fünf-/Sechs-Achs Transformation Maschinentypen 4 Zweiachs-Schwenkkopf mit drehbarem Werkzeug um sich selbst und Einachs- Drehtisch	Programmierung der Orientierungstransformation. Kinematik mit drei Linearachsen und drei orthogonalen Rundachsen. Die Rundachsen sind parallel zu zwei der drei Linearachsen. Die erste Rundachse wird von zwei kartesischen Linearachsen bewegt. Sie dreht die dritte Linearachse mit dem Werkzeug. Die zweite Rundachse dreht das Werkstück. Die Grundorientierung des Werkzeugs kann durch eine zusätzliche Drehung um sich selbst mit dem Drehwinkel THETA programmiert werden.

Beim Aufruf der "Generischen Drei-, Vier- und Fünf-/Sechs-Achs Transformation" kann zusätzlich die Grundorientierung des Werkzeugs übergeben werden. Es gelten die Einschränkungen bezüglich der Richtungen der Rundachsen nicht mehr. Wenn die Rundachsen nicht exakt senkrecht aufeinander stehen oder vorhandene Rundachsen nicht exakt parallel zu den Linearachsen stehen, kann die "Generische Fünf-/Sechs-Achs Transformation" bessere Ergebnisse der Werkzeugorientierung liefern.

Kinematische Transformationen TRANSMIT, TRACYL und TRAANG

Für Fräsbearbeitungen an Drehmaschinen oder einer schräg zustellbaren Achse beim Schleifen gelten abhängig von der Transformation im Standardfall folgende Achsanordnungen:

TRANSMIT	Aktivierung der Polar-Transformation
stirnseitige Bearbeitung in der Drehaufspannung	eine Rundachse eine Zustellachse senkrecht zur Drehachse eine Längsachse parallel zur Drehachse
TRACYL	Aktivierung der Zylindermanteltransformation
Bearbeitung von beliebig verlaufenden Nuten an den zylindrischen Körper	eine Rundachse eine Zustellachse senkrecht zur Drehachse eine Längsachse parallel zur Drehachse
TRAANG	Aktivierung der Transformation Schräge Achse
Bearbeitung mit schräger Zustellachse	eine Rundachse eine Zustellachse mit parametrierbaren Winkel eine Längsachse parallel zur Drehachse

Kartesisches PTP-Fahren

Die Bewegung der Maschine erfolgt in Maschinenkoordinaten und wird programmiert mit:

TRAORI	Aktivierung der Transformation
PTP Punkt zu Punkt fahren	Position im kartesischen Koordinatensystem (MKS) anfahren
CP	Bahnbewegung der kartesischen Achsen im (BKS)
STAT	Stellung der Gelenke ist abhängig von der Transformation
TU	Um welchen Winkel die Achsen auf den kürzesten Weg verfahren

PTP-Fahren bei generischer 5/6-AchsTransformation

Die Bewegung der Maschine erfolgt in Maschinenkoordinaten und die Werkzeugorientierung kann sowohl mit Rundachspositionen als auch mit von der Kinematik unabhängigen Vektoren Euler bzw. RPY-Winkel oder den Richtungsvektoren programmiert werden.

Dabei sind Rundachsinterpolation, Vektoreninterpolation mit Großkreisinterpolation oder Interpolation des Orientierungsvektors auf einer Kegelmantelfläche möglich.

Beispiel Drei-, bis Fünf-Achs-Transformation bei einen Kardanischen Fräskopf

Die Werkzeugmaschine hat mindestens 5 Achsen, davon

- Drei translatorische Achsen für geradlinige Bewegungen, die den Arbeitspunkt an jede beliebige Position im Arbeitsraum bewegen.
- Zwei rotatorische Schwenkachsen, die unter einem projektierbaren Winkel (meist 45 Grad) angeordnet sind, ermöglichen dem Werkzeug Orientierungen im Raum einzunehmen, die sich bei 45 Grad Anordnung auf eine Halbkugel beschränken.

6.1.2 Übersicht der Orientierungstransformation TRAORI

Mögliche Programmierungsarten im Zusammenhang mit TRAORI

Maschinentyp	Programmierung bei aktiver Transformation TRAORI
<p>Maschinentypen 1, 2 oder 3 Zweiachs-Schwenkkopf oder Zweiachs-Drehtisch oder eine Kombination von jeweils einachsigen Schwenkkopf und Drehtisch.</p>	<p>Achsfolge der Orientierungsachsen und die Orientierungsrichtung des Werkzeugs ist und entweder maschinenbezogen projektierbar über Maschinendaten abhängig von der Maschinenkinematik oder werkstückbezogen mit programmierbarer Orientierung unabhängig von der Maschinenkinematik Die Drehrichtungen der Orientierungsachsen im Bezugssystem wird programmiert mit: - ORIMKS Bezugssystem = Maschinenkoordinatensystem - ORIWKS Bezugssystem = Werkstückkoordinatensystem Die Grundeinstellung ist ORIWKS. Programmierung der Orientierungsachsen mit: A, B, C der Maschinenachsenpositionen direkt A2, B2, C2 Winkelprogrammierung virtueller Achsen mit - ORIEULER über Euler-Winkel (Standard) - ORIRPY über RPY-Winkel - ORIVIRT1 über virtuelle Orientierungsachsen 1. Definition - ORIVIRT2 über virtuelle Orientierungsachsen 2. Definition mit Unterscheidung der Interpolationsart: lineare Interpolation - ORIAxes von Orientierungsachsen oder Maschinenachsen Großkreisinterpolation (Interpolation des Orientierungsvektors) - ORIVECT von Orientierungsachsen Programmierung der Orientierungsachsen durch Angabe A3, B3, C3 der Vektorkomponenten (Richtung-/Flächennormale) Programmierung der resultierenden Werkzeugorientierung A4, B4, C4 des Flächennormalvektors am Satzanfang A5, B5, C5 des Flächennormalvektors am Satzende LEAD Voreilwinkel für die Werkzeugorientierung TILT Seitwärtswinkel für die Werkzeugorientierung</p>
	<p>Interpolation des Orientierungsvektors auf einer Kegelmantelfläche Orientierungsänderungen auf einer beliebig im Raum befindlichen Kegelmantelfläche durch Interpolation: - ORIPLANE in der Ebene (Großkreisinterpolation) - ORICONCW auf einer Kegelmantelfläche im Uhrzeigersinn - ORICONCCW auf einer Kegelmantelfläche gegen Uhrzeigersinn A6, B6, C6 Richtungsvektors (Drehachse des Kegels) - OICONIO Interpolation auf einer Kegelmantelfläche mit: A7, B7, C7 Zwischenvektoren (Start- und Endorientierung) oder - ORICONTO auf einer Kegelmantelfläche tangentialer Übergang Orientierungsänderungen bezogen auf eine Bahn mit - ORICURVE Vorgabe der Bewegung zweier Kontaktpunkte über PO[XH]=(xe, x2, x3, x4, x5) Orientierungspolynome bis 5.Grades PO[YH]=(ye, y2, y3, y4, y5) Orientierungspolynome bis 5.Grades PO[ZH]=(ze, z2, z3, z4, z5) Orientierungspolynome bis 5.Grades - ORIPATHS Glättung des Orientierungsverlaufs mit A8, B8, C8 Umorientierungsphase des Werkzeugs entspricht: Richtung und Weglänge des Werkzeugs bei der Abhebebewegung</p>

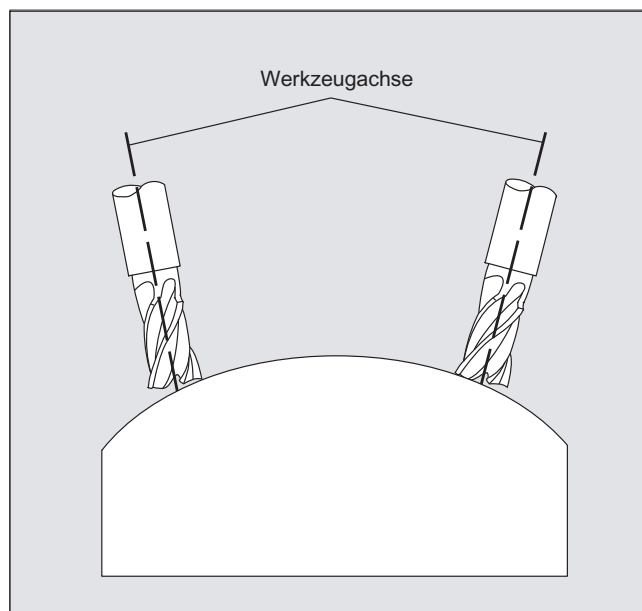
Maschinentyp	Programmierung bei aktiver Transformation TRAORI
<p>Maschinentypen 1 und 3</p> <p>Weitere Maschinentypen mit zusätzlicher Drehung des Werkzeugs um sich selbst erfordern eine 3. Rundachse</p> <p>Orientierungstransformation, wie z.B. generische 6-Achs-Transformation. Drehungen des Orientierungsvektors.</p>	<p>Programmierung der Drehungen der Werkzeugorientierung mit LEAD Voreilwinkel Winkel relativ zum Flächennormalenvektor PO[PHI] Programmierung eines Polynoms bis 5.Grades TILT Seitwärtswinkel Drehung um Bahntangente (Z-Richtung) PO[PSI] Programmierung eines Polynoms bis 5.Grades THETA Drehwinkel (Drehung um die Werkzeugrichtung in Z) THETA= Wert der am Satzende erreicht wird THETA=AC(...) Satzweise auf Maßangabe absolut umschalten THETA=IC(...) Satzweise auf Kettenmaßangabe umschalten THETA=Θ_e Programmierter Winkel G90/G91 interpolieren PO[THT]=(..) Programmierung eines Polynoms bis 5.Grades Programmierung des Drehvektors - ORIOTA Drehung absolut - ORIOTR relativer Drehvektor - ORIOTT tangentialer Drehvektor</p>
<p>Bahnrelative Orientierung für Orientierungsänderungen relativ zur Bahn oder Drehung des Drehvektors tangential zur Bahn</p>	<p>Orientierungsänderungen relativ zur Bahn mit - ORIPATH Werkzeugorientierung bezogen auf die Bahn - ORIPATHS zusätzlich bei einen Knick im Orientierungsverlauf Programmierung des Drehvektors - ORIOTC tangentialer Drehvektor, Drehung zur Bahntangente</p>

6.2 Drei-, Vier- und Fünf-Achs-Transformation (TRAORI)

6.2.1 Allgemeine Zusammenhänge Kardanischer Werkzeugkopf

Funktion

Um optimale Schnittbedingungen beim Bearbeiten räumlich gekrümmter Flächen zu erzielen, muss der Anstellwinkel des Werkzeugs veränderbar sein.

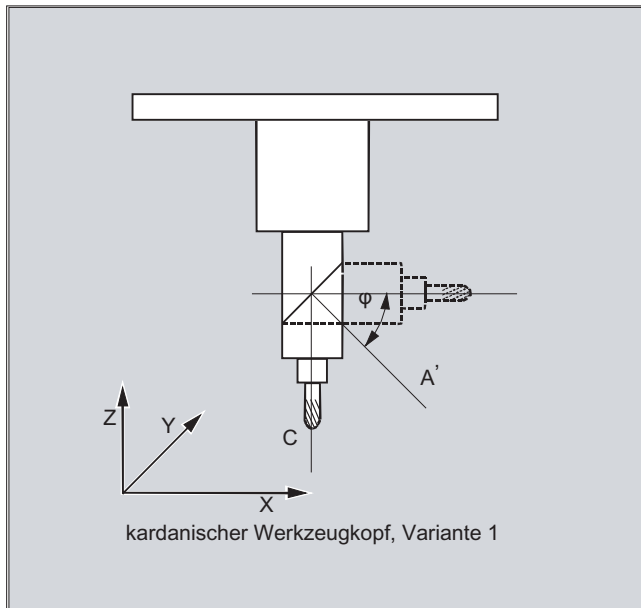


Mit welcher Maschinenkonstruktion dies erreicht wird, ist in den Achsdaten hinterlegt.

5-Achs-Transformation

Kardanischer Werkzeugkopf

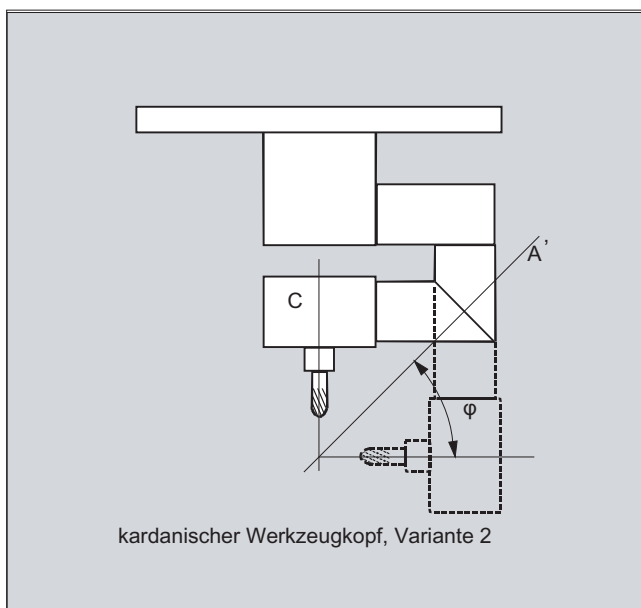
Hier legen drei Linearachsen (X, Y, Z) und zwei Orientierungsachsen (C, A) den Anstellwinkel und Arbeitspunkt des Werkzeugs fest. Eine der beiden Orientierungsachsen ist als Schrägachse angelegt, hier im Beispiel A' - in vielen Fällen als 45°-Anordnung.



In den hier gezeigten Beispielen sehen Sie die Anordnungen am Beispiel mit dem Kardanischen Werkzeugkopf der Maschinenkinematik CA!

Maschinenhersteller

Die Achsfolge der Orientierungsachsen und die Orientierungsrichtung des Werkzeugs kann abhängig von der Maschinenkinematik über Maschinendaten eingestellt werden.



In diesem Beispiel liegt A' unter dem Winkel φ zur X-Achse

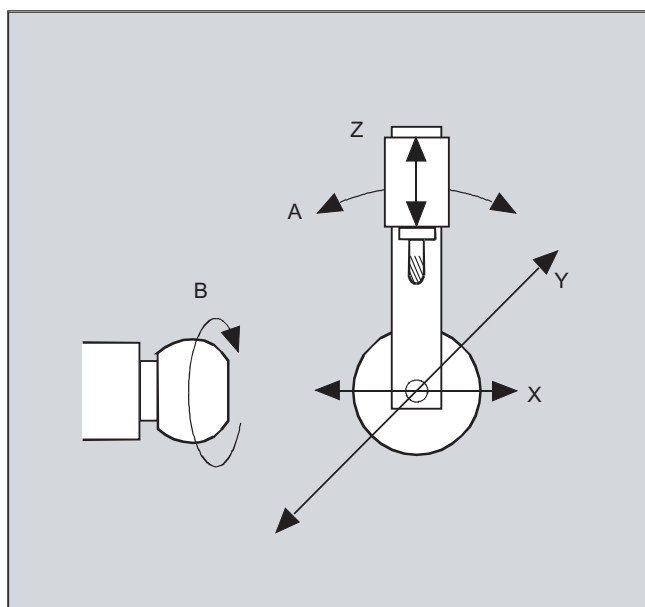
Allgemein gelten folgende mögliche Zusammenhänge:

A' liegt unter dem Winkel φ zur	X-Achse
B' liegt unter dem Winkel φ zur	Y-Achse
C' liegt unter dem Winkel φ zur	Z-Achse

Der Winkel φ kann im Bereich 0° bis $+89^\circ$ über Maschinendaten projiziert werden.

Mit schwenkbarer Linearachse

Hierbei handelt es sich um eine Anordnung mit bewegtem Werkstück und bewegtem Werkzeug. Die Kinematik setzt sich aus drei Linearachsen (X, Y, Z) und zwei rechtwinklig angeordneten Drehachsen zusammen. Die erste Rundachse wird z. B. über einen Kreuzschlitten von zwei Linearachsen bewegt, das Werkzeug steht parallel zur dritten Linearachse. Die zweite Drehachse dreht das Werkstück. Die dritte Linearachse (Schwenkachse) liegt in der Ebene des Kreuzschlittens.



Die Achsfolge der rotatorischen Achsen und die Orientierungsrichtung des Werkzeugs kann abhängig von der Maschinenkinematik über Maschinendaten eingestellt werden.

Es gelten folgende mögliche Zusammenhänge:

Achsen:	Achsfolgen:
1. Rundachse	A A B B C C
2. Rundachse	B C A C A B
Geschwenkte Linearachse	Z Y Z X Y X

Weitere Erläuterungen zu konfigurierbaren Achsfolgen für die Orientierungsrichtung des Werkzeugs siehe

Literatur: /FB3/ Funktionshandbuch Sonderfunktionen; 3- bis 5-Achs-Transformationen (F2), Kapitel Kardanischer Fräskopf, "Parametrierung".

6.2.2 Drei, Vier, und Fünf- Achs-Transformation (TRAORI)

Funktion

Der Anwender kann zwei bzw. drei translatorische Achsen und eine rotatorische Achse projektieren. Die Transformationen gehen davon aus, dass die rotatorische Achse orthogonal auf der Orientierungsebene steht.

Die Orientierung des Werkzeugs ist nur in der Ebene möglich, die senkrecht zur rotatorischen Achse ist. Die Transformation unterstützt die Maschinentypen mit beweglichem Werkzeug und beweglichem Werkstück.

Die Projektierung und Programmierung der Drei- und Vier-Achs-Transformationen sind analog zu den Fünf-Achs-Transformationen.

Literatur:

Funktionshandbuch Sonderfunktionen; Mehrachstransformationen (F2)

Syntax

```
TRAORI (<n>)  
TRAORI (<n>, <X>, <Y>, <Z>, <A>, <B>)  
TRAFOOF
```

Bedeutung

TRAORI:	Aktiviert die erste vereinbarte Orientierungstransformation
TRAORI (<n>):	Aktiviert die mit n vereinbarte Orientierungstransformation
<n>:	Nummer der Transformation
	Wert: 1 oder 2
	Beispiel:
	TRAORI(1) aktiviert Orientierungstransformation 1
<X>, <Y>, <Z>:	Komponente des Orientierungsvektors, in die das Werkzeug zeigt.
<A>, :	Programmierbarer Offset für die Rundachsen
TRAFOOF:	Transformation ausschalten

Werkzeugorientierung

Abhängig von der gewählten Orientierungsrichtung des Werkzeugs muss im NC-Programm die aktive Arbeitsebene (G17, G18, G19) so eingestellt werden, dass die Werkzeuglängtenkorrektur in Richtung der Werkzeugorientierung wirkt.

Hinweis

Nach dem Einschalten der Transformation beziehen sich Positionsangaben (X, Y, Z) immer auf die Spitze des Werkzeugs. Änderung der Positionen der an der Transformation beteiligten Rundachsen führen zu Ausgleichsbewegungen der übrigen Maschinenachsen, wodurch die Position der Werkzeugschneidspitze unverändert bleibt.

Die Orientierungstransformation ist immer von der Werkzeugspitze zur Werkzeugaufnahme gerichtet.

Offset für Orientierungsachsen

Bei Aktivierung der Orientierungstransformation kann ein zusätzlicher Offset für Orientierungsachsen direkt programmiert werden.

Es dürfen Parameter weggelassen werden, wenn bei der Programmierung die richtige Reihenfolge eingehalten wird.

Beispiel:

TRAORI (, , , , A,B) ; wenn nur ein einziger Offset eingegeben werden soll

Alternativ zur direkten Programmierung kann der zusätzliche Offset für Orientierungsachsen auch aus der momentan aktiven Nullpunktverschiebung automatisch übernommen werden. Die Übernahme wird über Maschinendaten projiziert.

Beispiele

TRAORI (1,0,0,1)	; Die Grundorientierung des Werkzeugs zeigt in Z-Richtung
TRAORI (1,0,1,0)	; Die Grundorientierung des Werkzeugs zeigt in Z-Richtung
TRAORI (1,0,1,1)	; Die Grundorientierung des Werkzeugs zeigt in Y/Z-Richtung (entspricht Stellung -45°)

6.2.3 Varianten der Orientierungsprogrammierung und Grundstellung (ORIRESET)

Orientierungsprogrammierung der Werkzeugorientierung bei TRAORI

In Verbindung mit einer programmierbaren Orientierungstransformation TRAORI können zusätzlich zu den Linearachsen X, Y, Z auch über die Rundachsbezeichner A..., B..., C... Achspositionen oder virtuelle Achsen mit Winkeln oder Vektorkomponenten programmiert werden. Für Orientierungs- und Maschinenachsen sind verschiedene Interpolationsarten möglich. Unabhängig davon, welche Orientierungspolynome PO[Winkel] und Achspolynome PO[Achse] gerade aktiv sind, können mehrere unterschiedliche Polynomarten wie z.B. G1, G2, G3, CIP oder POLY programmiert sein.

Die Änderung der Orientierung des Werkzeuges kann auch über Orientierungsvektoren programmiert werden. Hierbei kann die Endorientierung jedes Satzes entweder durch direkte Programmierung des Vektors oder durch Programmierung der Rundachspositionen erfolgen.

Hinweis

Varianten der Orientierungsprogrammierung bei Drei- bis Fünf-Achs-Transformationen

Bei der Drei- bis Fünf-Achs-Transformation schließen sich die Varianten

1. A, B, C direkte Angabe der Maschinenachspalten
2. A2, B2, C2 Winkelprogrammierung virtueller Achsen über Eulerwinkel oder RPY-Winkel
3. A3, B3, C3 Angabe der Vektorkomponenten
4. LEAD, TILT Angabe der Voreil- und Seitwärtswinkel bezogen auf die Bahn und Oberfläche
5. A4, B4, C4 und A5, B5, C5 Flächennormalenvektor am Satzanfang und am Satzende
6. A6, B6, C6 und A7, B7, C7 Interpolation des Orientierungsvektors auf einer Kegelmantelfläche
7. A8, B8, C8 Umorientierung des Werkzeuges, Richtung und Weglänge der Abhebebewegung

gegenseitig aus.

Gemischt programmierte Werte werden durch Alarmmeldungen verhindert.

Grundstellung der Werkzeugorientierung ORIRESET

Durch Programmierung von ORIRESET(A, B, C) werden Orientierungsachsen linear und synchron von ihrer momentanen Position zu der angegebenen Grundstellungsposition gefahren.

Wird für eine Achse keine Grundstellungsposition programmiert, dann wird definierte Position aus dem dazugehörigen Maschinendatum \$MC_TRAFO5_ROT_AX_OFFSET_1/2 verwendet. Eventuell aktive Frames der Rundachsen werden dabei nicht berücksichtigt.

Hinweis

Nur wenn eine Orientierungstransformation mit TRAORI(...) aktiv ist, kann eine Grundstellung der Werkzeugorientierung kinematikunabhängig mit ORIRESET(...) ohne Alarm 14101 programmiert werden.

Beispiele

```
1. Beispiel für Maschinenkinematik CA (Kanalachsnamen C, A)
ORIRESET(90, 45)      ;C auf 90 Grad, A auf 45 Grad
ORIRESET(, 30)       ;C auf $MC_TRAFO5_ROT_AX_OFFSET_1/2[0], A auf 30 Grad
ORIRESET( )          ;C auf $MC_TRAFO5_ROT_AX_OFFSET_1/2[0],
                    ;A auf $MC_TRAFO5_ROT_AX_OFFSET_1/2[1]

2. Beispiel für Maschinenkinematik CAC (Kanalachsnamen C, A, B)
ORIRESET(90, 45, 90) ;C auf 90 Grad, A auf 45 Grad, B auf 90 Grad
ORIRESET( )          ;C auf $MC_TRAFO5_ROT_AX_OFFSET_1/2[0],
                    ;A auf $MC_TRAFO5_ROT_AX_OFFSET_1/2[1],
                    ;B auf $MC_TRAFO5_ROT_AX_OFFSET_1/2[2]
```

Programmierung der Drehungen LEAD, TILT und THETA

Die Drehungen der Werkzeugorientierung werden bei der Drei- bis Fünf-Achs-Transformation mit den Voreilwinkel LEAD und den Seitwärtswinkel TILT programmiert.

Bei einer Transformation mit dritter Rundachse sind sowohl für die Orientierung mit Vektorkomponenten als auch mit Angabe der Winkel LEAD, TILT zusätzliche Programmierungen von C2 (Verdrehungen des Orientierungsvektors) erlaubt.

Mit einer zusätzlichen dritten Rundachse kann die Drehung des Werkzeugs um sich selbst mit dem Drehwinkel THETA programmiert werden.

6.2.4 Programmierung der Werkzeugorientierung (A..., B..., C..., LEAD, TILT)

Funktion

Für die Programmierung der Orientierung des Werkzeugs gibt es folgende Möglichkeiten:

1. Direkte Programmierung der Bewegung der Rundachsen. Die Orientierungsänderung erfolgt immer im Basis- bzw. Maschinen-Koordinatensystem. Die Orientierungsachsen werden als Synchronachsen verfahren.
2. Programmierung in Euler- oder RPY-Winkeln gemäß Winkeldefinition über A_2 , B_2 , C_2 .
3. Programmierung des Richtungsvektors über A_3 , B_3 , C_3 . Der Richtungsvektor zeigt von der Werkzeugspitze in Richtung Werkzeugaufnahme.
4. Programmierung des Flächennormalenvektors am Satzanfang mit A_4 , B_4 , C_4 und am Statzende mit A_5 , B_5 , C_5 (Stirnfräsen).
5. Programmierung über Voreilwinkel $LEAD$ und Seitwärtswinkel $TILT$
6. Programmierung der Drehachse des Kegels als normierter Vektor über A_6 , B_6 , C_6 oder der Zwischenorientierung auf der Kegelmantelfläche über A_7 , B_7 , C_7 , siehe Kapitel "Orientierungsprogrammierung entlang einer Kegelmantelfläche (ORIPLANE, ORICONxx)".
7. Programmierung der Umoorientierung, Richtung und Weglänge des Werkzeugs während der Abhebebewegung über A_8 , B_8 , C_8 , siehe Kapitel "Glättung des Orientierungsverlaufs (ORIPATHS A8=, B8=, C8=)"

Hinweis

In allen Fällen ist die Orientierungsprogrammierung nur zulässig, wenn eine Orientierungstransformation eingeschaltet ist.

Vorteil: Diese Programme sind auf jede Maschinenkinematik portierbar.

Definition der Werkzeugorientierung über G-Code

Hinweis

Maschinenhersteller

Über Maschinendatum kann zwischen Euler- oder RPY-Winkeln umgeschaltet werden. Bei entsprechenden Maschinendaten Einstellungen ist eine Umschaltung sowohl abhängig als auch unabhängig vom aktiven G-Code der Gruppe 50 möglich. Folgende Einstellmöglichkeiten stehen zur Auswahl:

1. Wenn beide Maschinendaten für die Definition der Orientierungsachsen und Definition der Orientierungswinkel über G-Code auf Null gesetzt sind:
Die mit A_2 , B_2 , C_2 programmierten Winkel werden **abhängig vom Maschinendatum** Winkeldefinition der Orientierungsprogrammierung entweder als Euler- oder RPY-Winkeln interpretiert.
2. Wenn das Maschinendatum für die Definition der Orientierungsachsen über G-Code auf Eins gesetzt ist, erfolgt Umschaltung **abhängig** vom aktiven G-Code der Gruppe 50:
Die mit A_2 , B_2 , C_2 programmierten Winkel werden gemäß eines der aktiven G-Codes `ORIEULER`, `ORIRPY`, `ORIVIRT1`, `ORIVIRT2`, `ORIXPOS` und `ORIPY2` interpretiert. Die mit den Orientierungsachsen programmierten Werte werden entsprechend dem aktiven G-Code der Gruppe 50 auch als Orientierungswinkel interpretiert.
3. Wenn das Maschinendatum für die Definition der Orientierungswinkel über G-Code auf Eins und das Maschinendatum für die Definition der Orientierungsachsen über G-Code auf Null gesetzt ist erfolgt Umschaltung **unabhängig** vom aktiven G-Code der Gruppe 50:
Die mit A_2 , B_2 , C_2 programmierten Winkel werden gemäß eines der aktiven G-Codes `ORIEULER`, `ORIRPY`, `ORIVIRT1`, `ORIVIRT2`, `ORIXPOS` und `ORIPY2` interpretiert. Die mit den Orientierungsachsen programmierten Werte werden unabhängig vom aktiven G-Code der Gruppe 50 immer als Rundachspositionen interpretiert.

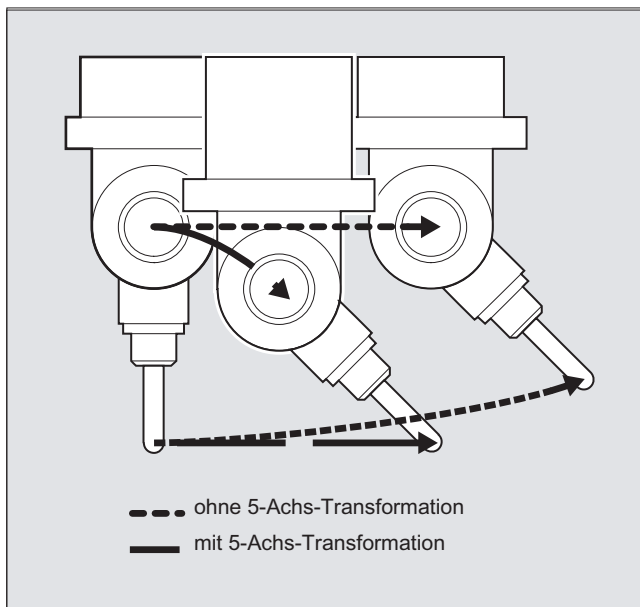
Programmierung

<code>G1 X Y Z A B C</code>	Programmierung der Bewegung der Rundachsen
<code>G1 X Y Z A2= B2= C2=</code>	Programmierung in Eulerwinkeln
<code>G1 X Y Z A3== B3== C3==</code>	Programmierung des Richtungsvektors
<code>G1 X Y Z A4== B4== C4==</code>	Programmierung des Flächennormalenvektors am Satzanfang
<code>G1 X Y Z A5== B5== C5==</code>	Programmierung des Flächennormalenvektors am Satzende
<code>LEAD=</code>	Voreilwinkel für die Programmierung der Werkzeugorientierung
<code>TILT=</code>	Seitwärtswinkel für die Programmierung der Werkzeugorientierung

Parameter

G...	Angabe der Bewegungsart der Rundachsen
X Y Z	Angabe der Linearachsen
A B C	Angabe der Maschinenachspalten der Rundachsen
A2 B2 C2	Winkelprogrammierung (Euler- oder RPY-Winkel) virtueller Achsen bzw. Orientierungsachsen
A3 B3 C3	Angabe der Vektorkomponenten Richtungsvektors
A4 B4 C4	Angabe z. B. beim Stirnfräsen die Komponente des Flächennormalenvektors am Satzanfang
A5 B5 C5	Angabe z. B. beim Stirnfräsen die Komponente des Flächennormalenvektors am Satzende
LEAD	Winkel relativ zum Flächennormalenvektor, in der von Bahntangente und Flächennormalenvektor aufgespannten Ebene
TILT	Winkel in der Ebene, senkrecht zur Bahntangente relativ zum Flächennormalenvektor

Beispiel Gegenüberstellung ohne und mit 5-Achs-Transformation



Weitere Informationen

In der Regel werden 5-Achs-Programme von CAD/CAM-Systemen erzeugt und nicht an der Steuerung eingegeben. Deshalb wenden sich die folgenden Erklärungen hauptsächlich an Programmierer von Postprozessoren.

Die Art der Orientierungsprogrammierung wird in der G-Code Gruppe 50 festgelegt:

G-Funktion	Orientierungsprogrammierung
ORIEULER	über Euler-Winkel
ORIRPY	über RPY-Winkel (Drehreihenfolge ZYX)
ORIVIRT1	über virtuelle Orientierungsachsen (Definition 1)
ORIVIRT2	über virtuelle Orientierungsachsen (Definition 2)
ORIXPOS	über virtuelle Orientierungsachsen mit Rundachspositionen
ORIPY2	über RPY-Winkel (Drehreihenfolge XYZ)

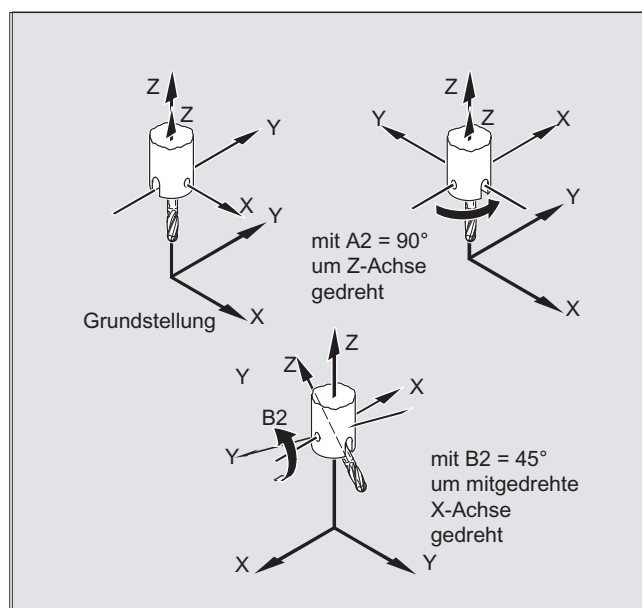
Hinweis

Über Maschinendaten können vom Maschinenhersteller verschiedene Varianten definiert werden. Bitte beachten Sie die Angaben des Maschinenherstellers.

Programmierung in Eulerwinkeln ORIEULER

Die bei der Orientierungsprogrammierung mit A_2 , B_2 , C_2 programmierten Werte werden als Eulerwinkel (in Grad) interpretiert.

Der Orientierungsvektor ergibt sich, indem ein Vektor in Z-Richtung zunächst mit A_2 um die Z-Achse, dann mit B_2 um die neue X-Achse und zuletzt mit C_2 um die neue Z-Achse gedreht wird.



In diesem Fall ist der Wert von C_2 (Drehung um neue Z-Achse) bedeutungslos und muss nicht programmiert werden.

Programmierung in RPY-Winkeln ORIRPY

Die bei der Orientierungsprogrammierung mit A_2 , B_2 , C_2 programmierten Werte werden als RPY-Winkel (in Grad) interpretiert.

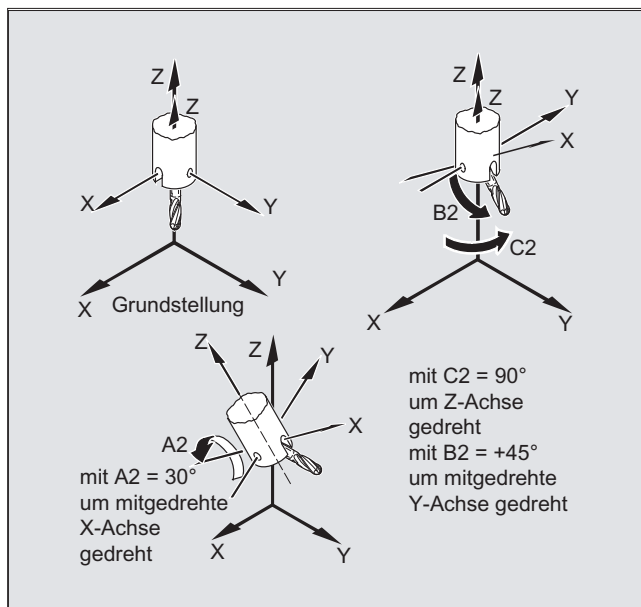
Hinweis

Im Gegensatz zur Eulerwinkel-Programmierung haben hier alle drei Werte Einfluss auf den Orientierungsvektor.

Bei Winkeldefinition mit Orientierungswinkel über RPY-Winkel gilt für die Orientierungsachsen:

$\$MC_ORI_DEF_WITH_G_CODE = 0$

Der Orientierungsvektor ergibt sich, indem ein Vektor in Z-Richtung zunächst mit C_2 um die Z-Achse, dann mit B_2 um die neue Y-Achse und zuletzt mit A_2 um die neue X-Achse gedreht wird.



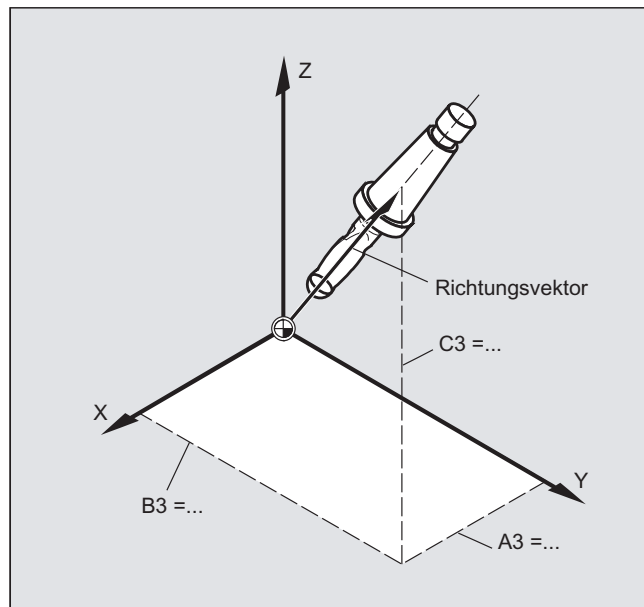
Ist das Maschinendatum zur über Definition der Orientierungsachsen über G-Code $\$MC_ORI_DEF_WITH_G_CODE = 1$, dann gilt:

Der Orientierungsvektor ergibt sich, indem ein Vektor in Z-Richtung zunächst mit A_2 um die Z-Achse, dann mit B_2 um die neue Y-Achse und zuletzt mit C_2 um die neue X-Achse gedreht wird.

Programmierung des Richtungsvektors

Die Komponenten des Richtungsvektors werden mit A_3 , B_3 , C_3 programmiert. Der Vektor zeigt in Richtung Werkzeugaufnahme; die Länge des Vektors ist dabei ohne Bedeutung.

Nicht programmierte Vektorkomponenten werden gleich Null gesetzt.



Programmierung der Werkzeugorientierung mit LEAD= und TILT=

Die resultierende Werkzeugorientierung wird ermittelt aus:

- Bahntangente
- Flächennormalenvektor
am Satzanfang A_4, B_4, C_4 und am Satzende A_5, B_6, C_5
- Voreilwinkel $LEAD$
in der von Bahntangente und Flächennormalenvektor aufgespannten Ebene
- Seitwärtswinkel $TILT$ am Satzende
senkrecht zur Bahntangente und relativ zum Flächennormalenvektor

Verhalten bei Innenecken (bei 3D-WZK):

Wenn der Satz an einer Innenecke verkürzt wird, wird die resultierende Werkzeugorientierung ebenso am Satzende erreicht.

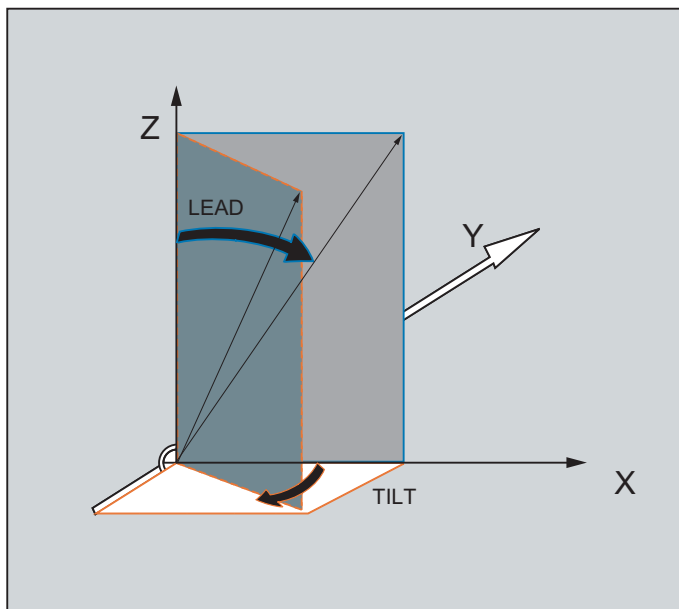
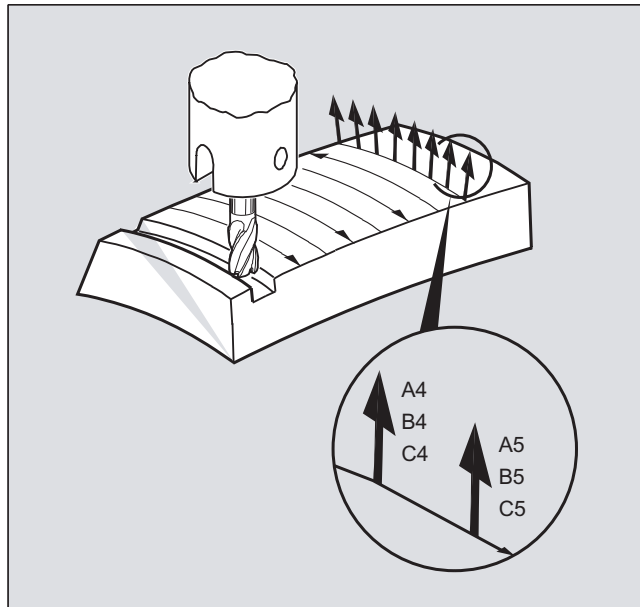


Bild 6-1 Definition der Werkzeugorientierung mit LEAD= und TILT=

6.2.5 Stirnfräsen (A4, B4, C4, A5, B5, C5)

Funktion

Stirnfräsen dient zur Bearbeitung beliebig gekrümmter Oberflächen.



Für diese Art des 3D-FräSENS benötigen Sie die zeilenweise Beschreibung der 3D-Bahnen auf der Werkstückoberfläche.

Die Berechnungen werden unter Berücksichtigung der Werkzeugform und Werkzeugabmessungen üblicherweise im CAM durchgeführt. Die fertig berechneten NC-Sätze werden dann über Postprozessoren in die Steuerung eingelesen.

Programmierung der Bahnkrümmung

Beschreibung der Flächen

Die Beschreibung der Bahnkrümmung erfolgt über Flächennormalenvektoren mit folgenden Komponenten:

A4, B4, C4 Startvektor am Satzanfang

A5, B5, C5 Endvektor am Satzende

Steht in einem Satz nur der Startvektor, bleibt der Flächennormalenvektor über den ganzen Satz konstant. Steht in einem Satz nur der Endvektor, so wird vom Endwert des vorherigen Satzes über Großkreisinterpolation zum programmierten Endwert interpoliert.

Sind Start- und Endvektor programmiert, so wird zwischen beiden Richtungen ebenfalls über Großkreisinterpolation interpoliert. Hierdurch lassen sich kontinuierlich glatte Bahnwege erzeugen.

In der Grundstellung zeigen Flächennormalenvektoren unabhängig von der aktiven Ebene G17 bis G19 in Z-Richtung.

Die Länge eines Vektors ist ohne Bedeutung.

Nicht programmierte Vektorkomponenten werden zu Null gesetzt.

Bei aktivem ORIWKs (siehe "Bezug der Orientierungsachsen (ORIWKs, ORIMKS) (Seite 346)") beziehen sich die Flächennormalenvektoren auf den aktiven Frame und werden bei Frame-Drehung mitgedreht.

Maschinenhersteller

Der Flächennormalenvektor muss innerhalb eines über Maschinendatum einstellbaren Grenzwertes senkrecht zur Bahntangente stehen, ansonsten wird Alarm ausgegeben.

6.2.6 Bezug der Orientierungsachsen (ORIWKs, ORIMKS)

Funktion

Bei Orientierungsprogrammierung im Werkstückkoordinatensystem über

- Euler- bzw. RPY-Winkel oder
- Orientierungsvektor

kann der Verlauf der Drehbewegung über ORIMKS/ORIWKs eingestellt werden.

Hinweis

Maschinenhersteller

Die Interpolationsart für die Orientierung wird festgelegt mit dem Maschinendatum:

MD21104 \$MC_ORI_IPO_WITH_G_CODE

= FALSE: Bezug sind die G-Funktionen ORIWKs und ORIMKS

= TRUE: Bezug sind die G- Funktionen der 51. Gruppe (ORIXES, ORIVECT, ORIPLANE, ...)

Syntax

ORIMKS= . . .

ORIWKs= . . .

Bedeutung

ORIMKS	Drehung im Maschinenkoordinatensystem
ORIWKs	Drehung im Werkstückkoordinatensystem

Hinweis

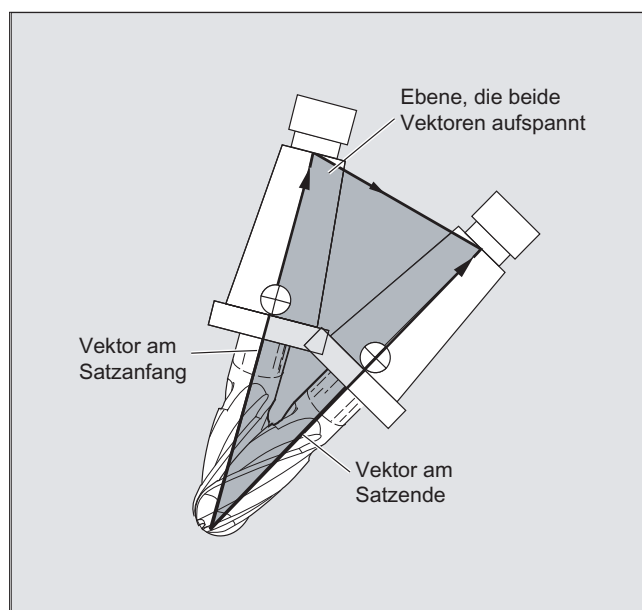
ORIWKs ist Grundeinstellung. Ist bei einem Fünf-Achs-Programm nicht von vornherein klar, auf welcher Maschine es ablaufen soll, so ist grundsätzlich ORIWKs zu wählen. Welche Bewegungen die Maschine tatsächlich ausführt, hängt von der Maschinenkinematik ab.

Mit **ORIMKS** können tatsächliche Maschinenbewegungen programmiert werden, z. B. um Kollisionen mit Vorrichtungen o. ä. zu vermeiden.

Beschreibung

Bei **ORIMKS** ist die ausgeführte Werkzeugbewegung von der Maschinenkinematik **abhängig**. Bei Orientierungsänderung mit raumfester Werkzeugspitze wird zwischen den Rundachpositionen linear interpoliert.

Bei **ORIWKS** ist die Werkzeugbewegung von der Maschinenkinematik **unabhängig**. Bei Orientierungsänderung mit raumfester Werkzeugspitze bewegt sich das Werkzeug in der vom Anfangs- und Endvektor aufgespannten Ebene.



Singuläre Stellungen

Hinweis

ORIWKS

Orientierungsbewegungen im Bereich der singulären Stellung der Fünf-Achs-Maschine erfordern große Bewegungen der Maschinenachsen. (Beispielsweise sind bei einem Drehschwenkkopf mit C als Drehachse und A als Schwenkachse alle Stellungen mit $A=0$ singulär.)

Maschinenhersteller

Um die Maschinenachsen nicht zu überlasten, senkt die Geschwindigkeitsführung die Bahngeschwindigkeit in der Nähe der singulären Stellen stark ab.

Mit den Maschinendaten

\$MC_TRAFO5_NON_POLE_LIMIT

\$MC_TRAFO5_POLE_LIMIT

kann die Transformation so parametrisiert werden, dass Orientierungsbewegungen in der Nähe des Pols durch den Pol gelegt werden und eine zügige Bearbeitung möglich ist.

Singuläre Stellen werden nur mit dem MD \$MC_TRAFO5_POLE_LIMIT behandelt.

Literatur:

/FB3/ Funktionshandbuch Sonderfunktionen; 3- bis 5-Achs-Transformation (F2), Kapitel "Singuläre Stellen und ihre Behandlung".

6.2.7 Programmierung der Orientierungsachsen (ORIAxes, ORIVect, ORIEuler, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2)

Funktion

Die Funktion "Orientierungsachsen" beschreibt die Orientierung des Werkzeugs im Raum und wird durch Programmierung der Offsets für die Rundachsen erreicht. Ein weiterer dritter Freiheitsgrad kann durch die zusätzliche Drehung des Werkzeugs um sich selbst erzielt werden. Diese Werkzeugorientierung erfolgt beliebig im Raum über eine dritte Rundachse und erfordert die Sechs-Achs-Transformation. Die Eigendrehung des Werkzeugs um sich selbst wird abhängig von der Interpolationsart der Drehvektoren mit dem Drehwinkel THETA festgelegt (siehe "Drehungen der Werkzeugorientierung (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) (Seite 359)".

Orientierungsachsen werden über die Achsbezeichner A2, B2, C2 programmiert.

Syntax

```
N... ORIAxes/ORIVect           ; Lineare oder Großkreisinterpolation
N... G1 X Y Z A B C

N... ORIPlane                 ; Orientierungsinterpolation der Ebene

N... ORIEuler/ORIRPY/ORIRPY2  ; Orientierungswinkel Euler-/RPY-Winkel
N... G1 X Y Z A2= B2= C2=     ; Winkelprogrammierung virtueller Achsen

N... ORIVIRT1/ORIVIRT2       ; virtuelle Orientierungsachsen Def. 1/2
N... G1 X Y Z A3= B3= C3=     ; Richtungsvektorprogrammierung
```

Hinweis

Für Orientierungsänderungen entlang einer im Raum befindlichen Kegelmantelfläche können weitere Rundachsoffsets der Orientierungsachsen programmiert werden (siehe "Orientierungsprogrammierung entlang einer Kegelmantelfläche (ORIPlane, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) (Seite 351)".

Bedeutung

ORIXES:	Lineare Interpolation der Maschinen- oder Orientierungsachsen
ORIVECT:	Großkreisinterpolation (identisch mit ORIPLANE)
ORIMKS:	Drehung im Maschinenkoordinatensystem
ORIWKS:	Drehung im Werkstückkoordinatensystem
	Beschreibung siehe "Bezug der Orientierungsachsen (ORIWKS, ORIMKS) (Seite 346)".
A= B= C=:	Programmierung der Maschinenachsposition
ORIEULER:	Orientierungsprogrammierung über Euler-Winkel
ORIRPY:	Orientierungsprogrammierung über RPY-Winkel
	Die Drehreihenfolge ist XYZ, wobei gilt:
	<ul style="list-style-type: none"> • A2 ist der Drehwinkel um X • B2 ist der Drehwinkel um Y • C2 ist der Drehwinkel um Z
ORIRPY2:	Orientierungsprogrammierung über RPY-Winkel
	Die Drehreihenfolge ist ZYX, wobei gilt:
	<ul style="list-style-type: none"> • A2 ist der Drehwinkel um Z • B2 ist der Drehwinkel um Y • C2 ist der Drehwinkel um X
A2= B2= C2=:	Winkelprogrammierung virtueller Achsen
ORIVIRT1/ORIVIRT2:	Orientierungsprogrammierung über virtuelle Orientierungsachsen
	Definition 1:
	Festlegung nach MD21120 \$MC_ORIAX_TURN_TAB_1
	Definition 2:
	Festlegung nach MD21130 \$MC_ORIAX_TURN_TAB_2
A3= B3= C3=:	Richtungsvektorprogrammierung der Richtungsachse

Beschreibung

Maschinenhersteller

Mit MD21102 \$MC_ORI_DEF_WITH_G_CODE wird festgelegt, wie die programmierten Winkel A2, B2, C2 definiert werden:

Definition erfolgt nach MD21100 \$MC_ORIENTATION_IS_EULER (Standard) oder Definition erfolgt nach G-Gruppe 50 (ORIEULER, ORIRPY, ORIVIRT1, ORIVIRT2).

Mit MD21104 \$MC_ORI_IPO_WITH_G_CODE wird festgelegt, welche Interpolationsart wirksam ist: ORIWKS/ORIMKS oder ORIAXES/ORIVECT.

Betriebsart JOG

Die Orientierungswinkel werden in dieser Betriebsart immer linear interpoliert. Beim kontinuierlichen und inkrementellen Verfahren über Verfahrtaasten kann nur eine Orientierungsachse verfahren werden. Über die Handräder können die Orientierungsachsen gleichzeitig verfahren werden.

Für das Handverfahren von Orientierungsachsen wirkt der kanalspezifische Vorschub-Korrekturschalter bzw. der Eilgang-Korrekturschalter bei Eilgangüberlagerung.

Mit folgenden Maschinendaten ist eine separate Geschwindigkeitsvorgabe möglich:

MD21160 \$MC_JOG_VELO_RAPID_GEO

MD21165 \$MC_JOG_VELO_GEO

MD21150 \$MC_JOG_VELO_RAPID_ORI

MD21155 \$MC_JOG_VELO_ORI

Hinweis

SINUMERIK 840D sl mit "Transformationspaket Handling"

Mit der Funktion "Kartesisches Handverfahren" kann im JOG-Betrieb die Translation von Geometrieachsen in den Bezugssystemen MKS, WKS und TKS getrennt voneinander eingestellt werden.

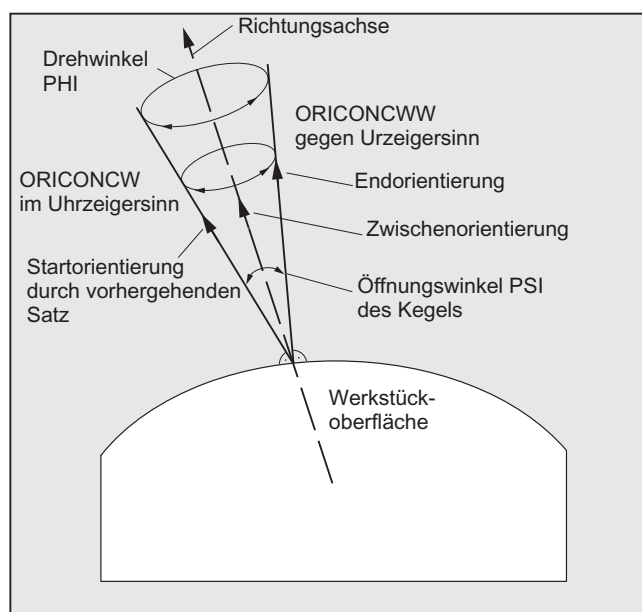
Literatur:

Funktionshandbuch Erweiterungsfunktionen; Kinematische Transformation (M1)

6.2.8 Orientierungsprogrammierung entlang einer Kegelmantelfläche (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO)

Funktion

Mit der erweiterten Orientierung ist es möglich, Orientierungsänderungen entlang sich einer im Raum befindlichen Kegelmantelfläche auszuführen. Die Interpolation des Orientierungsvektors auf einer Kegelmantelfläche erfolgt mit den modalen Befehlen ORICONxx. Für die Interpolation in einer Ebene kann die Endorientierung mit ORIPLANE programmiert werden. Generell wird die Startorientierung durch die vorhergehenden Sätze festgelegt.



Programmierung

Die Endorientierung wird entweder durch Angabe der Winkelprogrammierung in Euler- oder RPY-Winkel mit A2, B2, C2 oder durch Programmierung der Rundachspositionen mit A, B, C festgelegt. Für die Orientierungsachsen entlang der Kegelmantelfläche sind weitere Programmierangaben erforderlich:

- Drehachse des Kegels als Vektor mit A_6, B_6, C_6
- Öffnungswinkel ψ mit den Bezeichner NUT
- Zwischenorientierung im Kegelmantel mit A_7, B_7, C_7

Hinweis

Programmierung des Richtungsvektor A_6, B_6, C_6 für die Drehachse des Kegels

Die Programmierung einer Endorientierung ist nicht unbedingt erforderlich. Ist keine Endorientierung angegeben, dann wird ein voller Kegelmantel mit 360 Grad interpoliert.

Programmierung des Öffnungswinkel des Kegels mit NUT=winkel

Die Angabe einer Endorientierung ist zwingend erforderlich.

Ein vollständiger Kegelmantel mit 360 Grad kann auf diese Weise nicht interpoliert werden.

Programmierung der Zwischenorientierung A_7, B_7, C_7 im Kegelmantel

Die Angabe einer Endorientierung ist zwingend erforderlich. Die Orientierungsänderung und Drehrichtung wird eindeutig durch die drei Vektoren Start-, End- und Zwischenorientierung festgelegt. Alle drei Vektoren müssen hierbei voneinander unterschiedlich sein. Ist die programmierte Zwischenorientierung parallel zur Start- oder Endorientierung, dann wird eine lineare Großkreisinterpolation der Orientierung in der Ebene, die von Start- und Endvektor aufgespannt wird, durchgeführt.

Erweiterte Orientierungsinterpolation auf einer Kegelmantelfläche

```
N... ORICONCW oder ORICONCCW
N... A6= B6= C6= A3= B3= C3=
oder
N... ORICONTO
N... G1 X Y Z A6= B6= C6=
oder
N... ORICONIO
N... G1 X Y Z A7= B7= C7=
N... PO[PHI]=(a2, a3, a4, a5)
N... PO[PSI]=(b2, b3, b4, b5)
```

Interpolation auf einen Kegelmantel mit Richtungsvektor im/gegen Uhrzeigersinn des Kegels und Endorientierung oder tangentialem Übergang und Angabe der Endorientierung
oder
Angabe der Endorientierung und einer Zwischenorientierung im Kegelmantel mit Polynome für Drehwinkel und Polynome für Öffnungswinkel

Parameter

ORIPLANE	Interpolation in der Ebene (Großkreisinterpolation)
ORICONCW	Interpolation auf einer Kegelmantelfläche im Uhrzeigersinn
ORICONCCW	Interpolation auf einer Kegelmantelfläche gegen Uhrzeigersinn
ORICONTO	Interpolation auf einer Kegelmantelfläche tangentialer Übergang
A6= B6= C6=	Programmierung der Drehachse des Kegels (normierter Vektor)
NUT=winkel	Öffnungswinkel des Kegels in Grad
NUT=+179	Verfahrwinkel kleiner oder gleich 180 Grad
NUT=-181	Verfahrwinkel größer oder gleich 180 Grad
ORICONIO	Interpolation auf einer Kegelmantelfläche
A7= B7= C7=	Zwischenorientierung (Programmierung als normierter Vektor)
PHI	Drehwinkel der Orientierung um die Richtungsachse des Kegels
PSI	Öffnungswinkel des Kegels
mögliche Polynome	Außer den jeweiligen Winkeln sind auch Polynome maximal 5. Grades programmierbar
PO[PHI]=(a2, a3, a4, a5)	
PO[PSI]=(b2, b3, b4, b5)	

Beispiel: Unterschiedliche Orientierungsänderungen

Programmcode	Kommentar
...	
N10 G1 X0 Y0 F5000	
N20 TRAORI(1)	; Orientierungstransformation ein.
N30 ORIVECT	; Werkzeug-Orientierung als Vektor interpolieren.
...	; Werkzeugorientierung in der Ebene.
N40 ORIPLANE	; Großkreisinterpolation auswählen.
N50 A3=0 B3=0 C3=1	
N60 A3=0 B3=1 C3=1	; Orientierung in der Y/Z-Ebene um 45 Grad gedreht, am Satzende wird die Orientierung $(0, 1/\sqrt{2}, 1/\sqrt{2})$ erreicht.
...	
N70 ORICONCW	; Orientierungsprogrammierung auf Kegelmantel:
N80 A6=0 B6=0 C6=1 A3=0 B3=0 C3=1	; Der Orientierungsvektor wird auf einem Kegelmantel mit der Richtung $(0, 0, 1)$ bis zur Orientierung $(1/\sqrt{2}, 0, 1/\sqrt{2})$ im Uhrzeigersinn interpoliert, der Drehwinkel beträgt hierbei 270 Grad.
N90 A6=0 B6=0 C6=1	; Die Werkzeugorientierung durchläuft eine volle Umdrehung auf demselben Kegelmantel.

Beschreibung

Sollen Orientierungsänderungen auf einer beliebig im Raum liegenden Kegelmantelfläche beschrieben werden, dann muss der Vektor um den die Werkzeugorientierung gedreht werden soll, bekannt sein. Außerdem müssen die Start- und Endorientierung vorgegeben werden. Die Startorientierung ergibt sich aus den vorhergehenden Satz und die Endorientierung muss entweder programmiert oder durch andere Bedingungen festgelegt werden.

Programmierung in der Ebene ORIPLANE entspricht ORIVECT

Die Programmierung der Großkreisinterpolation zusammen mit Winkelpolynomen entspricht der Linear- und Polynominterpolation von Konturen. Die Werkzeugorientierung wird in einer Ebene interpoliert, die von der Start- und Endorientierung aufgespannt wird. Werden zusätzlich Polynome programmiert, dann kann der Orientierungsvektor auch aus der Ebene gekippt werden.

Programmierung von Kreisen in einer Ebene G2/G3, CIP und CT

Die erweiterte Orientierung entspricht der Interpolation von Kreisen in einer Ebene. Zu den entsprechenden Programmiermöglichkeiten von Kreisen mit Mittelpunktangabe oder Radiusangabe wie G2/G3, Kreis über Zwischenpunkt CIP und Tangentialkreise CT siehe

Literatur: Programmierhandbuch Grundlagen, "Wegbefehle programmieren".

Orientierungsprogrammierung

Interpolation des Orientierungsvektors auf einer Kegelmantelfläche ORICONxx

Für die Interpolation von Orientierungen auf einer Kegelmantelfläche können vier verschiedene Interpolationsarten aus der G-Code Gruppe 51 ausgewählt werden:

1. Interpolation auf einen Kegelmantel im Uhrzeigersinn `ORICONCW` mit Angabe der Endorientierung und der Kegelrichtung oder des Öffnungswinkels. Der Richtungsvektor wird mit den Bezeichnern `A6`, `B6`, `C6` und der Öffnungswinkel des Kegels wird mit dem Bezeichner `NUT=` Wertebereich im Intervall 0 bis 180 Grad programmiert.
2. Interpolation auf einen Kegelmantel gegen Uhrzeigersinn `ORICONCWW` mit Angabe der Endorientierung und der Kegelrichtung oder des Öffnungswinkels. Der Richtungsvektor wird mit den Bezeichnern `A6`, `B6`, `C6` und der Öffnungswinkel des Kegels wird mit dem Bezeichner `NUT=` Wertebereich im Intervall 0 bis 180 Grad programmiert.
3. Interpolation auf einen Kegelmantel `ORICONIO` mit Angabe der Endorientierung und einer Zwischenorientierung, die mit den Bezeichnern `A7`, `B7`, `C7` programmiert wird.
4. Interpolation auf einen Kegelmantel `ORICONTO` mit tangentialem Übergang und Angabe der Endorientierung. Der Richtungsvektor wird mit den Bezeichnern `A6`, `B6`, `C6` programmiert.

6.2.9 Orientierungsvorgabe zweier Kontaktpunkte (ORICURVE, PO[XH]=, PO[YH]=, PO[ZH]=)

Funktion

Programmierung der Orientierungsänderung durch die zweite Raumkurve ORICURVE

Eine weitere Möglichkeit der Programmierung von Orientierungsänderungen besteht darin, außer der Werkzeugspitze entlang einer Raumkurve auch die Bewegung eines zweiten Kontaktpunktes des Werkzeugs mit `ORICURVE` zu programmieren. Damit können Orientierungsänderungen des Werkzeugs, wie bei der Programmierung des Werkzeugvektors selber, eindeutig festgelegt werden.

Maschinenhersteller

Beachten Sie bitte die Hinweise des Maschinenherstellers zu über Maschinendatum einstellbare Achsbezeichner für die Programmierung der 2. Orientierungsbahn des Werkzeugs.

Programmierung

Bei dieser Interpolationsart können für die beiden Raumkurven Punkte mit `G1` bzw. Polynome mit `POLY` programmiert werden. Kreise und Evolventen sind nicht zulässig. Zusätzlich kann eine Spline-Interpolation mit `BSPLINE` und die Funktion "Zusammenfassung kurzer Spline-Sätze" aktiviert werden.

Literatur:

Funktionshandbuch Grundfunktionen; Bahnsteuerbetrieb, Genauhalt, Look Ahead (B1), Kapitel: Zusammenfassung kurzer Spline-Sätze

Die anderen Splinarten `ASPLINE` und `CSPLINE` sowie die Aktivierung eines Kompressors mit `COMPON`, `COMPCURV` oder `COMPCAD` sind nicht zulässig.

Die Bewegung der zwei Kontaktpunkte des Werkzeugs kann bei der Programmierung der Orientierungspolynome für Koordinaten bis maximal 5.Grades vorgegeben werden.

Erweiterte Orientierungsinterpolation mit zusätzlicher Raumkurve und Polynome für Koordinaten

```
N... ORICURVE
N... PO[XH]=(xe, x2, x3, x4, x5)
N... PO[YH]=(ye, y2, y3, y4, y5)
N... PO[ZH]=(ze, z2, z3, z4, z5)
```

Angabe der Bewegung des zweiten Kontaktpunktes des Werkzeugs und zusätzliche Polynome der jeweiligen Koordinaten

Parameter

ORICURVE	Interpolation der Orientierung mit Vorgabe der Bewegung zweier Kontaktpunkte des Werkzeuges.
XH YH ZH	Bezeichner der Koordinaten des zweiten Kontaktpunktes des Werkzeugs der zusätzlichen Kontur als Raumkurve
mögliche Polynome	Außer den jeweiligen Endpunkten sind die Raumkurven zusätzlich mit Polynomen programmierbar.
PO[XH]=(xe, x2, x3, x4, x5) PO[YH]=(ye, y2, y3, y4, y5) PO[ZH]=(ze, z2, z3, z4, z5)	
xe, ye, ze	Endpunkte der Raumkurve
xi, yi, zi	Koeffizienten der Polynome maximal 5. Grades

Hinweis

Bezeichner XH YH ZH für die Programmierung einer 2. Orientierungsbahn

Die Bezeichner müssen so gewählt werden, dass kein Konflikt mit anderen Bezeichnern der Linearachsen

X Y Z Achsen

und Rundachsen wie

A2 B2 C2 Eulerwinkel bzw. RPY-Winkel

A3 B3 C3 Richtungsvektoren

A4 B4 C4 bzw. A5 B5 C5 Flächennormalenvektoren

A6 B6 C6 Drehvektoren bzw. A7 B7 C7 Zwischenpunktkoordinaten

oder anderen Interpolationsparameter entsteht.

6.3 Orientierungspolynome (PO[Winkel], PO[Koordinate])

Funktion

Unabhängig davon, welche Polynominterpolation der G-Code Gruppe 1 gerade aktiv ist, können zwei verschiedene Typen von Orientierungspolynomen bis maximal 5. Grades bei einer Drei- bis Fünf-Achs-Transformation programmiert werden.

1. Polynome für **Winkel**: Voreilwinkel LEAD, Seitwärtswinkel TILT in Bezug auf die Ebene, die von Start- und Endorientierung aufgespannt wird.
2. Polynome für **Koordinaten**: XH, YH, ZH der zweiten Raumkurve für die Werkzeugorientierung eines Bezugspunktes auf dem Werkzeug.

Bei einer Sechs-Achs-Transformation kann zur Werkzeugorientierung zusätzlich die Drehung des Drehvektors THT mit Polynomen bis maximal 5. Grades für Drehungen des Werkzeugs selbst programmiert werden.

Syntax

Orientierungspolynome vom Typ 1 für Winkel

N... PO[PHI] = (a2, a3, a4, a5)

Drei- bis Fünf-Achs-Transformation

N... PO[PSI] = (b2, b3, b4, b5)

Drei- bis Fünf-Achs-Transformation

Orientierungspolynome vom Typ 2 für Koordinaten

N... PO[XH] = (xe, x2, x3, x4, x5)

N... PO[YH] = (ye, y2, y3, y4, y5)

N... PO[ZH] = (ze, z2, z3, z4, z5)

Bezeichner für die Koordinaten der zweiten Orientierungsbahn für die Werkzeugorientierung

Zusätzlich kann in beiden Fällen ein Polynom für die **Drehung** bei Sechs-Achs-Transformationen mit

N... PO[THT] = (c2, c3, c4, c5)

oder

N... PO[THT] = (d2, d3, d4, d5)

Bahnrelative Interpolation der Drehung

absoluter, relative und tangentielle Interpolation zur Orientierungsänderung

des Orientierungsvektors programmiert werden. Dies ist dann möglich, wenn die Transformation einen Drehvektor mit einem durch den Drehwinkel THETA programmierbaren und interpolierbaren Offset unterstützt.

Bedeutung

PO [PHI]	Winkel in der Ebene zwischen Start- und Endorientierung
PO [PSI]	Winkel der die Auskippung der Orientierung aus der Ebene zwischen Start- und Endorientierung beschreibt
PO [THT]	Drehwinkel der durch Drehung des Drehvektors einer der mit THETA programmierten G-Codes der Gruppe 54
PHI	Voreilwinkel LEAD
PSI	Seitwärtswinkel TILT
THETA	Drehung um die Werkzeugrichtung in Z
PO [XH]	X-Koordinate des Bezugspunktes auf dem Werkzeug
PO [YH]	Y-Koordinate des Bezugspunktes auf dem Werkzeug
PO [ZH]	Z-Koordinate des Bezugspunktes auf dem Werkzeug

Beschreibung

Orientierungspolynomen können nicht programmiert werden

- wenn die Splineinterpolationen ASPLINE, BSPLINE, CSPLINE aktiv sind. Polynome vom Typ1 für Orientierungswinkel sind für jede Interpolationsart außer Spline d.h. bei Linearinterpolation mit Eilgang G00 bzw. mit Vorschub G01 bei Polynominterpolation mit POLY und bei Kreis- bzw. Evolventeninterpolation mit G02, G03, CIP, CT, INVCW und INCCCW möglich. Polynome vom Typ2 für Orientierungskordinaten sind dagegen nur möglich, wenn Linearinterpolation mit Eilgang G00 bzw. mit Vorschub G01 oder Polynominterpolation mit POLY aktiv ist.
- wenn die Orientierung mittels Achsinterpolation ORIAxes interpoliert wird. In diesem Fall können direkt Polynome mit PO[A] und PO[B] für die Orientierungsachsen A und B programmiert werden.

Orientierungspolynome vom Typ 1 mit ORIVECT, ORIPLANE und ORICONxx

Bei Großkreisinterpolation und Kegelmantelinterpolation mit ORIVECT, ORIPLANE und ORICONxx sind nur Orientierungspolynome vom Typ 1 möglich.

Orientierungspolynome vom Typ 2 mit ORICURVE

Ist die Interpolation mit zusätzlicher Raumkurve ORICURVE aktiv, werden die kartesischen Komponenten des Orientierungsvektors interpoliert und es sind nur Orientierungspolynome vom Typ 2 möglich.

6.4 Drehungen der Werkzeugorientierung (ORIROTA, ORIROT, ORIROT, ORIROT, THETA)

Funktion

Soll bei Maschinentypen mit beweglichem Werkzeug auch die Orientierung des Werkzeugs veränderbar sein, so wird jeder Satz mit einer Endorientierung programmiert. Abhängig von der Maschinenkinematik können entweder die Orientierungsrichtung der Orientierungsachsen oder die Drehrichtung des Orientierungsvektors THETA programmiert werden. Für diese Drehvektoren sind verschiedene Interpolationsarten programmierbar:

- ORIROTA: Drehwinkel zu einer absolut vorgegebenen Drehrichtung.
- ORIROT: Drehwinkel relativ zur Ebene zwischen Start- und Endorientierung.
- ORIROT: Drehwinkel relativ zur Änderung des Orientierungsvektors.
- ORIROT: Tangentialer Drehwinkel zur Bahntangente.

Syntax

Nur wenn die Interpolationsart ORIROTA aktiv ist, kann der Drehwinkel oder der Drehvektor auf die vier möglichen Arten wie folgt programmiert werden:

1. Direkt die Rundachspositionen A , B , C
2. Eulerwinkel (in Grad) über A_2 , B_2 , C_2
3. RPY-Winkel (in Grad) über A_2 , B_2 , C_2
4. Richtungsvektor über A_3 , B_3 , C_3 (Drehwinkel mittels $THETA=<Wert>$)

Falls ORIROT oder ORIROT aktiv sind, kann der Drehwinkel nur noch direkt mit $THETA$ programmiert werden.

Eine Drehung kann auch allein in einem Satz programmiert werden, ohne dass eine Orientierungsänderung stattfindet. Dabei haben ORIROT und ORIROT keine Bedeutung. In diesem Fall wird der Drehwinkel immer in Bezug zur absoluten Richtung interpretiert (ORIROTA).

N... ORIROTA	Interpolation des Drehvektors festlegen
N... ORIROT	
N... ORIROT	
N... ORIROT	
N... A3= B3= C3= THETA=<Wert>	Drehung des Orientierungsvektors festlegen
N... PO[THT]=(d2, d3, d4, d5)	Drehwinkel mit Polynom 5. Grades interpolieren

Bedeutung

ORIOTA:	Drehwinkel zu einer absolut vorgegebenen Drehrichtung
ORIOTR:	Drehwinkel relativ zur Ebene zwischen Start- und Endorientierung
ORIOTT:	Drehwinkel als tangentialer Drehvektor zur Orientierungsänderung
ORIOTC:	Drehwinkel als tangentialer Drehvektor zur Bahntangente
THETA:	Drehung des Orientierungsvektors
THETA=<Wert>:	Drehwinkel in Grad, der am Satzende erreicht wird
THETA=@e:	Drehwinkel mit Endwinkel Θ_e des Drehvektors
THETA=AC (...):	Satzweise auf Maßangabe absolut umschalten
THETA=AC (...):	Satzweise auf Kettenmaßangabe umschalten
@e:	Endwinkel des Drehvektors sowohl absolut mit G90 als auch relativ mit G91 (Kettenmaßangabe) ist aktiv
PO [THT] = (...):	Polynom für den Drehwinkel

Beispiel: Drehungen der Orientierungen

Programmcode	Kommentar
N10 TRAORI	; Orientierungstransformation aktivieren
N20 G1 X0 Y0 Z0 F5000	; Orientierung des Werkzeugs
N30 A3=0 B3=0 C3=1 THETA=0	; in Z-Richtung mit Drehwinkel 0
N40 A3=1 B3=0 C3=0 THETA=90	; in X-Richtung und Drehung um 90 Grad
N50 A3=0 B3=1 C3=0 PO [THT]=(180,90)	; Orientierung
N60 A3=0 B3=1 C3=0 THETA=IC(-90)	; in Y-Richtung und Drehung auf 180 Grad
N70 ORIOTT	; bleibt konstant und Drehung auf 90 Grad
N80 A3=1 B3=0 C3=0 THETA=30	; Drehwinkel relativ zur Orientierungsänderung ; Drehvektor im Winkel 30 Grad zur X-Y Ebene

Bei der Interpolation von Satz N40 wird der Drehwinkel vom Startwert 0 Grad zum Endwert 90 Grad linear interpoliert. Im Satz N50 ändert sich der Drehwinkel von 90 Grad auf 180 Grad gemäß der Parabel $\theta(u) = +90u^2$. In N60 kann auch eine Drehung ausgeführt werden, ohne dass eine Orientierungsänderung stattfindet.

Bei N80 wird die Werkzeugorientierung von der Y-Richtung in X-Richtung gedreht. Dabei liegt die Orientierungsänderung in der X-Y Ebene und der Drehvektor bildet zu dieser Ebene einen Winkel von 30 Grad.

Beschreibung**ORIROTA**

Der Drehwinkel THETA wird bezüglich einer absolut festgelegten Richtung im Raum interpoliert. Die Grunddrehrichtung erfolgt über Maschinendaten

ORIROTR

Der Drehwinkel THETA wird relativ zur Ebene, die von der Start- und Endorientierung aufgespannt wird, interpretiert.

ORIROTT

Der Drehwinkel THETA wird relativ zur Orientierungsänderung interpretiert. Für $\text{THETA}=0$ wird der Drehvektor tangential zur Orientierungsänderung interpoliert und unterscheidet sich nur dann zu **ORIROTR**, wenn für die Orientierung mindestens ein Polynom für den "Kippwinkel PSI " programmiert wurde. Damit ergibt sich eine Orientierungsänderung, die nicht in der Ebene abläuft. Durch einen zusätzlich programmierten Drehwinkel THETA kann dann z. B. der Drehvektor so interpoliert werden, dass er immer einen bestimmten Wert zur Orientierungsänderung bildet.

ORIROTC

Der Drehvektor wird relativ zur Bahntangente mit einem durch den Winkel THETA programmierbaren Offset interpoliert. Für den Offsetwinkel kann dabei auch ein Polynom $\text{PO}[\text{THT}] = (\text{c2}, \text{c3}, \text{c4}, \text{c5})$ maximal 5. Grades programmiert werden.

6.5 Bahnrelative Orientierungen

6.5.1 Orientierungsarten relativ zur Bahn

Funktion

Mit dieser erweiterten Funktion wird die relative Orientierung nicht nur am Satzende, sondern über den gesamten Bahnverlauf erreicht. Es wird die im Vorgängersatz erreichte Orientierung mittels Großkreisinterpolation in die programmierte Endorientierung überführt. Grundsätzlich gibt es zwei Möglichkeiten die gewünschte Orientierung relativ zur Bahn zu programmieren:

1. Die Werkzeugorientierung als auch die Drehung des Werkzeugs wird mit ORIPATH, ORPATHS relativ zur Bahn interpoliert.
2. Der Orientierungsvektor wird wie bisher üblich programmiert und interpoliert. Mit ORIROTc wird die Drehung des Orientierungsvektors relativ zur Bahntangente angestellt.

Syntax

Die Interpolationsart der Orientierung und der Drehung des Werkzeugs wird programmiert mit:

N... ORIPATH	Bahnrelative Orientierung
N... ORIPATHS	Bahnrelative Orientierung mit Glättung des Orientierungsverlaufs
N... ORIROTc	Bahnrelative Interpolation des Drehvektors

Ein durch eine Ecke im Bahnverlauf hervorgerufener Knick der Orientierung kann mit ORIPATHS geglättet werden. Die Richtung und Weglänge der Abhebebewegung wird durch den Vektor mit den Komponenten $A8=X$, $B8=Y$ $C8=Z$ programmiert.

Mit ORIPATH/ORIPATHS können verschiedene Bezüge zur Bahntangente über die drei Winkel

- LEAD= Angabe Vorwärtswinkel bezogen auf die Bahn und Oberfläche
- TILT= Angabe von Seitwärtswinkel bezogen auf die Bahn und Oberfläche
- THETA= Drehwinkel

für den gesamten Bahnverlauf programmiert werden. Zum Drehwinkel THETA können mit $PO[THET] = (. . .)$ zusätzlich Polynome maximal 5. Grades programmiert werden.

Hinweis

Maschinenhersteller

Bitte beachten Sie die Angaben des Maschinenherstellers. Über projektierbare Maschinen- und Settingdaten können zur Bahnrelativen Orientierungsart weitere Einstellungen vorgenommen werden. Weitere Erläuterungen siehe

Literatur:

/FB3/ Funktionshandbuch Sonderfunktionen; 3- bis 5-Achs-Transformation (F2), Kapitel "Orientierung"

Bedeutung

Die Interpolation der Winkel `LEAD` und `TILT` ist über Maschinedatum unterschiedlich einstellbar:

- Der mit `LEAD` und `TILT` programmierte Bezug der Werkzeugorientierung wird über den ganzen Satz hinweg eingehalten.
- Vorwärtswinkel `LEAD`: Drehung um die Richtung senkrecht zur Tangente und Normalenvektor `TILT`: Drehung der Orientierung um den Normalenvektor.
- Vorwärtswinkel `LEAD`: Drehung um die Richtung senkrecht zur Tangente und Normalenvektor Seitwärtswinkel `TILT`: Drehung der Orientierung um die Richtung der Bahntangente.
- Drehwinkel `THETA`: Drehung des Werkzeugs um sich selbst mit einer zusätzlichen dritten Rundachse als Orientierungsachse bei Sechs-Achs-Transformation.

Hinweis

Bahnrelative Orientierung zusammen mit `OSC`, `OSS`, `OSSE`, `OSD`, `OST` unzulässig

Die bahnrelative Orientierungsinterpolation `ORIPATH` bzw. `ORIPATHS` und `ORIOTC` kann nicht zusammen mit der Glättung des Orientierungsverlaufs mit einen der G-Codes aus der Gruppe 34 programmiert werden. Hierfür muss `OSOF` aktiv sein.

6.5.2 Bahnrelative Drehung der Werkzeugorientierung (`ORIPATH`, `ORIPATHS`, Drehwinkel)

Funktion

Bei einer Sechs-Achs-Transformation kann zur Werkzeugorientierung beliebig im Raum auch das Werkzeug mit einer dritten Rundachse um sich selbst gedreht werden. Bei bahnrelativer Drehung der Werkzeugorientierung mit `ORIPATH` bzw. `ORIPATHS` kann die zusätzliche Drehung über den Drehwinkel `THETA` programmiert werden. Alternativ hierzu können die Winkel `LEAD` und `TILT` durch einen Vektor, der in der Ebene senkrecht zur Werkzeugrichtung liegt, programmiert werden.

Maschinenhersteller

Bitte beachten Sie die Angaben des Maschinenherstellers. Über Maschinedatum kann die Interpolation der Winkel `LEAD` und `TILT` unterschiedlich eingestellt werden.

Syntax

Drehung der Werkzeugorientierung und des Werkzeugs

Die Werkzeugorientierungsart relativ zur Bahn wird mit `ORIPATH` oder `ORIPATHS` aktiviert.

<code>N... ORIPATH</code>	Orientierungsart bezogen auf die Bahn aktivieren
<code>N... ORIPATHS</code>	Orientierungsart bezogen auf die Bahn mit Glättung des Orientierungsverlaufs aktivieren

Aktivierung der drei möglichen Winkel mit Drehwirkung:

- N... LEAD= Winkel für die programmierten Orientierung relativ zum Flächennormalenvektor
- N... TILT= Winkel für die programmierte Orientierung in der Ebene senkrecht zur Bahntangente relativ zum Flächennormalenvektor
- N... THETA= Drehwinkel relativ zur Orientierungsänderung um die Werkzeugrichtung der dritten Rundachse

Die Werte der Winkel am Satzende werden mit LEAD=Wert, TILT=Wert bzw. THETA=Wert programmiert. Zusätzlich zu den konstanten Winkeln können für alle drei Winkel Polynome maximal 5. Grades programmiert werden.

- N... PO[PHI]=(a2, a3, a4, a5) Polynom für den Voreilwinkel LEAD
- N... PO[PSI]=(b2, b3, b4, b5) Polynom für den Seitwärtswinkel TILT
- N... PO[THT]=(d2, d3, d4, d5) Polynom für den Drehwinkel THETA

Bei der Programmierung können die höheren Polynomkoeffizienten, die Null sind, weggelassen werden. Beispiel PO[PHI]=a2 ergibt für den Voreilwinkel LEAD eine Parabel.

Bedeutung

Bahnrelative Werkzeugorientierung

- ORIPATH Werkzeugorientierung bezogen auf die Bahn
- ORIPATHS Werkzeugorientierung bezogen auf die Bahn, Knick im Orientierungsverlauf wird geglättet
- LEAD Winkel relativ zum Flächennormalenvektor, in der von Bahntangente und Flächennormalenvektor aufgespannten Ebene
- TILT Drehung der Orientierung um die Z-Richtung bzw. Drehung um die Bahntangente
- THETA Drehung um die Werkzeugrichtung nach Z
- PO[PHI] Orientierungspolynom für den Voreilwinkel LEAD
- PO[PSI] Orientierungspolynom für den Seitwärtswinkel TILT
- PO[THT] Orientierungspolynom für den Drehwinkel THETA

Hinweis

Drehwinkel THETA

Für die Drehung des Werkzeugs mit dritter Rundachse als Orientierungsachse um sich selbst, ist eine Sechs-Achs-Transformation erforderlich.

6.5.3 Bahnrelative Interpolation der Werkzeugdrehung (ORIROTC, THETA)

Funktion

Interpolation mit Drehvektoren

Zur mit ORIROTC programmierten Drehung des Werkzeugs relativ zur Bahntangenten kann der Drehvektor auch mit einem durch den Drehwinkel THETA programmierbaren Offset interpoliert werden. Dabei kann für den Offsetwinkel mit PO[THT] ein Polynom bis maximal 5. Grades programmiert werden.

Syntax

N... ORIROTC	Drehung des Werkzeugs relativ zur Bahntangente anstellen
N... A3= B3= C3= THETA=Wert	Drehung des Orientierungsvektors festlegen
N... A3= B3= C3= PO[THT]=(c2, c3, c4, c5)	Offsetwinkel mit Polynom maximal 5. Grades interpolieren

Eine Drehung kann auch allein in einem Satz programmiert werden, ohne dass eine Orientierungsänderung stattfindet.

Bedeutung

Bahnrelative Interpolation der Drehung des Werkzeugs bei Sechs-Achs-Transformation

ORIROTC	tangentialer Drehvektor zur Bahntangente anstellen
THETA=Wert	Drehwinkel in Grad, der am Satzende erreicht wird
THETA=θe	Drehwinkel mit Endwinkel Θ_e des Drehvektors
THETA=AC (...)	Satzweise auf Maßangabe absolut umschalten
THETA=IC (...)	Satzweise auf Kettenmaßangabe umschalten
PO[THT]=(c2, c3, c4, c5)	Offsetwinkel mit Polynom 5. Grades interpolieren

Hinweis

Interpolation des Drehvektors ORIROTC

Soll gegen die Orientierungsrichtung des Werkzeugs auch die Drehung des Werkzeugs relativ zur Bahntangente angestellt werden, dann ist dies nur bei einer Sechs-Achs-Transformation möglich.

Bei aktiven ORIROTC

Der Drehvektor ORIROTA kann nicht programmiert werden. Im Falle einer Programmierung wird der ALARM 14128 "Absolutprogrammierung der Werkzeugdrehung bei aktivem ORIROTC" ausgegeben.

Orientierungsrichtung des Werkzeugs bei Drei- bis Fünf-Achs-Transformation

Die Orientierungsrichtung des Werkzeugs kann wie bei der Drei- bis Fünf-Achs-Transformation gewohnt über Eulerwinkel bzw. RPY-Winkel oder des Richtungsvektoren programmiert werden. Auch sind Orientierungsänderungen des Werkzeugs im Raum durch Programmierung der Großkreisinterpolation ORIVECT, der linearen Interpolation der Orientierungsachsen ORIAxes, alle Interpolationen auf einer Kegelmantelfläche ORICONxx sowie der Interpolation zusätzlich zur Raumkurve mit zwei Kontaktpunkten des Werkzeugs ORICURVE möglich.

G	Angabe der Bewegungsart der Rundachsen
X Y Z	Angabe der Linearachsen
ORIAxes	Lineare Interpolation der Maschinen- oder Orientierungsachsen
ORIVECT	Großkreisinterpolation (identisch mit ORIPLANE)
ORIMKS	Drehung im Maschinenkoordinatensystem
ORIWKS	Drehung im Werkstückkoordinatensystem
	Beschreibung siehe Kap. Drehungen der Werkzeugorientierung
A= B= C=	Programmierung der Maschinenachspannung
ORIEULER	Orientierungsprogrammierung über Euler-Winkel
ORIRPY	Orientierungsprogrammierung über RPY-Winkel
A2= B2= C2=	Winkelprogrammierung virtueller Achsen
ORIVIRT1	Orientierungsprogrammierung über virtuelle Orientierungsachsen
ORIVIRT2	(Definition 1), Festlegung nach MD \$MC_ORIAX_TURN_TAB_1 (Definition 2), Festlegung nach MD \$MC_ORIAX_TURN_TAB_2
A3= B3= C3=	Richtungsvektorprogrammierung der Richtungsachse
ORIPLANE	Interpolation in der Ebene (Großkreisinterpolation)
ORICONCW	Interpolation auf einer Kegelmantelfläche im Uhrzeigersinn
ORICONCCW	Interpolation auf einer Kegelmantelfläche gegen Uhrzeigersinn
ORICONTO	Interpolation auf einer Kegelmantelfläche tangentialer Übergang
A6= B6= C6=	Programmierung der Drehachse des Kegels (normierter Vektor)
NUT=winkel	Öffnungswinkel des Kegels in Grad
NUT=+179	Verfahrwinkel kleiner oder gleich 180 Grad
NUT=-181	Verfahrwinkel größer oder gleich 180 Grad
ORICONIO	Interpolation auf einer Kegelmantelfläche
A7= B7= C7=	Zwischenorientierung (Programmierung als normierter Vektor)
ORICURVE	Interpolation der Orientierung mit Vorgabe der Bewegung zweier Kontaktpunkte des Werkzeuges. Außer den jeweiligen Endpunkten sind zusätzliche Raumkurven Polynome programmierbar.
XH YH ZH z.B. mit Polynome PO[XH]=(xe, x2, x3, x4, x5)	

Hinweis

Wird die Werkzeugorientierung mit aktiven ORIAxes über die Orientierungsachsen interpoliert, dann wird die bahnrelative Anstellung des Drehwinkels nur am Satzende erfüllt.

6.5.4 Glättung des Orientierungsverlaufs (ORIPATHS A8=, B8=, C8=)

Funktion

Bei beschleunigungsstetigen Orientierungsänderungen an der Kontur sind Unterbrechungen der Bahnbewegungen, die besonders an einer Ecke der Kontur auftreten können unerwünscht. Der sich hieraus ergebene Knick im Orientierungsverlauf kann durch Einfügen eines eigenen Zwischensatzes geglättet werden. Die Orientierungsänderung erfolgt dann beschleunigungsstetig, wenn während der Umorientierung auch ORIPATHS aktiv ist. In dieser Phase kann eine Abhebebewegung des Werkzeugs durchgeführt werden.

Maschinenhersteller

Beachten Sie bitte die Hinweise des Maschinenherstellers zu gegebenenfalls vordefinierten Maschinendaten und Settingdaten mit denen diese Funktion aktiviert wird.

Über Maschinendatum ist einstellbar, wie der Abhebevektor interpretiert wird:

1. Im Werkzeugkoordinatensystem wird die Z-Koordinate durch die Werkzeugrichtung definiert.
2. Im Werkstückkoordinatensystem wird die Z-Koordinate durch die aktive Ebene definiert.

Weitere Erläuterungen zur Funktion "Bahnrelative Orientierung" siehe

Literatur:

Funktionshandbuch Sonderfunktionen; Mehrachstransformationen (F2)

Syntax

Für stetige Werkzeugorientierungen bezogen auf die gesamte Bahn sind an einer Ecke der Kontur weitere Programmierangaben erforderlich. Die Richtung und die Weglänge dieser Bewegung wird durch den Vektor mit den Komponenten A8=X, B8=Y, C8=Z programmiert:

```
N... ORIPATHS A8=X B8=Y C8=Z
```

Bedeutung

ORIPATHS	Werkzeugorientierung bezogen auf die Bahn, ein Knick im Orientierungsverlauf wird geglättet.
A8= B8= C8=	Vektorkomponenten für Richtung und Weglänge
X, Y, Z	Abhebebewegung in Werkzeugrichtung

Hinweis

Programmierung des Richtungsvektors A8, B8, C8

Ist die Länge dieses Vektors gleich Null erfolgt keine Abhebebewegung.

ORIPATHS

Die bahnbezogene Werkzeugorientierung wird mit ORIPATHS aktiv. Anderenfalls wird die Orientierung mittels linearer Großkreisinterpolation von der Start- zur Endorientierung überführt.

6.6 Komprimierung der Orientierung (COMPON, COMPCURV, COMPCAD)

Funktion

NC-Programme, in denen eine Orientierungstransformation ($_{TRAORI}$) aktiv ist und Werkzeugorientierungen (egal welcher Art) programmiert sind, können unter Einhaltung von vorgegeben Toleranzen komprimiert werden.

Programmierung

Werkzeugorientierung

Falls eine Orientierungstransformation ($_{TRAORI}$) aktiv ist, kann bei 5-Achs Maschinen die Werkzeugorientierung folgendermaßen (kinematikunabhängig) programmiert werden:

- Programmierung des Richtungsvektors über:

A3=<...> B3=<...> C3=<...>

- Programmierung der Eulerwinkel bzw. RPY-Winkel über:

A2=<...> B2=<...> C2=<...>

Drehung des Werkzeugs

Bei **6-Achs** Maschinen kann zusätzlich zur Werkzeugorientierung noch die Drehung des Werkzeugs programmiert werden.

Die Programmierung des Drehwinkels erfolgt mit:

THETA=<...>

Siehe "Drehungen der Werkzeugorientierung (Seite 359)".

Hinweis

NC-Sätze, in denen zusätzlich eine Drehung programmiert ist, sind nur dann komprimierbar, falls sich der Drehwinkel **linear** ändert. D. h. für den Drehwinkel darf kein Polynom mit $PO[THI]=(\dots)$ programmiert sein.

Allgemeine Form eines komprimierbaren NC-Satzes

Die allgemeine Form eines komprimierbaren NC-Satzes kann daher wie folgt aussehen:

N... X=<...> Y=<...> Z=<...> A3=<...> B3=<...> C3=<...> THETA=<...> F=<...>

bzw.

N... X=<...> Y=<...> Z=<...> A2=<...> B2=<...> C2=<...> THETA=<...> F=<...>

Hinweis

Die Positionswerte können direkt (z. B. X90) oder indirekt über Parameterzuweisungen (z. B. X=R1*(R2+R3)) angegeben werden.

Programmierung der Werkzeugorientierung durch Rundachspositionen

Die Werkzeugorientierung kann auch durch Rundachspositionen angegeben sein, z. B. in der Form:

N... X=<...> Y=<...> Z=<...> A=<...> B=<...> C=<...> THETA=<...> F=<...>

In diesem Fall wird die Komprimierung auf zwei unterschiedliche Arten durchgeführt, abhängig davon ob eine Großkreisinterpolation durchgeführt wird oder nicht. Wenn keine Großkreisinterpolation stattfindet, dann wird die komprimierte Orientierungsänderung durch axiale Polynome für die Rundachsen in üblicher Weise dargestellt.

Konturgenauigkeit

Abhängig vom eingestellten Kompressionsmodus (MD20482 \$MC_COMPRESSOR_MODE) werden für die Geometrieachsen und Orientierungsachsen bei der Komprimierung entweder die projizierten achsspezifischen Toleranzen (MD33100 \$MA_COMPRESS_POS_TOL) oder die folgenden über Settingdaten einstellbaren kanalspezifischen Toleranzen wirksam:

SD42475 \$SC_COMPRESS_CONTUR_TOL (Maximale Konturabweichung)

SD42476 \$SC_COMPRESS_ORI_TOL (Maximale Winkelabweichung für die Werkzeugorientierung)

SD42477 \$SC_COMPRESS_ORI_ROT_TOL (Maximale Winkelabweichung für den Drehwinkel des Werkzeugs) (nur bei 6-Achs Maschinen verfügbar)

Literatur:

Funktionshandbuch Grundfunktionen; 3- bis 5-Achs-Transformation (F2),
Kapitel: "Komprimierung der Orientierung"

Aktivierung / Deaktivierung

Kompressor-Funktionen werden eingeschaltet durch die modalen G-Codes COMPON, COMPCURV bzw. COMPCAD.

Beendet wird die Kompressor-Funktion mit COMPOF.

Siehe " NC-Satz-Kompression (COMPON, COMPCURV, COMPCAD) (Seite 262) ".

Hinweis

Die Orientierungsbewegung wird nur komprimiert bei aktiver Großkreisinterpolation (d. h. die Änderung der Werkzeugorientierung erfolgt in der Ebene, die von Start- und Endorientierung aufgespannt wird).

Eine Großkreisinterpolation wird unter den folgenden Bedingungen durchgeführt:

- MD21104 \$MC_ORI_IPO_WITH_G_CODE = 0,
ORIWKS ist aktiv und
Orientierung ist mittels Vektoren programmiert (mit A3, B3, C3 bzw. A2, B2, C2).
- MD21104 \$MC_ORI_IPO_WITH_G_CODE = 1 und
ORIVECT bzw. ORIPLANE ist aktiv.

Die Werkzeugorientierung kann entweder als Richtungsvektor oder mit Rundachspositionen programmiert sein. Ist einer der G-Codes ORICONxx oder ORICURVE aktiv oder sind Polynome für die Orientierungswinkel ($PO[PHI]$ und $PO[PSI]$) programmiert, wird keine Großkreisinterpolation durchgeführt.

Beispiel

Im nachfolgenden Programmbeispiel wird ein Kreis, der durch einen Polygonzug angenähert ist, komprimiert. Die Werkzeugorientierung bewegt sich dabei synchron dazu auf einem Kegelmantel. Obwohl die aufeinanderfolgenden programmierten Orientierungsänderungen unstetig verlaufen, generiert die Kompressor-Funktion einen glatten Verlauf der Orientierung.

Programmierung	Kommentar
DEF INT ANZAHL=60	
DEF REAL RADIUS=20	
DEF INT COUNTER	
DEF REAL WINKEL	
N10 G1 X0 Y0 F5000 G64	
\$SC_COMPRESS_CONTUR_TOL=0.05	; Maximale Abweichung der Kontur = 0.05 mm
\$SC_COMPRESS_ORI_TOL=5	; Maximale Abweichung der Orientierung = 5 Grad
TRAORI	
COMPCURV	
	; Es wird ein Kreis gefahren, der aus Polygonen gebildet wird. Die Orientierung bewegt sich dabei auf einem Kegel um die Z-Achse mit einem Öffnungswinkel von 45 Grad.
N100 X0 Y0 A3=0 B3=-1 C3=1	
N110 FOR COUNTER=0 TO ANZAHL	
N120 WINKEL=360*COUNTER/ANZAHL	
N130 X=RADIUS*cos(WINKEL) Y=RADIUS*sin(WINKEL)	
A3=sin(WINKEL) B3=-cos(WINKEL) C3=1	
N140 ENDFOR	

6.7 Glättung des Orientierungsverlaufs (ORISON, ORISOF)

Funktion

Mit der Funktion "Glättung des Orientierungsverlaufs (ORISON)" können Schwankungen der Orientierung über mehrere Sätze hinweg geglättet werden. Dadurch wird ein glatter Verlauf sowohl der Orientierung als auch der Kontur erzielt.

Voraussetzung

Die Funktion "Glättung des Orientierungsverlaufs (ORISON)" ist nur in Systemen mit 5/6-Achs-Transformation verfügbar.

Syntax

```
ORISON  
...  
ORISOF
```

Bedeutung

ORISON:	Glättung des Orientierungsverlaufs EIN
	Wirksamkeit: modal
ORISOF:	Glättung des Orientierungsverlaufs AUS
	Wirksamkeit: modal

Settingdaten

Die Glättung des Orientierungsverlaufs erfolgt unter Einhaltung:

- einer vorgegebenen maximalen Toleranz (maximale Winkelabweichung der Werkzeugorientierung in Grad)

und

- eines vorgegebenen maximalen Bahnwegs.

Diese Vorgaben werden über Settingdaten definiert:

- SD42678 \$SC_ORISON_TOL (Toleranz für die Glättung des Orientierungsverlaufs)
- SD42680 O\$SC_ORISON_DIST (Bahnweg für die Glättung des Orientierungsverlaufs)

Beispiel

Programmcode	Kommentar
...	
TRAORI()	; Einschalten der Orientierungstransformation.
ORISON	; Einschalten der Orientierungsglättung.
\$SC_ORISON_TOL=1.0	; Toleranz der Orientierungsglättung = 1,0 Grad.
G91	
X10 A3=1 B3=0 C3=1	
X10 A3=-1 B3=0 C3=1	
X10 A3=1 B3=0 C3=1	
X10 A3=-1 B3=0 C3=1	
X10 A3=1 B3=0 C3=1	
X10 A3=-1 B3=0 C3=1	
X10 A3=1 B3=0 C3=1	
X10 A3=-1 B3=0 C3=1	
X10 A3=1 B3=0 C3=1	
X10 A3=-1 B3=0 C3=1	
...	
ORISOF	; Ausschalten der Orientierungsglättung.
...	

Die Orientierung wird um 90 Grad in der XZ–Ebene von -45 bis +45 Grad geschwenkt. Durch die Glättung des Orientierungsverlaufs erreicht die Orientierung nicht mehr die maximalen Winkelwerte von -45 bzw. +45 Grad.

Weitere Informationen

Anzahl der Sätze

Die Glättung des Orientierungsverlaufs erfolgt über eine projektierte Anzahl von Sätzen, die im Maschinendatum MD28590 \$MC_MM_ORISON_BLOCKS hinterlegt ist.

Hinweis

Wird die Glättung des Orientierungsverlaufs mit ORISON aktiviert, ohne dass ausreichend Satzspeicher dafür projektiert wurde (MD28590 < 4), dann erfolgt eine Alarmmeldung und die Funktion kann nicht ausgeführt werden.

Maximale Satzweglänge

Der Orientierungsverlauf wird nur in solchen Sätzen geglättet, deren Verfahrensweg kleiner ist als die projektierte maximale Satzweglänge (MD20178 \$MC_ORISON_BLOCK_PATH_LIMIT). Sätze mit längeren Verfahrenswegen unterbrechen die Glättung und werden wie programmiert abgefahren.

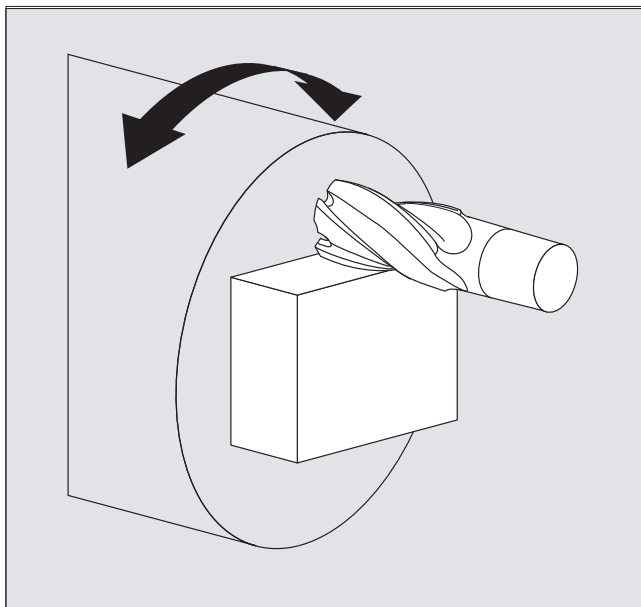
6.8 Kinematische Transformation

6.8.1 Fräsbearbeitung an Drehteilen (TRANSMIT)

Funktion

Die Funktion TRANSMIT ermöglicht folgende Leistungen:

- Stirnseitige Bearbeitung an Drehteilen in der Drehaufspannung (Bohrungen, Konturen).
- Für die Programmierung dieser Bearbeitungen kann ein kartesisches Koordinatensystem benutzt werden.
- Die Steuerung transformiert die programmierten Verfahrbewegungen des kartesischen Koordinatensystems auf die Verfahrbewegungen der realen Maschinenachsen (Standardfall):
 - Rundachse
 - Zustellachse senkrecht zur Drehachse
 - Längsachse parallel zur Drehachse
 - Die Linearachsen stehen senkrecht aufeinander.
- Werkzeugmittenversatz relativ zur Drehmitte ist zulässig.
- Die Geschwindigkeitsführung berücksichtigt die für die Drehbewegungen definierten Begrenzungen.



TRANSMIT Transformationstypen

Für TRANSMIT-Bearbeitungen gibt es zwei einstellbare Ausprägungen:

- TRANSMIT im Standardfall mit (TRAFO_TYPE_n = 256)
- TRANSMIT mit zusätzlicher Y-Linearachse (TRAFO_TYPE_n = 257)

Der erweiterte Transformationstyp 257 kann dazu verwendet werden, um z. B. Aufspannkorrekturen eines Werkzeugs mit realer Y-Achse zu kompensieren.

Syntax

TRANSMIT oder TRANSMIT(n)

TRAFOOF

Rundachse

Die Rundachse kann nicht programmiert werden, da sie von einer Geometrie-Achse belegt wird und somit als Kanalachse nicht direkt programmierbar ist.

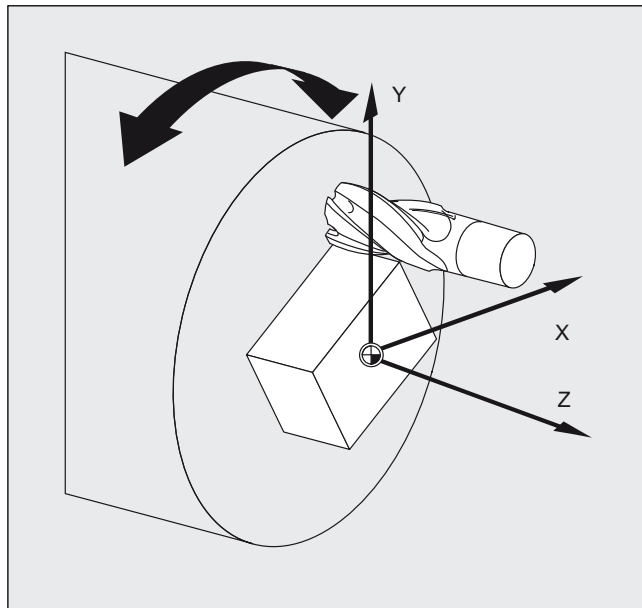
Bedeutung

TRANSMIT:	Aktiviert die erste vereinbarte TRANSMIT-Funktion. Diese Funktion wird auch als Polar-Transformation bezeichnet.
TRANSMIT(n):	Aktiviert die n. vereinbarte TRANSMIT-Funktion; n darf maximal 2 sein (TRANSMIT(1) entspricht TRANSMIT).
TRAFOOF:	Schaltet eine aktive Transformation aus
OFFN:	Offset Kontur-normal: Abstand der stirnseitigen Bearbeitung von der programmierten Bezugskontur

Hinweis

Eine aktive Transformation TRANSMIT wird ebenfalls ausgeschaltet, wenn im jeweiligen Kanal eine der übrigen Transformationen aktiviert wird (z. B. TRACYL, TRAANG, TRAORI).

Beispiel



Programmcode	Kommentar
N10 T1 D1 G54 G17 G90 F5000 G94	; Werkzeuganwahl
N20 G0 X20 Z10 SPOS=45	; Anfahren der Ausgangsstellung
N30 TRANSMIT	; TRANSMIT-Funktion aktivieren
N40 ROT RPL=-45	; Frame einstellen
N50 ATRANS X-2 Y10	
N60 G1 X10 Y-10 G41 OFFN=1OFFN	; Vierkant schrappen; Aufmaß 1 mm
N70 X-10	
N80 Y10	
N90 X10	
N100 Y-10	
N110 G0 Z20 G40 OFFN=0	; Werkzeugwechsel
N120 T2 D1 X15 Y-15	
N130 Z10 G41	
N140 G1 X10 Y-10	; Vierkant schlichten
N150 X-10	
N160 Y10	
N170 X10	
N180 Y-10	
N190 Z20 G40	; Frame abwählen
N200 TRANS	
N210 TRAFOOF	
N220 G0 X20 Z10 SPOS=45	; Anfahren der Ausgangsstellung
N230 M30	

Beschreibung

Pol

Zum Durchfahren des Pols gibt es zwei Möglichkeiten:

- Verfahren der Linearachse allein
- Verfahren in den Pol mit Drehung der Rundachse im Pol und Fahren aus dem Pol

Die Auswahl erfolgt über die MD 24911 und 24951.

TRANSMIT mit zusätzlicher Y-Linearachse (Transformationstyp 257):

Diese Transformationsvariante der Polar-Transformation nützt bei einer Maschine mit einer weiteren Linearachse die Redundanz aus, um eine verbesserte Werkzeugkorrektur durchzuführen. Für die zweite Linearachse gilt dann:

- ein kleinerer Arbeitsbereich und
- dass die zweite Linearachse für das Abfahren des Teileprogramms nicht genutzt werden soll.

Für das Teileprogramm und die Zuordnung der entsprechenden Achsen im BKS oder MKS werden bestimmte Maschineneinstellungen vorausgesetzt, siehe

Literatur

/FB2/ Funktionshandbuch Erweiterungsfunktionen; Kinematische Transformationen (M1)

6.8.2 Zylindermanteltransformation (TRACYL)

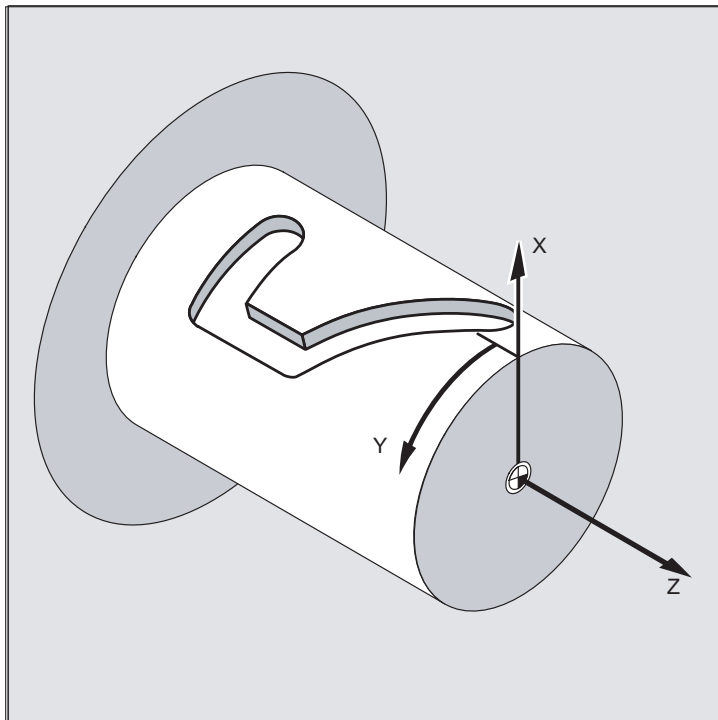
Funktion

Die Zylindermantelkurventransformation TRACYL ermöglicht folgende Leistungen:

Bearbeitung von

- Längsnuten an zylindrischen Körpern,
- Quernuten an zylindrischen Körpern,
- beliebig verlaufende Nuten an zylindrischen Körpern.

Der Verlauf der Nuten wird bezogen auf die abgewickelte, ebene Zylindermantelfläche programmiert.



TRACYL Transformationstypen

Die Zylindermantelkoordinatentransformation gibt es in drei Ausprägungen:

- TRACYL ohne Nutwandkorrektur: (TRAFO_TYPE_n=512)
- TRACYL mit Nutwandkorrektur: (TRAFO_TYPE_n=513)
- TRACYL mit zusätzlicher Linearachse und mit Nutwandkorrektur: (TRAFO_TYPE_n=514)
Die Nutwandkorrektur wird mit TRACYL über den dritten Parameter parametrier.

Bei Zylindermantelkurventransformation mit Nutwandkorrektur sollte die für die Korrektur verwendete Achse auf Null ($y=0$) stehen, damit die Nut mittig zur programmierten Nutmittellinie gefertigt wird.

Achsnutzung

Folgende Achsen können nicht als Positionierachse bzw. Pendelachse verwendet werden:

- die Geometrieachse in Umfangsrichtung der Zylindermantelfläche (Y-Achse)
- die zusätzliche Linearachse bei Nutwandkorrektur (Z-Achse)

Syntax

TRACYL(d) oder TRACYL(d, n) oder

für Transformationstyp 514

TRACYL(d, n, Nutwandkorrektur)

TRAFOOF

Rundachse

Die Rundachse kann nicht programmiert werden, da sie von einer Geometrie-Achse belegt wird und somit als Kanalachse nicht direkt programmierbar ist.

Bedeutung

TRACYL (d)	Aktiviert die erste in den Kanalmaschinen Daten vereinbarte TRACYL-Funktion. d Parameter für den Arbeitsdurchmesser.
TRACYL (d, n)	Aktiviert die n. in den Kanalmaschinen Daten vereinbarte TRACYL-Funktion. n darf maximal 2 sein, TRACYL(d,1) entspricht TRACYL(d).
D	Wert für den Arbeitsdurchmesser. Der Arbeitsdurchmesser ist der doppelte Abstand zwischen Werkzeugspitze und Drehmitte. Dieser Durchmesser muss immer angegeben werden und größer als 1 sein.
n	Optionaler 2. Parameter für den TRACYL-Datensatz 1 (vorangewählt) oder 2.
Nutwandkorrektur	Optionaler 3. Parameter dessen Wert für TRACYL aus den Mode von Maschinen Daten vorangewählt wird. Wertebereich: 0: Transformationstyp 514 ohne Nutwandkorrektur wie bisher 1: Transformationstyp 514 mit Nutwandkorrektur
TRAFOOF	Transformation aus (BKS und MKS sind wieder identisch).
OFFN	Offset Kontur-normal: Abstand der Nutwand von der programmierten Bezugskontur

Hinweis

Eine aktive Transformation TRACYL wird ebenfalls ausgeschaltet, wenn im jeweiligen Kanal eine der übrigen Transformationen aktiviert wird (z. B. TRANSMIT, TRAANG, TRAORI).

Beispiel: Definition des Werkzeugs

Folgendes Beispiel ist geeignet, die Parametrierung der Zylindertransformation TRACYL zu testen:

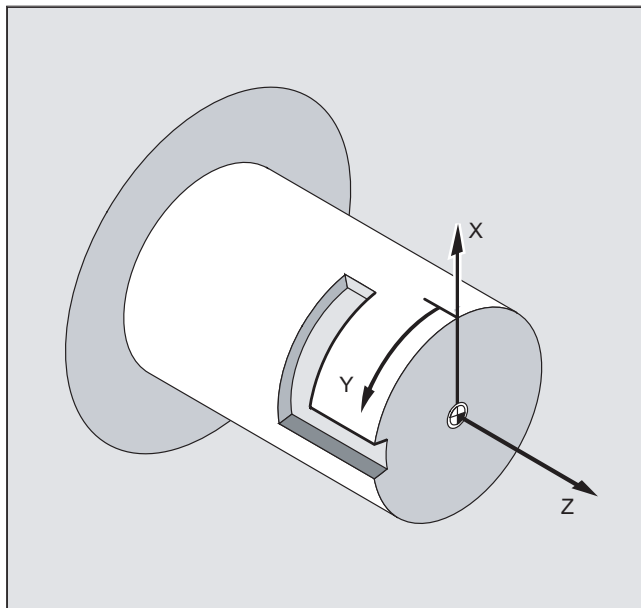
Programmcode	Kommentar	
Werkzeugparameter	Bedeutung	Bemerkung
Nummer (DP)		
\$TC_DP1 [1,1]=120	Werkzeugtyp	Fräser
\$TC_DP2 [1,1]=0	Schneidenlage	nur für Drehwerkzeuge

Programmcode	Kommentar	
Geometrie	Längenkorrektur	
\$TC_DP3 [1,1]=8.	Längenkorrekturvektor	Verrechnung nach Typ
\$TC_DP4 [1,1]=9.		und Ebene
\$TC_DP5 [1,1]=7.		

Programmcode	Kommentar	
Geometrie	Radius	
\$TC_DP6 [1,1]=6.	Radius	Werkzeugradius
\$TC_DP7 [1,1]=0	Nutbreite b für Nutsäge, Verrundungsradius für Fräswerkzeuge	
\$TC_DP8 [1,1]=0	Überstand k	nur für Nutsäge
\$TC_DP9 [1,1]=0		
\$TC_DP10 [1,1]=0		
\$TC_DP11 [1,1]=0	Winkel für kegelige Fräswerkzeuge	

Programmcode	Kommentar	
Verschleiß	Längen- und Radiuskorrektur	
\$TC_DP12 [1,1]=0	Die restlichen Parameter bis \$TC_DP24=0	Basismaß/Adapter

Beispiel: Fertigen einer hakenförmigen Nut



Zylindermanteltransformation einschalten:

Programmcode	Kommentar
N10 T1 D1 G54 G90 F5000 G94	; Werkzeuganwahl, Aufspannkomensation
N20 SPOS=0	; Anfahren der Ausgangsstellung
N30 G0 X25 Y0 Z105 CC=200	
N40 TRACYL (40)	; Zylindermantelkurventransformation einschalten
N50 G19	; Ebenenanwahl

Hakenförmige Nut fertigen:

Programmcode	Kommentar
N60 G1 X20	; Werkzeug auf Nutgrund zustellen
N70 OFFN=12	; Nutwandabstand 12 mm relativ zur Nutmitte festlegen
N80 G1 Z100 G42	; Anfahren der rechten Nutwand
N90 G1 Z50	; Nutabschnitt parallel zur Zylinderachse
N100 G1 Y10	; Nutabschnitt parallel zum Umfang
N110 OFFN=4 G42	; Anfahren der linken Nutwand; Nutwandabstand 4 mm relativ zur Nutmitte festlegen
N120 G1 Y70	; Nutabschnitt parallel zum Umfang
N130 G1 Z100	; Nutabschnitt parallel zur Zylinderachse
N140 G1 Z105 G40	; Abfahren von der Nutwand
N150 G1 X25	; Freifahren
N160 TRAFOOF	
N170 G0 X25 Y0 Z105 CC=200	; Anfahren der Ausgangsstellung
N180 M30	

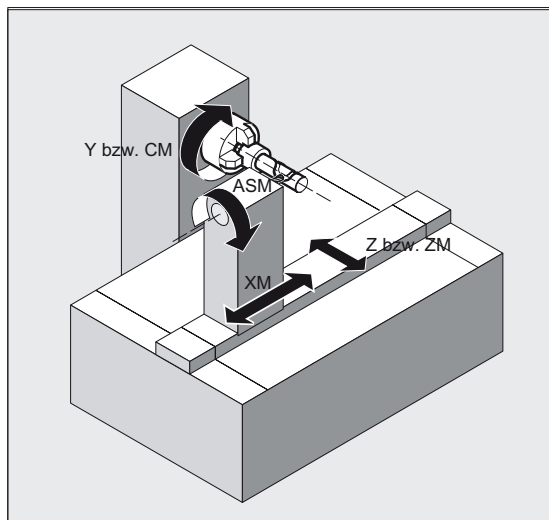
Beschreibung

Ohne Nutwandkorrektur (Transformationstyp 512):

Die Steuerung transformiert die programmierten Verfahrbewegungen des Zylinder-Koordinatensystems auf die Verfahrbewegungen der realen Maschinenachsen:

- Rundachse
- Zustellachse senkrecht zur Drehachse
- Längsachse parallel zur Drehachse

Die Linearachsen stehen senkrecht aufeinander. Die Zustellachse schneidet die Rundachse.

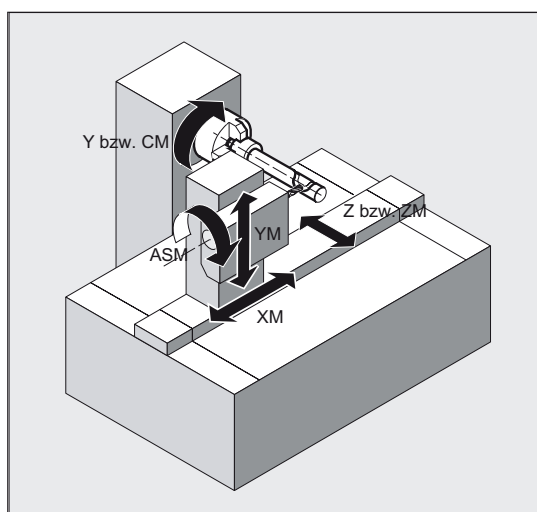


Mit Nutwandkorrektur (Transformationstyp 513):

Kinematik wie oben, aber zusätzlich –Längsachse parallel zur Umfangsrichtung

Die Linearachsen stehen senkrecht aufeinander.

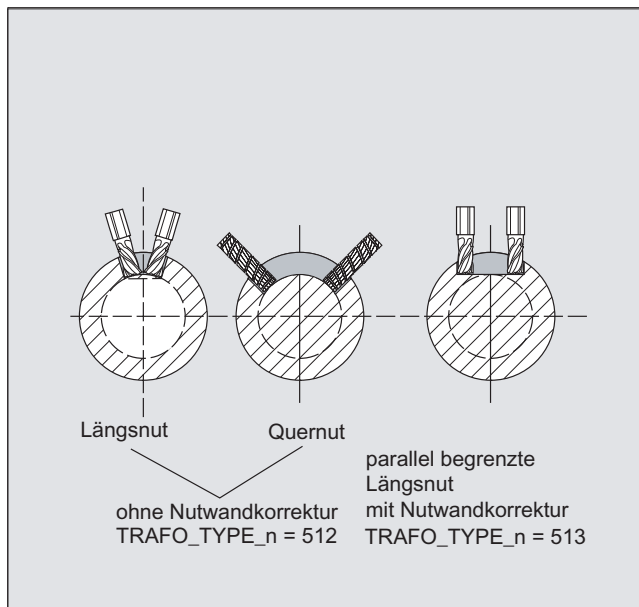
Die Geschwindigkeitsführung berücksichtigt die für die Drehbewegungen definierten Begrenzungen.



Nutquerschnitt

Bei Achskonfiguration 1 sind Nuten längs zur Rundachse nur dann parallel begrenzt, wenn die Nutbreite genau dem Werkzeugradius entspricht.

Nuten parallel zum Umfang (Quernuten) sind an Anfang und Ende nicht parallel.



Mit zusätzlicher Linearachse und mit Nutwandkorrektur (Transformationstyp 514):

Diese Transformationsvariante nützt bei einer Maschine mit einer weiteren Linearachse die Redundanz aus, um eine verbesserte Werkzeugkorrektur durchzuführen. Für die zweite Linearachse gilt dann:

- ein kleinerer Arbeitsbereich und
- dass die zweite Linearachse für das Abfahren des Teileprogramms nicht genutzt werden soll.

Für das Teileprogramm und die Zuordnung der entsprechenden Achsen im BKS oder MKS werden bestimmte Maschinendateneinstellungen vorausgesetzt, siehe

Literatur

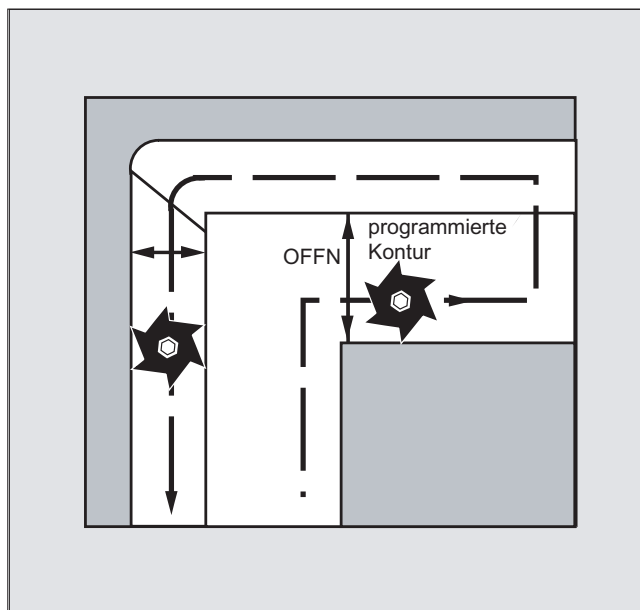
/FB2/ Funktionshandbuch Erweiterungsfunktionen; Kinematische Transformationen (M1)

Offset Kontur-normal OFFN (Transformationstyp 513)

Um mit `TRACYL` Nuten zu fräsen, wird im

- Teileprogramm die Nutmittenlinie,
- über `OFFN` **die halbe Nutbreite** programmiert.

`OFFN` wird erst mit angewählter Werkzeugradiuskorrektur wirksam, um eine Beschädigung der Nutwand zu vermeiden). **Ferner sollte $OFFN \geq \text{Werkzeugradius}$ sein, um eine Beschädigung der gegenüberliegenden Nutwand auszuschließen.**



Ein Teileprogramm zum Fräsen einer Nut besteht in der Regel aus folgenden Schritten:

1. Werkzeug anwählen
2. `TRACYL` anwählen
3. Passende Koordinatenverschiebung (`FRAME`) anwählen
4. Positionieren
5. `OFFN` programmieren
6. `WRK` anwählen
7. Anfahrsatz (Einfahren der `WRK` und Anfahren der Nutwand)
8. Kontur der Nutmittenlinie
9. `WRK` abwählen
10. Abfahrsatz (Ausfahren der `WRK` und Wegfahren von der Nutwand)
11. Positionieren
12. `TRAFOOF`
13. Ursprüngliche Koordinatenverschiebung (`FRAME`) wieder anwählen

Besonderheiten

- WRK-Anwahl:

WRK wird nicht hinsichtlich der Nutwand, sondern relativ zur programmierten Nutmittellinie programmiert. Damit das Werkzeug links von der Nutwand fährt, wird G42 eingegeben (anstatt G41). Sie vermeiden dies, wenn in `OFFN` die Nutbreite mit negativem Vorzeichen eingetragen wird.

- `OFFN` mit `TRACYL` wirkt sich anders aus als ohne `TRACYL`. Da `OFFN` auch ohne `TRACYL` bei aktiver WRK eingerechnet wird, sollte `OFFN` nach `TRAFOOF` wieder zu Null gesetzt werden.
- Eine Änderung von `OFFN` innerhalb des Teileprogramms ist möglich. Damit könnte die Nutmittellinie aus der Mitte verschoben werden (siehe Bild).
- Führungsnuten:

Mit `TRACYL` wird nicht dieselbe Nut bei Führungsnuten erzeugt als wäre diese mit einem Werkzeug gefertigt worden, dessen Durchmesser die Nutbreite aufweist. Es ist prinzipiell nicht möglich, mit einem kleineren zylindrischen Werkzeug dieselbe Nutwandgeometrie zu erzeugen wie mit einem größeren. `TRACYL` minimiert den Fehler. Um Genauigkeitsprobleme zu vermeiden, sollte der Werkzeugradius nur wenig kleiner als die halbe Nutbreite sein.

Hinweis

OFFN und WRK

Bei `TRAFO_TYPE_n = 512` wirkt der Wert unter `OFFN` als Aufmass zur WRK.

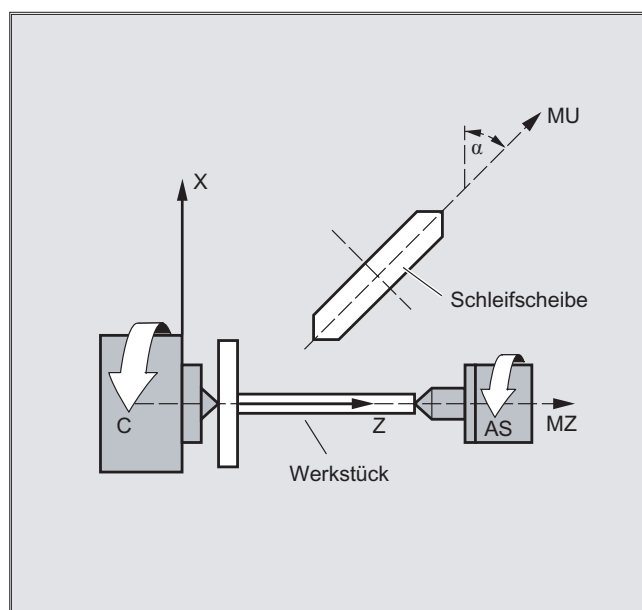
Bei `TRAFO_TYPE_n = 513` wird im `OFFN` die halbe Nutbreite programmiert. Die Kontur wird mit `OFFN-WRK` abgefahren.

6.8.3 Schräge Achse (TRAANG)

Funktion

Die Funktion "Schräge Achse" ist für die Technologie "Schleifen" gedacht und ermöglicht folgende Leistungen:

- Bearbeitung mit schräger Zustellachse
- Für die Programmierung kann ein kartesisches Koordinatensystem verwendet werden.
- Die Steuerung transformiert die programmierten Verfahrbewegungen des kartesischen Koordinatensystems auf die Verfahrbewegungen der realen Maschinenachsen (Standardfall): schräge Zustellachse.



Syntax

```
TRAANG( $\alpha$ ) oder TRAANG( $\alpha$ , <n>)  
...  
TRAFOOF
```

Bedeutung

Element	Beschreibung
TRAANG () oder TRAANG (, n):	Transformation mit der Parametrierung der vorhergehenden Anwahl aktivieren
TRAANG (α):	Aktiviert die erste vereinbarte Transformation "Schräge Achse"
TRAANG (α, <n>):	Aktiviert die n-te vereinbarte Transformation "Schräge Achse". Hinweis: TRAANG (α, 1) entspricht TRAANG (α).
α:	Winkel der schrägstehenden Achse
	Wertebereich: -90 Grad < α < + 90 Grad
<n>:	Anzahl vereinbarte Transformationen
	Wertebereich: 1, 2
TRAFOOF:	Transformation aus

Winkel α weglassen oder Null

Wird der Winkel α weggelassen (z. B. TRAANG (), TRAANG (, n)), wird die Transformation mit der Parametrierung der vorhergehenden Anwahl aktiviert. Bei der ersten Anwahl gilt die Vorbelegung gemäß den Maschinendaten.

Ein Winkel α = 0 (z. B. TRAANG (0), TRAANG (0, n)) ist eine gültige Parametrierung und entspricht nicht mehr dem Weglassen des Parameters bei älteren Versionen.

Beispiel

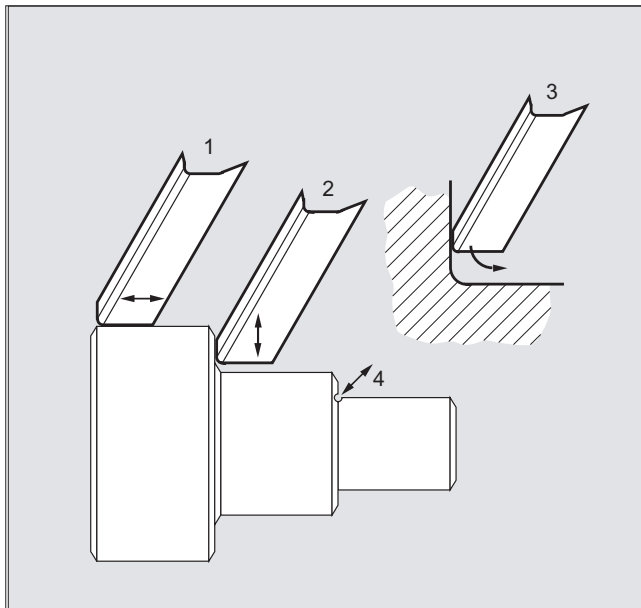
Programmcode	Kommentar
N10 G0 G90 Z0 MU=10 G54 F5000 G18 G64 T1 D1	; Werkzeuganwahl, Aufspannkompensation, Ebenenanwahl.
N20 TRAANG(45)	; Transformation Schräge Achse einschalten.
N30 G0 Z10 X5	; Anfahren der Ausgangsstellung.
N40 WAITP(Z)	; Achsen zum Pendeln freigeben.
N50 OSP[Z]=10 OSP2[Z]=5 OST1[Z]=-2 OST2[Z]=-2 FA[Z]=5000	; Pendeln, bis Maß erreicht.
N60 OS[Z]=1	
N70 POS[X]=4.5 FA[X]=50	
N80 OS[Z]=0	
N90 WAITP(Z)	; Pendelachsen als Positionierachsen freigeben.
N100 TRAFOOF	; Transformation ausschalten.
N110 G0 Z10 MU=10	; Freifahren.
N120 M30	

Weitere Informationen

Anwendungen

Folgende Bearbeitungen sind möglich:

1. Längsschleifen
2. Planschleifen
3. Schleifen einer bestimmten Kontur
4. Schrägeinstechschleifen



Maschinenhersteller

Folgende Einstellungen werden über Maschinendatum festgelegt:

- der Winkel zwischen einer Maschinenachse und der schrägen Achse,
- die Lage des Werkzeugnullpunktes bezogen auf den Ursprung des bei der Funktion "Schräge Achse" vereinbarten Koordinatensystems,
- die Geschwindigkeitsreserve, die auf der parallelen Achse für die Ausgleichsbewegung bereitgehalten wird,
- die Achsbeschleunigungsreserve, die auf der parallelen Achse für die Ausgleichsbewegung bereitgehalten wird.

Achskonfiguration

Um im kartesischen Koordinatensystem programmieren zu können, muss der Steuerung der Zusammenhang zwischen diesem Koordinatensystem und den tatsächlich existierenden Maschinenachsen (MU, MZ) mitgeteilt werden:

- Benennung der Geometrieachsen
- Zuordnung der Geometrieachsen zu Kanalachsen
 - allgemeiner Fall (Schräge Achse nicht aktiv)
 - Schräge Achse aktiv
- Zuordnung der Kanalachsen zu den Maschinenachsnummern
- Kennzeichnung der Spindeln
- Zuweisung von Maschinenachsamen

Das Vorgehen entspricht mit Ausnahme von "Schräge Achse aktiv" dem Vorgehen bei der normalen Achskonfiguration.

6.8.4 Schräge Achse programmieren (G5, G7)

Funktion

Im JOG-Betrieb kann die Schleifscheibe wahlweise kartesisch oder in Richtung der Schrägen Achse bewegt (Anzeige bleibt kartesisch) werden. Es bewegt sich nur die reale U-Achse, die Anzeige der Z-Achse wird aktualisiert.

REPOS-Verschiebungen müssen im Jog-Betrieb kartesisch zurückgefahren werden.

Das Überfahren der kartesischen Arbeitsfeldbegrenzung wird im JOG-Betrieb bei aktivem "PTP-Fahren" überwacht, die entsprechende Achse wird vorher gebremst. Ist "PTP-Fahren" nicht aktiv, kann die Achse exakt bis zur Arbeitsfeldbegrenzung gefahren werden.

Literatur

Funktionshandbuch Erweiterungsfunktionen; Kinematische Transformation (M1)

Syntax

G7

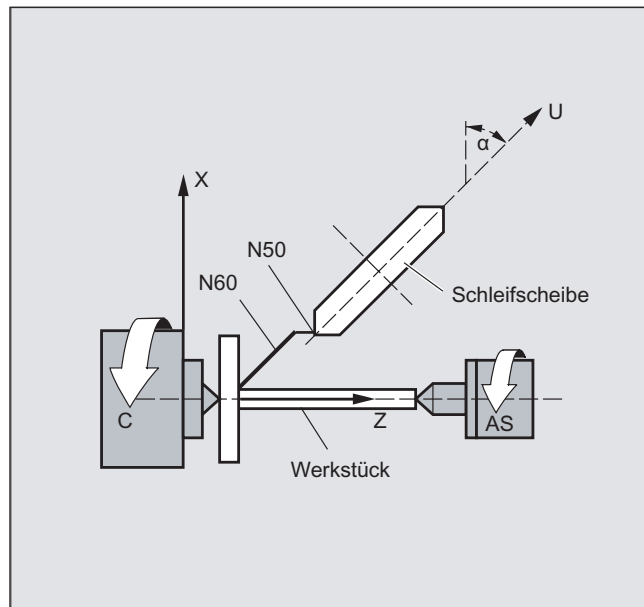
G5

Die Befehle $G7/G5$ dienen der Erleichterung der Programmierung der Schrägen Achse. Dabei können Positionen im kartesischen Koordinatensystem programmiert und angezeigt werden. Die Werkzeugkorrektur und Nullpunktverschiebung werden kartesisch eingerechnet. Nach der Programmierung des Winkels für die Schräge Achse im NC-Programm kann die Startposition angefahren werden ($G7$) und danach das Schrägeinstecken ($G5$) vollzogen werden.

Bedeutung

- G7 Startposition anfahren
- G5 Aktiviert Schrägeinstechen

Beispiel



Programmierung	Kommentar
N.. G18	; Winkel für die Schräge Achse programmieren
N50 G7 X70 Z40 F4000	; Startposition anfahren
N60 G5 X70 F100	; Schräg einstechen
N70 ...	;

6.9 Kartesisches PTP-Fahren

Funktion

Mit dieser Funktion kann eine Position in einem kartesischen Koordinatensystem programmiert werden, die Bewegung der Maschine erfolgt aber in Maschinenkoordinaten. Die Funktion kann beispielsweise beim Wechseln der Gelenkstellung angewendet werden, wenn dabei die Bewegung durch eine Singularität führt.

Hinweis

Die Funktion ist nur in Verbindung mit einer aktiven Transformation sinnvoll. Weiterhin ist das "PTP-Fahren" nur in Verbindung mit G0 und G1 zulässig.

Syntax

```
N... TRAORI
N... STAT='B10' TU='B100' PTP
N... CP
```

PTP-Fahren bei generischer 5/6-Achs Transformation

Wird bei aktiver generischer 5/6-Achs-Transformation mit PTP ein Punkt-zu-Punkt Fahren im Maschinenkoordinatensystem (`ORIMKS`) aktiviert, kann die Werkzeugorientierung mit Rundachspositionen, mit von der Kinematik unabhängigen Vektoren Euler- bzw. RPY-Winkel oder den Richtungsvektoren programmiert werden:

- Rundachspositionen: `N... G1 X Y Z A B C`
- Euler-Winkel in ZY'X"-Konvention (RPY-Winkel) oder ZX'Z"-Konvention:
`N... ORIRPY` bzw. `N... ORIEULER`
`N... G1 X Y Z A2 B2 C2`
- Richtungsvektoren: `N... G1 X Y Z A3 B3 C3`

Dabei kann sowohl Rundachsinterpolation als auch Vektorinterpolation mit Großkreisinterpolation `ORIVECT` oder Interpolation des Orientierungsvektors auf einer Kegelmantelfläche `ORICONxx` aktiv sein.

Mehrdeutigkeiten der Orientierung mit Vektoren

Bei der Programmierung der Orientierung mit Vektoren gibt es eine Mehrdeutigkeit in den möglichen Rundachspositionen. Die anzufahrenden Rundachspositionen können dabei durch die Programmierung von `STAT = <...>` ausgewählt werden. Wird

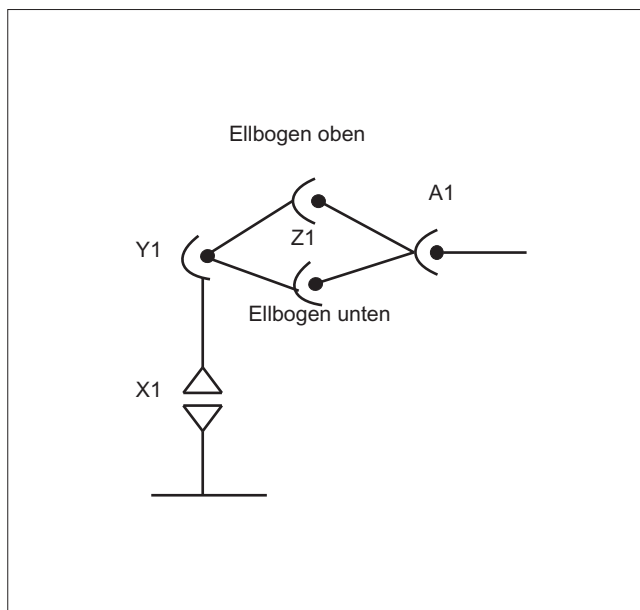
`STAT = 0` programmiert wird (dies entspricht der Standardeinstellung), werden die Positionen, die den kürzesten Abstand zu den Startpositionen haben angefahren. Wenn

`STAT = 1` programmiert wird, werden die Positionen, die den längeren Abstand zu den Startpositionen haben angefahren.

Bedeutung

Befehl	Bedeutung
PTP:	Point to Point (Punkt zu Punkt Bewegung) Wirksamkeit: modal
CP:	Continuous Path (kartesische Bahnbewegung) Wirksamkeit: modal
STAT=:	Stellung der Gelenke. Der Wert ist abhängig von der Transformation. Wirksamkeit: modal Ein STAT-Wert wirkt nur bei Vektorinterpolation.
TU=:	Achswinkel Wertebereich: ± 360 Grad Wirksamkeit: satzweise Ein TU-Wert wirkt bei Vektor- und Rundachsinterpolation.

Beispiel



Programmcode	Kommentar
N10 G0 X0 Y-30 Z60 A-30 F10000	; Ausgangsstellung → Ellbogen oben
N20 TRAORI(1)	; Transformation ein
N30 X1000 Y0 Z400 A0	
N40 X1000 Z500 A0 STAT='B10'	; Umorientierung ohne Transformation
TU='B100' PTP	; → Ellbogen unten
N50 X1200 Z400 CP	; Transformation wieder aktiv
N60 X1000 Z500 A20	
N70 M30	

Beispiel PTP-Fahren bei generischer 5-Achs Transformation

Annahme: Es liegt eine rechtwinklige CA-Kinematik zu Grunde.

Programmcode	Kommentar
TRAORI	; Transformation CA-Kinematik ein
PTP	; PTP-Fahren einschalten
N10 A3=0 B3=0 C3=1	; Rundachspositionen C = 0 A = 0
N20 A3=1 B3=0 C3=1	; Rundachspositionen C = 90 A = 45
N30 A3=1 B3=0 C3=0	; Rundachspositionen C = 90 A = 90
N40 A3=1 B3=0 C3=1 STAT=1	; Rundachspositionen C = 270 A = -45

Eindeutige Anfahrstellung der Rundachsposition auswählen:

Im Satz N40 fahren dabei die Rundachsen durch die Programmierung von $STAT = 1$ den längeren Weg von ihrem Startpunkt (C=90, A=90) zum Endpunkt (C=270, A=-45), anstatt wie es bei $STAT = 0$ der Fall wäre den kürzesten Weg zum Endpunkt (C=90, A=45).

Beschreibung

Die Umschaltung zwischen dem kartesischen Verfahren und dem Verfahren der Maschinenachsen erfolgt durch die Befehle `PTP` und `CP`.

PTP-Fahren bei generischer 5/6-Achs Transformation

Beim PTP-Fahren bleibt im Gegensatz zur 5/6-Achs Transformation die TCP allgemein nicht ortsfest, falls sich nur die Orientierung ändert. Es werden die transformierten Endpositionen aller Transformationsachsen (3 Linearachsen und bis zu 3 Rundachsen) linear angefahren, ohne dass dabei die Transformation im eigentlichen Sinn noch aktiv ist.

Das PTP-Fahren wird durch Programmierung des modalen G-Codes `CP` ausgeschaltet.

Die unterschiedlichen Transformationen sind enthalten in der Druckschrift: /FB3/ Funktionshandbuch Sonderfunktionen; Transformationspaket Handling (TE4).

Programmierung der Stellung (STAT=)

Eine Maschinenstellung ist allein durch die Positionsangabe mit kartesischen Koordinaten und der Orientierung des Werkzeugs nicht eindeutig bestimmt. Je nachdem, um welche Kinematik es sich handelt, existieren bis zu 8 unterschiedliche bzw. unterscheidende Gelenkstellungen. Diese sind damit transformationsspezifisch. Um eine kartesische Position eindeutig in die Achswinkel umrechnen zu können, muss die Stellung der Gelenke mit dem Befehl `STAT=` angegeben werden. Der Befehl "STAT" enthält als Binärwert für jede der möglichen Stellungen ein Bit.

Die Stellungsbits, welche bei "STAT" zu programmieren sind, siehe: /FB2/ Funktionshandbuch Erweiterungsfunktionen; Kinematische Transformation (M1), Kapitel "Kartesisches PTP-Fahren".

Programmierung der Achswinkel (TU=)

Um Achswinkel $< \pm 360$ Grad eindeutig anfahren zu können, muss diese Information mit dem Befehl "TU=" programmiert werden.

Die Achsen verfahren auf kürzestem Weg:

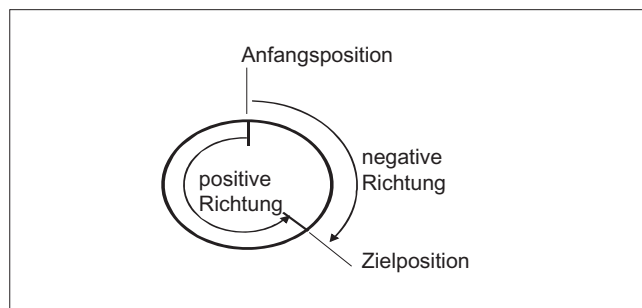
- wenn bei einer Position kein TU programmiert wird,
- bei Achsen, welche einen Verfahrbereich $> \pm 360$ Grad besitzen.

Beispiel:

Die im Bild angegebene Zielposition kann in negativer oder in positiver Richtung angefahren werden. Unter der Adresse A1 wird die Richtung programmiert.

A1=225°, TU=Bit 0, → positive Richtung

A1=-135°, TU=Bit 1, → negative Richtung



Beispiel Auswertung von TU für generische 5/6-Achs Transformation und Zielpositionen

Die Variable TU enthält für jede Achse, die in die Transformation eingeht, ein Bit, das die Verfahrrichtung anzeigt. Die Zuordnung der TU-Bits entspricht der Kanalachssicht der Rundachsen. Die TU-Information wird nur bei den bis zu 3 möglichen Rundachsen, die in die Transformation eingehen, ausgewertet:

Bit0: Achse 1, TU-Bit = 0 : 0 Grad \leq Rundachswinkel $<$ 360 Grad

Bit1: Achse 2, TU-Bit = 1: -360 Grad $<$ Rundachswinkel $<$ 0 Grad

Die Startposition einer Rundachse ist C = 0, durch die Programmierung von C = 270 fährt die Rundachse auf folgende Zielpositionen:

C = 270: TU-Bit 0, positive Drehrichtung

C = -90: TU-Bit 1, negative Drehrichtung

Weiteres Verhalten

Betriebsartenwechsel

Die Funktion "Kartesisches PTP-Fahren" ist nur in den Betriebsarten AUTO und MDA sinnvoll. Beim Wechsel der Betriebsart nach JOG bleibt die aktuelle Einstellung erhalten.

Wenn der G-Code `PTP` eingestellt ist, werden die Achsen im MKS verfahren. Wenn der G-Code `CP` eingestellt ist, werden die Achsen im WKS verfahren.

Power On/RESET

Nach Power On oder nach RESET ist die Einstellung abhängig vom Maschinendatum `$MC_GCODE_REST_VALUES[48]`. Standardmäßig ist die Verfahart "`CP`" eingestellt.

REPOS

War während des Unterbrechungssatzes die Funktion "Kartesisches PTP-Fahren" eingestellt, wird auch mit `PTP` rückpositioniert.

Überlagerte Bewegungen

DRF-Verschiebung oder externe Nullpunktverschiebung sind beim kartesischen PTP-Fahren nur eingeschränkt möglich. Beim Wechsel von einer PTP- nach einer CP-Bewegung dürfen keine Überlagerungen im BKS vorhanden sind.

Überschleifen zwischen CP- und PTP-Bewegungen

Zwischen den Sätzen ist mit `G641` ein programmierbares Übergangverschleifen möglich.

Die Größe des Verschleifbereiches ist der Bahnweg in mm oder Inch, ab dem bzw. zu dem der Satzübergang verschliffen wird. Die Größe ist wie folgt anzugeben:

- für G0-Sätze mit `ADISPOS`
- für alle anderen Wegbefehle mit `ADIS`

Die Bahnwegberechnung entspricht der Berücksichtigung der F-Adressen bei Nicht-G0-Sätzen. Der Vorschub wird auf die in `FGROUP(...)` angegebenen Achsen eingehalten.

Vorschubberechnung

Für CP-Sätze werden die kartesischen Achsen des Basiskoordinatensystems zur Berechnung verwendet.

Für PTP-Sätze werden die entsprechenden Achsen des Maschinenkoordinatensystems zur Berechnung verwendet.

6.9.1 PTP bei TRANSMIT

Funktion

Mit PTP bei TRANSMIT können G0- und G1-Sätze zeitoptimiert angefahren werden. Anstatt die Achsen des Basiskoordinatensystems linear zu verfahren (CP), werden die Maschinenachsen linear verfahren (PTP). Dadurch wirkt sich der Maschinenachsverlauf in Polnähe so aus, dass der Satzendpunkt erheblich schneller erreicht werden kann.

Das Teileprogramm wird weiterhin im kartesischen Werkstückkoordinatensystem geschrieben und alle Koordinatenverschiebungen, Drehungen und Frameprogrammierungen bleiben gültig. Die Simulation auf HMI, wird ebenfalls im kartesischen Werkstückkoordinatensystem angezeigt.

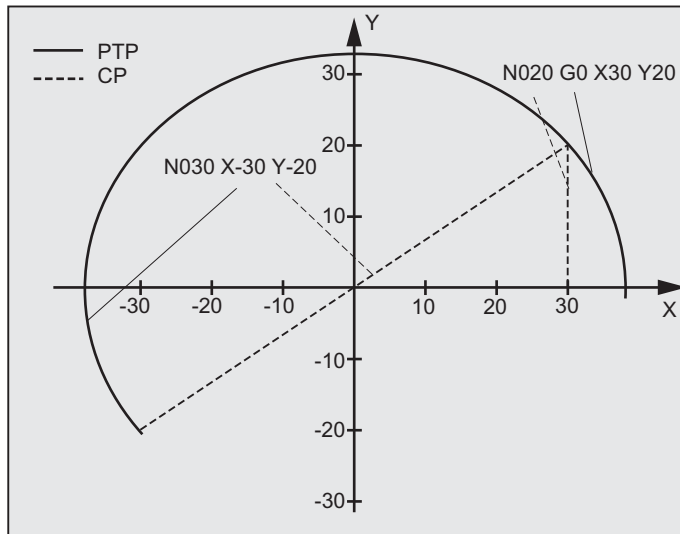
Syntax

```
N... TRANSMIT  
N... PTPG0  
N... G0 ...  
...  
N... G1 ...
```

Bedeutung

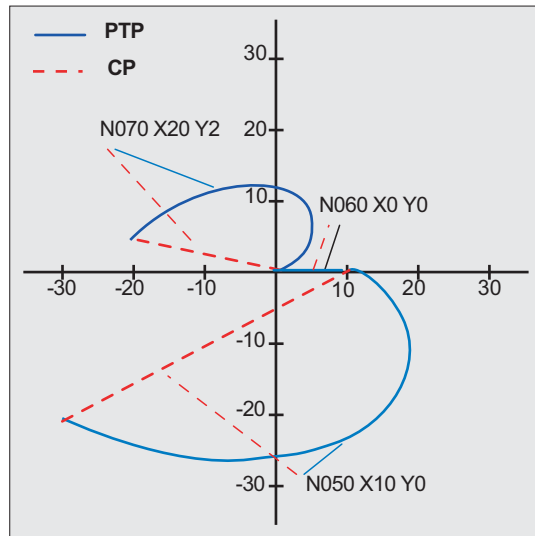
TRANSMIT	Aktiviert die erste vereinbarte TRANSMIT-Funktion (siehe Kapitel "Fräsbearbeitungen an Drehteilen: TRANSMIT")
PTPG0	Point to Point G0 (Punkt zu Punkt Bewegung automatisch zu jedem G0-Satz und danach wieder CP setzen) Da STAT und TU modal sind, gilt immer der zuletzt programmierte Wert.
PTP	Point to Point (Punkt zu Punkt Bewegung) Für TRANSMIT bedeutet PTP, dass im Kartesischen auf Archimedischen Spiralen entweder um den Pol oder aus dem Pol herausgefahren wird. Die hieraus resultierenden Werkzeugbewegungen verlaufen deutlich anders als bei CP und sind in den jeweiligen Programmierbeispielen dargestellt.
STAT=	Auflösen der Mehrdeutigkeit hinsichtlich des Pols.
TU=	TU ist bei PTP bei TRANSMIT nicht relevant

Beispiel Umfahren des Poles mit PTP und TRANSMIT



Programmcode	Kommentar
N001 G0 X30 Z0 F10000 T1 D1 G90	; Ausgangsstellung Absolutmaß
N002 SPOS=0	
N003 TRANSMIT	; Transformation TRANSMIT
N010 PTPG0	; zu jedem G0-Satz automatisch PTP und danach wieder CP
N020 G0 X30 Y20	
N030 X-30 Y-20	
N120 G1 X30 Y20	
N110 X30 Y0	
M30	

Beispiel Herausfahren aus dem Pol mit PTP und TRANSMIT



Programmierung	Kommentar
N001 G0 X90 Z0 F10000 T1 D1 G90	; Ausgangsstellung
N002 SPOS=0	
N003 TRANSMIT	; Transformation TRANSMIT
N010 PTPG0	; zu jedem G0-Satz automatisch PTP und danach wieder CP
N020 G0 X90 Y60	
N030 X-90 Y-60	
N040 X-30 Y-20	
N050 X10 Y0	
N060 X0 Y0	
N070 X-20 Y2	
N170 G1 X0 Y0	
N160 X10 Y0	
N150 X-30 Y-20	
M30	

Beschreibung

PTP und PTPG0

PTPG0 wird bei allen Transformationen berücksichtigt, die PTP abarbeiten können. In allen anderen Fällen ist PTPG0 nicht relevant.

G0-Sätze werden im CP-Mode abgefahren.

Die Anwahl von PTP bzw. PTPG0 erfolgt im Teileprogramm oder durch die Abwahl von CP im Maschinendatum \$MC_GCODE_RESET_VALUES[48].

VORSICHT

Randbedingungen

Hinsichtlich der Werkzeugbewegungen und Kollision gelten mehrere Randbedingungen und bestimmte Funktionsausschlüsse wie:

Mit PTP darf keine Werkzeugradiuskorrektur (WRK) aktiv sein.

Mit PTPG0 wird bei aktiver Werkzeugradiuskorrektur (WRK) per CP gefahren.

Mit PTP ist Weiches An- und Abfahren (WAB) nicht möglich.

Mit PTPG0 wird bei Weichem An- und Abfahren (WAB) per CP gefahren.

Mit PTP sind Abspanzyklen (CONTPRON, CONTDCON) nicht möglich.

Mit PTPG0 wird in Abspanzyklen (CONTPRON, CONTDCON) per CP gefahren.

Fase (CHF, CHR) und Rundung (RND, RNDM) werden ignoriert.

Kompressor ist nicht verträglich mit PTP und wird automatisch in PTP-Sätzen abgewählt.

Eine Achsüberlagerung in der Interpolation darf sich während des PTP-Abschnittes nicht ändern.

Bei G643 wird automatisch nach Überschleifen mit axialer Genauigkeit G642 umgeschaltet.

Bei aktivem PTP können Achsen der Transformation nicht gleichzeitig Positionierachsen sein.

Literatur:

/FB2/ Funktionshandbuch Erweiterungsfunktionen; Kinematische Transformation (M1), Kapitel "Kartesisches PTP-Fahren"

PTP bei TRACON:

PTP kann auch mit TRACON genutzt werden, wenn die erste verkettete Transformation PTP unterstützt.

Bedeutung von STAT= und TU= bei TRANSMIT

Soll die Rundachse um 180 Grad drehen, bzw. die Kontur bei CP durch den Pol führen, können Rundachsen abhängig vom Maschinendatum \$MC_TRANSMIT_POLE_SIDE_FIX_1/2 [48] um +/- 180 Grad gedreht und im oder gegen den Uhrzeigersinn verfahren werden. Ebenso kann eingestellt werden, ob durch den Pol gefahren wird, oder um den Pol gedreht wird.

6.10 Randbedingungen bei der Anwahl einer Transformation

Funktion

Die Anwahl von Transformationen ist über Teileprogramm bzw. MDA möglich. Dabei ist zu beachten:

- Ein Bewegungszwischensatz wird nicht eingefügt (Fasen/Radien).
- Eine Spline-Satzfolge muss abgeschlossen sein; wenn nicht, erscheint eine Meldung.
- Werkzeugfeinkorrektur muss abgewählt sein (FTOCOF); wenn nicht, erscheint eine Meldung.
- Werkzeugradiuskorrektur muss abgewählt sein (G40); wenn nicht, erscheint eine Meldung.
- Eine aktivierte Werkzeuglängenkorrektur wird von der Steuerung in die Transformation übernommen.
- Der vor der Transformation wirksame aktuelle Frame wird von der Steuerung abgewählt.
- Eine aktive Arbeitsfeldbegrenzung wird für die von der Transformation betroffenen Achsen von der Steuerung abgewählt (entspricht WALIMOF).
- Schutzbereichsüberwachung wird abgewählt.
- Bahnsteuerbetrieb und Überschleifen werden unterbrochen.
- Alle in dem Maschinendatum angegebenen Achsen müssen satzbezogen synchronisiert sein.
- Getauschte Achsen werden zurückgetauscht; wenn nicht, erscheint eine Meldung.
- Bei abhängigen Achsen wird eine Meldung ausgegeben.

Werkzeugwechsel

Ein Werkzeugwechsel ist nur bei abgewählter Werkzeugradiuskorrektur zulässig.

Ein Wechsel der Werkzeuglängenkorrektur und eine An-/Abwahl der Werkzeugradiuskorrektur darf nicht im selben Satz programmiert sein.

Framewechsel

Alle Anweisungen, die sich nur auf das Basis-Koordinatensystem beziehen, sind erlaubt (FRAME, Werkzeugradiuskorrektur). Ein Framewechsel bei G91 (Kettenmaß) wird aber – anders als bei inaktiver Transformation – nicht gesondert behandelt. Das zu fahrende Inkrement wird im Werkstück-Koordinatensystem des neuen Frames ausgewertet – unabhängig davon, welches Frame im Vorgängersatz wirkte.

Ausschlüsse

Von der Transformation betroffene Achsen können nicht verwendet werden:

- als Preset-Achse (Alarm),
- für das Fixpunktfahren (Alarm),
- zum Referieren (Alarm).

6.11 Transformation abwählen (TRAFOOF)

Funktion

Mit dem Befehl `TRAFOOF` werden alle aktiven Transformationen und Frames ausgeschaltet.

Hinweis

Danach benötigte Frames müssen durch erneute Programmierung aktiv geschaltet werden.

Dabei ist zu beachten:

Für die Abwahl der Transformation gelten dieselben Randbedingungen wie für die Anwahl (siehe Kapitel "Randbedingungen bei der Anwahl einer Transformation").

Syntax

`TRAFOOF`

Bedeutung

`TRAFOOF` Befehl zum Ausschalten aller aktiven Transformationen/Frames

6.12 Verkettete Transformationen (TRACON, TRAFOOF)

Funktion

Jeweils **zwei** Transformationen können hintereinander geschaltet (verkettet) werden, so dass die Bewegungsanteile für die Achsen aus der ersten Transformation Eingangsdaten für die verkettete zweite Transformation sind. Die Bewegungsanteile aus der zweiten Transformation wirken auf die Maschinenachsen.

Die Kette darf **zwei** Transformationen umfassen.

Hinweis

Ein Werkzeug wird immer der ersten Transformation einer Kette zugeordnet. Die nachfolgende Transformation verhält sich dann so, als ob die aktive Werkzeuglänge Null wäre. Es sind nur die über Maschinendaten eingestellten Basislängen eines Werkzeuges (`_BASE_TOOL_`) für die erste Transformation der Kette wirksam.

Maschinenhersteller

Beachten Sie die Hinweise des Maschinenherstellers zu ggf. durch Maschinendaten vordefinierten Transformationen.

Transformationen und verkettete Transformationen sind Optionen. Über die Verfügbarkeit bestimmter Transformationen in der Kette in bestimmten Steuerungen gibt jeweils der aktuelle Katalog Auskunft.

Anwendungen

- Schleifen von Konturen, die als Mantellinie einer Zylinderabwicklung programmiert wurden (TRACYL) mit einer schräg stehenden Schleifscheibe z. B. Werkzeugschleifen.
- Feinbearbeitung einer mit TRANSMIT erzeugten nicht runden Kontur mit schräg stehender Schleifscheibe.

Syntax

TRACON(*trf, par*)

TRAFOOF

Eine verkettete Transformation wird eingeschaltet.

Bedeutung

TRACON	Die verkettete Transformation wird eingeschaltet. Eine zuvor aktivierte andere Transformation wird durch TRACON() implizit ausgeschaltet.
TRAFOOF	Die zuletzt eingeschaltete (verkettete) Transformation wird ausgeschaltet.
trf	Nummer der verketteten Transformation: 0 oder 1 für erste/einzige verkettete Transformation. Ist an dieser Stelle nichts programmiert, ist das gleichbedeutend mit der Angabe des Wertes 0 oder 1, d. h. es wird die erste/einzige Transformation aktiviert. 2 für die zweite verkettete Transformation. (Werte ungleich 0 - 2 erzeugen einen Fehleralarm).
par	Ein oder mehrere durch Komma getrennte Parameter für die Transformationen in der Verkettung, die Parameter erwarten, z. B. Winkel der schrägen Achse. Bei nicht gesetzten Parametern werden die Voreinstellungen oder die zuletzt benutzten Parameter wirksam. Durch Kommasetzung muss dafür gesorgt werden, dass die angegebenen Parameter in der Reihenfolge ausgewertet werden, in der sie erwartet werden, wenn für vorher stehende Parameter Voreinstellungen wirken sollen. Insbesondere muss bei Angabe mindestens eines Parameters vor diesem ein Komma stehen, auch wenn die Angabe von trf nicht notwendig ist, also beispielsweise TRACON(, 3.7).

Voraussetzung

Die **zweite** Transformation muss "**Schräge Achse**" (TRAANG) sein. Als erste Transformation sind möglich:

- Orientierungstransformationen (TRAORI), einschließlich Kardanischer Fräskopf
- TRANSMIT
- TRACYL
- TRAANG

Für die Benutzung des Einschaltbefehles für eine verkettete Transformation ist Voraussetzung, dass die einzelnen zu verkettenden Transformationen und die zu aktivierende verkettete Transformation durch Maschinendaten definiert sind.

Die in den Einzelbeschreibungen für die Transformationen angegebenen Randbedingungen und Sonderfälle sind auch bei der Benutzung innerhalb einer Verkettung zu beachten.

Informationen zur Projektierung der Maschinendaten der Transformationen finden Sie im:

/FB2/ Funktionshandbuch Erweiterungsfunktionen; Kinematische Transformationen (M1) und
/FB3/ Funktionshandbuch Sonderfunktionen; 3- bis 5-Achs-Transformationen (F2).

Werkzeugkorrekturen

7.1 Korrekturspeicher

Funktion

Aufbau des Korrekturspeichers

Jedes Datenfeld ist mit einer T- und D-Nummer aufrufbar (außer "Flache D-Nr.") und enthält neben den geometrischen Angaben für das Werkzeug noch weitere Einträge, z. B. den Werkzeugtyp.

Flache D-Nummern-Struktur

Die "flache D-Nummern-Struktur" wird verwendet, wenn die Werkzeugverwaltung außerhalb des NCK erfolgt. In diesem Fall werden die D-Nummern mit den zugehörigen Werkzeugkorrektursätzen ohne Zuordnung zu Werkzeugen angelegt.

Im Teileprogramm kann weiterhin T programmiert werden. Dieses T hat aber keinen Bezug zur programmierten D-Nummer.

Anwender-Schneidendaten

Über Maschinendatum können Anwender-Schneidendaten konfiguriert werden. Bitte beachten Sie die Angaben des Maschinenherstellers.

Werkzeugparameter

Hinweis

Einzelne Werte im Korrekturspeicher

Die einzelnen Werte des Korrekturspeichers P1 bis P25 sind über Systemvariable vom Programm les- und schreibbar. Alle übrigen Parameter sind reserviert.

Die Werkzeugparameter \$TC_DP6 bis \$TC_DP8, \$TC_DP10 und \$TC_DP11 sowie \$TC_DP15 bis \$TC_DP17, \$TC_DP19 und \$TC_DP20 haben abhängig vom Werkzeugtyp eine andere Bedeutung.

Werkzeugparameter Nummer (DP)	Bedeutung der Systemvariablen	Bemerkung
\$TC_DP1	Werkzeugtyp	Übersicht siehe Liste
\$TC_DP2	Schneidenlage	nur für Drehwerkzeuge
Geometrie	Längenkorrektur	
\$TC_DP3	Länge 1	Verrechnung nach
\$TC_DP4	Länge 2	Typ und Ebene
\$TC_DP5	Länge 3	
Geometrie	Radius	
\$TC_DP6 ¹⁾ \$TC_DP6 ²⁾	Radius 1 / Länge 1 Durchmesser d	Fräs-/Dreh-/Schleifwerkz. Nutsäge
\$TC_DP7 ¹⁾ \$TC_DP7 ²⁾	Länge 2 / Eckenradius kegelige Fräser Nutbreite b Eckenradius	Fräswerkzeuge Nutsäge
\$TC_DP8 ¹⁾ \$TC_DP8 ²⁾	Verrundungsradius 1 für Fräswerkzeuge Überstand k	Fräswerkzeuge Nutsäge
\$TC_DP9 ^{1) 3)}	Verrundungsradius 2	reserviert
\$TC_DP10 ¹⁾	Winkel 1 Stirnseite des Werkzeugs	kegelige Fräswerkzeuge
\$TC_DP11 ¹⁾	Winkel 2 Werkzeug-Längsachse	kegelige Fräswerkzeuge
Verschleiß	Längen- und Radiuskorrektur	
\$TC_DP12	Länge 1	
\$TC_DP13	Länge 2	
\$TC_DP14	Länge 3	
\$TC_DP15 ¹⁾ \$TC_DP15 ²⁾	Radius 1 / Länge 1 Durchmesser d	Fräs-/Dreh-/Schleifwerkz. Nutsäge
\$TC_DP16 ¹⁾ \$TC_DP16 ³⁾	Länge 2 / Eckenradius kegelige Fräser Nutbreite b Eckenradius	Fräswerkzeuge Nutsäge
\$TC_DP17 ¹⁾ \$TC_DP17 ²⁾	Verrundungsradius 1 für Fräswerkzeuge Überstand k	Fräsen / 3D Stirnfräsen Nutsäge
\$TC_DP18 ^{1) 3)}	Verrundungsradius 2	reserviert
\$TC_DP19 ¹⁾	Winkel 1 Stirnseite des Werkzeugs	kegelige Fräswerkzeuge
\$TC_DP20 ¹⁾	Winkel 2 Werkzeug-Längsachse	kegelige Fräswerkzeuge
Basismaß/Adapter	Längenkorrekturen	
\$TC_DP21	Länge 1	
\$TC_DP22	Länge 2	
\$TC_DP23	Länge 3	
Technologie		
\$TC_DP24	Freiwinkel	nur für Drehwerkzeuge
\$TC_DP25		reserviert

- 1) Gilt auch bei Fräswerkzeugen für das 3D-Stirnfräsen
- 2) Bei Nutsäge Werkzeugtyp
- 3) reserviert (wird von SINUMERIK 840D sl nicht benutzt)

Anmerkungen

Für die geometrischen Größen (z. B. Länge 1 oder Radius) bestehen mehrere Eintragskomponenten. Diese werden zu einer resultierenden Größe additiv verrechnet (z. B. Gesamtlänge 1, Gesamtradius), die dann zur Wirkung kommt.

Nicht benötigte Korrekturen sind mit dem Wert Null zu belegen.

Werkzeugparameter \$TC-DP1 bis \$TC-DP23 mit Konturwerkzeugen**Hinweis**

Die Werkzeugparameter, die in der Tabelle nicht aufgeführt sind wie z. B. \$TC_DP7, werden nicht ausgewertet, d. h. ihr Inhalt ist bedeutungslos.

Werkzeugparameter Nummer (DP)	Bedeutung	Schneiden Dn		Bemerkung
\$TC_DP1	Werkzeugtyp			400 bis 599
\$TC_DP2	Schneidenlage			
Geometrie	Längenkorrektur			
\$TC_DP3	Länge 1			
\$TC_DP4	Länge 2			
\$TC_DP5	Länge 3			
Geometrie	Radius			
\$TC_DP6	Radius			
Geometrie	Grenzwinkel			
\$TC_DP10	minimaler Grenzwinkel			
\$TC_DP11	maximaler Grenzwinkel			
Verschleiß	Längen- und Radiuskorrektur			
\$TC_DP12	Verschleiß Länge 1			
\$TC_DP13	Verschleiß Länge 2			
\$TC_DP14	Verschleiß Länge 3			
\$TC_DP15	Verschleiß Radius			
Verschleiß	Grenzwinkel			
\$TC_DP19	Verschleiß min. Grenzwinkel			
\$TC_DP20	Verschleiß max. Grenzwinkel			
Basismaß/Adapter	Längenkorrekturen			
\$TC_DP21	Länge 1			
\$TC_DP22	Länge 2			
\$TC_DP23	Länge 3			

Grundwert und Verschleißwert

Die resultierenden Größen ergeben sich jeweils als Summe aus Grundwert und Verschleißwert (z. B. \$TC_DP6 + \$TC_DP15 für den Radius). Zur Werkzeuglänge der ersten Schneide wird außerdem noch das Basismaß (\$TC_DP21 – \$TC_DP23) addiert. Zusätzlich wirken auf diese Werkzeuglänge alle anderen Größen, die auch bei einem herkömmlichen Werkzeug die effektive Werkzeuglänge beeinflussen können (Adapter, orientierbarer Werkzeugträger, Settingdaten).

Grenzwinkel 1 und 2

Die Grenzwinkel 1 bzw. 2 beziehen sich jeweils auf den Vektor vom Schneidenmittelpunkt zum Schneidenbezugspunkt und werden im Gegenuhrzeigersinn gezählt.

7.2 Additive Korrekturen

7.2.1 Additive Korrekturen anwählen (DL)

Funktion

Additive Korrekturen können als in der Bearbeitung programmierbare Prozesskorrekturen betrachtet werden. Sie beziehen sich auf die geometrischen Daten einer Schneide und sind somit Bestandteil der Werkzeugschneidendaten.

Die Daten einer additiven Korrektur werden über eine DL-Nummer angesprochen (DL: Location dependent; Korrekturen bezüglich des jeweiligen Einsatzorts) und über die Bedienoberfläche eingegeben.

Anwendung

Durch additive Korrekturen können einsatzortbedingte Maßfehler ausgeglichen werden.

Syntax

DL=<Nummer>

Bedeutung

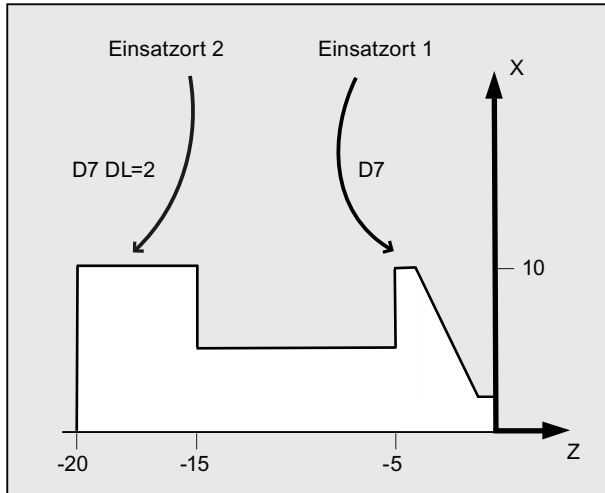
Element	Beschreibung
DL:	Befehl zur Aktivierung einer additiven Korrektur
<Nummer>:	Über den Parameter <Nummer> wird der zu aktivierende additive Werkzeugkorrekturdatensatz angegeben

Hinweis

Die Festlegung von Anzahl und Aktivierung der additiven Korrekturen erfolgt über Maschinendaten (→ Angaben des Maschinenherstellers beachten!).

Beispiel

Die gleiche Schneide wird für 2 Lagersitze verwendet:



Programmcode	Kommentar
N110 T7 D7	; Der Revolver wird auf Platz 7 positioniert. D7 und DL=1 werden aktiviert und im nächsten Satz herausgefahren.
N120 G0 X10 Z1	
N130 G1 Z-6	
N140 G0 DL=2 Z-14	; Additiv zu D7 wird DL=2 aktiviert und im nächsten Satz herausgefahren.
N150 G1 Z-21	
N160 G0 X200 Z200	; Werkzeugwechsellpunkt anfahren.
...	

7.2.2 Verschleiß- und Einrichtewerte festlegen (\$TC_SCPxy[t,d], \$TC_ECPxy[t,d])

Funktion

Verschleiß- und Einrichtewerte können über Systemvariablen gelesen und geschrieben werden. Dabei orientiert sich die Logik an der Logik der entsprechenden Systemvariablen für Werkzeuge und Schneiden.

Systemvariablen

Systemvariable	Bedeutung
\$TC_SCPxy[<t>,<d>]	Verschleißwerte, die über xy dem jeweiligen Geometrieparameter zugeordnet sind, wobei x die Nummer des Verschleißwerts entspricht und y den Bezug zum Geometrieparameter herstellt.
\$TC_ECPxy[<t>,<d>]	Einrichtewerte, die über xy dem jeweiligen Geometrieparameter zugeordnet sind, wobei x die Nummer des Einrichtewerts entspricht und y den Bezug zum Geometrieparameter herstellt.
<t>: T-Nummer des Werkzeugs <d>: D-Nummer der Schneide des Werkzeugs	

Hinweis

Die festgelegten Verschleiß- und Einrichtewerte werden zu den Geometrieparametern und den übrigen Korrekturparametern (D-Nummer) addiert.

Beispiel

Der Verschleißwert der Länge 1 wird für die Schneide <d> des Werkzeugs <t> auf den Wert 1.0 festgelegt.

Parameter: \$TC_DP3 (Länge 1, bei Drehwerkzeugen)

Verschleißwerte: \$TC_SCP13 bis \$TC_SCP63

Einrichtewerte: \$TC_ECP13 bis \$TC_ECP63

\$TC_SCP43 [<t>,<d>] = 1.0

7.2.3 Additive Korrekturen löschen (DELDL)

Funktion

Mit dem Befehl `DELDL` werden additive Korrekturen für die Schneide eines Werkzeugs gelöscht (Freigabe von Speicher). Dabei werden sowohl die festgelegten Verschleißwerte als auch die Einrichtewerte gelöscht.

Syntax

```
DELDL [<t>, <d>]  
DELDL [<t>]  
DELDL  
<Status>=DELDL [<t>, <d>]
```

Bedeutung

<code>DELDL</code>	Befehl zum Löschen additiver Korrekturen
<code><t></code>	T-Nummer des Werkzeugs
<code><d></code>	D-Nummer der Schneide des Werkzeugs
<code>DELDL [<t>, <d>]</code>	Es werden alle additiven Korrekturen der Schneide <code><d></code> des Werkzeugs <code><t></code> gelöscht.
<code>DELDL [<t>]</code>	Es werden alle additiven Korrekturen aller Schneiden des Werkzeugs <code><t></code> gelöscht.
<code>DELDL</code>	Es werden alle additiven Korrekturen aller Schneiden aller Werkzeuge der TO-Einheit gelöscht (für den Kanal, in dem der Befehl programmiert wird).
<code><Status></code>	Lösch-Status
Wert:	Bedeutung:
0	Das Löschen wurde erfolgreich durchgeführt.
-	Das Löschen wurde nicht durchgeführt (wenn die Parametrierung genau eine Schneide bezeichnet), oder das Löschen erfolgte nicht vollständig (wenn die Parametrierung mehrere Schneiden bezeichnet).

Hinweis

Verschleiß- und Einrichtewerte aktiver Werkzeuge können nicht gelöscht werden (verhält sich analog zum Löschverhalten von `D` bzw. Werkzeugdaten).

7.3 Werkzeugkorrektur - Sonderbehandlung

Funktion

Mit den Settingdaten SD42900 bis SD42960 lässt sich die Bewertung der Vorzeichen für Werkzeuglänge und Verschleiß steuern.

Das gilt ebenfalls für das Verhalten der Verschleißkomponenten beim Spiegeln von Geometrieachsen oder beim Wechsel der Bearbeitungsebene und auch zur Temperaturkompensation in Werkzeugrichtung.

Verschleißwerte

Wenn im Folgenden auf Verschleißwerte Bezug genommen wird, ist darunter jeweils die Summe aus den eigentlichen Verschleißwerten (\$TC_DP12 bis \$TC_DP20) und den Summenkorrekturen mit den Verschleißwerten (\$SCPX3 bis \$SCPX11) und Einrichtewerten (\$ECPX3 bis \$ECPX11) zu verstehen.

Näheres zu den Summenkorrekturen siehe:

Literatur:

Funktionshandbuch Werkzeugverwaltung

Settingdaten

Settingdatum	Bedeutung
SD42900 \$SC_MIRROR_TOOL_LENGTH	Spiegeln von Werkzeuglängenkomponenten und Komponenten des Basismaßes.
SD42910 \$SC_MIRROR_TOOL_WEAR	Spiegeln von Verschleißwerten der Werkzeuglängenkomponenten.
SD42920 \$SC_WEAR_SIGN_CUTPOS	Vorzeichenbewertung der Verschleißkomponenten in Abhängigkeit von der Schneidenlage.
SD42930 \$SC_WEAR_SIGN	Invertiert die Vorzeichen der Verschleißmaße.
SD42935 \$SC_WEAR_TRANSFORM	Transformation der Verschleißwerte.
SD42940 \$SC_TOOL_LENGTH_CONST	Zuordnung der Werkzeuglängenkomponenten zu den Geometrieachsen.
SD42950 \$SC_TOOL_LENGTH_TYPE	Zuordnung der Werkzeuglängenkomponenten unabhängig vom Werkzeugtyp.
SD42960 \$SC_TOOL_TEMP_COMP	Temperaturkompensationswert in Werkzeugrichtung. Ist auch bei vorhandener Werkzeugorientierung wirksam.

Literatur

Funktionshandbuch Grundfunktionen; Werkzeugkorrektur (W1)

Weitere Informationen

Wirksamwerden der veränderten Settingdaten

Die Neubewertung von Werkzeugkomponenten bei einer Änderung der beschriebenen Settingdaten wird erst wirksam, wenn das nächste Mal eine Werkzeugschneide angewählt wird. Ist ein Werkzeug bereits aktiv und die Bewertung der Daten dieses Werkzeugs soll verändert wirksam werden, muss dieses Werkzeug erneut angewählt werden.

Entsprechendes gilt für den Fall, dass sich die resultierende Werkzeuglänge ändert, weil der Spiegelungszustand einer Achse geändert wurde. Das Werkzeug muss nach dem Spiegelbefehl erneut angewählt werden, damit die geänderten Werkzeuglängenkomponenten wirksam werden.

Orientierbare Werkzeugträger und neue Settingdaten

Die Settingdaten SD42900 bis SD42940 wirken nicht auf die Komponenten eines eventuell aktiven orientierbaren Werkzeugträgers. Ein Werkzeug geht jedoch immer mit seiner gesamten resultierenden Länge (Werkzeuglänge + Verschleiß + Basismaß) in die Berechnung mit einem orientierbaren Werkzeugträger ein. Bei der Berechnung der resultierenden Gesamtlänge werden alle Änderungen berücksichtigt, die durch die Settingdaten verursacht wurden; d.h. Vektoren des orientierbaren Werkzeugträgers sind unabhängig von der Bearbeitungsebene.

Hinweis

Häufig wird es beim Einsatz orientierbarer Werkzeugträger sinnvoll sein, alle Werkzeuge für ein nicht gespiegeltes Grundsystem zu definieren, auch diejenigen, die nur bei Spiegelbearbeitung verwendet werden. Bei Bearbeitung mit gespiegelten Achsen wird dann der Werkzeugträger so gedreht, dass die tatsächliche Lage des Werkzeugs richtig beschrieben wird. Alle Werkzeuglängenkomponenten wirken dann automatisch in der richtigen Richtung, so dass sich eine Steuerung der Bewertung einzelner Komponenten über Settingdaten abhängig vom Spiegelungszustand einzelner Achsen erübrigt.

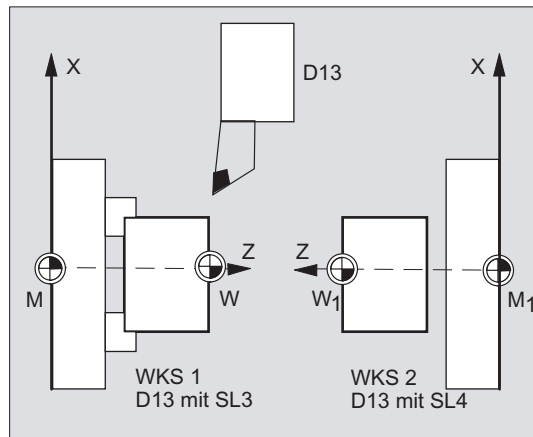
Weitere Anwendungsmöglichkeiten

Die Verwendung der Funktionalität orientierbarer Werkzeugträger kann auch dann sinnvoll sein, wenn an der Maschine physikalisch keine Möglichkeit vorgesehen ist, Werkzeuge zu drehen, Werkzeuge aber mit verschiedenen Orientierungen fest installiert sind. Die Werkzeugvermaßung kann dann einheitlich in einer Grundorientierung vorgenommen werden, und die für die Bearbeitung relevanten Maße ergeben sich durch Drehungen eines virtuellen Werkzeugträgers.

7.3.1 Werkzeuglängen spiegeln

Funktion

Mit gesetzten Settingdaten SD42900 \$SC_MIRROR_TOOL_LENGTH und SD42910 \$SC_MIRROR_TOOL_WEAR ungleich Null können Sie Werkzeuglängenkomponenten und Komponenten der Basismaße mit Verschleißwerten deren zugehörigen Achsen spiegeln.



SD42900 \$SC_MIRROR_TOOL_LENGTH

Settingdatum **ungleich** Null:

Es werden die Werkzeuglängenkomponenten (\$TC_DP3, \$TC_DP4 und \$TC_DP5) und die Komponenten der Basismaße (\$TC_DP21, \$TC_DP22 und \$TC_DP23), deren zugehörige Achsen gespiegelt sind, ebenfalls gespiegelt - durch Vorzeicheninvertierung.

Die Verschleißwerte werden **nicht** mitgespiegelt. Sollen diese ebenfalls gespiegelt werden, muss das Settingdatum SD42910 \$SC_MIRROR_TOOL_WEAR gesetzt sein.

SD42910 \$SC_MIRROR_TOOL_WEAR

Settingdatum **ungleich** Null:

Es werden die Verschleißwerte der Werkzeuglängenkomponenten, deren zugehörige Achsen gespiegelt sind, ebenfalls gespiegelt - durch Vorzeicheninvertierung.

7.3.2 Vorzeichenbewertung Verschleiß

Funktion

Mit gesetzten Settingdaten SD42920 \$SC_WEAR_SIGN_CUTPOS und SD42930 \$SC_WEAR_SIGN ungleich Null können Sie die Vorzeichenbewertung der Verschleißkomponenten invertieren.

SD42920 \$SC_WEAR_SIGN_CUTPOS

Settingdatum **ungleich** Null:

Bei Werkzeugen mit relevanter Schneidenlage (Dreh- und Schleifwerkzeuge, Werkzeugtypen 400) hängt die Vorzeichenbewertung der Verschleißkomponenten in der Bearbeitungsebene von der Schneidenlage ab. Bei Werkzeugtypen ohne relevanter Schneidenlage ist dieses Settingdatum bedeutungslos.

In folgender Tabelle sind die Maße durch ein X gekennzeichnet, deren Vorzeichen über das SD42920 (ungleich 0) invertiert wird:

Schneidenlage	Länge 1	Länge 2
1		
2		X
3	X	X
4	X	
5		
6		
7		X
8	X	
9		

Hinweis

Die Vorzeichenbewertung durch SD42920 und SD42910 sind voneinander unabhängig. Wenn z. B. das Vorzeichen einer Maßangabe durch beide Settingdaten geändert wird, bleibt das resultierende Vorzeichen unverändert.

SD42930 \$SC_WEAR_SIGN

Settingdatum **ungleich** Null:

Das Vorzeichen aller Verschleißmaße wird invertiert. Es wirkt sowohl auf die Werkzeuglänge als auch auf die übrigen Größen wie Werkzeugradius, Verrundungsradius usw.

Wird ein positives Verschleißmaß eingegeben, wird somit das Werkzeug "kürzer" und "dünner", siehe Kapitel "Werkzeugkorrektur, Sonderbehandlung", "Wirksamwerden der veränderten Settingdaten".

7.3.3 Koordinatensystem der aktiven Bearbeitung (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS)

Funktion

Abhängig von der Kinematik der Maschine oder vom Vorhandensein eines orientierbaren Werkzeugträgers werden die in einem dieser Koordinatensysteme gemessenen Verschleißwerte in ein geeignetes Koordinatensystem überführt bzw. transformiert.

Koordinatensysteme der aktiven Bearbeitung

Aus den folgenden Koordinatensystemen können Offsets der Werkzeuglänge hervorgehen, welche die Werkzeuglängenkomponente Verschleiß über den entsprechenden G-Code der Gruppe 56 in ein aktives Werkzeug eingerechnet werden.

- Maschinenkoordinatensystem (MKS)
- Basiskoordinatensystem (BKS)
- Werkstückkoordinatensystem (WKS)
- Werkzeugkoordinatensystem (TCS)
- Werkzeugkoordinatensystem der kinematischen Transformation (KCS)

Syntax

TOWSTD
TOWMCS
TOWWCS
TOWBCS
TOWTCS
TOWKCS

Bedeutung

TOWSTD	Grundstellungswert für Korrekturen in der Werkzeuglänge Verschleißwert
TOWMCS	Korrekturen in der Werkzeuglänge im MKS
TOWWCS	Korrekturen in der Werkzeuglänge im WKS
TOWBCS	Korrekturen in der Werkzeuglänge im BKS
TOWTCS	Korrekturen der Werkzeuglänge am Werkzeugträgerbezugspunkt (orientierbarer Werkzeugträger)
TOWKCS	Korrekturen der Werkzeuglänge des Werkzeugkopfes (kinematischer Transformation)

Weitere Informationen

Unterscheidungsmerkmale

In der folgenden Tabelle sind die wichtigsten Unterscheidungsmerkmale dargestellt:

G-Code	Verschleißwert	Aktiver orientierbarer Werkzeugträger
TOWSTD	Grundstellungswert, Werkzeuglänge	Verschleißwerte unterliegen der Drehung.
TOWMCS	Verschleißwert Im MKS. TOWMCS ist mit TOWSTD identisch, wenn kein orientierbarer WZ-Träger aktiv ist.	Es dreht nur der Vektor der resultierenden Werkzeuglänge ohne Berücksichtigung des Verschleißes.
TOWWCS	Der Verschleißwert wird Im WKS auf das MKS umgerechnet.	Der Werkzeugvektor wird ohne Berücksichtigung des Verschleißes wie bei TOWMCS berechnet.
TOWBCS	Der Verschleißwert wird Im BKS auf das MKS umgerechnet.	Der Werkzeugvektor wird ohne Berücksichtigung des Verschleißes wie bei TOWMCS berechnet.
TOWTCS	Der Verschleißwert wird Im Werkzeugkoordinatensystem auf das MKS umgerechnet.	Der Werkzeugvektor wird ohne Berücksichtigung des Verschleißes wie bei TOWMCS berechnet.

TOWWCS , TOWBCS, TOWTCS: Der Verschleißvektor wird zum Werkzeugvektor addiert.

Lineare Transformation

Die Werkzeuglänge ist im MKS nur sinnvoll definierbar, wenn das MKS aus dem BKS durch eine lineare Transformation hervorgeht.

Nicht lineare Transformation

Ist z. B. mit TRANSMIT eine nicht lineare Transformation aktiv, dann wird bei Angabe des MKS als gewünschtes Koordinatensystem automatisch das BKS verwendet.

Keine kinematische Transformation und kein orientierbarer Werkzeugträger

Ist weder eine kinematische Transformation noch ein orientierbarer Werkzeugträger aktiv, dann fallen bis auf das WKS alle weiteren vier Koordinatensysteme zusammen. Damit unterscheidet sich nur das WKS von den übrigen. Da ausschließlich Werkzeuglängen zu bewerten sind, haben Translationen zwischen den Koordinatensystemen keine Bedeutung.

Literatur:

Weitere Informationen zur Werkzeugkorrektur siehe:
 Funktionshandbuch Grundfunktionen; Werkzeugkorrektur (W1)

Einrechnung der Verschleißwerte

Das Settingdatum **SD42935 \$SC_WEAR_TRANSFORM** legt fest, welche der drei Verschleißkomponenten:

- Verschleiß
- Summenkorrekturen fein
- Summenkorrekturen grob

einer Drehung durch eine Adaptertransformation oder einen orientierbaren Werkzeugträger unterworfen werden soll, wenn einer der folgenden G-Codes aktiv ist:

- **TOWSTD** Grundstellung
für Korrekturen in der Werkzeuglänge
- **TOWMCS** Verschleißwerte
im Maschinenkoordinatensystem (MKS)
- **TOWWCS** Verschleißwerte
im Werkstückkoordinatensystem (WKS)
- **TOWBCS** Verschleißwerte (BKS)
im Basiskoordinatensystem
- **TOWTCS** Verschleißwerte im Werkzeugkoordinatensystem an der Werkzeughalteraufnahme (T Werkzeugträgerbezug)
- **TOWKCS** Verschleißwerte im Koordinatensystem des Werkzeugkopfes bei kinetischer Transformation

Hinweis

Die Bewertung der einzelnen Verschleißkomponenten (Zuordnung zu den Geometrieachsen, Vorzeichenbewertung) wird beeinflusst durch:

- die aktive Ebene
 - die Adaptertransformation
 - folgende Settingdaten:
 - SD42910 \$SC_MIRROR_TOOL_WEAR
 - SD42920 \$SC_WEAR_SIGN_CUTPOS
 - SD42930 \$SC_WEAR_SIGN
 - SD42940 \$SC_TOOL_LENGTH_CONST
 - SD42950 \$SC_TOOL_LENGTH_TYPE
-

7.3.4 Werkzeuglänge und Ebenenwechsel

Funktion

Mit gesetzten Settingdaten SD42940 \$SC_TOOL_LENGTH_CONST ungleich Null können Sie Werkzeuglängenkomponenten wie Länge, Verschleiß und Basismaß zu den Geometrieachsen für Dreh- und Schleifwerkzeuge bei einem Ebenenwechsel zuordnen.

SD42940 \$SC_TOOL_LENGTH_CONST

Settingdatum **ungleich** Null:

Die Zuordnung der Werkzeuglängenkomponenten (Länge, Verschleiß und Basismaß) zu den Geometrieachsen beim Wechsel der Bearbeitungsebene (G17 - G19) wird nicht verändert.

Folgende Tabelle zeigt die Zuordnung der Werkzeuglängenkomponenten zu den Geometrieachsen für Dreh- und Schleifwerkzeuge (WZ-Typ 400 bis 599):

Inhalt	Länge 1	Länge 2	Länge 3
17	Y	X	Z
*)	X	Z	Y
19	Z	Y	X
-17	X	Y	Z
-18	Z	X	Y
-19	Y	Z	X

*) Jeder Wert ungleich 0, der nicht gleich einem der sechs aufgeführten Werte ist, wird wie der Wert 18 bewertet.

Folgende Tabelle zeigt die Zuordnung der Werkzeuglängenkomponenten zu den Geometrieachsen für alle anderen Werkzeuge (WZ-Typ < 400 bzw. > 599):

Bearbeitungsebene	Länge 1	Länge 2	Länge 3
*)	Z	Y	X
18	Y	X	Z
19	X	Z	Y
-17	Z	X	Y
-18	Y	Z	X
-19	X	Y	Z

*) Jeder Wert ungleich 0, der nicht gleich einem der sechs aufgeführten Werte ist, wird wie der Wert 17 bewertet.

Hinweis

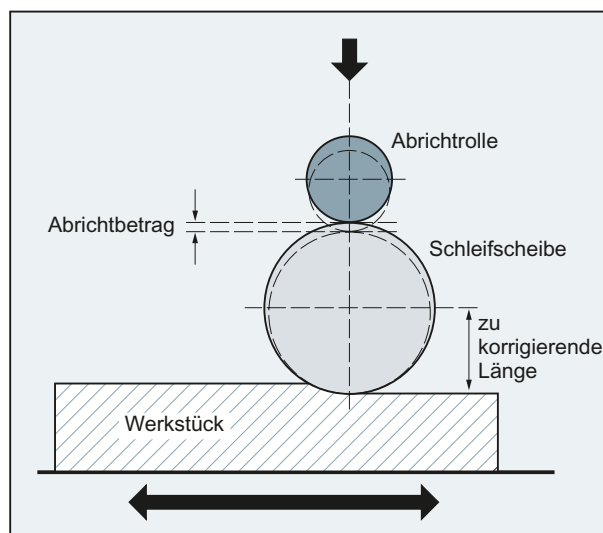
Bei der Darstellung in den Tabellen wird davon ausgegangen, dass die Geometrieachsen bis 3 mit X, Y, Z bezeichnet werden. Für die Zuordnung einer Korrektur zu einer Achse ist nicht der Achsbezeichner, sondern die Achsreihenfolge maßgebend.

7.4 Online-Werkzeugkorrektur (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF)

Funktion

Mit aktiver Funktion "Online-Werkzeugkorrektur" wird bei Schleifwerkzeugen eine Werkzeuglängenkorrektur, die sich aus der Bearbeitung ergibt, sofort eingerechnet.

Ein Anwendungsbeispiel ist das CD-Abrichten, bei dem die Schleifscheibe parallel zur Bearbeitung abgerichtet wird:



Die Werkzeuglängenkorrektur kann aus dem Bearbeitungskanal oder einem parallelen Kanal (Abrichterkanal) verändert werden.

Zum Schreiben der Online-WZK werden je nach gewünschtem Zeitpunkt des Abrichtvorgangs unterschiedliche Funktionen verwendet:

- Schreiben kontinuierlich satzweise (PUTFTOCF)

Mit PUTFTOCF erfolgt der Abrichtvorgang zeitgleich mit der Bearbeitung.

Die Werkzeugkorrektur wird im Bearbeitungskanal kontinuierlich nach einer Polynomfunktion 1., 2. oder 3. Grades geändert, die vorher mit FCTDEF definiert werden muss.

PUTFTOCF wirkt immer satzweise, d. h. im anschließenden Verfahrssatz.

- Schreiben kontinuierlich modal: ID=1 DO FTOC (siehe "Synchronaktionen (Seite 569)")
- Schreiben diskret (PUTFTOC)

Mit PUTFTOC erfolgt der Abrichtvorgang nicht zeitgleich mit der Bearbeitung aus einem parallelen Kanal. Der mit PUTFTOC angegebene Korrekturwert wird im Zielkanal sofort wirksam.

Hinweis

Die Online-WZK kann nur bei Schleifwerkzeugen angewendet werden.

Syntax

Online-WZK im Zielkanal ein-/ausschalten:

```
FTCON
...
FTCOF
```

Online-WZK schreiben:

- Kontinuierlich satzweise:

```
FCTDEF (<Funktion>, <LLimit>, <ULimit>, <a0>, <a1>, <a2>, <a3>)
PUTFTOCF (<Funktion>, <Bezugswert>, <WZ-Parameter>, <Kanal>, <Spindel>)
...
```

- diskret:

```
PUTFTOC (<Korrekturwert>, <WZ-Parameter>, <Kanal>, <Spindel>)
...
```

Bedeutung

FTCON:	<p>Online-WZK einschalten</p> <p>FTCON muss in dem Kanal programmiert werden, in dem die Online-WZK wirksam werden soll.</p>																
FTCOF:	<p>Online-WZK abbrechen</p> <p>Mit FTCOF wird die Korrektur nicht weiter herausgefahren, in den schneidenspezifischen Korrekturdaten ist jedoch der komplette mit PUTFTOC/PUTFTOCF geschriebene Betrag korrigiert.</p> <p>Hinweis: Zum endgültigen Deaktivieren der Online-WZK muss nach FTCOF noch eine An-/Abwahl des Werkzeugs (T...) erfolgen.</p>																
FCTDEF:	<p>Mit FCTDEF wird die Polynom-Funktion für PUTFTOCF definiert.</p> <p>Parameter:</p> <table border="0" style="margin-left: 20px;"> <tr> <td style="padding-right: 20px;"><Funktion>:</td> <td>Nummer der Polynom-Funktion</td> </tr> <tr> <td></td> <td>Typ: INT</td> </tr> <tr> <td style="padding-right: 20px;"><LLimit>:</td> <td>Unterer Grenzwert</td> </tr> <tr> <td></td> <td>Typ: REAL</td> </tr> <tr> <td style="padding-right: 20px;"><ULimit>:</td> <td>Oberer Grenzwert</td> </tr> <tr> <td></td> <td>Typ: REAL</td> </tr> <tr> <td style="padding-right: 20px;"><a0> ... <a3>:</td> <td>Koeffizienten der Polynom-Funktion</td> </tr> <tr> <td></td> <td>Typ: REAL</td> </tr> </table>	<Funktion>:	Nummer der Polynom-Funktion		Typ: INT	<LLimit>:	Unterer Grenzwert		Typ: REAL	<ULimit>:	Oberer Grenzwert		Typ: REAL	<a0> ... <a3>:	Koeffizienten der Polynom-Funktion		Typ: REAL
<Funktion>:	Nummer der Polynom-Funktion																
	Typ: INT																
<LLimit>:	Unterer Grenzwert																
	Typ: REAL																
<ULimit>:	Oberer Grenzwert																
	Typ: REAL																
<a0> ... <a3>:	Koeffizienten der Polynom-Funktion																
	Typ: REAL																

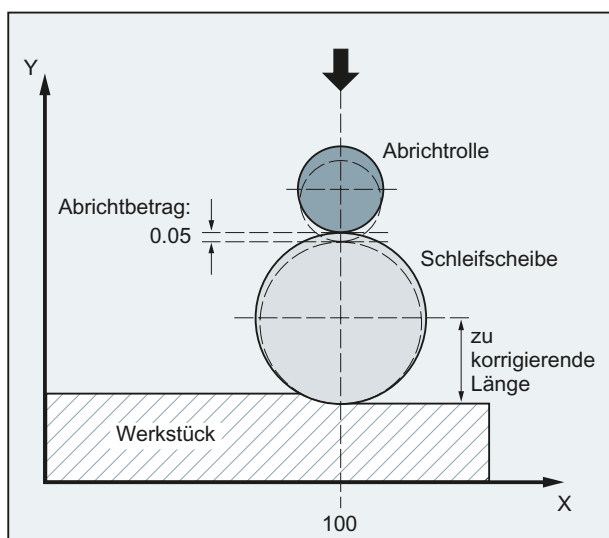
PUTFTOCF:	<p>Funktion "Online-WZK schreiben kontinuierlich satzweise" aufrufen</p> <p>Parameter:</p> <p><Funktion>: Nummer der Polynom-Funktion Typ: INT Hinweis: Muss mit der Angabe bei FCTDEF übereinstimmen.</p> <p><Bezugswert>: Variabler Bezugswert, von dem die Korrektur abgeleitet werden soll (z. B. sich verändernder Istwert). Typ: VAR REAL</p> <p><WZ-Parameter>: Nummer des Verschleißparameters (Länge 1, 2 oder 3), in dem der Korrekturwert addiert werden soll. Typ: INT</p> <p><Kanal>: Nummer des Kanals, in dem die Online-WZK wirksam werden soll. Typ: INT Hinweis: Eine Angabe ist nur erforderlich, wenn die Korrektur nicht im aktiven Kanal wirksam werden soll.</p> <p><Spindel>: Nummer der Spindel, für die die Online-WZK wirksam werden soll. Typ: INT Hinweis: Eine Angabe ist nur erforderlich, wenn statt dem aktiven, im Einsatz befindlichen Werkzeug eine nicht aktive Schleifscheibe korrigiert werden soll.</p>
PUTFTOC:	<p>Funktion "Online-WZK schreiben diskret" aufrufen</p> <p>Parameter:</p> <p><Korrekturwert>: Korrekturwert, der im Verschleißparameter addiert werden soll. Typ: REAL</p> <p><WZ-Parameter>: siehe PUTFTOCF</p> <p><Kanal>: Nummer des Kanals, in dem die Online-WZK wirksam werden soll. Typ: INT</p> <p><Spindel>: siehe PUTFTOCF</p>

Beispiel

Flachschleifmaschine mit:

- Y: Zustellachse für die Schleifscheibe
- V: Zustellachse für die Abrichtrolle
- Bearbeitungskanal: Kanal 1 mit den Achsen X, Z, Y
- Abrichtkanal: Kanal 2 mit Achse V

Nach Beginn der Schleifbewegung soll bei X100 die Schleifscheibe um den Betrag 0,05 abgerichtet werden. Der Abrichtbetrag soll mit "Online-WZK schreiben kontinuierlich" beim Schleifwerkzeug wirksam werden.



Bearbeitungsprogramm in Kanal 1:

Programmcode	Kommentar
...	
N110 G1 G18 F10 G90	; Grundstellung.
N120 T1 D1	; Aktuelles Werkzeug anwählen.
N130 S100 M3 X100	; Spindel ein, Fahren auf Ausgangsposition.
N140 INIT(2, "ABRICHT", "S")	; Anwahl des Abrichtprogramms in Kanal 2.
N150 START(2)	; Starten des Abrichtprogramms in Kanal 2.
N160 X200	; Fahren auf Zielposition.
N170 FTOCON	; Online-Korrektur einschalten.
N... G1 X100	; Weitere Bearbeitung.
N... M30	

Abrichtprogramm in Kanal 2:

Programmcode	Kommentar
...	
N40 FCTDEF(1,-1000,1000,-\$AA_IW[V],1)	; Funktion definieren: Gerade mit Steigung=1.
N50 PUTFTOCF(1,\$AA_IW[V],3,1)	; Online-WZK schreiben kontinuierlich: abgeleitet von der Bewegung der V-Achse wird die Länge 3 der aktuellen Schleifscheibe in Kanal 1 korrigiert.
N60 V-0.05 G1 F0.01 G91	; Zustellbewegung zum Abrichten, nur in diesem Satz ist PUTFTOCF wirksam.
...	
N... M30	

Weitere Informationen**Allgemeines zur Online-WZK**

Beim kontinuierlichen Schreiben (je IPO-Takt) wird nach dem Einschalten der Auswertefunktion jede Änderung additiv im Verschleißspeicher verrechnet (um Sollwertsprünge zu vermeiden).

In jedem Fall gilt: Die Online-WZK kann in jedem Kanal für jede Spindel und die Länge 1, 2 oder 3 der Verschleißparameter wirken.

Die Zuordnung der Längen zu den Geometrieachsen erfolgt anhand der aktuellen Arbeitsebene.

Die Zuordnung der Spindel zum Werkzeug erfolgt durch die Werkzeugdaten bei `GWPSON` bzw. `TMON`, sofern es sich nicht um die aktive Schleifscheibe handelt.

Korrigiert wird immer der Verschleißparameter für die aktuelle Scheibenseite bzw. die linke Scheibenseite bei nicht aktiven Werkzeugen.

Hinweis

Bei identischer Korrektur für mehrere Scheibenseiten ist über Verkettungsvorschrift dafür zu sorgen, dass die Werte automatisch für die zweite Scheibenseite übernommen werden.

Werden für einen Bearbeitungskanal Online-Korrekturen vorgegeben, so dürfen die Verschleißwerte für das aktuelle Werkzeug in diesem Kanal nicht aus dem Bearbeitungsprogramm oder über Bedienung geändert werden.

Die Online-WZK wird auch für die konstante Scheibenumfangsgeschwindigkeit (`SUG`) sowie Werkzeugüberwachung (`TMON`) berücksichtigt.

7.5 Aktivierung 3D-Werkzeugkorrekturen (CUT3DC..., CUT3DF...)

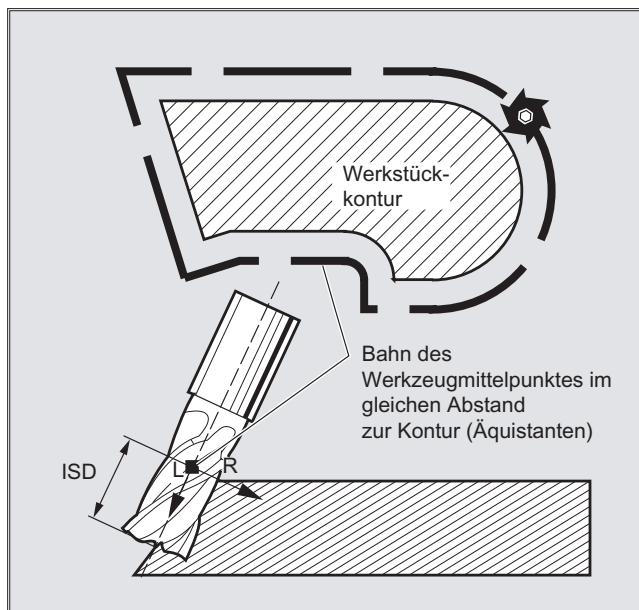
7.5.1 Aktivierung von 3D-Werkzeugkorrekturen (CUT3DC, CUT3DF, CUT3DFS, CUT3DFF, ISD)

Funktion

Bei der Werkzeugradiuskorrektur für zylindrische Werkzeuge wird die veränderliche Werkzeugorientierung berücksichtigt.

Für die Anwahl der 3D-Werkzeugradiuskorrektur gelten die gleichen Programmbefehle wie bei der 2D-Werkzeugradiuskorrektur. Mit $G41/G42$ wird die Korrektur links/rechts in Bewegungsrichtung angegeben. Das Anfahrverhalten ist immer `NORM`. Die 3D-Werkzeugradiuskorrektur wirkt nur bei angewählter 5-Achs-Transformation.

Die 3D-Werkzeugradiuskorrektur wird auch als 5D-Korrektur bezeichnet, da in diesem Fall 5 Freiheitsgrade für die Lage des Werkzeugs im Raum zur Verfügung stehen.



Unterschied zwischen 2 1/2D- und 3D-Werkzeugradiuskorrektur

Bei der 3D-Werkzeugradiuskorrektur ist die Werkzeug-Orientierung veränderlich. Bei der 2 1/2D-Werkzeugradiuskorrektur wird nur mit einem Werkzeug mit konstanter Orientierung gerechnet.

Syntax

```
CUT3DC
CUT3DFS
CUT3DFF
CUT3DF
ISD=<Wert>
```


Bedeutung

CUT3DC	Aktivierung der 3D-Radiuskorrektur für das Umfangsfräsen
CUT3DFS	D-Werkzeugkorrektur für das Stirnfräsen mit konstanter Orientierung. Die Werkzeugorientierung ist durch $G_{17} - G_{19}$ festgelegt und wird durch Frames nicht beeinflusst.
CUT3DFE	D-Werkzeugkorrektur für das Stirnfräsen mit konstanter Orientierung. Die Werkzeugorientierung ist die durch $G_{17} - G_{19}$ festgelegte und gegebenenfalls durch einen Frame gedrehte Richtung.
CUT3DF	D-Werkzeugkorrektur für das Stirnfräsen mit Orientierungsänderung (nur bei aktiver 5-Achs-Transformation).
G40 X... Y... Z...	Zum Ausschalten: Linearsatz G_0/G_1 mit Geometrieachsen
ISD	Eintauchtiefe

Hinweis

Die Befehle sind modal wirksam und stehen in der gleichen Gruppe wie `CUT2D` und `CUT2DF`. Die Abwahl findet erst mit der nächsten Bewegung in der aktuellen Ebene statt. Dies gilt immer für `G40` und ist unabhängig vom `CUT`-Befehl.

Zwischensätze bei aktiver 3D-Werkzeugradiuskorrektur sind erlaubt. Es gelten die Festlegungen der 2 1/2D-Werkzeugradiuskorrektur.

Randbedingungen

- **G450/G451 und DISC**

An Außenecken wird immer ein Kreissatz eingefügt. `G450/G451` haben keine Bedeutung. Der Befehl `DISC` wird nicht ausgewertet.

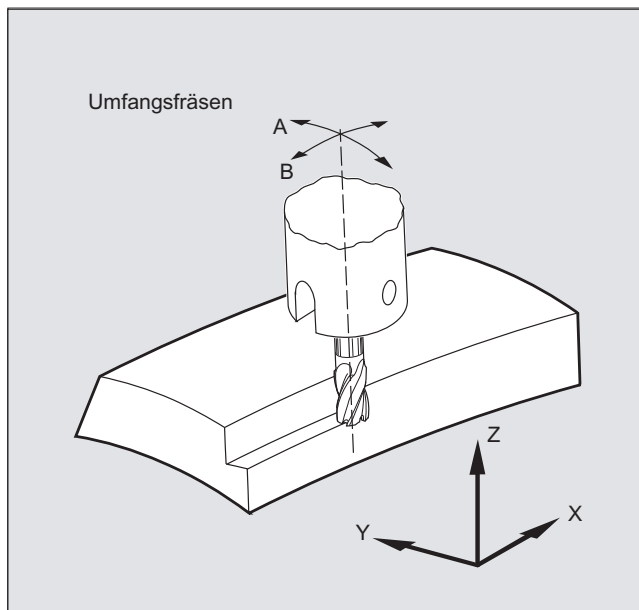
Beispiel

Programmcode	Kommentar
N10 A0 B0 X0 Y0 Z0 F5000	
N20 T1 D1	; Werkzeugaufruf, Werkzeugkorrekturwerte aufrufen.
N30 TRAORI(1)	; Transformationsanwahl
N40 CUT3DC	; 3D-Werkzeugradiuskorrektur-Anwahl
N50 G42 X10 Y10	; Werkzeugradiuskorrektur-Anwahl
N60 X60	
N70 ...	

7.5.2 3D-Werkzeugkorrektur: Umfangfräsen, Stirnfräsen

Umfangsfräsen

Die hier benutzte Variante des Umfangfräsens ist durch die Vorgabe einer Bahn (Leitlinie) und der zugehörigen Orientierung realisiert. Bei dieser Art der Bearbeitung ist auf der Bahn die Werkzeugform ohne Bedeutung. Entscheidend ist allein der Radius am Werkzeugeingriffspunkt.

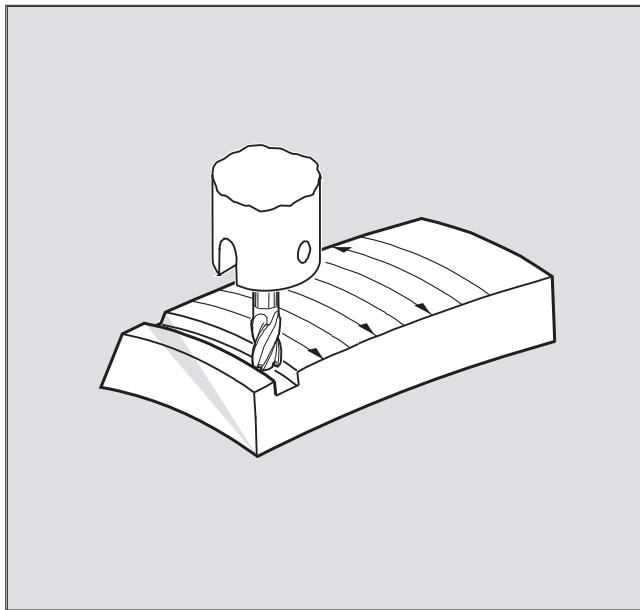


Hinweis

Die Funktion 3D-WRK beschränkt sich auf zylindrische Werkzeuge.

Stirnfräsen

Für diese Art des 3D-FräSENS benötigen Sie die zeilenweise Beschreibung der 3D-Bahnen auf der Werkstückoberfläche. Die Berechnungen werden unter Berücksichtigung der Werkzeugform und Werkzeugabmessungen - üblicherweise im CAM durchgeführt. Der Postprozessor schreibt in das Teileprogramm - neben den NC-Sätzen - die Werkzeugorientierungen (bei aktiver 5-Achstransformation) und den G-Code für die gewünschte 3D-Werkzeugkorrektur. Hierdurch hat der Maschinenbediener die Möglichkeit - abweichend von dem für die Berechnung der NC-Bahnen verwendeten Werkzeug -, geringfügig kleinere Werkzeuge einzusetzen.



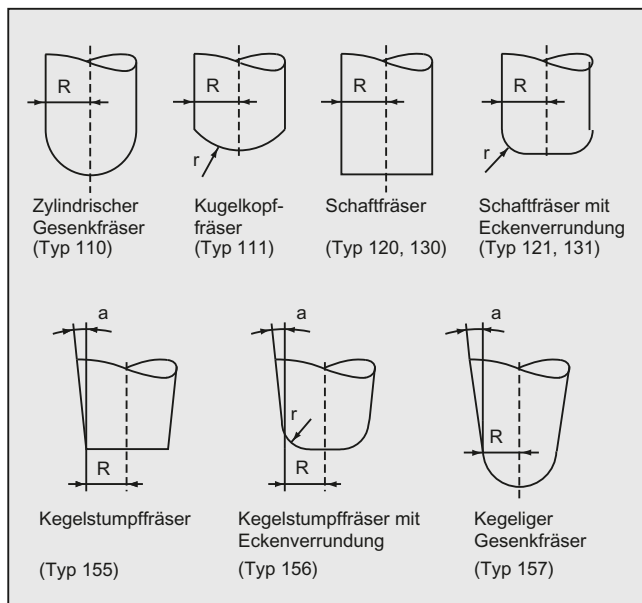
Beispiel:

NC-Sätze wurden mit Fräser 10 mm berechnet. Hier könnte auch mit Fräserdurchmesser 9,9 mm gefertigt werden, wobei dann mit verändertem Rauheitsprofil zu rechnen ist.

7.5.3 3D-Werkzeugkorrektur: Werkzeugformen und Werkzeugdaten für Stirnfräsen

Fräserformen, Werkzeugdaten

Im Folgenden sind die für Stirnfräsen möglichen Werkzeugformen und Grenzwerte der Werkzeugdaten zusammengestellt. Die Form des Werkzeugschafts wird nicht berücksichtigt. Die Werkzeugtypen 120 und 156 sind in ihrer Wirkung identisch.



Wird im NC-Programm eine andere als in der Abbildung gezeigte Typ-Nummer angegeben, verwendet das System automatisch den Werkzeugtyp 110 (Zylindrischer Gesenkfräser). Bei Verletzung der Grenzwerte für die Werkzeugdaten wird ein Alarm ausgegeben.

Fräsertyp	Typ-Nr.	R	r	a
Zylindrischer Gesenkfräser	110	> 0	-	-
Kugelkopffräser	111	> 0	> R	-
Schafffräser, Winkelkopffräser	120, 130	> 0	-	-
Schafffräser, Winkelkopffräser mit Eckenverrundung	121, 131	> r	> 0	-
Kugelstumpffräser	155	> 0	-	> 0
Kugelstumpffräser mit Eckenverrundung	156	> 0	> 0	> 0
Kegeliger Gesenkfräser	157	> 0	-	> 0

- R = Schaftradius (Werkzeugradius)
- r = Eckenradius
- a = Winkel zwischen Werkzeugschaftsachse und oberem Ende der Torusfläche
- = wird nicht ausgewertet

Werkzeugdaten	Werkzeugparameter	
Werkzeugmaße	Geometrie	Verschleiß
R	\$TC_DP6	\$TC_DP15
r	\$TC_DP7	\$TC_DP16
a	\$TC_DP11	\$TC_DP20

Werkzeuglängenkorrektur

Als Bezugspunkt für die Längenkorrektur gilt die Werkzeugspitze (Schnittpunkt Längsachse/Oberfläche).

3D-Werkzeugkorrektur, Werkzeugwechsel

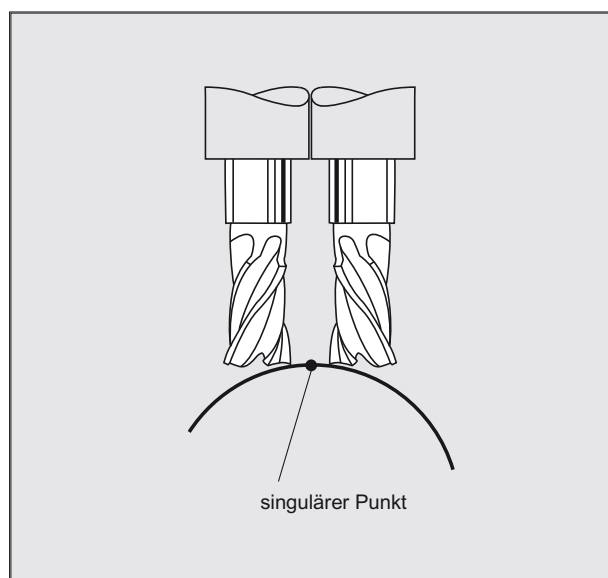
Ein neues Werkzeug mit veränderten Abmessungen (R, r, a) oder anderer Form darf nur mit Programmierung von G41 bzw. G42 angegeben werden (Übergang G40 nach G41 bzw. G42, erneute Programmierung von G41 bzw. G42). Alle anderen Werkzeugdaten, z. B. Werkzeuglängen, bleiben von dieser Regel unberücksichtigt, so dass solche Werkzeuge auch ohne erneutes G41 bzw. G42 eingewechselt werden können.

7.5.4 3D-Werkzeugkorrektur: Korrektur auf der Bahn, Bahnkrümmung, Eintauchtiefe (CUT3DC, ISD)

Funktion

Korrektur auf der Bahn

Beim Stirnfräsen muss der Fall betrachtet werden, dass der Berührungspunkt auf der Werkzeugoberfläche springt. Wie in diesem Beispiel bei der Bearbeitung einer konvexen Fläche mit senkrecht stehendem Werkzeug. Die im Bild gezeigte Anwendung kann als Grenzfall betrachtet werden.



Dieser Grenzfall wird von der Steuerung überwacht, indem auf Basis der Winkelstellungen zwischen Werkzeug und Flächennormalenvektoren sprunghafte Änderungen des Bearbeitungspunktes erkannt werden. An diesen Stellen fügt die Steuerung Linearsätze ein, so dass die Bewegung ausgeführt werden kann.

Für die Berechnung der Linearsätze sind in Maschinendaten für den Seitwärtswinkel zulässige Winkelbereiche hinterlegt. Falls die in Maschinendaten festgelegten Grenzwerte für zugelassene Winkelbereiche überschritten werden, meldet das System Alarm.

Bahnkrümmung

Die Bahnkrümmung wird nicht überwacht. Auch hier empfiehlt es sich, nur solche Werkzeuge zu verwenden, mit denen ohne Konturverletzung gearbeitet werden kann.

Eintauchtiefe (ISD)

Die Eintauchtiefe ISD wird nur bei aktiver 3D-Werkzeugradiuskorrektur ausgewertet.

Mit dem Programmbefehl `ISD` (Insertion Depth) wird die Eintauchtiefe des Werkzeugs beim Umfangsfräsen programmiert. Damit ist es möglich, die Lage des Bearbeitungspunktes auf der Mantelfläche des Werkzeugs zu verändern.

Syntax

3D-Werkzeugkorrektur Umfangsfräsen
CUT3DC
ISD=<Wert>

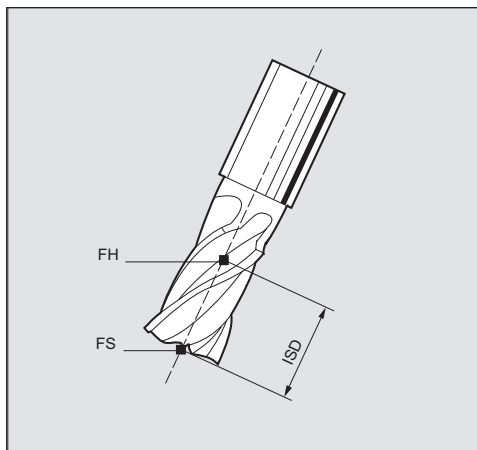
Bedeutung

CUT3DC 3D-Werkzeugkorrektur für das Umfangsfräsen aktivieren, z. B. für Taschenfräsen mit schrägen Seitenwänden.

ISD Mit dem Befehl `ISD` wird der Abstand (<Wert>) zwischen Fräser Spitze (FS) und dem Fräserhilfspunkt (FH) angegeben.

Fräserhilfspunkt

Der Fräserhilfspunkt (FH) entsteht durch Projektion des programmierten Bearbeitungspunktes auf die Werkzeugachse.



Weitere Informationen

Taschenfräsen mit schrägen Seitenwänden für Umfangsfräsen mit CUT3DC

Bei dieser 3D-Werkzeugradiuskorrektur wird eine Abweichung des Fräserradius kompensiert, indem in Richtung der Flächennormalen der zu bearbeitenden Fläche zugestellt wird. Dabei bleibt die Ebene, in der die Stirnseite des Fräasers liegt unverändert, wenn die Eintauchtiefe I_{SD} gleich geblieben ist. Ein Fräser mit z. B. kleinerem Radius gegenüber einem Normwerkzeug würde dann den Taschenboden, der auch die Begrenzungsfläche darstellt, nicht erreicht werden. Für eine automatische Zustellung des Werkzeugs muss der Steuerung diese Begrenzungsfläche bekannt sein, siehe Kapitel "3D-Umfangsfräsen mit Begrenzungsflächen".

Weitere Informationen zur Kollisionsüberwachung siehe:

Literatur:

Programmierhandbuch Grundlagen; Kapitel "Werkzeugkorrekturen".

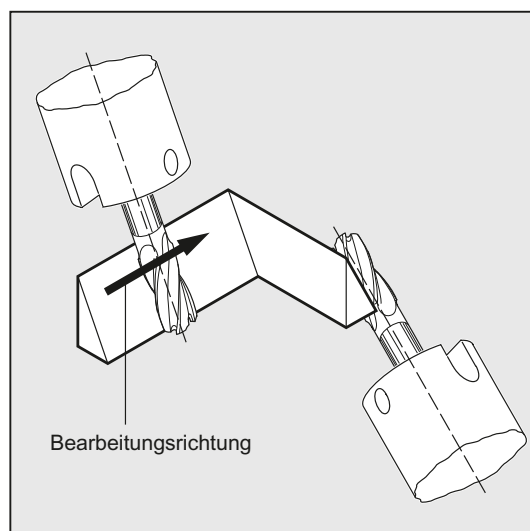
7.5.5 3D-Werkzeugkorrektur: Innenecken/Außenecken und Schnittpunktverfahren (G450/G451)

Funktion

Innenecken/Außenecken

Außen- und Innenecken werden getrennt behandelt. Die Bezeichnung Innen- oder Außenecke ist abhängig von der Werkzeugorientierung.

Bei Orientierungsänderungen an einer Ecke kann der Fall auftreten, dass sich der Eckentyp während der Bearbeitung ändert. Tritt dieser Fall auf, wird die Bearbeitung mit einer Fehlermeldung abgebrochen.



Syntax

G450
G451

Bedeutung

G450 Übergangskreis (Werkzeug umfährt Werkstückecken auf einer Kreisbahn)
G451 Schnittpunkt der Äquidistanten (Werkzeug schneidet in der Werkstückecke frei)

Weitere Informationen

Schnittpunktverfahren für 3D-Korrektur

Bei 3D-Umfangsfräsen wird jetzt an Außenecken der G-Code G450/G451 ausgewertet, d. h. es kann der Schnittpunkt der Offset-Kurven angefahren werden. Bis SW 4 wurde an Außenecken immer ein Kreis eingefügt. Das verfügbare Schnittpunktverfahren ist bei typischen CAD-erzeugten 3D-Programmen besonders vorteilhaft. Diese bestehen häufig aus kurzen Geradensätzen (zur Approximation glatter Kurven), bei denen die Übergänge zwischen benachbarten Sätzen nahezu tangential sind.

Bei Werkzeugradiuskorrektur an der Außenseite der Kontur wurden bislang grundsätzlich Kreise zum Umfahren der Außenecken eingefügt. Da diese Sätze bei nahezu tangentialen Übergängen sehr kurz werden, ergeben sich unerwünschte Geschwindigkeitseinbrüche.

In diesen Fällen werden analog zur 2 ½ D-Radiuskorrektur die beiden beteiligten Kurven verlängert, der Schnittpunkt der beiden verlängerten Kurven wird angefahren.

Der Schnittpunkt wird bestimmt, indem die Offsetkurven der beiden beteiligten Sätze verlängert werden und deren Schnittpunkt in der Ebene senkrecht zur Werkzeugorientierung an der Ecke bestimmt wird. Existiert kein derartiger Schnittpunkt, wird die Ecke wie bisher behandelt, d. h. es wird ein Kreis eingefügt.

Weitere Informationen zum Schnittpunktverfahren siehe:

Literatur:

Funktionshandbuch Sonderfunktionen; 3D-Werkzeugradiuskorrektur (W5)

7.5.6 3D-Werkzeugkorrektur: 3D-Umfangsfräsen mit Begrenzungsflächen

Anpassungen von 3D-Umfangsfräsen an Gegebenheiten von CAD-Programmen

Von CAD-Systemen generierte NC-Programme approximieren in der Regel die Mittelpunktsbahn eines Normwerkzeuges mit einer großen Anzahl kurzer Linearsätze. Damit diese so erzeugten Sätze vieler Teilekonturen die ursprüngliche Originalkontur möglichst genau nachbilden, ist es notwendig im Teileprogramm gewisse Anpassungen vorzunehmen.

Wichtige Informationen, die für eine optimale Korrektur erforderlich wären, aber im Teilprogramm nicht mehr zur Verfügung stehen, müssen durch geeignete Maßnahmen ersetzt werden. Nachfolgend werden typische Methoden dargestellt, um kritische Übergänge entweder direkt im Teileprogramm oder bei Ermittlung der realen Kontur (z. B. durch Zustellung des Werkzeugs) auszugleichen.

Anwendungen

Zusätzlich zu den typischen Anwendungsfällen, bei denen anstelle des Normwerkzeugs ein reales Werkzeug die Mittelpunktsbahn beschreibt, werden auch zylindrische Werkzeuge mit 3D-Werkzeugkorrektur behandelt. Hierbei bezieht sich die programmierte Bahn auf die Kontur an der Bearbeitungsfläche. Die hierfür zutreffende Begrenzungsfläche ist werkzeugunabhängig. Es wird wie bei der herkömmlichen Werkzeugradiuskorrektur der Gesamtradius zur Berechnung des senkrechten Offsets zur Begrenzungsfläche herangezogen.

7.5.7 3D-Werkzeugkorrektur: Berücksichtigung einer Begrenzungsfläche (CUT3DCC, CUT3DCCD)

Funktion

3D-Umfangsfräsen mit realen Werkzeugen

Beim 3D-Umfangsfräsen mit kontinuierlicher oder konstanter Veränderung der Werkzeugorientierung wird häufig die Werkzeugmittelpunktsbahn für ein definiertes Normwerkzeug programmiert. Da in der Praxis oft nicht die passenden Normwerkzeuge zur Verfügung stehen, kann ein von einem Normwerkzeug nicht allzu stark abweichendes Werkzeug eingesetzt werden.

Mit CUT3DCCD wird für ein reales Differenzwerkzeug eine Begrenzungsfläche berücksichtigt, die das programmierte Normwerkzeug beschreiben würde. Das NC-Programm beschreibt die Mittelpunktsbahn des Normwerkzeuges.

Mit CUT3DCC wird bei Verwendung von zylindrischen Werkzeugen eine Begrenzungsfläche berücksichtigt, die das programmierte Normwerkzeug erreicht hätte. Das NC-Programm beschreibt die Kontur auf der Bearbeitungsfläche.

Syntax

CUT3DCCD
CUT3DCC

Bedeutung

CUT3DCCD Aktivierung der 3D-Werkzeugkorrektur für das Umfangfräsen mit Begrenzungsflächen mit Differenzwerkzeug auf der Werkzeugmittelpunktbahn: Zustellung zur Begrenzungsfläche.

CUT3DCC Aktivierung der 3D-Werkzeugkorrektur für das Umfangfräsen mit Begrenzungsflächen mit 3D-Radiuskorrektur: Kontur an der Bearbeitungsfläche

Hinweis

Werkzugradiuskorrektur mit G41, G42

Für die Werkzeugradiuskorrektur mit G41, G42 bei aktivem CUT3DCCD oder CUT3DCC muss die Option "Orientierungstransformation" vorhanden sein.

Normwerkzeuge mit Eckenverrundung

Die Eckenverrundung des Normwerkzeugs wird durch den Werkzeugparameter \$TC_DP7 beschrieben. Aus den Werkzeugparameter \$TC_DP16 ergibt sich die Abweichung der Eckenverrundung des realen Werkzeugs gegenüber dem Normwerkzeug.

Beispiel

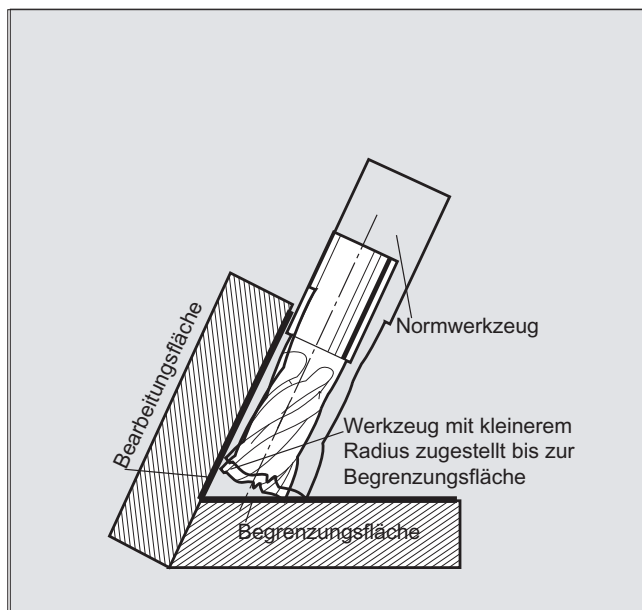
Werkzeugabmessungen für einen Torusfräser mit verringertem Radius gegenüber dem Normwerkzeug.

Werkzeugtyp	R = Schafradius	r = Eckenradius
Normwerkzeug mit Eckenverrundung	$R = \$TC_DP6$	$r = \$TC_DP7$
Reales Werkzeug mit Eckenverrundung: Werkzeugtypen 121 und 131 Torusfräser (Schaftfräser)	$R' = \$TC_DP6 + \$TC_DP15 + OFFN$	$r' = \$TC_DP7 + \TC_DP16
In diesem Beispiel sind sowohl \$TC_DP15 + OFFN als auch \$TC_DP16 negativ. Der Werkzeugtyp (\$TC_DP1) wird ausgewertet.		
Zugelassen sind nur Fräsertypen mit zylindrischen Schaft (Zylinder- oder Schaftfräser) sowie Torusfräser (Typ 121 und 131) und im Grenzfall der zylindrische Gesenkfräser (Typ 110).	Bei diesen zugelassenen Fräsertypen ist der Eckenradius r gleich dem Schafradius R. Alle anderen zugelassenen Werkzeugtypen werden als Zylinderfräser interpretiert und ein eventuell angegebenes Maß für die Eckenverrundung wird nicht ausgewertet.	
Zugelassen sind alle Werkzeugtypen der Nummern 1 – 399 mit Ausnahme der Nummern 111 und 155 bis 157.		

Weitere Informationen

Werkzeugmittelpunktsbahn mit Zustellung bis zur Begrenzungsfläche CUT3DCCD

Wird ein Werkzeug verwendet, welches im Vergleich zum passenden Normwerkzeug einen kleineren Radius aufweist, dann wird ein in Längsrichtung zugestellter Fräser soweit weiter geführt, bis dieser den Taschenboden wieder berührt. Damit wird die Ecke, die von der Bearbeitungs- und der Begrenzungsfläche gebildet wird so weit ausgeräumt, wie dies das Werkzeug zulässt. Es handelt sich dabei um eine gemischte Bearbeitungsweise aus Umfangs- und Stirnfräsen. Analog zu einem Werkzeug mit verringertem Radius, wird beim Werkzeug mit vergrößerten Radius, in die entgegengesetzte Richtung entsprechend zugestellt.



Gegenüber allen anderen Werkzeugkorrekturen der G-Code Gruppe 22 hat ein für CUT3DCCD angegebener Werkzeugparameter $\$TC_DP6$ keine Bedeutung für den Werkzeugradius und beeinflusst die resultierende Korrektur nicht.

Der Korrekturoffset ergibt sich aus der Summe von:

- Verschleißwert des Werkzeugradius (Werkzeugparameter $\$TC_DP15$)
- und einem zur Berechnung des senkrechten Offsets zur Begrenzungsfläche programmierten Werkzeugoffset $OFFN$.

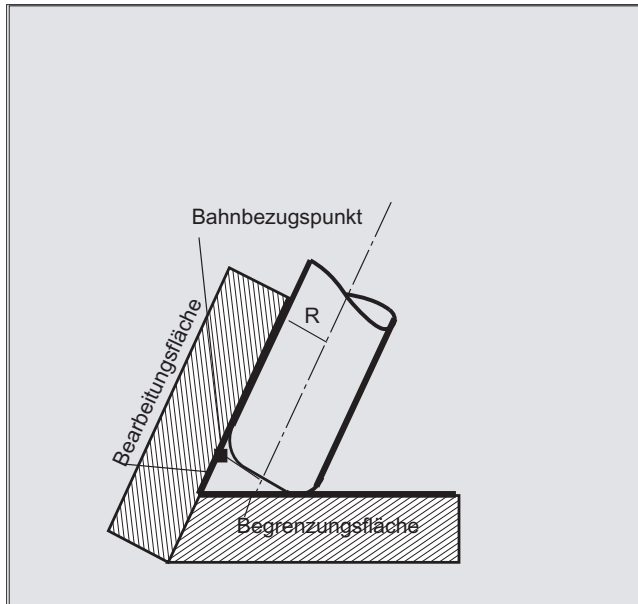
Ob die zu bearbeitende Fläche links oder rechts von der Bahn liegt, kann aus dem erzeugten Teileprogramm nicht entnommen werden. Es wird deshalb von einem positiven Radius und einem negativen Verschleißwert des Originalwerkzeuges ausgegangen. Ein negativer Verschleißwert beschreibt immer ein Werkzeug mit verringertem Durchmesser.

Verwendung von zylindrischen Werkzeugen

Bei der Verwendung von zylindrischen Werkzeugen ist eine Zustellung nur dann erforderlich, wenn die Bearbeitungsfläche und die Begrenzungsfläche einen spitzen Winkel (kleiner als 90 Grad) bilden. Werden Torusfräser (Zylinder mit Eckverrundung) verwendet, dann erfordert dies sowohl bei spitzen als auch bei stumpfen Winkeln eine Zustellung in Längsrichtung des Werkzeugs.

3D-Radiuskorrektur mit CUT3DCC, Kontur an der Bearbeitungsfläche

Ist `CUT3DCC` mit einem Torusfräser aktiv, so bezieht sich die programmierte Bahn auf einen fiktiven Zylinderfäser gleichen Durchmessers. Der hieraus resultierende Bahnbezugspunkt ist bei Verwendung eines Torusfräser im folgenden Bild dargestellt.



Es ist zulässig, dass der Winkel zwischen Bearbeitungs- und Begrenzungsfläche auch innerhalb eines Satzes von einem spitzen in einem stumpfen Winkel oder umgekehrt übergeht.

Gegenüber dem Normwerkzeug darf das verwendete reale Werkzeug sowohl größer als auch kleiner sein. Dabei darf der resultierende Eckenradius nicht negativ werden und das Vorzeichen des resultierenden Werkzeugradius muss erhalten bleiben.

Bei `CUT3DCC` bezieht sich das NC-Teileprogramm auf die Kontur an der Bearbeitungsfläche. Es wird hierbei wie bei der herkömmlichen Werkzeugradienkorrektur der Gesamtradius herangezogen, der sich zusammensetzt aus der Summe von:

- Werkzeugradius (Werkzeugparameter `$TC_DP6`)
- Verschleißwert (Werkzeugparameter `$TC_DP15`)
- und einem zur Berechnung des senkrechten Offsets zur Begrenzungsfläche programmierten Werkzeugoffset `OFFN`.

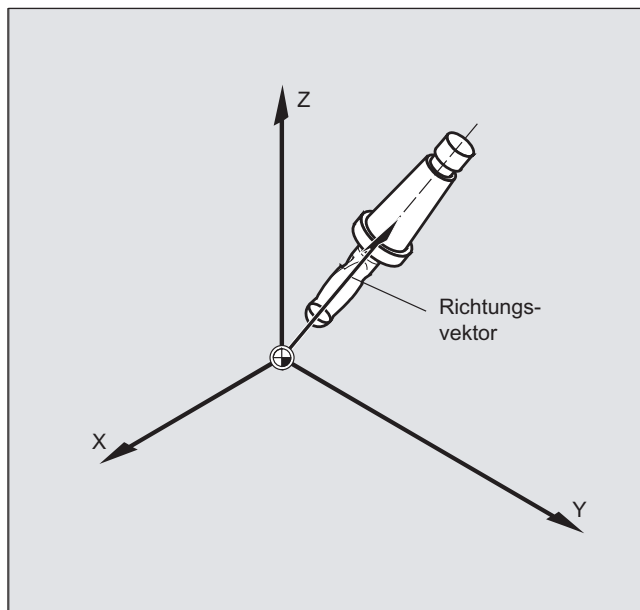
Die Lage der Begrenzungsfläche wird bestimmt aus der Differenz der beiden Werte:

- Abmessungen des Normwerkzeugs
- Werkzeugradius (WZ-Parameter `$TC_DP6`)

7.6 Werkzeugorientierung (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST)

Funktion

Unter Werkzeugorientierung versteht man die geometrische Ausrichtung des Werkzeugs im Raum. Bei einer 5-Achs-Bearbeitungsmaschine ist die Werkzeugorientierung über Programmbefehle einstellbar.



Mit `OSD` und `OST` aktivierte Überschiefbewegungen der Orientierung werden je nach Interpolationsart für die Werkzeugorientierung unterschiedlich gebildet.

Bei aktiver Vektorinterpolation wird der geglättete Orientierungsverlauf auch mittels Vektorinterpolation interpoliert. Dagegen wird bei aktiver Rundachsinterpolation die Orientierung direkt mittels Rundachsbewegungen geglättet.

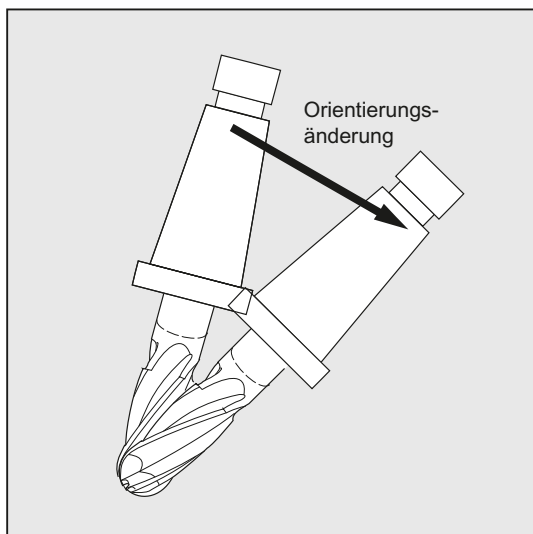
Programmierung

Programmierung der Orientierungsänderung:

Eine Orientierungsänderung des Werkzeugs kann programmiert werden durch:

- direkte Programmierung der Rundachsen A , B , C (Rundachsinterpolation)
- Euler- oder RPY-Winkel
- Richtungsvektor (Vektorinterpolation durch Angabe von A_3 oder B_3 oder C_3)
- `LEAD/TILT` (Stirnfräsen)

Das Bezugskordinatensystem ist entweder das Maschinenkoordinatensystem (`ORIMKS`) oder das aktuelle Werkstückkoordinatensystem (`ORIWKS`).



Programmierung der Werkzeugorientierung:

Befehl	Bedeutung
ORIC:	Orientierung und Bahnbewegung parallel
ORID:	Orientierung und Bahnbewegung nacheinander
OSOF:	keine Orientierungsglättung
OSC:	Orientierung konstant
OSS:	Orientierungsglättung nur am Satzanfang
OSSE:	Orientierungsglättung am Satzanfang und -ende
ORIS:	Geschwindigkeit der Orientierungsänderung bei eingeschalteter Orientierungsglättung in Grad pro mm (gilt für <i>OSS</i> und <i>OSSE</i>)
OSD:	Überschleifen der Orientierung durch Vorgabe der Überschleiflänge mit dem Settingdatum: SD42674 \$SC_ORI_SMOOTH_DIST
OST:	Überschleifen der Orientierung durch Vorgabe der Winkeltoleranz in Grad bei Vektorinterpolation mit dem Settingdatum: SD42676 \$SC_ORI_SMOOTH_TOL Bei Rundachsinterpolation wird die vorgegebene Toleranz als maximale Abweichung der Orientierungsachsen angenommen.

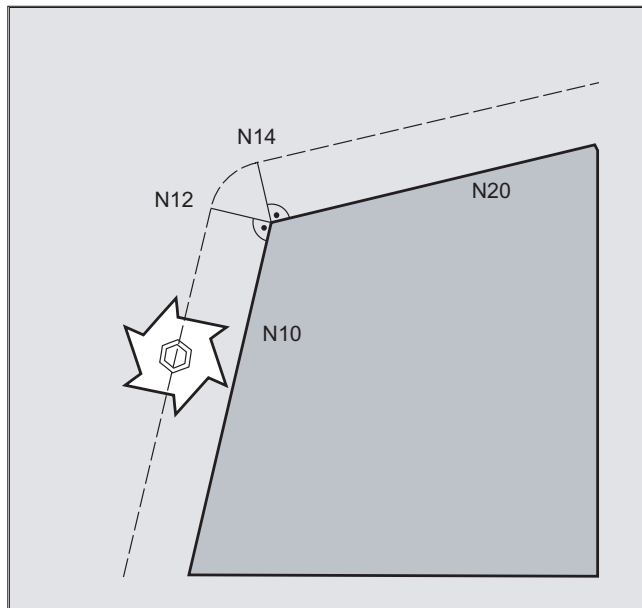
Hinweis

Alle Befehle zum Überschleifen der Werkzeugorientierung (*OSOF*, *OSC*, *OSS*, *OSSE*, *OSD* und *OST*) sind in der G-Funktionsgruppe 34 zusammengefasst. Sie sind modal wirksam, d. h. es kann immer nur einer dieser Befehle wirken.

Beispiele

Beispiel 1: ORIC

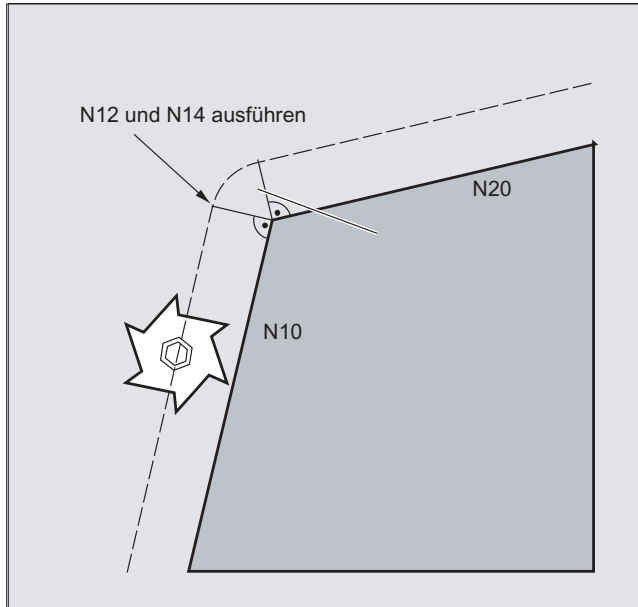
Sind zwischen den Verfahrssätzen N10 und N20 zwei oder mehrere Sätze mit Orientierungsänderungen (z. B. A2=... B2=... C2=...) programmiert und ORIC ist aktiv, so wird der eingefügte Kreissatz entsprechend dem Betrag der Winkeländerungen auf diese Zwischensätze aufgeteilt.



Programmcode	Kommentar
ORIC	
N8 A2=... B2=... C2=...	
N10 X... Y... Z...	
N12 C2=... B2=...	; Der Kreissatz, der an der Außenecke eingefügt wird, verteilt sich auf N12 und N14, entsprechend der Orientierungsänderung. Kreisbewegung und Orientierungsänderung werden hierbei parallel ausgeführt.
N14 C2=... B2=...	
N20 X =...Y=... Z=... G1 F200	

Beispiel 2: ORID

Ist ORID aktiv, so werden alle Sätze zwischen den beiden Verfahrssätzen am Ende des ersten Verfahrssatzes ausgeführt. Der Kreissatz mit konstanter Orientierung wird unmittelbar vor dem zweiten Verfahrssatz ausgeführt.

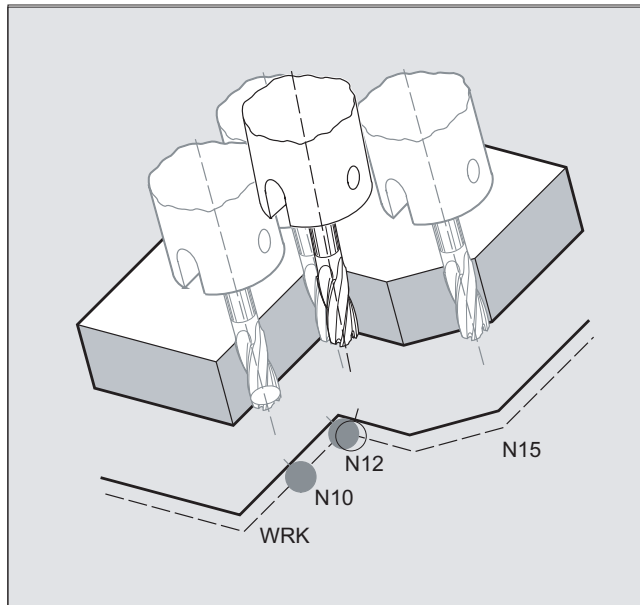


Programmcode	Kommentar
ORID	
N8 A2=... B2=... C2=...	
N10 X... Y... Z...	
N12 A2=... B2=... C2=...	; Der Satz N12 und N14 wird am Ende von N10 ausgeführt. Danach wird der Kreissatz mit der aktuellen Orientierung ausgefahren.
N14 M20	; Hilfsfunktionen etc.
N20 X... Y... Z...	

Hinweis

Für die Art der Orientierungsänderung an einer Außenecke ist der Programmbefehl maßgebend, welcher im ersten Verfahrssatz einer Außenecke aktiv ist.

Ohne Orientierungsänderung: Wird die Orientierung an der Satzgrenze nicht verändert, so ist der Werkzeugquerschnitt ein Kreis, der die beiden Konturen berührt.

Beispiel 3: Änderung der Orientierung an einer Innenecke**Programmcode**

```

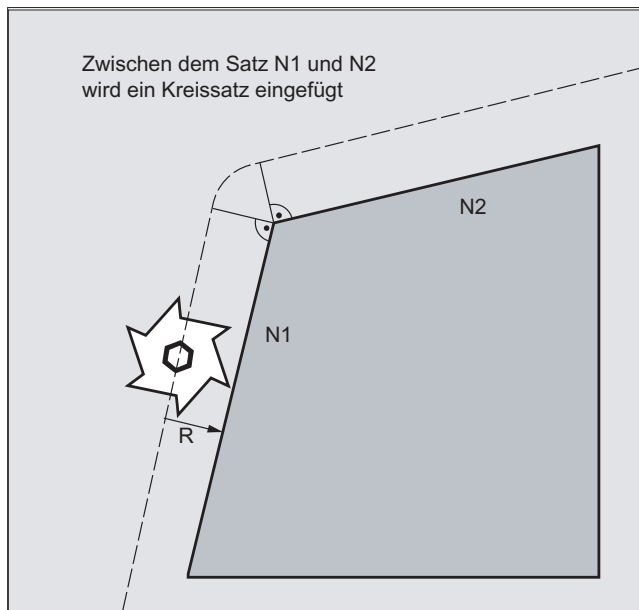
ORIC
N10 X ...Y... Z... G1 F500
N12 X ...Y... Z... A2=... B2=... C2=...
N15 X ...Y... Z... A2=... B2=... C2=...

```

Weitere Informationen**Verhalten an Außenecken**

An einer Außenecke wird immer ein Kreissatz mit dem Radius des Fräsers eingefügt.

Mit den Programmbefehlen `ORIC` bzw. `ORID` kann festgelegt werden, ob Orientierungsänderungen, die zwischen Satz `N1` und `N2` programmiert wurden, vor Beginn des eingefügten Kreissatzes oder gleichzeitig mit diesem ausgeführt werden.



Ist an Außenecken eine Orientierungsänderung notwendig, so kann diese wahlweise parallel zur Interpolation oder getrennt mit der Bahnbewegung erfolgen.

Bei `ORID` werden zunächst die eingefügten Sätze ohne Bahnbewegung ausgeführt. Der Kreissatz wird unmittelbar vor dem zweiten der beiden Verfahrssätze eingefügt, durch welche die Ecke gebildet wird.

Sind an einer Außenecke mehrere Orientierungssätze eingefügt und `ORIC` ist angewählt, so wird die Kreisbewegung entsprechend den Beträgen der Orientierungsänderungen der einzelnen eingefügten Sätze auf diese verteilt.

Überschleifen der Orientierung mit `OSD` bzw. `OST`

Beim Überschleifen mit `G642` kann die maximale Abweichung für die Konturachsen und die Orientierungsachsen nicht sehr unterschiedlich sein. Die kleinere Toleranz von beiden bestimmt die Form der Überschleifbewegung bzw. Winkeltoleranz, den Orientierungsverlauf relativ stark zu glätten, ohne dabei größere Konturabweichungen hinnehmen zu müssen.

Durch Aktivierung von `OSD` bzw. `OST` ist es möglich, mit einer vorgegebenen Überschleiflänge bzw. Winkeltoleranz sehr geringe Abweichungen des Orientierungsverlaufs ohne gravierende Konturabweichungen "großzügig" zu glätten.

Hinweis

Im Unterschied zum Überschleifen der Kontur (und dem Orientierungsverlauf) mit `G642` wird beim Überschleifen der Orientierung mit `OSD` bzw. `OST` kein eigener Satz gebildet, sondern die Überschleifbewegung wird direkt in die programmierten Originalsätze eingefügt.

Mit `OSD` bzw. `OST` können keine Satzübergänge überschleifen werden bei denen ein Wechsel der Interpolationsart für die Werkzeugorientierung (Vektor → Rundachse, Rundachse → Vektor) stattfindet. Diese Satzübergänge können gegebenenfalls mit den herkömmlichen Überschleiffunktionen `G641`, `G642` bzw. `G643` überschleifen werden.

7.7 Freie D-Nummernvergabe, Schneidenummer

7.7.1 Freie D-Nummernvergabe, Schneidenummer (Adresse CE)

D-Nummer

Die D-Nummern können als Korrekturnummern verwendet werden. Zusätzlich kann über die Adresse CE die Nummer der Schneide adressiert werden. Über die Systemvariable \$TC_DPCE kann die Schneidenummer beschrieben werden.

Voreinstellung: Korrekturr. == Schneidennr.

Über Maschinendaten werden die maximale Anzahl der D-Nummern (Schneidenummern) und die maximale Schneidenanzahl pro Werkzeug festgelegt (→ Maschinenhersteller). Die folgenden Befehle sind nur sinnvoll, wenn die maximale Schneidenummer (MD18105) größer als die Anzahl der Schneiden pro Werkzeug (MD18106) festgelegt wurde. Beachten Sie die Angaben des Maschinenherstellers.

Hinweis

Neben der relativen D-Nummernvergabe können die D-Nummern auch als "flache" bzw. "absolute" D-Nummern (1-32000) ohne Bezug zu einer T-Nummer vergeben werden (innerhalb der Funktion "Flache D-Nummernstruktur").

Literatur

Funktionshandbuch Grundfunktionen; Werkzeugkorrektur (W1)

7.7.2 Freie D-Nummernvergabe: D-Nummern prüfen (CHKDNO)

Funktion

Mit dem Befehl `CHKDNO` prüfen Sie, ob die vorhandenen D-Nummern eindeutig vergeben worden sind. Die D-Nummern aller innerhalb einer TO-Einheit definierten Werkzeuge dürfen nur einmal auftreten. Ersatzwerkzeuge werden dabei nicht berücksichtigt.

Syntax

```
state=CHKDNO (Tno1, Tno2, Dno)
```

Bedeutung

<code>state</code>	=TRUE:	Die D-Nummern wurden für den überprüften Bereich eindeutig vergeben.
	=FALSE:	Es erfolgte eine D-Nummernkollision oder die Parametrierung ist ungültig. Über <code>Tno1</code> , <code>Tno2</code> und <code>Dno</code> werden die Parameter übergeben, die zur Kollision führten. Diese Daten können im Teileprogramm ausgewertet werden.
<code>CHKDNO (Tno1, Tno2)</code>		Es werden alle D-Nummern der genannten Werkzeuge geprüft.
<code>CHKDNO (Tno1)</code>		Es werden alle D-Nummern von <code>Tno1</code> gegen alle anderen Werkzeuge geprüft.
<code>CHKDNO</code>		Es werden alle D-Nummern aller Werkzeuge gegen alle anderen Werkzeuge geprüft.

7.7.3 Freie D-Nummernvergabe: D-Nummern umbenennen (GETDNO, SETDNO)

Funktion

D-Nummern müssen eindeutig vergeben werden. Zwei verschiedene Schneiden eines Werkzeuges können nicht dieselbe D-Nummer haben.

GETDNO

Dieser Befehl liefert die D-Nummer einer bestimmten Schneide (ce) eines Werkzeuges mit der T-Nummer t. Existiert keine D-Nummer zu den eingegebenen Parametern, wird d=0 gesetzt. Ist die D-Nummer ungültig wird ein Wert größer 32000 zurückgegeben.

SETDNO

Mit diesem Befehl weisen Sie den Wert d der D-Nummer einer Schneide ce des Werkzeuges t zu. Über state wird das Ergebnis dieser Anweisung zurückgegeben (TRUE oder FALSE). Existiert kein Datensatz zu den eingegebenen Parametern wird FALSE zurückgegeben. Syntaxfehler erzeugen einen Alarm. Die D-Nummer kann nicht explizit auf 0 gesetzt werden.

Syntax

```
d = GETDNO (t,ce)
state = SETDNO (t,ce,d)
```

Bedeutung

d D-Nummer der Schneide des Werkzeuges
t T-Nummer des Werkzeuges
ce Schneidenummer (CE-Nummer) des Werkzeuges
state Gibt an, ob der Befehl fehlerfrei ausgeführt werden konnte (TRUE oder FALSE).

Beispiel Umbenennen einer D-Nummer

Programmierung	Kommentar
\$TC_DP2 [1,2] = 120	;
\$TC_DP3 [1,2] = 5.5	;
\$TC_DPCE[1,2] = 3	; Schneidenummer CE
...	;
N10 def int DNrAlt, DNrNeu = 17	;
N20 DNrAlt = GETDNO(1,3)	;
N30 SETDNO(1,3, DNrNeu)	;

Damit wird der Schneide CE=3 der neue D-Wert 17 zugewiesen. Jetzt werden die Daten dieser Schneide über die D-Nummer 17 angesprochen; sowohl über die Systemvariablen als auch in der Programmierung mit der NC-Adresse.

7.7.4 Freie D-Nummernvergabe: T-Nummer zur vorgegebenen D-Nummer ermitteln (GETACTTD)

Funktion

Mit dem Befehl `GETACTTD` ermitteln Sie zu einer absoluten D-Nummer die dazugehörige T-Nummer. Es erfolgt keine Prüfung auf Eindeutigkeit. Gibt es mehrere gleiche D-Nummern innerhalb einer TO-Einheit, wird die T-Nummer des ersten gefundenen Werkzeugs zurückgegeben. Bei Verwendung "flacher" D-Nummern ist die Verwendung des Befehls nicht sinnvoll, da hier immer der Wert "1" zurückgegeben wird (keine T-Nummer in der Datenhaltung).

Syntax

```
status=GETACTTD (Tnr, Dnr)
```

Bedeutung

Dnr	D-Nummer, für die die T-Nummer gesucht werden soll.
Tnr	Gefundene T-Nummer
status	Wert: Bedeutung:
0	Die T-Nummer wurde gefunden. Tnr erhält den Wert der T-Nummer.
-1	Zur angegebenen D-Nummer existiert keine T-Nummer; Tnr=0.
-2	Die D-Nummer ist nicht absolut. Tnr erhält den Wert des ersten gefundenen Werkzeugs, das die D-Nummer mit dem Wert Dnr enthält.
-5	Die Funktion konnte aus einem anderen Grund nicht ausgeführt werden.

7.7.5 Freie D-Nummernvergabe: D-Nummern ungültig setzen (DZERO)

Funktion

Der Befehl `DZERO` dient zur Unterstützung während dem Umrüsten. So gekennzeichnete Korrekturdatensätze werden nicht mehr vom Befehl `CHKDNO` geprüft. Um sie wieder zugänglich zu machen, muss die D-Nummer wieder mit `SETDNO` gesetzt werden.

Syntax

```
DZERO
```

Bedeutung

DZERO	Kennzeichnet alle D-Nummern der TO-Einheit als ungültig.
-------	--

7.8 Werkzeugträgerkinematik

Voraussetzungen

Ein Werkzeugträger kann ein Werkzeug nur dann in alle möglichen Raumrichtungen orientieren, wenn

- zwei Drehachsen v_1 und v_2 vorhanden sind.
- die Drehachsen aufeinander senkrecht stehen.
- die Werkzeuglängsachse senkrecht auf der zweiten Drehachse v_2 steht.

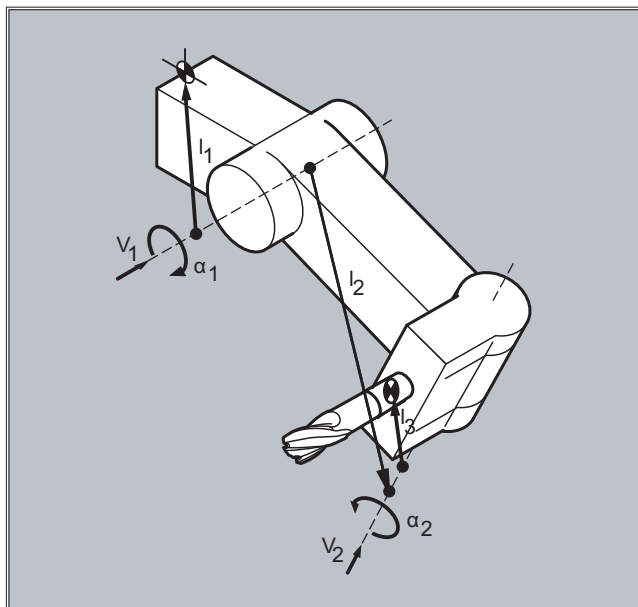
Zusätzlich gilt bei Maschinen, bei denen alle möglichen Orientierungen einstellbar sein müssen, folgende Forderung:

- die Werkzeugorientierung muss senkrecht auf der ersten Drehachse v_1 stehen.

Funktion

Die Werkzeugträgerkinematik mit maximal zwei Drehachsen v_1 oder v_2 wird über die 17 Systemvariablen $\$TC_CARR1[m]$ bis $\$TC_CARR17[m]$ beschrieben. Die Beschreibung des Werkzeugträgers besteht aus:

- dem vektoriellen Abstand von der ersten Drehachse zum Bezugspunkt des Werkzeugträgers I_1 , dem vektoriellen Abstand von erster zu zweiter Drehachse I_2 , dem vektoriellen Abstand von zweiter Drehachse zum Bezugspunkt des Werkzeugs I_3 .
- den Richtungsvektoren beider Drehachsen v_1 , v_2 .
- den Drehwinkeln α_1 , α_2 um die beiden Achsen. Die Drehwinkel werden mit Blickrichtung in Richtung der Drehachsvektoren im Uhrzeigersinn positiv gezählt.



Für Maschinen mit **aufgelöster Kinematik** (sowohl Werkzeug als auch Werkstück sind drehbar) wurden die Systemvariablen um die Einträge $\$TC_CARR18[m]$ bis $\$TC_CARR23[m]$ erweitert.

Parameter

Funktion der Systemvariablen für orientierbare Werkzeugträger			
Bezeichnung	x-Komponente	y-Komponente	z-Komponente
l ₁ Offsetvector	\$TC_CARR1[m]	\$TC_CARR2[m]	\$TC_CARR3[m]
l ₂ Offsetvector	\$TC_CARR4[m]	\$TC_CARR5[m]	\$TC_CARR6[m]
v ₁ Drehachse	\$TC_CARR7[m]	\$TC_CARR8[m]	\$TC_CARR9[m]
v ₂ Drehachse	\$TC_CARR10[m]	\$TC_CARR11[m]	\$TC_CARR12[m]
α ₁ Drehwinkel	\$TC_CARR13[m]		
α ₂ Drehwinkel	\$TC_CARR14[m]		
l ₃ Offsetvector	\$TC_CARR15[m]	\$TC_CARR16[m]	\$TC_CARR17[m]

Erweiterungen der Systemvariablen für orientierbare Werkzeugträger			
Bezeichnung	x-Komponente	y-Komponente	z-Komponente
l ₄ Offsetvector	\$TC_CARR18[m]	\$TC_CARR19[m]	\$TC_CARR20[m]
Achsbezeichner Drehachse v ₁ Drehachse v ₂	Achsbezeichner der Drehachsen v ₁ und v ₂ (Vorbelegung ist Null) \$TC_CARR21[m] \$TC_CARR22[m]		
Kinematiktyp	\$TC_CARR23[m]		
Tool	Kinematiktyp-T ->	Kinematiktyp-P ->	Kinematiktyp-M
Part	Nur das Werkzeug ist drehbar (Vorbelegung)	Nur das Werkstück ist drehbar	Werkstück & Werkzeug sind drehbar
Mixed mode			
Offset der Drehachse v ₁ Drehachse v ₂	Winkel in Grad der Drehachsen v ₁ und v ₂ bei Einnahme der Grundstellung \$TC_CARR24[m] \$TC_CARR25[m]		
Winkeloffset der Drehachse v ₁ Drehachse v ₂	Offset der Hirth-Verzahnung in Grad der Drehachsen v ₁ und v ₂ \$TC_CARR26[m] \$TC_CARR27[m]		
Winkelinkrem. v ₁ Drehachse v ₂ Drehachse	Inkrement der Hirth-Verzahnung in Grad der Drehachsen v ₁ und v ₂ \$TC_CARR28[m] \$TC_CARR29[m]		
Min.-Position Drehachse v ₁ Drehachse v ₂	Software-Limit für Minimalposition der Drehachsen v ₁ und v ₂ \$TC_CARR30[m] \$TC_CARR31[m]		
Max.-Position Drehachse v ₁ Drehachse v ₂	Software-Limits für Maximalposition der Drehachsen v ₁ und v ₂ \$TC_CARR32[m] \$TC_CARR33[m]		
Werkzeugträger Name	Anstelle einer Zahl kann ein Werkzeugträger einem Namen bekommen. \$TC_CARR34[m]		
Anwender: Achsnamen 1 Achsnamen 2 Kennung	Beabsichtigte Verwendung innerhalb der Messzyklen vom Anwender. \$TC_CARR35[m] \$TC_CARR36[m] \$TC_CARR37[m]		
Position	\$TC_CARR38[m]	\$TC_CARR39[m]	\$TC_CARR40[m]

Erweiterungen der Systemvariablen für orientierbare Werkzeugträger			
Feinver- schiebung	Parameter, die zu den Werten in den Basisparametern addiert werden können.		
l ₁ Offsetvector	\$TC_CARR41[m]	\$TC_CARR42[m]	\$TC_CARR43[m]
l ₂ Offsetvector	\$TC_CARR44[m]	\$TC_CARR45[m]	\$TC_CARR46[m]
l ₃ Offsetvector	\$TC_CARR55[m]	\$TC_CARR56[m]	\$TC_CARR57[m]
l ₄ Offsetvector	\$TC_CARR58[m]	\$TC_CARR59[m]	\$TC_CARR60[m]
v ₁ Drehachse	\$TC_CARR64[m]		
v ₂ Drehachse	\$TC_CARR65[m]		

Hinweis

Erklärungen zu den Parametern

Mit "m" wird jeweils die Nummer des zu beschreibenden Werkzeugträgers angegeben.

\$TC_CARR47 bis \$TC_CARR54 sowie \$TC_CARR61 bis \$TC_CARR63 sind nicht definiert und führen beim Versuch hierauf lesend oder schreiben zuzugreifen, zu einem Alarm.

Die Anfangs- bzw. Endpunkte der Abstandsvektoren auf den Achsen können frei gewählt werden. Die Drehwinke α_1, α_2 um die beiden Achsen werden im Grundzustand des Werkzeugträgers mit 0° definiert. Die Kinematik eines Werkzeugträgers kann so auf beliebig viele Möglichkeiten beschrieben werden.

Werkzeugträger mit nur einer oder keiner Drehachse können durch Nullsetzen der Richtungsvektoren einer oder beider Drehachsen beschrieben werden.

Bei einem Werkzeugträger ohne Drehachse wirken die Abstandsvektoren wie zusätzliche Werkzeugkorrekturen, deren Komponenten beim Umschalten der Bearbeitungsebenen (G17 bis G19)° nicht beeinflusst werden.

Erweiterungen der Parameter

Parameter der Drehachsen

Die Systemvariablen wurden um die Einträge \$TC_CARR24[m] bis \$TC_CARR33[m] erweitert und wie folgt beschrieben:

Den Offset der Drehachsen v ₁ , v ₂	Veränderung der Position der Drehachse v ₁ oder v ₂ bei Grundstellung des orientierbaren Werkzeugträgers.
Den Winkeloffset/ Winkelinkrement Drehachsen v ₁ , v ₂	Offset oder Winkelinkrement der Hirth-Verzahnung der Drehachsen v ₁ und v ₂ . Programmierter oder berechneter Winkel wird auf den nächstliegenden Wert gerundet, der sich bei ganzzahligem n aus $\phi = s + n \cdot d$ ergibt.
Minimal- und Maximalposition Drehachsen v ₁ , v ₂	Der Minimalposition/Maximalposition der Drehachse Grenzwinkel (Software-Limit) der Drehachse v ₁ und v ₂ .

Parameter für den Anwender

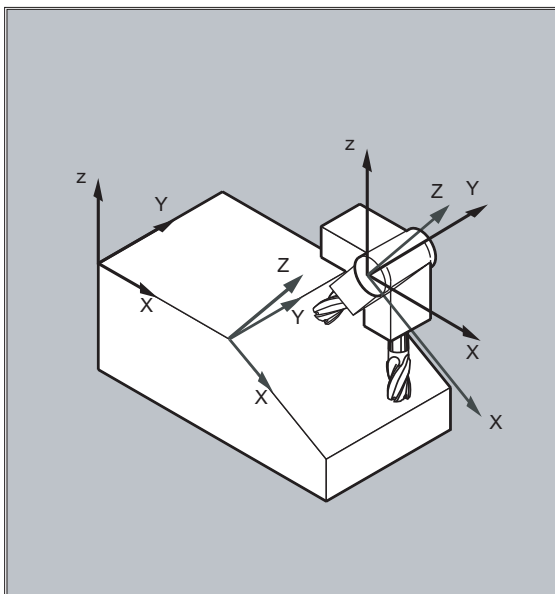
\$TC_CARR34 bis \$TC_CARR40 enthalten Parameter, die den Anwender zur freien Verfügung stehen und bis zum SW 6.4 standardmäßig innerhalb der NCK nicht weiter ausgewertet werden oder keine Bedeutung haben.

Parameter der Feinverschiebung

\$TC_CARR41 bis \$TC_CARR65 enthalten Feinverschiebungsparameter, die zu den Werten in den Basisparametern addiert werden können. Der einem Basisparameter zugeordnete Feinverschiebungswert ergibt sich, wenn zur Parameternummer der Wert 40 addiert wird.

Beispiel

Der im folgenden Beispiel verwendete Werkzeugträger lässt sich durch eine Drehung um die Y-Achse vollständig beschreiben.



Programmcode	Kommentar
N10 \$TC_CARR8[1]=1	; Definition der Y-Komponente der ersten Drehachse des Werkzeugträgers 1.
N20 \$TC_DP1[1,1]=120	; Definition eines Schaftfräasers.
N30 \$TC_DP3[1,1]=20	; Definition eines Schaftfräasers mit Länge 20 mm.
N40 \$TC_DP6[1,1]=5	; Definition eines Schaftfräasers mit Radius 5 mm.
N50 ROT Y37	; Framedefinition mit Drehung von 37° um die Y-Achse.
N60 X0 Y0 Z0 F10000	; Ausgangsposition anfahren.
N70 G42 CUT2DF TCOFR TCARR=1 T1 D1 X10	; Radiuskorrektur, Werkzeuglängenkorrektur im gedrehten Frame einstellen, Werkzeugträger 1, Werkzeug 1 anwählen.
N80 X40	; Bearbeitung unter einer Drehung von 37° durchführen.
N90 Y40	
N100 X0	
N110 Y0	
N120 M30	

Weitere Informationen

Aufgelöste Kinematik

Für Maschinen mit aufgelöster Kinematik (sowohl Werkzeug als auch Werkstück sind drehbar) wurden die Systemvariablen um die Einträge $\$TC_CARR18$ [m] bis $\$TC_CARR23$ [m] erweitert und wie folgt beschrieben:

Der drehbare Werkzeuggestisch bestehend aus:

- dem vektoriellen Abstand der zweiten Drehachse v_2 zum Bezugspunkt eines drehbaren Werkzeuggestisches I_4 der dritten Drehachse.

Die Rundachsen bestehend aus:

- den beiden Kanalbezeichnern für den Bezug der Drehachsen v_1 und v_2 , auf deren Position gegebenenfalls bei der Bestimmung der Orientierung des orientierbaren Werkzeugträgers zugegriffen wird.

Der Kinematiktyp mit einem der Werte T, P oder M:

- Kinematiktyp T: Nur das Werkzeug ist drehbar.
- Kinematiktyp P: Nur das Werkstück ist drehbar.
- Kinematiktyp M: Werkzeug und Werkstück sind drehbar.

Löschen der Werkzeugträgerdaten

Mit $\$TC_CARR1 [0] = 0$ können die Daten aller Werkzeugträgerdatensätze gelöscht werden.

Der Kinematiktyp $\$TC_CARR23 [T] = T$ muss mit einem der drei zulässigen Groß- oder Kleinbuchstaben (T,P,M) belegt werden und sollte aus diesen Grund nicht gelöscht werden.

Ändern der Werkzeugträgerdaten

Jeder der beschriebenen Werte kann durch Zuweisung eines neuen Wertes im Teileprogramm verändert werden. Jedes andere Zeichen als T, P oder M führt bei dem Versuch, den orientierbaren Werkzeugträger zu aktivieren, zu einem Alarm.

Lesen der Werkzeugträgerdaten

Jeder der beschriebenen Werte kann durch Zuweisung an eine Variable im Teileprogramm gelesen werden.

Feinverschiebungen

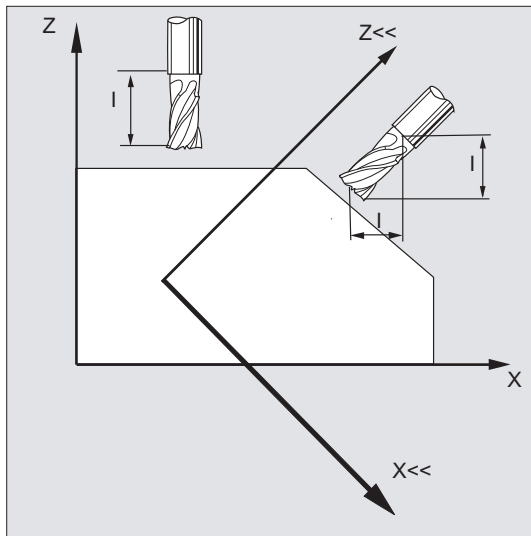
Ein unzulässiger Feinverschiebungswert wird erst erkannt, wenn ein orientierbarer Werkzeugträger aktiviert wird, der solch einen Wert enthält und gleichzeitig das Settingdatum SD42974 $\$SC_TOCARR_FINE_CORRECTION = TRUE$ ist.

Der Betrag der zulässigen Feinverschiebung wird über Maschinendaten auf einen maximal zulässigen Wert begrenzt.

7.9 Werkzeuglängenkorrektur für orientierbare Werkzeugträger (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ)

Funktion

Mit veränderter Raumorientierung des Werkzeugs ändern sich auch dessen Werkzeuglängenkomponenten.



Nach Umrüsten, z. B. durch manuelle Einstellung oder Wechsel des Werkzeugträgers mit fester räumlicher Ausrichtung, müssen daher die Werkzeuglängenkomponenten neu ermittelt werden. Dies erfolgt mit den Wegbefehlen `TCOABS` und `TCOFR`.

Bei einem orientierbaren Werkzeugträger eines aktiven Frames kann bei Werkzeuganwahl mit `TCOFRZ`, `TCOFRY` und `TCOFRX` die Richtung, in die das Werkzeug zeigen soll, bestimmt werden.

Syntax

```
TCARR= [<m>]  
TCOABS  
TCOFR  
TCOFRZ  
TCOFRY  
TCOFRX
```

Bedeutung

TCARR= [<m>]:	Werkzeugträger mit der Nummer "m" anfordern
TCOABS:	Werkzeuglängenkomponenten aus der aktuellen Werkzeugträgerorientierung berechnen

7.9 Werkzeuglängenkorrektur für orientierbare Werkzeugträger (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ)

TCOFR:	Werkzeuglängenkomponenten aus der Orientierung des aktiven Frames bestimmen
TCOFRZ:	Orientierbarer Werkzeugträger aus aktiven Frame, dessen Werkzeug in Z-Richtung zeigt
TCOFRY:	Orientierbarer Werkzeugträger aus aktiven Frame, dessen Werkzeug in Y-Richtung zeigt
TCOFRX:	Orientierbarer Werkzeugträger aus aktiven Frame, dessen Werkzeug in X-Richtung zeigt

Weitere Informationen

Werkzeuglängenkorrektur aus Trägerorientierung (TCOABS)

TCOABS berechnet die Werkzeuglängenkorrektur aus den aktuellen Orientierungswinkeln des Werkzeugträgers; abgelegt in den Systemvariablen \$TC_CARR13 und \$TC_CARR14.

Zur Definition der Werkzeugträgerkinematik mit Systemvariablen siehe " Werkzeugträgerkinematik (Seite 447) ".

Zur Neuberechnung der Werkzeuglängenkorrektur bei Frame-Wechsel muss das Werkzeug nochmals ausgewählt werden.

Werkzeugrichtung aus aktiven Frame

Der orientierbare Werkzeugträger kann so eingestellt werden, dass das Werkzeug in folgende Richtungen zeigt:

- mit TCOFR bzw. TCOFRZ in Z-Richtung
- mit TCOFRY in Y-Richtung
- mit TCOFRX in X-Richtung

Ein Umschalten zwischen TCOFR und TCOABS bewirkt eine Neuberechnung der Werkzeuglängenkorrektur.

Werkzeugträger anfordern (TCARR)

Mit TCARR werden mit der Werkzeugträgernummer m dessen Geometriedaten angefordert (Korrekturspeicher).

Mit m=0 wird der aktive Werkzeugträger ausgewählt.

Die Geometriedaten des Werkzeugträgers werden erst nach Aufruf eines Werkzeugs aktiv. Das angewählte Werkzeug bleibt über den Wechsel eines Werkzeugträgers hinaus aktiv.

Die aktuellen Geometriedaten des Werkzeugträgers können auch im Teileprogramm über die entsprechenden Systemvariablen definiert werden.

Neuberechnung der Werkzeuglängenkorrektur (TCOABS) bei Frame-Wechsel

Zur Neuberechnung der Werkzeuglängenkorrektur bei Frame-Wechsel muss das Werkzeug nochmals ausgewählt werden.

Hinweis

Die Werkzeugorientierung muss dem aktiven Frame manuell angepasst werden.

Bei der Berechnung der Werkzeuglängenkorrektur werden in einem Zwischenschritt auch die Drehwinkel des Werkzeugträgers berechnet. Da bei Werkzeugträgern mit zwei Drehachsen im Allgemeinen zwei Drehwinkelpaare existieren, mit denen die Werkzeugorientierung dem aktiven Frame angepasst werden kann, müssen die in den Systemvariablen abgelegten Drehwinkelwerte zumindest annähernd den mechanisch eingestellten Drehwinkeln entsprechen.

Hinweis

Werkzeugorientierung

Die Steuerung kann die über die Frame-Orientierung berechneten Verdrehwinkel nicht auf die Einstellbarkeit an der Maschine überprüfen.

Sind die Drehachsen des Werkzeugträgers konstruktiv so angeordnet, dass die durch die Frame-Orientierung berechnete Werkzeugorientierung nicht erreicht werden kann, wird ein Alarm ausgegeben.

Die Kombination von Werkzeugfeinkorrektur und den Funktionalitäten zur Werkzeuglängenkorrektur bei beweglichen Werkzeugträgern ist nicht zulässig. Beim Versuch beide Funktionen gleichzeitig aufzurufen, erfolgt eine Fehlermeldung.

Mit `TOFRAME` ist es möglich, einen Frame aufgrund der Orientierungsrichtung des angewählten Werkzeugträgers zu definieren. Genauere Informationen siehe Kapitel "Frames".

Bei aktiver Orientierungstransformation (3-, 4-, 5-Achstransformation) kann ein Werkzeugträger mit von der Null-Lage abweichender Orientierung angewählt werden, ohne dass dabei ein Alarm ausgegeben wird.

Übergabeparameter von Standard- und Messzyklen

Für die Übergabeparameter von Standard- und Messzyklen gelten definierte Wertebereiche.

Bei Winkelwerten ist der Wertebereich wie folgt festgelegt:

- Drehung um 1. Geometrieachse: -180 Grad bis +180 Grad
- Drehung um 2. Geometrieachse: -90 Grad bis +90 Grad
- Drehung um 3. Geometrieachse: -180 Grad bis +180 Grad

Siehe Kapitel Frames, "Programmierbare Drehung (ROT, AROT, RPL)".

Hinweis

Bei der Übergabe von Winkelwerten an einen Standard- oder Messzyklus ist zu beachten:

Werte kleiner als die Rechenfeinheit der NC sind auf Null zu runden!

Die Rechenfeinheit der NC für Winkelpositionen ist festgelegt im Maschinendatum:

MD10210 \$MN_INT_INCR_PER_DEG

7.10 Online-Werkzeuglängenkorrektur (TOFFON, TOFFOF)

Funktion

Über die Systemvariable \$AA_TOFF[<n>] können die effektiven Werkzeuglängen entsprechend der drei Werkzeugrichtungen dreidimensional in Echtzeit überlagert werden.

Als Index <n> werden die drei Geometrieachsbezeichner verwendet. Damit ist die Anzahl der aktiven Korrekturrichtungen durch die zur selben Zeit aktiven Geometrieachsen festgelegt.

Alle Korrekturen können gleichzeitig aktiv sein.

Die Funktion Online-Werkzeuglängenkorrektur ist anwendbar bei:

- Orientierungstransformation TRAORI
- Orientierbare Werkzeugträger TCARR

Hinweis

Die Online-Werkzeuglängenkorrektur ist eine **Option**, die vorher frei geschaltet werden muss. Nur in Verbindung mit einer aktiven Orientierungstransformation oder einem aktiven orientierbaren Werkzeugträger ist diese Funktion sinnvoll.

Syntax

```
TRAORI
TOFFON (<Korrekturrichtung> [, <Offsetwert>])
WHEN TRUE DO $AA_TOFF [<Korrekturrichtung>]           ; In Synchronaktionen.
...
TOFFOF (<Korrekturrichtung>)
```

Weitere Erläuterungen zur Programmierung der Online-Werkzeuglängenkorrektur in Bewegungssynchronaktionen siehe "Synchronaktionen (Seite 569)".

Bedeutung

TOFFON:	Online-Werkzeuglängenkorrektur aktivieren
<Korrekturrichtung>:	Werkzeugrichtung (x, y, z), in der die Online-Werkzeuglängenkorrektur wirksam sein soll.
<Offsetwert>:	Bei der Aktivierung kann für die entsprechende Korrekturrichtung ein Offsetwert angegeben werden, der sofort herausgefahren wird.
TOFFOF:	Online-Werkzeuglängenkorrektur zurücksetzen
	Die Korrekturwerte in der angegebenen Korrekturrichtung werden zurückgesetzt und es wird ein Vorlaufstopp ausgelöst.

Beispiele

Beispiel 1: Anwahl der Werkzeiglängenkorrektur

Programmcode	Kommentar
MD21190 \$MC_TOFF_MODE =1	; Absolute Werte werden angefahren.
MD21194 \$MC_TOFF_VELO[0] =1000	
MD21196 \$MC_TOFF_VELO[1] =1000	
MD21194 \$MC_TOFF_VELO[2] =1000	
MD21196 \$MC_TOFF_ACCEL[0] =1	
MD21196 \$MC_TOFF_ACCEL[1] =1	
MD21196 \$MC_TOFF_ACCEL[2] =1	
N5 DEF REAL XOFFSET	
N10 TRAORI(1)	; Transformation ein.
N20 TOFFON(Z)	; Aktivierung der Online-WZL-Korrektur für die Z-Werkzeugrichtung.
N30 WHEN TRUE DO \$AA_TOFF[Z]=10 G4 F5	; Für die Z-Werkzeugrichtung wird eine WZL-Korrektur von 10 interpoliert.
...	
N100 XOFFSET=\$AA_TOFF_VAL[X]	; Aktuelle Korrektur in X-Richtung zuweisen.
N120 TOFFON(X,-XOFFSET) G4 F5	; Für die X-Werkzeugrichtung wird die WZL-Korrektur wieder zu 0 zurückgefahren.

Beispiel 2: Abwahl der Werkzeiglängenkorrektur

Programmcode	Kommentar
N10 TRAORI(1)	; Transformation ein.
N20 TOFFON(X)	; Aktivierung der Online-WZL-Korrektur für die X-Werkzeugrichtung.
N30 WHEN TRUE DO \$AA_TOFF[X]=10 G4 F5	; Für die X-Werkzeugrichtung wird eine WZL-Korrektur von 10 interpoliert.
...	
N80 TOFFOF(X)	; Positionsoffset der X-Werkzeugrichtung wird gelöscht: ...\$AA_TOFF[X]=0 Es wird keine Achse verfahren. Zur aktuellen Position im WKS wird der Positionsoffset entsprechend der aktuellen Orientierung hinzugerechnet.

Weitere Informationen

Satzaufbereitung

Bei der Satzaufbereitung im Vorlauf wird der im Hauptlauf wirksame aktuelle Werkzeiglängenoffset mit berücksichtigt. Um die maximal zulässigen Achsgeschwindigkeiten weitgehend ausnutzen zu können, ist es erforderlich, die Satzaufbereitung mit einem Vorlaufstopp `STOPRE` anzuhalten, während ein Werkzeugoffset aufgebaut wird.

Der Werkzeugoffset ist zum Vorlaufzeitpunkt auch immer dann bekannt, wenn die Werkzeiglängenkorrekturen nach Programmstart nicht mehr verändert werden, oder wenn nach einer Veränderung der Werkzeiglängenkorrekturen mehr Sätze abgearbeitet wurden als der IPO-Buffer zwischen Vorlauf und Hauptlauf aufnehmen kann.

Variable `$AA_TOFF_PREP_DIFF`

Das Maß für die Differenz zwischen der aktuellen im Interpolator wirksamen Korrektur und der Korrektur, die zum Zeitpunkt der Satzaufbereitung wirksam war, kann in der Variablen `$AA_TOFF_PREP_DIFF[<n>]` abgefragt werden.

Maschinendaten und Settingdaten einstellen

Für die Online-Werkzeiglängenkorrektur stehen folgende Systemdaten zur Verfügung:

- MD20610 `$MC_ADD_MOVE_ACCEL_RESERVE` (Beschleunigungsreserve für überlagerte Bewegung)
- MD21190 `$MC_TOFF_MODE`
Inhalt der Systemvariable `$AA_TOFF[<n>]` wird als absoluter Wert herausgefahren oder aufintegriert.
- MD21194 `$MC_TOFF_VELO` (Geschwindigkeit der Online-Werkzeiglängenkorrektur)
- MD21196 `$MC_TOFF_ACCEL` (Beschleunigung der Online-Werkzeiglängenkorrektur)
- Settingdatum zur Vorgabe von Grenzwerten:
SD42970 `$SC_TOFF_LIMIT` (Obergrenze des Werkzeiglängenkorrekturwertes)

Literatur:

Funktionshandbuch Sonderfunktionen; F2: Mehrachstransformationen

7.11 Schneidendaten-Modifikation bei drehbaren Werkzeugen (CUTMOD)

Funktion

Mit der Funktion "Schneidendaten-Modifikation bei drehbaren Werkzeugen" können die veränderten geometrischen Verhältnisse, die sich bei der Drehung von Werkzeugen (vorwiegend Drehwerkzeuge, aber auch Bohr- und Fräswerkzeuge) relativ zum bearbeiten Werkstück ergeben, bei der Werkzeugkorrektur berücksichtigt werden.

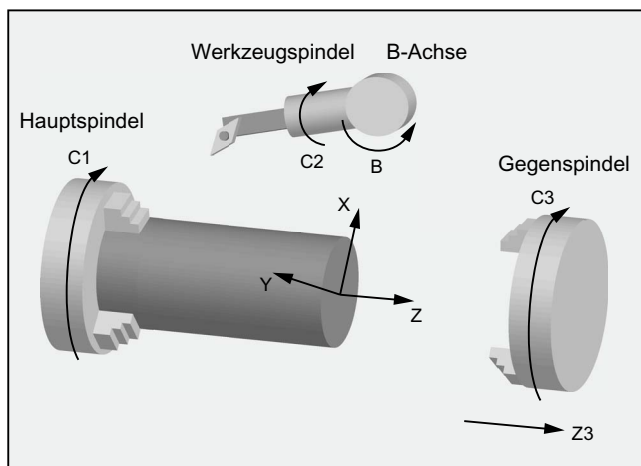


Bild 7-1 Drehbares Werkzeug bei einer Drehmaschine

Die aktuelle Drehung des Werkzeugs wird dabei immer aus einem aktuell aktiven orientierbaren Werkzeugträger (siehe " Werkzeuglängenkorrektur für orientierbare Werkzeugträger (Seite 452) ") ermittelt.

Die Funktion wird aktiviert mit dem Befehl `CUTMOD.`

Syntax

`CUTMOD=<Wert>`

Bedeutung

CUTMOD	Befehl zum Einschalten der Funktion "Schneidendaten-Modifikation bei drehbaren Werkzeugen"
<Wert>	Dem CUTMOD-Befehl können folgende Werte zugewiesen werden:
0	Die Funktion ist deaktiviert. Die von den Systemvariablen \$P_AD... gelieferten Werte sind gleich den korrespondierenden Werkzeugparametern.
> 0	Die Funktion wird aktiviert, falls ein orientierbarer Werkzeugträger mit der angegebenen Nummer aktiv ist, d. h. die Aktivierung ist an einen bestimmten orientierbaren Werkzeugträger gebunden. Die von den Systemvariablen \$P_AD... gelieferten Werte sind gegenüber den korrespondierenden Werkzeugparametern abhängig von der aktiven Drehung gegebenenfalls modifiziert. Die Deaktivierung des bezeichneten orientierbaren Werkzeugträgers deaktiviert die Funktion temporär, die Aktivierung eines anderen orientierbaren Werkzeugträgers deaktiviert sie permanent. Im ersten Fall wird die Funktion deshalb bei erneuter Anwahl des gleichen orientierbaren Werkzeugträgers wieder aktiviert, im zweiten Fall ist eine erneute Anwahl notwendig, auch dann wenn zu einem späteren Zeitpunkt der orientierbare Werkzeugträger mit der angegebenen Nummer erneut aktiviert wird. Die Funktion wird durch Reset nicht beeinflusst.
-1	Die Funktion wird immer aktiviert, falls ein orientierbarer Werkzeugträger aktiv ist. Beim Wechsel des Werkzeugträgers oder bei dessen Abwahl und einer späteren erneuten Anwahl muss CUTMOD nicht erneut gesetzt werden.
-2	Die Funktion wird immer aktiviert, falls ein orientierbarer Werkzeugträger aktiv ist, dessen Nummer gleich der des aktuell aktiven orientierbaren Werkzeugträgers ist. Ist kein orientierbarer Werkzeugträger aktiv, ist das gleichbedeutend mit CUTMOD=0. Ist ein orientierbarer Werkzeugträger aktiv, ist das gleichbedeutend mit der unmittelbaren Angabe der aktuellen Werkzeugträgernummer.
< -2	Werte kleiner -2 werden ignoriert, d. h. dieser Fall wird so behandelt, als wäre CUTMOD nicht programmiert. Hinweis: Dieser Wertebereich sollte nicht verwendet werden, weil er für eventuelle spätere Erweiterungen reserviert ist.

Hinweis

SD42984 \$SC_CUTDIRMOD

Die über den Befehl CUTMOD aktivierbare Funktion ersetzt die über das Settingdatum SD42984 \$SC_CUTDIRMOD aktivierbare Funktion. Diese Funktion steht jedoch weiterhin unverändert zur Verfügung. Da es aber nicht sinnvoll ist, beide Funktionen parallel zu nutzen, kann sie nur aktiviert werden, wenn CUTMOD gleich Null ist.

Beispiel

Das folgende Beispiel bezieht sich auf ein Werkzeug mit der Schneidenlage 3 und einem orientierbaren Werkzeugträger, der das Werkzeug um die B-Achse drehen kann.

Die Zahlenwerte in den Kommentaren geben jeweils die Satzendpositionen in Maschinenkoordinaten (MKS) in der Reihenfolge X, Y, Z an.

Programmcode	Kommentar		
N10 \$TC_DP1 [1,1]=500			
N20 \$TC_DP2 [1,1]=3	; Schneidenlage		
N30 \$TC_DP3 [1,1]=12			
N40 \$TC_DP4 [1,1]=1			
N50 \$TC_DP6 [1,1]=6			
N60 \$TC_DP10 [1,1]=110	; Halterwinkel		
N70 \$TC_DP11 [1,1]=3	; Schnitttrichtung		
N80 \$TC_DP24 [1,1]=25	; Freiwinkel		
N90 \$TC_CARR7 [2]=0 \$TC_CARR8 [2]=1 \$TC_CARR9 [2]=0	; B-Achse		
N100 \$TC_CARR10 [2]=0 \$TC_CARR11 [2]=0 \$TC_CARR12 [2]=1	; C-Achse		
N110 \$TC_CARR13 [2]=0			
N120 \$TC_CARR14 [2]=0			
N130 \$TC_CARR21 [2]=X			
N140 \$TC_CARR22 [2]=X			
N150 \$TC_CARR23 [2]="M"			
N160 TCOABS CUTMOD=0			
N170 G18 T1 D1 TCARR=2	X	Y	Z
N180 X0 Y0 Z0 F10000	; 12.000	0.000	1.000
N190 \$TC_CARR13 [2]=30			
N200 TCARR=2			
N210 X0 Y0 Z0	; 10.892	0.000	-5.134
N220 G42 Z-10	; 8.696	0.000	-17.330
N230 Z-20	; 8.696	0.000	-21.330
N240 X10	; 12.696	0.000	-21.330
N250 G40 X20 Z0	; 30.892	0.000	-5.134
N260 CUTMOD=2 X0 Y0 Z0	; 8.696	0.000	-7.330
N270 G42 Z-10	; 8.696	0.000	-17.330
N280 Z-20	; 8.696	0.000	-21.330
N290 X10	; 12.696	0.000	-21.330
N300 G40 X20 Z0	; 28.696	0.000	-7.330
N310 M30			

Erläuterungen:

In Satz N180 wird zunächst das Werkzeug bei CUTMOD=0 und nicht gedrehtem orientierbaren Werkzeugträger angewählt. Da alle Offsetvektoren des orientierbaren Werkzeugträgers 0 sind, wird die Position angefahren, die den in \$TC_DP3 [1,1] und \$TC_DP4 [1,1] angegebenen Werkzeuglängen entspricht.

In Satz N200 wird der orientierbare Werkzeugträger mit einer Drehung von 30° um die B-Achse aktiviert. Da die Schneidenlage wegen CUTMOD=0 nicht modifiziert wird, ist nach wie vor der alte Schneidenbezugspunkt maßgebend. Deshalb wird in Satz N210 die Position angefahren, die den alten Schneidenbezugspunkt im Nullpunkt beibehält (d. h. der Vektor (1, 12) wird in der Z/X-Ebene um 30° gedreht).

In Satz N260 ist im Unterschied zu Satz N200 CUTMOD=2 wirksam. Aufgrund der Drehung des orientierbaren Werkzeugträgers wird die modifizierte Schneidenlage 8. Daraus folgen auch abweichende Achspositionen.

In den Sätzen N220 bzw. N270 wird jeweils die Werkzeugradiuskorrektur (WRK) aktiviert. Die unterschiedliche Schneidenlage in beiden Programmstücken hat auf die Endpositionen der Sätze, in denen die WRK aktiv ist, keinen Einfluss, die entsprechenden Positionen sind deshalb identisch. Erst in den Abwahlsätzen N260 bzw. N300 wirken sich die unterschiedlichen Schneidenlagen wieder aus.

Weitere Informationen

Wirksamkeit der modifizierten Schneidendaten

Die modifizierte Schneidenlage und der modifizierte Schneidenbezugspunkt werden bei Programmierung auch für ein bereits aktives Werkzeug sofort wirksam. Eine Werkzeugneuanwahl ist dazu nicht notwendig.

Einfluss der aktiven Arbeitsebene

Für die Bestimmung von modifizierter Schneidenlage, Schnittrichtung und Halter- bzw. Freiwinkel ist die Betrachtung der Schneide in der jeweils aktiven Ebene (G17 - G19) maßgebend.

Enthält jedoch das Settingdatum SD42940 \$SC_TOOL_LENGTH_CONST (Wechsel der Werkzeuglängenkomponenten bei Ebenenwechsel) einen gültigen Wert ungleich Null (plus oder minus 17, 18 oder 19), so bestimmt dessen Inhalt die Ebene, in der die relevanten Größen betrachtet werden.

Systemvariablen

Folgende Systemvariablen stehen zur Verfügung:

Systemvariablen	Bedeutung
\$P_CUTMOD_ANG / \$AC_CUTMOD_ANG	Liefert den (nicht gerundeten) Winkel in der aktiven Bearbeitungsebene, der für die Modifikation der Schneidendaten (Schneidenlage, Schnittrichtung, Freiwinkel und Halterwinkel) bei den mit CUTMOD bzw. \$SC_CUTDIRMOD aktivierten Funktionen zugrunde gelegt wurde. \$P_CUTMOD_ANG bezieht sich auf den aktuellen Zustand im Vorlauf, \$AC_CUTMOD_ANG auf den aktuellen Hauptlaufsatz.

7.11 Schneidendaten-Modifikation bei drehbaren Werkzeugen (CUTMOD)

Systemvariablen	Bedeutung
\$P_CUTMOD / \$AC_CUTMOD	<p>Liest den aktuell gültigen Wert, der zuletzt mit dem Befehl CUTMOD programmiert wurde (Nummer des Werkzeugträgers, für den die Schneidendaten-Modifikation aktiviert werden soll).</p> <p>War der letzte programmierte CUTMOD-Wert = -2 (Aktivierung mit dem aktuell aktiven orientierbaren Werkzeugträger), dann wird in \$P_CUTMOD nicht der Wert -2, sondern die Nummer des zum Zeitpunkt der Programmierung aktiven orientierbaren Werkzeugträgers zurückgeliefert.</p> <p>\$P_CUTMOD bezieht sich auf den aktuellen Zustand im Vorlauf, \$AC_CUTMOD auf den aktuellen Hauptlaufsatz.</p>
\$P_CUT_INV / \$AC_CUT_INV	<p>Liefert den Wert TRUE, wenn das Werkzeug so gedreht ist, dass die Spindeldrehrichtung invertiert werden muss. Dazu müssen in dem Satz, auf den sich die jeweilige Leseoperation bezieht, die folgenden vier Bedingungen erfüllt sein:</p> <ol style="list-style-type: none"> 1. Es ist ein Dreh- oder Schleifwerkzeug aktiv (Werkzeugtypen 400 bis 599 und / oder SD42950 \$SC_TOOL_LENGTH_TYPE = 2). 2. Die Schneidenbeeinflussung wurde mit dem Sprachbefehl CUTMOD aktiviert. 3. Es ist ein orientierbarer Werkzeugträger aktiv, der durch den numerischen Wert von CUTMOD bezeichnet wurde. 4. Der orientierbare Werkzeugträger dreht das Werkzeug um eine Achse in der Bearbeitungsebene (typischerweise die C-Achse) so, dass die resultierende Normale der Werkzeugschneide gegenüber der Ausgangslage um mehr als 90° (typischerweise 180°) gedreht ist. <p>Ist mindestens eine der genannten vier Bedingungen nicht erfüllt, ist der Inhalt der Variablen FALSE. Für Werkzeuge, deren Schneidenlage nicht definiert ist, ist der Wert der Variablen immer FALSE.</p> <p>\$P_CUT_INV bezieht sich auf den aktuellen Zustand im Vorlauf und \$AC_CUT_INV auf den aktuellen Hauptlaufsatz.</p>

Alle Hauptlaufvariablen (\$AC_CUTMOD_ANG, \$AC_CUTMOD und \$AC_CUT_INV) können in Synchronaktionen gelesen werden. Ein Lesezugriff aus dem Vorlauf generiert einen Vorlaufstopp.

Modifizierte Schneidendaten:

Falls eine Werkzeugdrehung aktiv ist, werden die modifizierten Daten in den folgenden Systemvariablen zur Verfügung gestellt:

Systemvariable	Bedeutung
\$P_AD[2]	Schneidenlage
\$P_AD[10]	Halterwinkel
\$P_AD[11]	Schnittrichtung
\$P_AD[24]	Freiwinkel

Hinweis

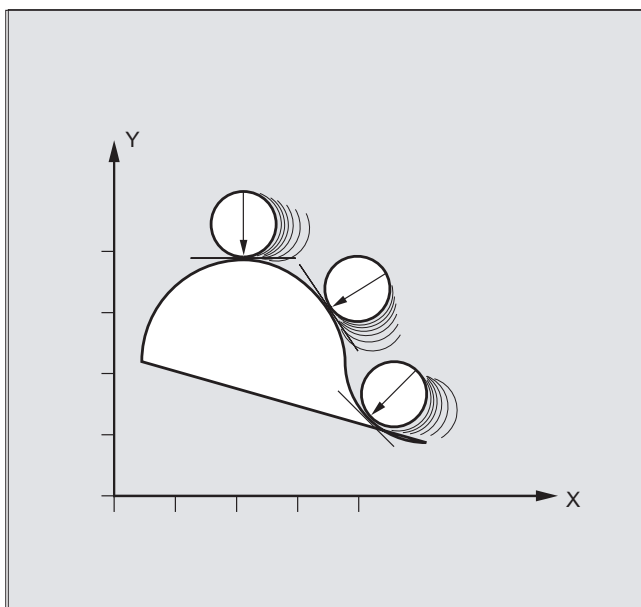
Die Daten sind gegenüber den korrespondierenden Werkzeugparametern (\$TC_DP2[... , ...] usw.) immer dann modifiziert, wenn die Funktion "Schneidendaten-Modifikation bei drehbaren Werkzeugen" mit dem Befehl CUTMOD aktiviert wurde und ein orientierbarer Werkzeugträger aktiv ist, der eine Werkzeugdrehung bewirkt.

Bahnverhalten

8.1 Tangentialsteuerung (TANG, TANGON, TANGOF, TLIFT, TANGDEL)

Funktion

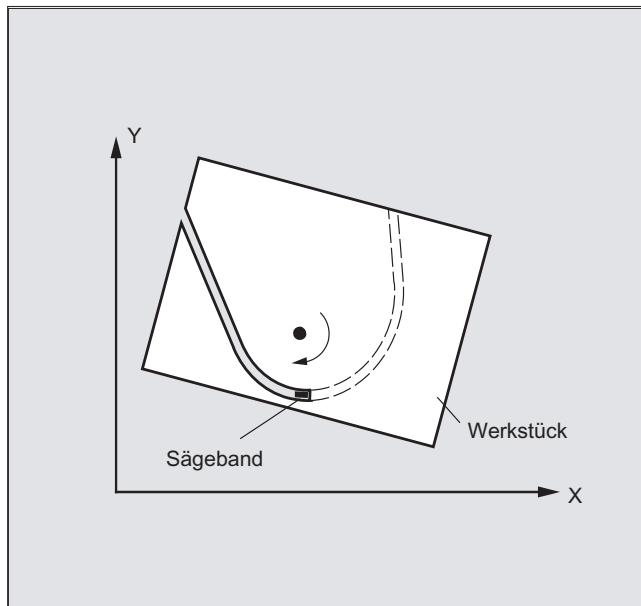
Die Folgeachse wird gemäß der Tangente an der durch die Leitachsen festgelegten Bahn nachgeführt. Dadurch kann ein Werkzeug parallel zur Kontur ausgerichtet werden. Durch den in der `TANGON`-Anweisung programmierten Winkel kann das Werkzeug relativ zur Tangente angestellt werden.



Anwendung

Die Tangentialsteuerung kann z. B. eingesetzt werden bei:

- Tangentiellem Anstellen eines drehbaren Werkzeugs beim Nibbeln
- Nachführen der Werkstückausrichtung bei einer Bandsäge (siehe folgende Abbildung)
- Anstellen eines Abrichtwerkzeugs an eine Schleifscheibe
- Anstellen eines Schneidrädchens zur Glas- oder Papierverarbeitung
- Tangentialer Zuführung eines Drahtes beim 5-achsigen Schweißen



Syntax

Tangentiale Nachführung definieren:

TANG (<Folgeachse>, <Leitachse1>, <Leitachse2>, <Koppelfaktor>, <KS>, <Opt>)

Tangentialsteuerung einschalten:

TANGON (<Folgeachse>, <Winkel>, <Dist>, <Winkeltoleranz>)

Tangentialsteuerung ausschalten:

TANGOF (<Folgeachse>)

Funktion "Zwischensatz an Konturrecken einfügen" einschalten:

TLIFT (<Folgeachse>)

Die TLIFT-Anweisung wird im Anschluss an die Achsenzuordnung mit TANG (...) angegeben.

Funktion "Zwischensatz an Konturrecken einfügen" ausschalten:

TANG (...) -Anweisung wiederholen ohne folgendes TLIFT (<Folgeachse>).

Definition einer Tangentialen Nachführung löschen:

TANGDEL (<Folgeachse>)

Eine bestehende anwenderdefinierte Tangentiale Nachführung muss gelöscht werden, wenn eine neue Tangentiale Nachführung mit der gleichen Folgeachse im Vorbereitungsauftrag TANG definiert werden soll. Ein Löschen ist nur möglich, wenn die Kopplung mit TANGOF (<Folgeachse>) ausgeschaltet ist.

Bedeutung

TANG:	Vorbereitende Anweisung für die Definition einer tangentialen Nachführung
TANGON:	Tangentialsteuerung für die angegebene Folgeachse einschalten
TANGOF:	Tangentialsteuerung für die angegebene Folgeachse ausschalten
TLIFT:	Funktion "Zwischensatz an Konturrecken einfügen" einschalten
TANGDEL:	Definition einer Tangentialen Nachführung löschen
<Folgeachse>:	Folgeachse: Tangential nachgeführte Zusatzrundachse
<Leitachse1>, <Leitachse2>:	Leitachsen: Bahnachsen, aus denen die Tangente für die Nachführung bestimmt wird
<Koppelfaktor>:	Koppelfaktor: Zusammenhang zwischen Winkeländerung der Tangente und der nachgeführten Achse Voreinstellung: 1 Hinweis: Ein Koppelfaktor von 1 muss nicht explizit programmiert werden.
<KS>:	Kennbuchstabe für Koordinatensystem "B": Basiskoordinatensystem (Voreinstellung) Hinweis: <KS> = "B" muss nicht explizit programmiert werden. "W": Werstückkoordinatensystem (nicht verfügbar)
<Opt>:	Optimierung "S": Standard (Voreinstellung) Hinweis: <Opt> = "S" muss nicht explizit programmiert werden. "P": Automatische Anpassung des Zeitverlaufs der tangentialen Achse und der Kontur Hinweis: Mit <Opt> = "P" wird die Dynamik der Folgeachse bei der Geschwindigkeitsbegrenzung der Leitachsen mitberücksichtigt. Diese Einstellung ist vor allem beim Einsatz von kinematischen Transformationen zu empfehlen.
<Winkel>:	Offsetwinkel der Folgeachse
<Dist>:	Überschleifweg der Folgeachse (erforderlich bei <Opt> = "P")
<Winkeltoleranz>:	Winkeltoleranz der Folgeachse (optional; Auswertung nur bei <Opt> = "P") Hinweis: Die Parameter <Dist> und <Winkeltoleranz> begrenzen gezielt den Fehler zwischen der nachgeführten Achse und der Tangente der Leitachsen.

Beispiele

Beispiel 1: Tangentiale Nachführung definieren und einschalten

Programmcode	Kommentar
N10 TANG(C,X,Y,1,"B","P")	; Definition einer tangentialen Nachführung: Rundachse C soll den Geometrieachsen X und Y folgen.
N20 TANGON(C,90)	; Die C-Achse ist Folgeachse. Sie wird bei jeder Bewegung der Bahnachsen in eine 90°-Position zur Bahn-Tangente gedreht.
...	

Hinweis

Vereinfachte Programmierung

TANG(C,X,Y,1,"B","P") kann vereinfacht programmiert werden als TANG(C,X,Y,,,"P").

Beispiel 2: Ebenenwechsel

Programmcode	Kommentar
N10 TANG(A,X,Y,1)	; 1.Definition der Tangentialen Nachführung.
N20 TANGON(A)	; Aktivierung der Kopplung.
N30 X10 Y20	; Radius
...	
N80 TANGOF(A)	; Ausschalten der 1.Kopplung.
N90 TANGDEL(A)	; Löschen der 1.Definition.
...	
TANG(A,X,Z)	; 2.Definition der Tangentialen Nachführung.
TANGON(A)	; Aktivierung der neuen Kopplung.
...	
N200 M30	

Beispiel 3: Geometrieachsumschaltung und TANGDEL

Es wird kein Alarm erzeugt.

Programmcode	Kommentar
N10 GEOAX(2,Y1)	; Y1 ist Geometrieachse 2.
N20 TANG(A,X,Y)	; 1.Definition der Tangentialen Nachführung.
N30 TANGON(A,90)	; Aktivierung der Nachführung mit Y1
N40 G2 F8000 X0 Y0 I0 J50	
N50 TANGOF(A)	; Deaktivierung der Nachführung mit Y1.
N60 TANGDEL(A)	; Löschen der 1.Definition.
N70 GEOAX(2,Y2)	; Y2 ist neue Geometrieachse 2.
N80 TANG(A,X,Y)	; 2.Definition der Tangentialen Nachführung.
N90 TANGON(A,90)	; Aktivierung der Nachführung mit Y2.
...	

Beispiel 4: Tangentiale Nachführung mit automatischer Optimierung

Y1 ist Geometrieachse 2.

Programmcode	Kommentar
...	
N80 G0 C0	
N100 F=50000	
N110 G1 X1000 Y500	
N120 TRAORI	
N130 G642	; Überschleifen unter Einhaltung der maximal erlaubten Bahnabweichung.
N171 TRANS X50 Y50	
N180 TANG(C,X,Y,1,, "P")	; Definition Tangentiale Nachführung mit automatischer Optimierung der Bahngeschwindigkeit.
N190 TANGON(C,0,5.0,2.0)	; Tangentiale Nachführung mit automatischer Optimierung einschalten: Überschleifweg 5 mm, Winkeltoleranz 2 Grad.
N210 G1 X1310 Y500	
N215 G1 X1420 Y500	
N220 G3 X1500 Y580 I=AC(1420) J=AC(580)	
N230 G1 X1500 Y760	
N240 G3 X1360 Y900 I=AC(1360) J=AC(760)	
N250 G1 X1000 Y900	
N280 TANGOF(C)	
N290 TRAFOOF	
N300 M02	

Weitere Informationen

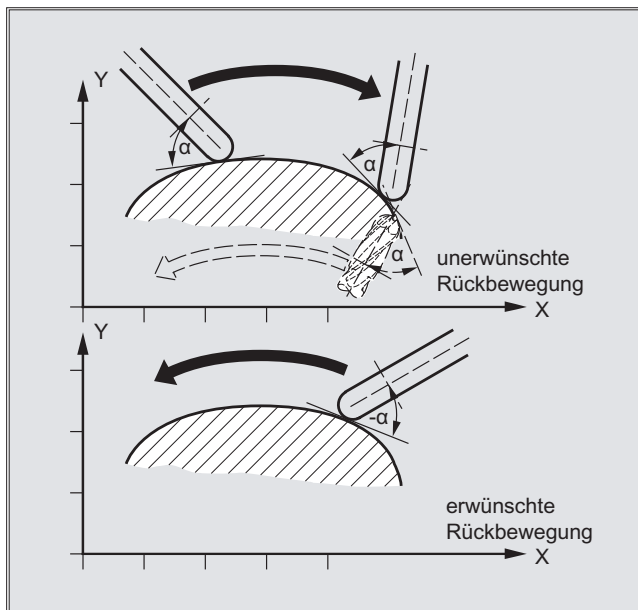
Folge- und Leitachse definieren

Die Definition von Folge- und Leitachsen erfolgt mit TANG.

Ein Koppelfaktor gibt den Zusammenhang zwischen einer Winkeländerung der Tangente und der nachgeführten Achse an. Sein Wert beträgt in der Regel 1 (Voreinstellung).

Grenzwinkel durch Arbeitsfeldbegrenzung

Bei hin- und hergeführten Bahnbewegungen springt die Tangente im Umkehrpunkt der Bahn um 180° um, entsprechend ändert sich die Ausrichtung der Folgeachse. In der Regel ist dieses Verhalten nicht sinnvoll: Die Rückbewegung soll im gleichen negativen Offsetwinkel wie die Hinbewegung abgefahren werden:



Dazu muss das Arbeitsfeld der Folgeachse begrenzt werden (G25, G26). Die Arbeitsfeldbegrenzung muss zum Zeitpunkt der Bahnumkehr aktiv sein (WALIMON). Liegt der Offsetwinkel außerhalb der Arbeitsfeldbegrenzung, wird versucht, mit negativem Offsetwinkel wieder in den zulässigen Arbeitsbereich zu kommen.

Zwischensatz an Konturecken einfügen (TLIFT)

An einer Ecke der Kontur ändert sich die Tangente und damit die Sollposition der nachgeführten Achse sprunghaft. Die Achse versucht normalerweise, diesen Sprung mit ihrer maximal möglichen Geschwindigkeit auszugleichen. Dabei ergibt sich jedoch über eine gewisse Strecke auf der Kontur nach der Ecke eine Abweichung zur gewünschten tangentiellen Anstellung. Wenn dies aus technologischen Gründen nicht tolerierbar ist, kann mit der Anweisung TLIFT die Steuerung dazu veranlasst werden, an der Ecke anzuhalten und in einem automatisch erzeugten Zwischensatz die nachgeführte Achse in die neue Tangentenrichtung zu drehen.

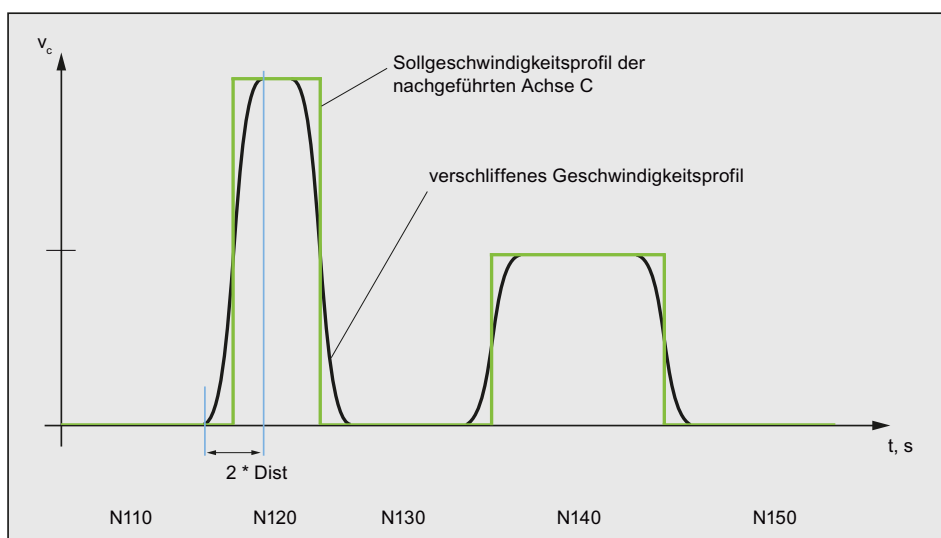
Die Drehung erfolgt mit der programmierten Bahnachse, wenn die nachgeführte Achse einmal als Bahnachse gefahren wurde. Durch die Funktion TFGREF [<Achse>]=0.001 kann hier eine maximale Achsgeschwindigkeit der nachgeführten Achse erreicht werden.

Wurde die nachgeführte Achse bisher nicht als Bahnachse verfahren, so wird diese Achse als Positionierachse verfahren. Die Geschwindigkeit ist dann abhängig von der im Maschinendatum hinterlegten Positioniergeschwindigkeit.

Die Drehung erfolgt mit der maximalen Geschwindigkeit der nachgeführten Achse.

Optimierungsmöglichkeit

Ist die automatische Optimierung angewählt (<Opt> = "P") und sind für die Folgeachse die Parameter Überschleifweg (<Dist>) und Winkeltoleranz (<Winkeltoleranz>) angegeben, dann werden beim Tangentialen Nachführen Geschwindigkeitssprünge der Folgeachse infolge von Sprüngen in der Leitachskontur überschleift bzw. geglättet. Dabei wird die Folgeachse vorausschauend geführt (siehe Diagramm), um die Abweichung möglichst klein zu halten.



Winkeländerung definieren

Die Winkeländerung, ab der ein automatischer Zwischensatz eingeführt wird, wird über das folgende Maschinendatum definiert:

MD37400 \$MA_EPS_TLIFT_TANG_STEP (Tangentenwinkel für Eckenerkennung)

Einfluss auf Transformationen

Die Position der nachgeführten Rundachse kann Eingangswert für eine Transformation sein.

Explizite Positionierung der Folgeachse

Wird eine ihren Leitachsen nachgeführte Folgeachse explizit positioniert, so wirkt die Positionsangabe additiv zum programmierten Offsetwinkel.

Zulässig sind alle Wegvorgaben (Bahn- und Positionierachsbewegungen).

Status der Kopplung

Im NC-Teileprogramm kann der Status der Kopplung mit der Systemvariablen \$AA_COUP_ACT[<Achse>] abgefragt werden:

Wert	Bedeutung
0	Keine Kopplung aktiv
1,2,3	Tangentiales Nachführen aktiv

8.2 Vorschubverlauf (FNORM, FLIN, FCUB, FPO)

Funktion

Zur flexibleren Vorgabe des Vorschubverlaufs wird die Vorschubprogrammierung nach DIN 66025 um lineare und kubische Verläufe erweitert.

Die kubischen Verläufe können direkt oder als interpolierende Splines programmiert werden. Hierdurch lassen sich - abhängig von der Krümmung des zu bearbeitenden Werkstücks - kontinuierlich glatte Geschwindigkeitsverläufe programmieren.

Diese Geschwindigkeitsverläufe ermöglichen ruckfreie Beschleunigungsänderungen und hierdurch Fertigung gleichmäßiger Werkstückoberflächen.

Syntax

F... FNORM
F... FLIN
F... FCUB
F=FPO (... , ... , ...)

Bedeutung

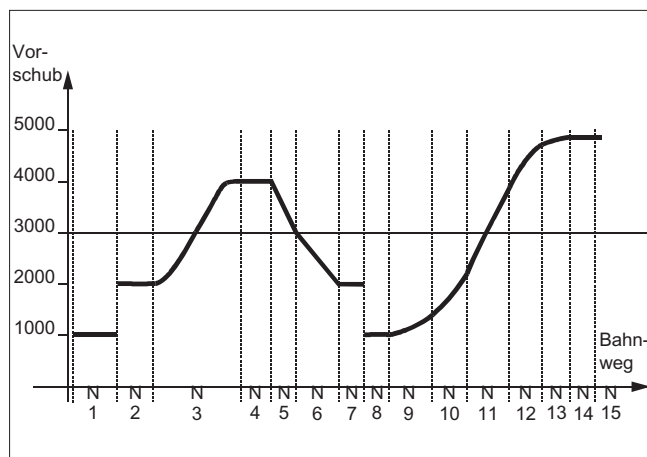
- FNORM** Grundeinstellung. Der Vorschubwert wird über den Bahnweg des Satzes vorgegeben und gilt danach als modaler Wert.
- FLIN** Bahngeschwindigkeitsprofil **linear**:
Der Vorschubwert wird vom aktuellen Wert am Satzanfang bis zum Satzende über den Bahnweg linear eingefahren und gilt danach als modaler Wert. Dieses Verhalten kann mit G93 und G94 kombiniert werden.
- FCUB** Bahngeschwindigkeitsprofil **kubisch**:
Die satzweise programmierten F-Werte werden - bezogen auf den Satzende - durch einen Spline verbunden. Der Spline beginnt und endet tangential zur vorhergehenden bzw. nachfolgenden Vorschubangabe und wirkt mit G93 und G94. Fehlt in einem Satz die F-Adresse, so wird hierfür der zuletzt programmierte F-Wert verwendet.
- F=FPO...** Bahngeschwindigkeitsprofil **über Polynom**:
Die F-Adresse bezeichnet den Vorschubverlauf über ein Polynom vom aktuellen Wert bis zum Satzende. Der Endwert gilt danach als modaler Wert.

Vorschuboptimierung bei gekrümmten Bahnstücken

Vorschub-Polynom $F=FPO$ und Vorschubspline $FCUB$ sollten immer mit konstanter Schnittgeschwindigkeit CFC abgefahren werden. Hierdurch lässt sich ein beschleunigungsstetiges Sollvorschubprofil erzeugen.

Beispiel: Verschiedene Vorschubprofile

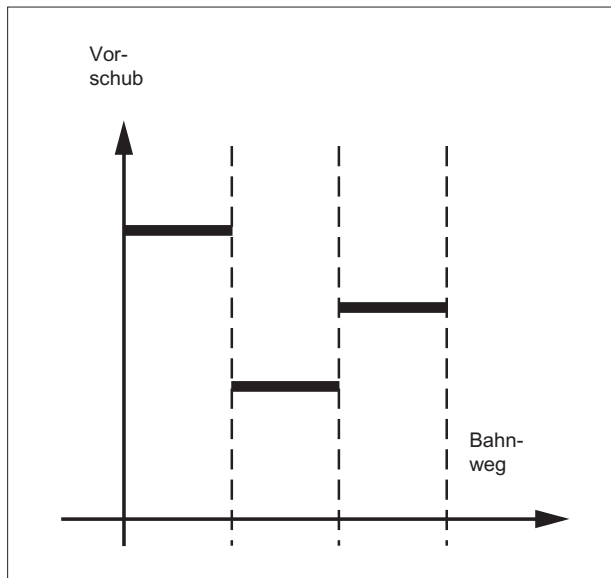
In diesem Beispiel finden Sie die Programmierung und grafische Darstellung verschiedener Vorschubprofile.



Programmcode	Kommentar
N1 F1000 FNORM G1 X8 G91 G64	; Konstantes Vorschubprofil, Kettenmaßangabe
N2 F2000 X7	; Sprunghafte Sollgeschwindigkeitsänderung
N3 F=FPO(4000, 6000, -4000)	; Vorschubprofil über Polynom mit Vorschub 4000 am Satzende
N4 X6	; Polynomvorschub 4000 gilt als modaler Wert
N5 F3000 FLIN X5	; Lineares Vorschubprofil
N6 F2000 X8	; Lineares Vorschubprofil
N7 X5	Linearer Vorschub gilt als modaler Wert
N8 F1000 FNORM X5	; Konstantes Vorschubprofil mit sprunghafter Beschleunigungsänderung
N9 F1400 FCUB X8	; Alle folgenden satzweise programmierten F-Werte werden mit Splines verbunden
N10 F2200 X6	
N11 F3900 X7	
N12 F4600 X7	
N13 F4900 X5	; Splineprofil ausschalten
N14 FNORM X5	
N15 X20	

FNORM

Die Vorschubadresse F bezeichnet den Bahnvorschub als konstanten Wert nach DIN 66025. Mehr Informationen hierzu finden Sie im Programmierhandbuch "Grundlagen".

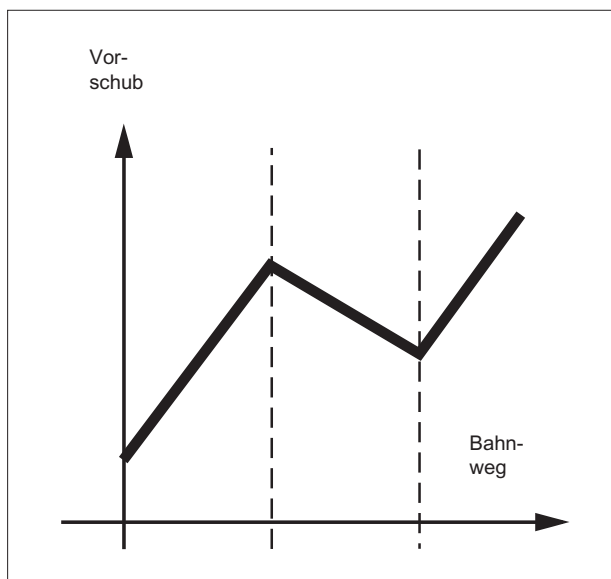


FLIN

Der Vorschubverlauf wird vom aktuellen Vorschubwert zum programmierten F-Wert linear bis Satzende eingefahren.

Beispiel:

N30 F1400 FLIN X50



FCUB

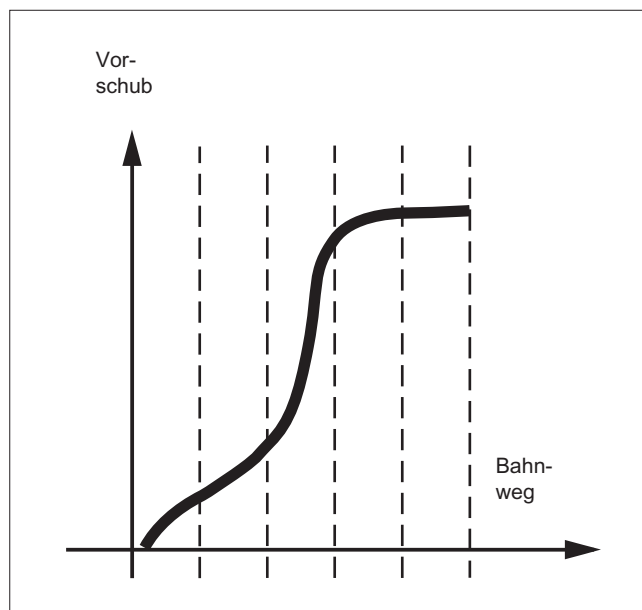
Der Vorschub wird vom aktuellen Vorschubwert zum programmierten F-Wert bis Satzende im kubischen Verlauf eingefahren. Die Steuerung verbindet alle mit aktivem FCUB satzweise programmierten Vorschubwerte durch Splines. Die Vorschubwerte dienen hier als Stützpunkte zur Berechnung der Splineinterpolation.

Beispiel:

N50 F1400 FCUB X50

N60 F2000 X47

N70 F3800 X52



F=FPO(.....)

Der Vorschubverlauf wird über ein Polynom direkt programmiert. Die Angabe der Polynomkoeffizienten erfolgt analog zur Polynominterpolation.

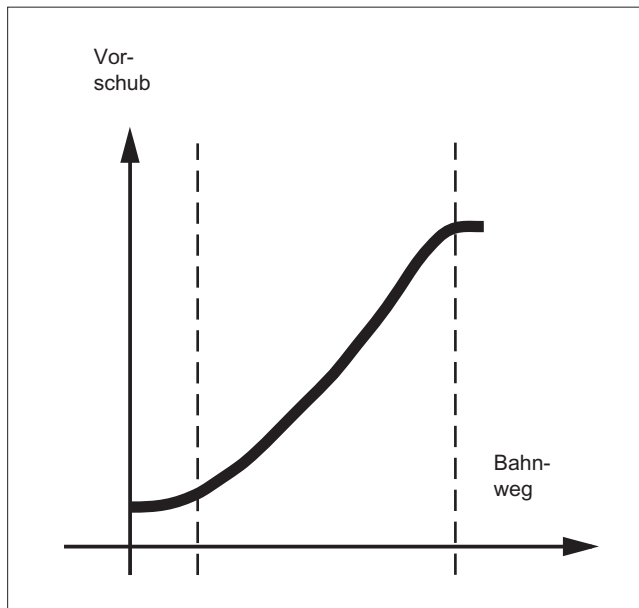
Beispiel:

F=FPO(endfeed, quadf, cubf)

endfeed, quadf und cubf sind vorher definierte Variable.

endfeed:	Vorschub am Satzende
quadf:	Quadratischer Polynomkoeffizient
cubf:	Kubischer Polynomkoeffizient

Bei aktivem FCUB schließt der Spline am Satzanfang und Satzende tangential an den über FPO festgelegten Verlauf an.



Randbedingungen

Unabhängig vom programmierten Vorschubverlauf gelten die Funktionen zur Programmierung des Bahnfahrverhaltens.

Der programmierbare Vorschubverlauf gilt grundsätzlich absolut - unabhängig von G90 oder G91.

Der Vorschubverlauf FLIN und FCUB wirkt mit

G93 und G94.

FLIN und FCUB **wirkt nicht bei**

G95, G96/G961 und G97/G971.

Aktiver Kompressor COMPON

Bei aktivem Kompressor COMPON gilt bei Zusammenfassung mehrerer Sätze zu einem Splinesegment:

FNORM:

Für das Splinesegment gilt das F-Wort des letzten zugehörigen Satzes.

FLIN:

Für das Splinesegment gilt das F-Wort des letzten zugehörigen Satzes.

Der programmierte F-Wert gilt zum Ende des Segments und wird dann linear angefahren.

FCUB:

Der erzeugte Vorschubspline weicht maximal um den im Maschinendatum MD20172 \$MC_COMPRESS_VELO_TOL definierten Wert von den programmierten Endpunkten ab.

F=FPO(.....)

Diese Sätze werden nicht komprimiert.

8.3 Beschleunigungsverhalten

8.3.1 Beschleunigungsmodus (BRISK, BRISKA, SOFT, SOFTA, DRIVE, DRIVEA)

Funktion

Zur Programmierung des Beschleunigungsmodus stehen folgende Teileprogrammbeefehle zur Verfügung:

- BRISK, BRISKA

Die Einzelachsen bzw. die Bahnachsen verfahren mit maximaler Beschleunigung bis zum Erreichen der programmierten Vorschubgeschwindigkeit (**Beschleunigung ohne Ruckbegrenzung**).

- SOFT, SOFTA

Die Einzelachsen bzw. die Bahnachsen verfahren mit stetiger Beschleunigung bis zum Erreichen der programmierten Vorschubgeschwindigkeit (**Beschleunigung mit Ruckbegrenzung**).

- DRIVE, DRIVEA

Die Einzelachsen bzw. die Bahnachsen verfahren mit maximaler Beschleunigung bis zu einer projektierten Geschwindigkeitsgrenze (MD-Einstellung!). Danach erfolgt eine Beschleunigungsreduktion (MD-Einstellung!) bis zum Erreichen der programmierten Vorschubgeschwindigkeit.

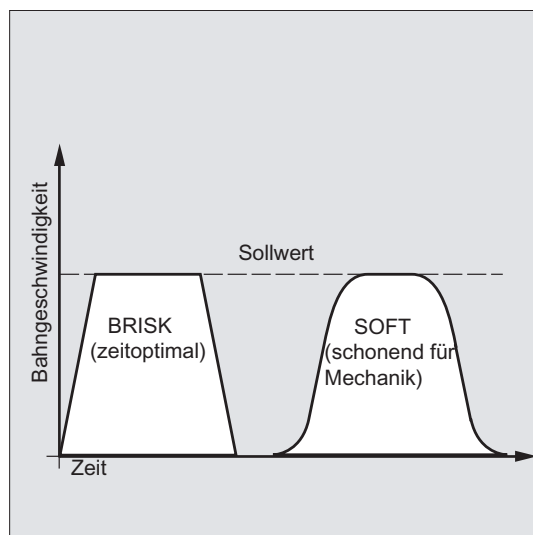


Bild 8-1 Verlauf der Bahnbeschwindigkeit bei BRISK und SOFT

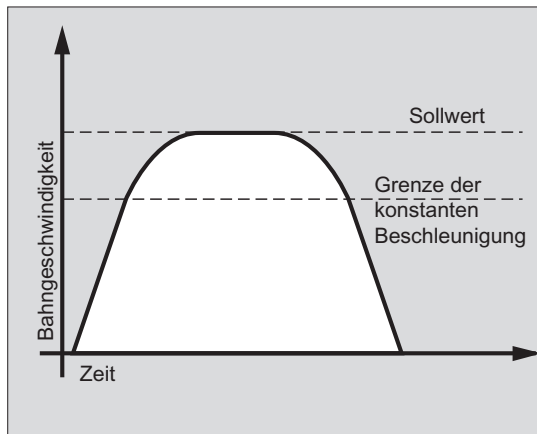


Bild 8-2 Verlauf der Bahngeschwindigkeit bei DRIVE

Syntax

```
BRISK  
BRISKA (<Achse1>, <Achse2>, ...)  
SOFT  
SOFTA (<Achse1>, <Achse2>, ...)  
DRIVE  
DRIVEA (<Achse1>, <Achse2>, ...)
```

Bedeutung

BRISK:	Befehl zum Einschalten der "Beschleunigung ohne Ruckbegrenzung" für die Bahnachsen.
BRISKA:	Befehl zum Einschalten der "Beschleunigung ohne Ruckbegrenzung" für Einzelachs-bewegungen (JOG, JOG/INC, Positionierachse, Pendelachse, etc.).
SOFT:	Befehl zum Einschalten der "Beschleunigung mit Ruckbegrenzung" für die Bahnachsen.
SOFTA:	Befehl zum Einschalten der "Beschleunigung mit Ruckbegrenzung" für Einzelachs-bewegungen (JOG, JOG/INC, Positionierachse, Pendelachse, etc.).
DRIVE:	Befehl zum Einschalten der reduzierten Beschleunigung oberhalb einer projektierten Geschwindigkeitsgrenze (MD35220 \$MA_ACCEL_REDUCTION_SPEED_POINT) für die Bahnachsen.
DRIVEA:	Befehl zum Einschalten der reduzierten Beschleunigung oberhalb einer projektierten Geschwindigkeitsgrenze (MD35220 \$MA_ACCEL_REDUCTION_SPEED_POINT) für Einzelachs-bewegungen (JOG, JOG/INC, Positionierachse, Pendelachse, etc.).
(<Achse1>, <Achse2>, ...):	Einzelachsen, für die der aufgerufene Beschleunigungsmodus gelten soll.

Randbedingungen

Wechsel des Beschleunigungsmodus während Bearbeitung

Wenn in einem Teileprogramm der Beschleunigungsmodus während der Bearbeitung gewechselt wird (BRISK ↔ SOFT), dann erfolgt auch bei Bahnsteuerbetrieb am Übergang ein Satzwechsel mit Genauhalt am Satzende.

Beispiele

Beispiel 1: SOFT und BRISKA

Programmcode

```
N10 G1 X... Y... F900 SOFT  
N20 BRISKA (AX5, AX6)  
...
```

Beispiel 2: DRIVE und DRIVEA

Programmcode

```
N05 DRIVE  
N10 G1 X... Y... F1000  
N20 DRIVEA (AX4, AX6)  
...
```

Literatur

Funktionshandbuch Grundfunktionen; Beschleunigung (B2)

8.3.2 Beeinflussung der Beschleunigung bei Folgeachsen (VELOLIMA, ACCLIMA, JERKLIMA)

Funktion

Bei Achskopplungen (Tangentiale Nachführung, Mitschleppen, Leitwertkopplung, Elektronisches Getriebe; siehe "Achskopplungen (Seite 515)") werden Folgeachsen/-spindeln abhängig von einer oder mehreren Leitachsen/-spindeln verfahren.

Die Dynamikbegrenzungen der Folgeachsen/-spindeln können mit den Funktionen VELOLIMA, ACCLIMA und JERKLIMA aus dem Teileprogramm oder aus Synchronaktionen beeinflusst werden, auch bei bereits aktiver Achskopplung.

Hinweis

Die Funktion JERKLIMA ist nicht für alle Kopplungsarten verfügbar.

Literatur:

- Funktionshandbuch Sonderfunktionen; Achskopplungen (M3)
- Funktionshandbuch Erweiterungsfunktionen; Synchronspindel (S3)

Hinweis

Verfügbarkeit bei SINUMERIK 828D

Die Funktionen VELOLIMA, ACCLIMA und JERKLIMA können bei SINUMERIK 828D nur in Verbindung mit der Funktion "Mitschleppen" verwendet werden!

Syntax

```
VELOLIMA (<Achse>) =<Wert>  
ACCLIMA (<Achse>) =<Wert>  
JERKLIMA (<Achse>) =<Wert>
```

Bedeutung

VELOLIMA:	Befehl zur Korrektur der parametrisierten Maximal geschwindigkeit
ACCLIMA:	Befehl zur Korrektur der parametrisierten Maximal beschleunigung
JERKLIMA:	Befehl zur Korrektur des parametrisierten Maximal rucks
<Achse>:	Folgeachse, deren Dynamikbegrenzungen korrigiert werden sollen
<Wert>:	Prozentualer Korrekturwert

Beispiele

Beispiel 1: Korrektur der Dynamikbegrenzungen für eine Folgeachse (AX4)

Programmcode	Kommentar
...	
VELOLIMA [AX4] =75	; Begrenzungskorrektur auf 75% der im Maschinendatum hinterlegten axialen Maximalgeschwindigkeit.
ACCLIMA [AX4] =50	; Begrenzungskorrektur auf 50% der im Maschinendatum hinterlegten axialen Maximalbeschleunigung.
JERKLIMA [AX4] =50	; Begrenzungskorrektur auf 50% des im Maschinendatum hinterlegten axialen Maximalrucks bei Bahnbewegung.
...	

Beispiel 2: Elektronisches Getriebe

Achse 4 wird über eine Kopplung "Elektronisches Getriebe" an Achse X gekoppelt. Das Beschleunigungsvermögen der Folgeachse wird auf 70 % der maximalen Beschleunigung begrenzt. Die maximal zulässige Geschwindigkeit wird auf 50 % der maximalen Geschwindigkeit begrenzt. Nach erfolgter Einschaltung der Kopplung wird die maximal zulässige Geschwindigkeit wieder auf 100 % gesetzt.

Programmcode	Kommentar
...	
N120 ACCLIMA [AX4] =70	; Reduzierte maximale Beschleunigung.
N130 VELOLIMA [AX4] =50	; Reduzierte maximale Geschwindigkeit.
...	
N150 EGON (AX4, "FINE", X, 1, 2)	; Einschalten der EG-Kopplung.
...	
N200 VELOLIMA [AX4] =100	; Volle Maximalgeschwindigkeit.
...	

Beispiel 3: Leitwertkopplung per statische Synchronaktion beeinflussen

Achse 4 wird mittels Leitwertkopplung an X gekoppelt. Das Beschleunigungsverhalten wird per statische Synchronaktion 2 ab Position 100 auf 80 % begrenzt.

Programmcode	Kommentar
...	
N120 IDS=2 WHENEVER \$AA_IM [AX4] > 100 DO ACCLIMA [AX4] =80	; Synchronaktion
N130 LEADON (AX4, X, 2)	; Leitwertkopplung ein
...	

8.3.3 Aktivierung von Technologie-spezifischen Dynamikwerten (DYNNORM, DYNPOS, DYNROUGH, DYNSEMIFIN, DYNFINISH)

Funktion

Mittels der G-Gruppe "Technologie" kann für 5 unterschiedliche technologische Bearbeitungsschritte die dazu passende Dynamik aktiviert werden.

Dynamikwerte und G-Befehle sind projektierbar und damit von Maschinendateneinstellungen abhängig (→ Maschinenhersteller!).

Literatur:

Funktionshandbuch Grundfunktionen; Bahnsteuerbetrieb, Genauhalt, LookAhead (B1)

Syntax

Dynamikwerte aktivieren:

DYNNORM
DYNPOS
DYNROUGH
DYNSEMIFIN
DYNFINISH

Hinweis

Die Dynamikwerte werden bereits in dem Satz wirksam, in dem der zugehörige G-Befehl programmiert wird. Es folgt kein Bearbeitungsstopp.

Bestimmtes Feldelement lesen oder schreiben:

R<m>=\$MA... [n, X]
\$MA... [n, X] =<Wert>

Bedeutung

DYNNORM:	G-Befehl zur Aktivierung der normalen Dynamik
DYNPOS:	G-Befehl zur Aktivierung der Dynamik für Positionierbetrieb, Gewindebohren
DYNROUGH:	G-Befehl zur Aktivierung der Dynamik für Schruppen
DYNSEMIFIN:	G-Befehl zur Aktivierung der Dynamik für Schlichten
DYNFINISH:	G-Befehl zur Aktivierung der Dynamik für Feinschlichten
R<m>:	Rechenparameter mit Nummer <m>
\$MA... [n, X]:	Maschinendatum mit dynamikbestimmendem Feldelement

<n>: Feldindex
 Wertebereich: 0 ... 4
 0 Normale Dynamik (DYNNORM)
 1 Dynamik für Positionierbetrieb (DYNPOS)
 2 Dynamik für Schruppen (DYNROUGH)
 3 Dynamik für Schlichten (DYNSEMIFIN)
 4 Dynamik für Feinschlichten (DYNFINISH)

<X> : Achsadresse
 <Wert>: Dynamikwert

Beispiele

Beispiel 1: Dynamikwerte aktivieren

Programmcode	Kommentar
DYNNORM G1 X10	; Grundstellung
DYNPOS G1 X10 Y20 Z30 F...	; Positionierbetrieb, Gewindebohren
DYNROUGH G1 X10 Y20 Z30 F10000	; Schruppen
DYNSEMIFIN G1 X10 Y20 Z30 F2000	; Schlichten
DYNFINISH G1 X10 Y20 Z30 F1000	; Feinschlichten

Beispiel 2: Bestimmtes Feldelement lesen oder schreiben

Maximale Beschleunigung für das Schruppen, Achse X.

Programmcode	Kommentar
R1=\$MA_MAX_AX_ACCEL[2,X]	; Lesen
\$MA_MAX_AX_ACCEL[2,X]=5	; Schreiben

8.4 Fahren mit Vorsteuerung (FFWON, FFWOF)

Funktion

Durch die Vorsteuerung wird der geschwindigkeitsabhängige Nachlaufweg beim Bahnfahren gegen Null reduziert. Fahren mit Vorsteuerung ermöglicht höhere Bahngenauigkeit und damit bessere Fertigungsergebnisse.

Syntax

FFWON

FFWOF

Bedeutung

FFWON: Befehl zum **E**inschalten der Vorsteuerung

FFWOF: Befehl zum **A**usschalten der Vorsteuerung

Hinweis

Über Maschinendaten wird die Art der Vorsteuerung festgelegt und welche Bahnachsen vorgesteuert verfahren werden sollen.

Standard: Geschwindigkeitsabhängige Vorsteuerung

Option: Beschleunigungsabhängige Vorsteuerung

Beispiel

Programmcode

N10 FFWON

N20 G1 X... Y... F900 SOFT

8.5 Programmierbare Konturgenauigkeit (CPRECON, CPRECOF)

Funktion

Die Funktion "Programmierbare Konturgenauigkeit" reduziert den Bahnfehler an gekrümmten Konturen durch automatische Anpassung der Geschwindigkeit.

Die einzuhaltende Konturgenauigkeit wird abhängig von der Projektierung der Maschine (MD20470 \$MC_MC_CPREC_WITH_FFW; siehe Angaben des Maschinenherstellers) entweder über das Settingdatum \$SC_CONTPREC oder über die programmierte Konturtoleranz CTOL vorgegeben. Je kleiner der Wert und je kleiner der K_V -Faktor der Geometrieachsen, umso stärker wird der Bahnvorschub auf gekrümmten Konturen abgesenkt.

Die Funktion "Programmierbare Konturgenauigkeit" wird über die Anweisungen `CPRECON` und `CPRECOF` im NC-Programm ein- bzw. ausgeschaltet.

Syntax

```
CPRECON
...
CPRECOF
```

Bedeutung

<code>CPRECON:</code>	G-Funktionsaufruf: "Programmierbare Konturgenauigkeit" einschalten Wirksamkeit: modal
<code>CPRECOF:</code>	G-Funktionsaufruf: "Programmierbare Konturgenauigkeit" ausschalten Wirksamkeit: modal

`CPRECON` und `CPRECOF` bilden zusammen die G-Funktionsgruppe 39 (Programmierbare Konturgenauigkeit).

Hinweis

Über das Settingdatum \$SC_MINFEED (Mindestbahnvorschub bei `CPRECON`) kann der Anwender eine Mindestgeschwindigkeit für den Bahnvorschub vorgeben.

Der Vorschub wird nicht unter diesen Wert begrenzt, es sei denn, ein niedrigerer F-Wert wurde programmiert oder die dynamischen Begrenzungen der Achsen erzwingen eine niedrigere Bahngeschwindigkeit.

Hinweis

Die Funktion "Programmierbare Konturgenauigkeit" betrachtet nur die Geometrieachsen der Bahn. Auf die Geschwindigkeiten von Positionierachsen hat sie keinen Einfluss.

Beispiel

Programmcode	Kommentar
N10 G0 X0 Y0	
N20 CPRECON	; Einschalten der "Programmierbaren Konturgenauigkeit".
N30 G1 G64 X100 F10000	; Bearbeitung mit 10 m/min im Bahnsteuerbetrieb.
N40 G3 Y20 J10	; Automatische Vorschubbegrenzung im Kreissatz.
N50 G1 X0	; Vorschub wieder ohne Begrenzung (10 m/min).
...	
N100 CPRECOF	; Ausschalten der "Programmierbaren Konturgenauigkeit".
N110 G0 ...	

Literatur

Zur Programmierung von CTOL siehe "Programmierbare Kontur-/Orientierungstoleranz (CTOL, OTOL, ATOL) (Seite 509)"

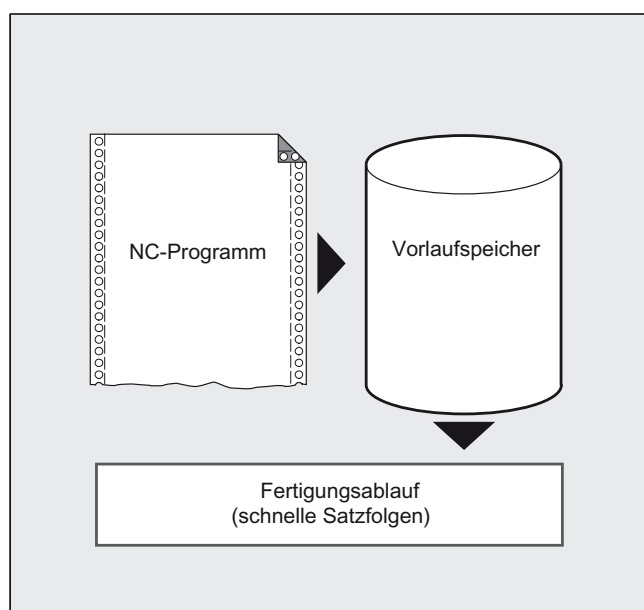
Für ausführlichere Informationen zur Funktion "Programmierbare Konturgenauigkeit" siehe:

Funktionshandbuch Sonderfunktionen; Konturtunnel-Überwachung (K6), Kapitel: "Programmierbare Konturgenauigkeit"

8.6 Programmablauf mit Vorlaufspeicher (STOPFIFO, STARTFIFO, FIFCTRL, STOPRE)

Funktion

Je nach Ausbaustufe verfügt die Steuerung über eine bestimmte Menge sog. Vorlaufspeicher, die fertig aufbereitete Sätze vor der Abarbeitung speichern und im Fertigungsablauf als schnelle Satzfolgen ausgeben. Hierdurch lassen sich kurze Wege mit hohen Geschwindigkeiten abfahren. Soweit die Restzeit der Steuerung es zulässt, wird der Vorlaufspeicher grundsätzlich gefüllt.



Bearbeitungsabschnitt kennzeichnen

Der Bearbeitungsabschnitt, der im Vorlaufspeicher zwischengespeichert werden soll, wird im Teileprogramm am Anfang mit `STOPFIFO` und am Ende mit `STARTFIFO` gekennzeichnet. Die Abarbeitung der aufbereiteten und zwischengespeicherten Sätze beginnt erst nach dem Befehl `STARTFIFO` oder wenn der Vorlaufspeicher voll ist.

Automatische Vorlaufspeichersteuerung

Die automatische Vorlaufspeichersteuerung wird mit dem Befehl `FIFCTRL` aufgerufen. `FIFCTRL` wirkt zunächst genauso wie `STOPFIFO`. Bei jeder Programmierung wird gewartet, bis der Vorlaufspeicher voll ist, dann beginnt die Abarbeitung. Unterschiedlich ist dagegen das Verhalten beim Leerlaufen des Vorlaufspeichers: mit `FIFCTRL` wird ab einem Füllstand von 2/3 die Bahngeschwindigkeit zunehmend reduziert, um ein komplettes Leerlaufen und ein Abbremsen bis zum Stillstand zu verhindern.

Vorlaufstopp

Die Satzaufbereitung und -zwischenspeicherung wird angehalten, wenn im Satz der Befehl `STOPRE` programmiert ist. Der nachfolgende Satz wird erst dann ausgeführt, wenn alle vorher aufbereiteten und gespeicherten Sätze vollständig abgearbeitet sind. Der vorherige Satz wird im Genauhalt angehalten (wie G9).

Syntax

Tabelle 8- 1 Bearbeitungsabschnitt kennzeichnen:

```
STOPFIFO  
...  
STARTFIFO
```

Tabelle 8- 2 Automatische Vorlaufspeichersteuerung:

```
...  
FIFCTRL  
...
```

Tabelle 8- 3 Vorlaufstopp:

```
...  
STOPRE  
...
```

Hinweis

Die Befehle `STOPFIFO`, `STARTFIFO`, `FIFCTRL` und `STOPRE` müssen im eigenen Satz programmiert werden.

Bedeutung

<code>STOPFIFO:</code>	<code>STOPFIFO</code> kennzeichnet den Beginn eines Bearbeitungsabschnitts, der im Vorlaufspeicher zwischengespeichert werden soll. Mit <code>STOPFIFO</code> wird die Bearbeitung angehalten und der Vorlaufspeicher gefüllt, bis: <ul style="list-style-type: none">• <code>STARTFIFO</code> oder <code>STOPRE</code> erkannt wirdoder• der Vorlaufspeicher voll istoder• das Programmende erreicht ist.
<code>STARTFIFO:</code>	Mit <code>STARTFIFO</code> startet die schnelle Abarbeitung des Bearbeitungsabschnitts, parallel dazu erfolgt das Auffüllen des Vorlaufspeichers
<code>FIFCTRL:</code>	Einschalten der automatischen Vorlaufspeichersteuerung
<code>STOPRE:</code>	Vorlauf stoppen

Hinweis

Das Auffüllen des Vorlaufspeichers wird nicht ausgeführt bzw. unterbrochen, wenn der Bearbeitungsabschnitt Befehle enthält, die einen ungepufferten Betrieb erzwingen (Referenzpunktfahren, Messfunktionen, ...).

Hinweis

Beim Zugriff auf Zustandsdaten der Maschine (\$SA...) erzeugt die Steuerung internen Vorlaufstopp.

**VORSICHT**

Bei eingeschalteter Werkzeugkorrektur und bei Spline-Interpolationen sollte kein `STOPRE` programmiert werden, da sonst zusammengehörige Satzfolgen unterbrochen werden.

Beispiel: Vorlauf stoppen

Programmcode	Kommentar
...	
N30 MEAW=1 G1 F1000 X100 Y100 Z50	; Messsatz mit Messtaster des ersten Messeingangs und Geradeninterpolation.
N40 STOPRE	; Vorlaufstopp.
...	

8.7 Bedingt unterbrechbare Programmabschnitte (DELAYFSTON, DELAYFSTOF)

Funktion

Bedingt unterbrechbare Teileprogrammabschnitte werden Stop-Delay-Bereiche genannt. Innerhalb bestimmter Programmabschnitte soll nicht **angehalten** werden und auch der **Vorschub** nicht verändert werden. Im Wesentlichen sollen kurze Programmabschnitte, die z. B. zur Herstellung eines Gewindes dienen, vor fast allen Stopp-Ereignissen geschützt werden. Ein etwaiger Stopp wirkt erst, nachdem der Programmabschnitt zu Ende bearbeitet worden ist.

Syntax

```
DELAYFSTON  
...  
DELAYFSTOF
```

Hinweis

Die Befehle DELAYFSTON und DELAYFSTOF stehen allein in einer Teileprogrammzeile.

Bedeutung

DELAYFSTON:	Beginn eines Bereichs definieren, in dem "sanfte" Stopps verzögert werden, bis das Ende des Stop-Delay-Bereichs erreicht wird
DELAYFSTOF:	Ende eines Stop-Delay-Bereichs definieren

Hinweis

DELAYFSTON und DELAYFSTOF sind nur in Teileprogrammen, nicht jedoch in Synchronaktionen zulässig.

Hinweis

Bei MD11550 \$MN_STOP_MODE_MASK Bit 0 = 0 (Default) wird ein Stop-Delay-Bereich implizit definiert, wenn G331/G332 aktiv ist und eine Bahnbewegung bzw. G4 programmiert ist.

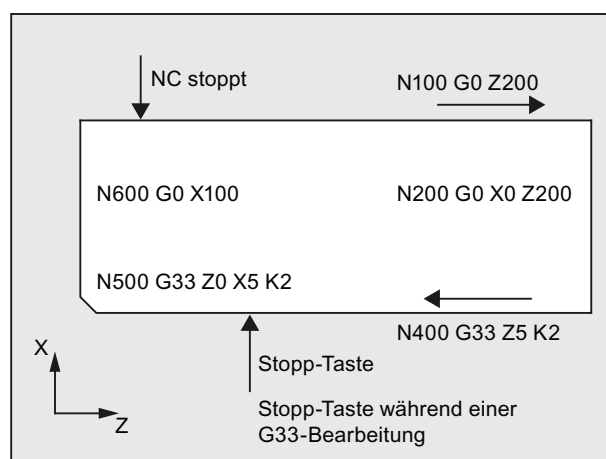
Beispiele

Beispiel 1: Verschachtelung von Stop-Delay-Bereichen in zwei Programmebenen

Programmcode	Kommentar
N10010 DELAYFSTON()	; Sätze mit N10xxx Programmebene 1.
N10020 R1 = R1 + 1	
N10030 G4 F1	; Stop-Delay-Bereich beginnt.
...	
N10040 Unterprogramm2	
...	
...	; Interpretation des Unterprogramms 2.
N20010 DELAYFSTON()	; Unwirksam, wiederholter Beginn, 2. Ebene.
...	
N20020 DELAYFSTOF()	; Unwirksam, Ende in anderer Ebene.
N20030 RET	
N10050 DELAYFSTOF()	; Stop-Delay-Bereichs-Ende in gleicher Ebene.
...	
N10060 R2 = R2 + 2	
N10070 G4 F1	; Stop-Delay-Bereich endet. Stopps wirken ab jetzt unmittelbar.

Beispiel 2

In einer Schleife wird folgender Programmblock wiederholt:



Im Bild ist erkennbar, dass der Anwender im Stop-Delay-Bereich "Stopp" drückt, und die NC beginnt den Bremsvorgang außerhalb des Stop-Delay-Bereichs, d. h. im Satz N100. Damit kommt die NC im vorderen Bereich von N100 zum Halten.

Programmcode
...
N99 MY_LOOP:
N100 G0 Z200

8.7 Bedingt unterbrechbare Programmabschnitte (DELAYFSTON, DELAYFSTOF)

```

Programmcode
N200 G0 X0 Z200
N300 DELAYFSTON()
N400 G33 Z5 K2 M3 S1000
N500 G33 Z0 X5 K3
N600 G0 X100
N700 DELAYFSTOF()
N800 GOTOB MY_LOOP
    
```

Weitere Informationen

Stopp-Ereignisse

Im Stop-Delay-Bereich wird eine Veränderung des Vorschubs und Vorschubsperrung ignoriert. Sie wirken erst nach dem Stop-Delay-Bereich.

Die Stopp-Ereignisse werden unterschieden in:

- "Sanfte" Stopp-Ereignisse Reaktion: delayed
- "Harte" Stopp-Ereignisse Reaktion: immediate

Auswahl einiger Stopp-Ereignisse, die zumindest kurzfristig stoppen:

Ereignisname	Reaktion	Unterbrechungsparameter
RESET	immediate	NST: DB21, ... DBX7.7 und DB11 DBX20.7
PROG_END	Alarm 16954	NC-Prog: M30
INTERRUPT	delayed	NST: FC-9 und ASUP DB10 DBB1
SINGLEBLOCKSTOP	delayed	Einzelsatzbetrieb im Stop-Delay-Bereich eingeschaltet: NC stoppt am Ende des 1. Satzes außerhalb des Stop-Delay-Bereichs. Einzelsatz bereits vor Stop-Delay-Bereich angewählt: NST: "NC-Stopp an der Satzgrenze" DB21, ... DBX7.2
STOPPROG	delayed	NST: DB21,... DBX7.3 und DB11 DBX20.5
PROG_STOP	Alarm 16954	NC-Prog: M0 und M1
WAITM	Alarm 16954	NC-Prog: WAITM
WAITE	Alarm 16954	NC-Prog: WAITE
STOP_ALARM	immediat	Alarm: Alarmprojektierung STOPBYALARM
RETREAT_MOVE_THREAD	Alarm 16954	NC-Prog: Alarm 16954 bei LFON (Stopp & Fastlift im G33 nicht möglich)
WAITMC	Alarm 16954	NC-Prog: WAITMC
NEWCONF_PREP_STOP	Alarm 16954	NC-Prog: NEWCONF
SYSTEM_SHUTDOWN	immediate	System-Shutdown bei 840Di sl
ESR	delayed	Erweitertes Stillsetzen und Rückziehen
EXT_ZERO_POINT	delayed	Externe Nullpunktverschiebung
STOPRUN	Alarm 16955	BTSS: PI "_N_FINDST" STOPRUN

Erklärung der Reaktionen:

immediate ("hartes" Stopp-Ereignis)	Stoppt sofort auch im Stop-Delay-Bereich
delayed ("sanftes" Stopp-Ereignis)	Stopp (auch ein kurzfristiger) erfolgt erst nach dem Stop-Delay-Bereich.
Alarm 16954	Programm wird abgebrochen, da im Stop-Delay-Bereich unerlaubte Programmbefehle verwendet worden sind.
Alarm 16955	Programm wird fortgesetzt, im Stop-Delay-Bereich hat eine unerlaubte Aktion stattgefunden hat.
Alarm 16957	Der Programmbereich (Stop-Delay-Bereich), der durch DELAYFSTON und DELAYFSTOF geklammert ist, konnte nicht aktiviert werden. Damit wirkt jeder Stopp im Bereich sofort und wird nicht verzögert.

Eine Zusammenfassung weiterer Reaktionen auf Stopp-Ereignisse siehe:

Literatur:

Funktionshandbuch Grundfunktionen; BAG, Kanal, Programmbetrieb, Reset-Verhalten (K1)

Vorteile des Stop-Delay-Bereichs

Ein Programmabschnitt wird ohne Geschwindigkeitseinbruch bearbeitet.

Bricht der Anwender, nachdem gestoppt ist, das Programm mit RESET ab, so ist der abgebrochene Programmsatz nach dem geschützten Bereich. Dieser Programmsatz eignet sich dann als Suchziel für einen nachfolgenden Suchlauf.

Solange ein Stop-Delay-Bereich bearbeitet wird werden folgende Hauptlaufachsen nicht gestoppt:

- Kommandoachsen und
- Positionierachsen die mit POSA verfahren

Der Teileprogrammbefehl G4 ist im Stop-Delay-Bereich zulässig, dagegen sind andere Teileprogrammbefehle, die zu einem vorübergehenden Stopp führen (z. B. WAITM) nicht zulässig.

G4 macht, wie eine Bahnbewegung, den Stop-Delay-Bereich wirksam bzw. hält seine Wirksamkeit aufrecht.

Beispiel: Vorschub Eingriffe

Wird der Override vor dem Stop-Delay-Bereich auf 6% gesenkt, so wird der Override im Stop-Delay-Bereich wirksam.

Wird der Override im Stop-Delay-Bereich von 100% auf 6% abgesenkt, so wird der Stop-Delay-Bereich mit 100% zu Ende gefahren und danach wird mit 6% weitergefahren.

Die Vorschubsperrung wirkt im Stop-Delay-Bereich nicht, erst nach dem Verlassen des Stop-Delay-Bereichs wird angehalten.

Überlappung / Schachtelung

Überschneiden sich zwei Stop-Delay-Bereiche, einer aus den Sprachbefehlen und der andere aus dem Maschinendatum MD11550 \$MN_STOP_MODE_MASK, so wird der größtmögliche Stop-Delay-Bereich gebildet.

Folgende Punkte regeln das Zusammenspiel der Sprachbefehle DELAYFSTON und DELAYFSTOF mit Verschachtelungen und dem Unterprogrammende:

- Mit dem Ende des Unterprogramms, in dem DELAYFSTON gerufen wurde, wird implizit DELAYFSTOF aktiviert.
- DELAYFSTON in einem Stop-Delay-Bereich bleibt ohne Wirkung.
- Ruft Unterprogramm 1 in einem Stop-Delay-Bereich Unterprogramm 2, so ist Unterprogramm 2 komplett ein Stop-Delay-Bereich. Insbesondere ist DELAYFSTOF in Unterprogramm 2 wirkungslos.

Hinweis

REPOSA ist ein Unterprogrammende und DELAYFSTON wird in jedem Fall abgewählt.

Trifft ein "hartes" Stopp-Ereignis auf den "Stop-Delay-Bereich", so wird der "Stop-Delay-Bereich" komplett abgewählt! Das heißt, tritt in diesem Programmabschnitt ein weiterer beliebiger Stopp auf, so wird sofort angehalten. Erst eine Neuprogrammierung (erneutes DELAYFSTON) lässt einen neuen Stop-Delay-Bereich beginnen.

Wird die Stopp-Taste vor dem Stop-Delay-Bereich gedrückt und der NCK muss zum Bremsen in den Stop-Delay-Bereich einfahren, so stoppt der NCK im Stop-Delay-Bereich und der Stop-Delay-Bereich bleibt abgewählt!

Wird ein Stop-Delay-Bereich mit Override 0% betreten, so wird der Stop-Delay-Bereich **nicht** akzeptiert!

Dies gilt für alle "sanften" Stopp-Ereignisse.

Mit STOPALL kann im Stop-Delay-Bereich gebremst werden. Mit einem STOPALL werden aber alle anderen Stopp-Ereignisse sofort aktiv, die bislang verzögert worden sind.

Systemvariablen

Ein Stop-Delay-Bereich kann mit \$P_DELAYFST im Teileprogramm erkannt werden. Ist Bit 0 der Systemvariablen auf 1 gesetzt, so befindet sich die Teileprogrammbearbeitung zu diesem Zeitpunkt in einem Stop-Delay-Bereich.

Ein Stop-Delay-Bereich kann mit \$AC_DELAYFST in Synchronaktionen erkannt werden. Ist Bit 0 der Systemvariablen auf 1 gesetzt, so befindet sich die Teileprogrammbearbeitung zu diesem Zeitpunkt in einem Stop-Delay-Bereich.

Kompatibilität

Die Vorbesetzung des Maschinendatums MD11550 \$MN_STOP_MODE_MASK Bit 0 = 0 bewirkt impliziten Stop-Delay-Bereich während der G-Code-Gruppe G331/G332 und wenn eine Bahnbewegung bzw. G4 programmiert ist.

Bit 0 = 1 ermöglicht Stopp während der G-Code-Gruppe G331/G332 und wenn eine Bahnbewegung bzw. G4 programmiert ist. Zur Definition eines Stop-Delay-Bereichs müssen die Befehle DELAYFSTON/DELAYFSTOF benutzt werden.

8.8 Programmstelle für SERUPRO verhindern (IPTRLOCK, IPTRUNLOCK)

Funktion

Für bestimmte komplizierte mechanische Situationen an der Maschine ist es erforderlich, den Satzsuchlauf SERUPRO zu verhindern.
Mit einem programmierbaren Unterbrechungszeiger besteht eine Eingriffsmöglichkeit, beim "Suchen auf der Unterbrechungsstelle", vor der suchunfähigen Stelle aufzusetzen.
Es können auch suchunfähige Bereiche in Teileprogrammgebieten definiert werden, in denen die NCK noch nicht wieder einsteigen kann. Mit dem Programmabbruch vermerkt der NCK den zuletzt verarbeiteten Satz, auf den über die Bedienoberfläche HMI gesucht werden kann.

Syntax

```
IPTRLOCK
IPTRUNLOCK
```

Die Befehle stehen allein in einer Teleprogrammzeile und ermöglichen einen programmierbaren Unterbrechungszeiger

Bedeutung

```
IPTRLOCK           Beginn des suchunfähigen Programmabschnitts
IPTRUNLOCK         Ende des suchunfähigen Programmabschnitts
```

Beide Befehle sind nur in Teileprogrammen, **nicht** jedoch in Synchronaktionen zulässig.

Beispiel

Verschachtelung suchunfähiger Programmabschnitte in zwei Programmebenen mit impliziten IPTRUNLOCK. Das implizite IPTRUNLOCK in Unterprogramm 1 beendet den suchunfähigen Bereich.

Programmcode	Kommentar
N10010 IPTRLOCK()	
N10020 R1 = R1 + 1	
N10030 G4 F1	; Haltesatz, der suchunfähige Programmabschnitt beginnt.
...	
N10040 Unterprogramm2	
...	; Interpretation des Unterprogramms 2.
N20010 IPTRLOCK ()	; Unwirksam, wiederholter Beginn.
...	
N20020 IPTRUNLOCK ()	; Unwirksam, Ende in anderer Ebene.
N20030 RET	
...	
N10060 R2 = R2 + 2	
N10070 RET	; Ende des suchunfähigen Programmabschnitts.
N100 G4 F2	; Hauptprogramm wird fortgesetzt.

Eine Unterbrechung auf 100 liefert dann wieder der Unterbrechungszeiger.

Suchunfähige Bereiche erfassen und suchen

Die suchunfähigen Programmabschnitte werden mit dem Sprachbefehlen `IPTRLOCK` und `IPTRUNLOCK` gekennzeichnet.

Der Befehl `IPTRLOCK` friert den Unterbrechungszeiger auf ein im Hauptlauf ausführbaren Einzelsatz (`SBL1`) ein. Dieser Satz wird im Folgenden als Haltesatz bezeichnet. Tritt nach `IPTRLOCK` ein Programmabbruch ein, so kann auf der Bedienoberfläche HMI nach diesen sogenannten Haltesatz gesucht werden.

Auf den aktuellen Satz wieder aufsetzen

Der Unterbrechungszeiger wird mit `IPTRUNLOCK` für den nachfolgenden Programmabschnitt auf den aktuellen Satz zum Unterbrechungspunkt gesetzt werden.

Nach einem gefundenen Suchziel kann mit dem selben Haltesatz ein neues Suchziel wiederholt werden.

Ein vom Benutzer editierter Unterbrechungszeiger, muss über HMI wieder entfernt werden.

Regeln bei Schachtelung

Folgende Punkte regeln das Zusammenspiel der Sprachbefehle `IPTRLOCK` und `IPTRUNLOCK` mit Verschachtelungen und dem Unterprogrammende:

1. Mit dem Ende des Unterprogramms, in dem `IPTRLOCK` gerufen wurde, wird implizit `IPTRUNLOCK` aktiviert.
2. `IPTRLOCK` in einem suchunfähigen Bereich bleibt ohne Wirkung.
3. Ruft Unterprogramm1 in einem suchunfähigen Bereich Unterprogramm2, so bleibt Unterprogramm2 komplett suchunfähig. Insbesondere ist `IPTRUNLOCK` in Unterprogramm2 wirkungslos.

Weitere Informationen hierzu siehe /FB1/ Funktionshandbuch Grundfunktionen; BAG, Kanal, Programmbetrieb (K1).

Systemvariable

Ein suchunfähiger Bereich kann mit `$P_IPTRLOCK` im Teileprogramm erkannt werden.

Automatischer Unterbrechungszeiger

Die Funktion automatischer Unterbrechungszeiger legt automatisch eine vorher festgelegte Kopplungsart als suchunfähig fest. Mittels Maschinendatum wird für

- Elektronisches Getriebe bei `EGON`
- Axiale Leitwertkopplung bei `LEADON`

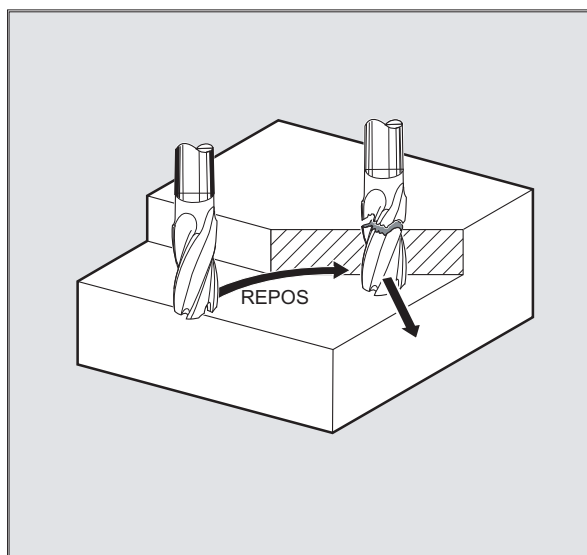
der automatische Unterbrechungszeiger aktiviert. Überschneiden sich der programmierte und der über Maschinendatum aktivierbare automatische Unterbrechungszeiger, so wird der größtmögliche suchunfähige Bereich gebildet.

8.9 Wiederanfahren an Kontur (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN)

Funktion

Wenn Sie während der Bearbeitung das laufende Programm unterbrechen und das Werkzeug freifahren – zum Beispiel wegen Werkzeugbruch oder weil Sie etwas nachmessen wollen – können Sie die Kontur an einem wählbaren Punkt programmgesteuert wieder anfahren.

Der REPOS-Befehl wirkt wie ein Unterprogramm-Rücksprung (z. B. über M17). Nachfolgende Sätze in der Interruptroutine werden nicht mehr ausgeführt.



Zur Unterbrechung des Programmlaufs siehe auch "Interruptroutine (ASUP) (Seite 130)".

Syntax

```
REPOSA RMI DISPR=...
REPOSA RMB
REPOSA RME
REPOSA RMN
REPOSL RMI DISPR=...
REPOSL RMB
REPOSL RME
REPOSL RMN
REPOSQ RMI DISPR=... DISR=...
REPOSQ RMB DISR=...
REPOSQ RME DISR=...
REPOSQA DISR=...
REPOSH RMI DISPR=... DISR=...
REPOSH RMB DISR=...
REPOSH RME DISR=...
REPOSHA DISR=...
```

Bedeutung

Anfahrweg

REPOSA	Anfahren auf einer Geraden mit allen Achsen
REPOSL	Anfahren auf einer Geraden
REPOSQ DISR=...	Anfahren auf einem Viertelkreis mit Radius DISR
REPOSQA DISR=...	Anfahren auf allen Achsen auf einem Viertelkreis mit Radius DISR
REPOSH DISR=...	Anfahren auf einem Halbkreis mit Durchmesser DISR
REPOSHA DISR=...	Anfahren auf allen Achsen auf einem Halbkreis mit Radius DISR

Wiederanfahrpunkt

RMI	Unterbrechungspunkt anfahren
RMI DISPR=...	Eintrittspunkt im Abstand DISPR in mm/inch vor Unterbrechungspunkt
RMB	Satzanfangspunkt anfahren
RME	Satzendpunkt anfahren
RME DISPR=...	Satzendpunkt anfahren im Abstand DISPR vor Endpunkt
RMN	An den nächstliegenden Bahnpunkt anfahren
A0 B0 C0	Achsen, in denen angefahren werden soll

Beispiel: Anfahren auf einer Geraden anfahren, REPOSA, REPOSL

Das Werkzeug fährt den Wiederanfahrpunkt direkt auf einer Geraden an.

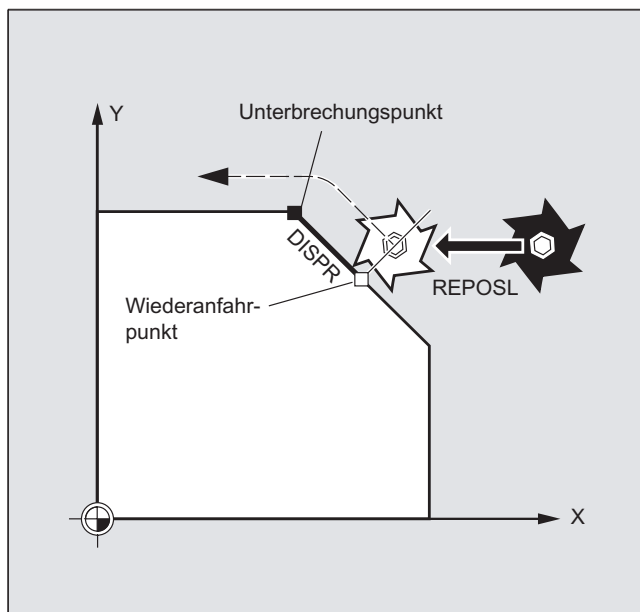
Mit REPOSA werden automatisch alle Achsen verfahren. Bei REPOSL können Sie die zu verfahrenen Achsen angeben.

Beispiel:

```
REPOSL RMI DISPR=6 F400
```

oder

```
REPOSA RMI DISPR=6 F400
```

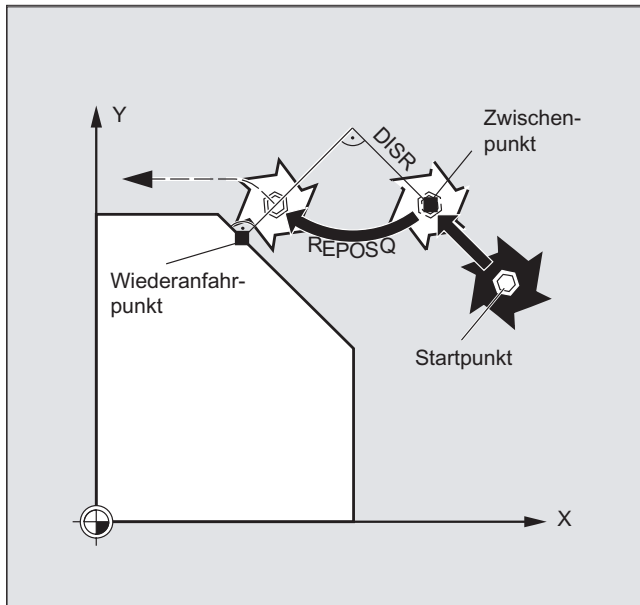


Beispiel: Anfahren im Viertelkreis anfahren, REPOSQ, REPOSQA

Das Werkzeug fährt den Wiederaufnahmepunkt auf einem Viertelkreis mit Radius $DISR = \dots$ an. Den notwendigen Zwischenpunkt zwischen Start- und Wiederaufnahmepunkt berechnet die Steuerung automatisch.

Beispiel:

REPOSQ RMI DISR=10 F400

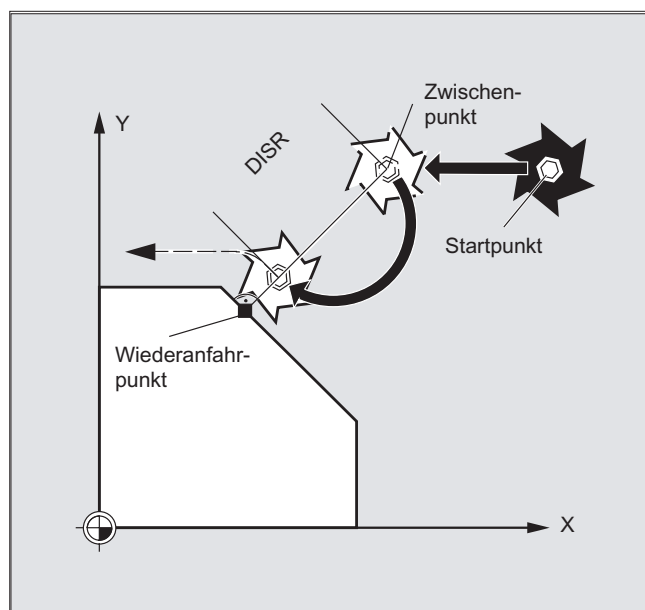


Beispiel: Werkzeug im Halbkreis anfahren, REPOSH, REPOSHA

Das Werkzeug fährt den Wiederauffahrtspunkt auf einem Halbkreis mit Durchmesser $DISR = \dots$ an. Den notwendigen Zwischenpunkt zwischen Start- und Wiederauffahrtspunkt berechnet die Steuerung automatisch.

Beispiel:

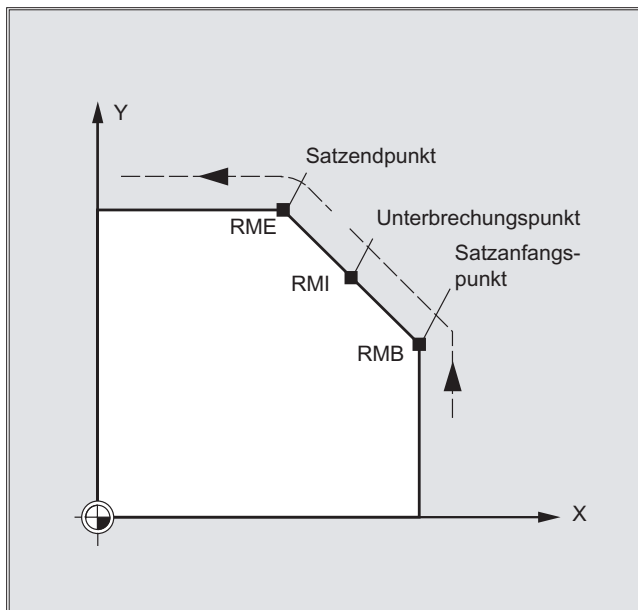
REPOSH RMI DISR=20 F400



Wiederanfahrpunkt festlegen (nicht für SERUPRO Anfahren mit RMN)

Bezogen auf den NC-Satz, in dem der Programm-Ablauf unterbrochen wurde, können Sie zwischen drei Wiederanfahrpunkten wählen:

- RMI, Unterbrechungspunkt
- RMB, Satzanfangspunkt bzw. letzter Endpunkt
- RME, Satzendpunkt



Mit RMI DISPR=... bzw. mit RME DISPR=... können Sie einen Wiederanfahrpunkt festlegen, der vor dem Unterbrechungspunkt bzw. vor dem Satzendpunkt liegt.

Mit DISPR=... beschreiben Sie den Konturweg in mm/inch, um den der Wiederanfahrpunkt vor dem Unterbrechungs- bzw. Endpunkt liegt. Dieser Punkt kann - auch für größere Werte - maximal im Satzanfangspunkt liegen.

Wird kein DISPR=... programmiert, gilt DISPR=0 und damit der Unterbrechungspunkt (bei RMI) bzw. der Satzendpunkt (bei RME).

Vorzeichen von DISPR

Das Vorzeichen von DISPR wird ausgewertet. Bei positivem Vorzeichen ist das Verhalten wie bisher.

Bei negativem Vorzeichen wird hinter dem Unterbrechungspunkt bzw. bei RMB hinter dem Startpunkt wieder aufgesetzt.

Der Abstand Unterbrechungspunkt-Aufsetzpunkt ergibt sich aus dem Betrag von DISPR. Dieser Punkt kann auch für betragsmäßig größere Werte maximal im Satzendpunkt liegen.

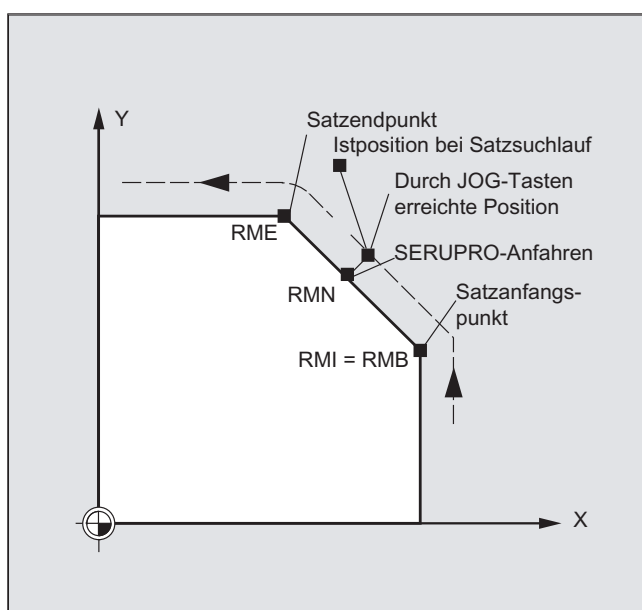
Anwendungsbeispiel:

Durch einen Sensor wird die Annäherung an eine Spannpratze erkannt. Es wird ein $ASUP$ ausgelöst, mit dem die Spannpratze umfahren wird.

Anschließend wird mit negativem DISPR auf einen Punkt hinter der Spannpratze repositioniert und das Programm fortgesetzt.

SERUPRO-Anfahren mit RMN

Wird bei der Bearbeitung an einer beliebigen Stelle ein Abbruch erzwungen, dann wird mit SERUPRO-Anfahren unter RMN der kürzeste Weg von der Abbruchstelle angefahren, um anschließend nur den Restweg abzuarbeiten. Dazu startet der Anwender ein SERUPRO-Vorgang auf den Unterbrechungssatz und positioniert mit den JOG-Tasten vor die schadhafte Stelle des Zielsatzes.



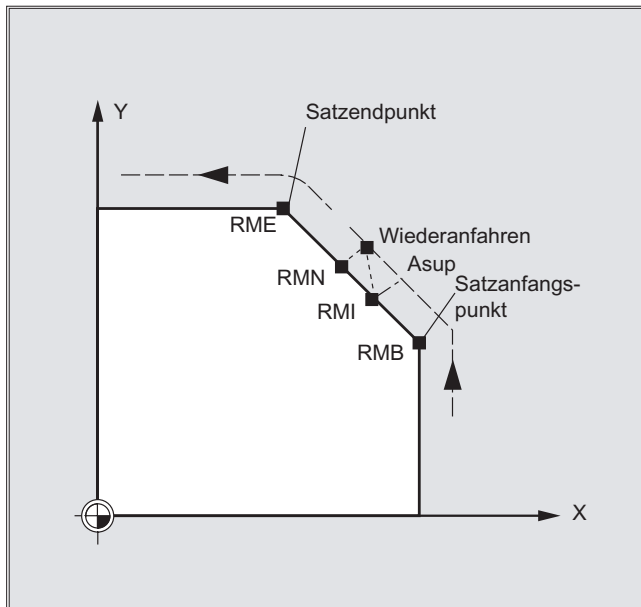
Hinweis

SERUPRO

Für SERUPRO ist RMI und RMB identisch. RMN ist nicht nur auf SERUPRO beschränkt, sondern allgemein gültig.

Anfahren vom nächstliegenden Bahnpunkt RMN

Zum Interpretationszeitpunkt von REPOSA wird nach einer Unterbrechung der Wiederanfahrtsatz mit RMN nicht noch einmal komplett begonnen, sondern nur der Restweg abgearbeitet. Es wird der nächstliegende Bahnpunkt des unterbrochenen Satzes angefahren.



Status für den gültigen REPOS-Mode

Der gültige REPOS-Mode des unterbrochenen Satzes kann über Synchronaktionen mit der Variablen \$AC_REPOS_PATH_MODE gelesen werden:

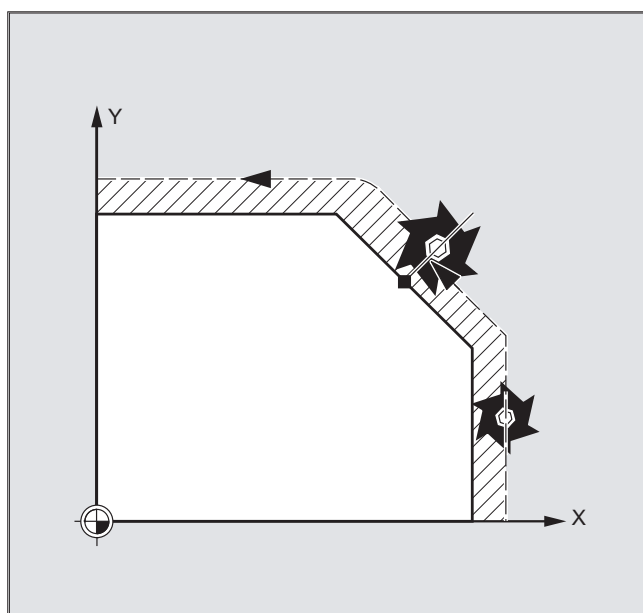
- 0 Anfahren nicht definiert
- 1 RMB: Anfahren auf den Beginn
- 2 RMI: Anfahren auf den Unterbrechungspunkt
- 3 RME: Anfahren auf den Satzendpunkt
- 4 RMN: Anfahren auf den nächstliegenden Bahnpunkt des unterbrochenen Satzes

Anfahren mit neuem Werkzeug

Falls Sie den Programmablauf wegen Werkzeugbruch gestoppt haben:

Mit Programmierung der neuen D-Nummer wird das Programm ab Wiederanfahrpunkt mit den geänderten Werkzeugkorrekturwerten fortgesetzt.

Bei geänderten Werkzeugkorrekturwerten kann der Unterbrechungspunkt möglicherweise nicht mehr angefahren werden. In diesem Fall wird der dem Unterbrechungspunkt nächstgelegene Punkt auf der neuen Kontur angefahren (gegebenenfalls um DISPR modifiziert).



Kontur anfahren

Die Bewegung, mit der das Werkzeug wieder an die Kontur heranfährt, ist programmierbar. Die Adressen der zu verfahrenen Achsen geben Sie mit Wert Null an.

Mit den Befehlen REPOSA, REPOSQA und REPOSHA werden automatisch alle Achsen repositioniert. Es ist keine Achsangabe notwendig.

Bei Programmierung von REPOSL, REPOSQ und REPOSH fahren alle Geometrieachsen automatisch, also auch ohne Angabe im Befehl, an. Alle anderen Achsen müssen im Befehl angegeben werden.

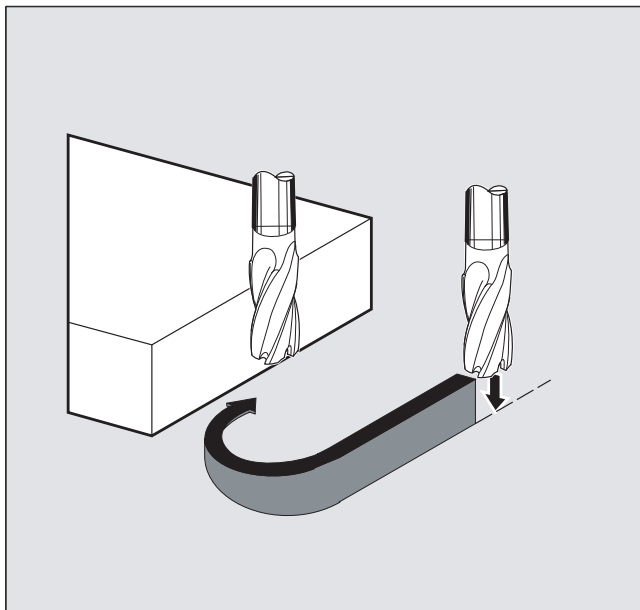
Für die Kreisbewegungen REPOSH und REPOSQ gilt:

Der Kreis wird in der angegebenen Arbeitsebene G17 bis G19 gefahren.

Falls Sie im Anfahrsatz die dritte Geometrieachse (Zustellrichtung) angeben, wird der Wiederanfahrpunkt für den Fall, dass Werkzeugposition und programmierte Position in Zustellrichtung nicht übereinstimmen, auf einer Schraubenlinie angefahren.

In folgenden Fällen wird automatisch auf lineares Anfahren REPOSL umgeschaltet:

- Sie haben keinen Wert für DISR angegeben.
- Es gibt keine definierte Anfahrrichtung (Programmunterbrechung in einem Satz ohne Verfahrinformation).
- Bei Anfahrrichtung senkrecht zur aktuellen Arbeitsebene.



8.10 Beeinflussung der Bewegungsführung

8.10.1 Prozentuale Ruckkorrektur (JERKLIM)

Funktion

Mit dem NC-Befehl `JERKLIM` kann der per Maschinendatum eingestellte maximal mögliche Ruck einer Achse bei Bahnbewegung in kritischen Programmabschnitten reduziert oder überhöht werden.

Voraussetzung

Der Beschleunigungsmodus `SOFT` muss aktiv sein.

Wirksamkeit

Die Funktion wirkt:

- in den AUTOMATIK-Betriebsarten.
- nur auf Bahnachsen.

Syntax

`JERKLIM [<Achse>] =<Wert>`

Bedeutung

<code>JERKLIM:</code>	Befehl zur Ruckkorrektur
<code><Achse>:</code>	Maschinenachse, deren Ruckgrenzwert angepasst werden soll.
<code><Wert>:</code>	Prozentualer Korrekturwert, bezogen auf den projektierten maximalen Achsruck bei Bahnbewegung (MD32431 <code>\$MA_MAX_AX_JERK</code>).
	Wertebereich: 1 ... 200
	Der Wert 100 bewirkt keine Beeinflussung des Rucks.

Hinweis

Das Verhalten von `JERKLIM` bei Teileprogrammende und Kanal-Reset wird projektiert mit Bit 0 im Maschinendatum MD32320 `$MA_DYN_LIMIT_RESET_MASK`:

- Bit 0 = 0:
Der programmierte Wert für `JERKLIM` wird mit Kanal-Reset/M30 auf 100 % zurückgesetzt.
 - Bit 0 = 1:
Der programmierte Wert für `JERKLIM` bleibt über Kanal-Reset/M30 hinaus erhalten.
-

Beispiel

Programmcode	Kommentar
...	
N60 JERKLIM[X]=75	; Der Achsschlitten in X-Richtung soll nur mit maximal 75% des für die Achse zulässigen Rucks beschleunigt/verzögert werden.
...	

8.10.2 Prozentuale Geschwindigkeitskorrektur (VELOLIM)

Funktion

Mit dem Befehl `VELOLIM` kann im Teileprogramm oder Synchronaktion die über Maschinendatum eingestellte maximal mögliche Geschwindigkeit einer Achse bzw. die maximal mögliche getriebestufenabhängige Drehzahl einer Spindel reduziert werden.

Wirksamkeit

Die Funktion wirkt:

- in den AUTOMATIK-Betriebsarten.
- auf Bahn- und Positionierachsen.
- auf Spindeln im Spindel-/Achsbetrieb

Syntax

`VELOLIM[<Achse/Spindel>]=<Wert>`

Bedeutung

`VELOLIM:` Befehl zur Geschwindigkeitskorrektur
`<Achse/Spindel>:` Achse oder Spindel, deren Geschwindigkeits- oder Drehzahlgrenzwert angepasst werden soll.

VELOLIM für Spindeln
 Über Maschinendatum (MD30455 \$MA_MISC_FUNCTION_MASK, Bit 6) kann für die Programmierung im Teileprogramm eingestellt werden, ob `VELOLIM` unabhängig von der aktuellen Verwendung als Spindel oder Achse wirkt (Bit 6 = 1) oder getrennt für jede Betriebsart programmierbar sein soll (Bit 6 = 0). Ist eine getrennte Wirkung projektiert, dann wird die Auswahl über den Bezeichner bei der Programmierung getroffen:

- Spindelbezeichner `S<n>` für Spindelbetriebsarten
- Achsbezeichner, z. B. "c", für den Achsbetrieb

<Wert>: Prozentualer Korrekturwert
 Der Korrekturwert bezieht sich:

- bei Achsen / Spindeln im Achsbetrieb (MD30455, Bit 6 == 0):
 auf die projektierte maximale Achsgeschwindigkeit
 (MD32000 \$MA_MAX_AX_VELO).
- bei Spindeln im Spindel- oder Achsbetrieb (MD30455
 Bit 6 == 1):
 auf die Maximaldrehzahl der aktiven Getriebestufe
 (MD35130 \$MA_GEAR_STEP_MAX_VELO_LIMIT[<n>])

Wertebereich: 1 ... 100
 Der Wert 100 bewirkt keine Beeinflussung der Geschwindigkeit
 bzw. Drehzahl.

Hinweis

Verhalten bei Teileprogrammende und Kanal-Reset

Das Verhalten von VELOLIM bei Teileprogrammende und Kanal-Reset ist einstellbar über das Maschinendatum: MD32320 \$MA_DYN_LIMIT_RESET_MASK, Bit 0

Erkennen einer aktiven Drehzahlbegrenzung im Spindelbetrieb

Eine Drehzahlbegrenzung durch VELOLIM (kleiner 100 %) kann im Spindelbetrieb über folgende Systemvariable erkannt werden:

- \$AC_SMAXVELO (maximal mögliche Spindeldrehzahl)
- \$AC_SMAXVELO_INFO (Kennung für die drehzahlbegrenzende Ursache)

Beispiele

Beispiel 1: Geschwindigkeitsbegrenzung Maschinenachse

Programmcode	Kommentar
...	
N70 VELOLIM[X]=80	; Der Achsschlitten in X-Richtung soll nur mit maximal 80 % der für die Achse zulässigen Geschwindigkeit verfahren werden.
...	

Beispiel 2: Drehzahlbegrenzung Spindel

Programmcode	Kommentar
N05 VELOLIM[S1]=90	; Begrenzung der Maximaldrehzahl von Spindel 1 auf 90% von 1000 U/min.
...	
N50 VELOLIM[C]=45	; Begrenzung der Drehzahl auf 45% von 1000 U/min, C sei der Achsbezeichner von S1.
...	

Maschinendateneinstellungen für Spindel 1 (AX5)

- Maximaldrehzahl der Getriebstufe 1 = 1000 U/min:
MD35130 \$MA_GEAR_STEP_MAX_VELO_LIMIT[1, AX5] = 1000
- Die Programmierung von VELOLIM wirkt gemeinsam für Spindel- und Achsbetrieb unabhängig vom programmierten Bezeichner:
MD30455 \$MA_MISC_FUNCTION_MASK[AX5], Bit 6 = 1

8.10.3 Programmbeispiel für JERKLIM und VELOLIM

Das folgende Programm stellt ein Anwendungsbeispiel für die prozentuale Ruck- und Geschwindigkeitsbegrenzung dar:

Programmcode	Kommentar
N1000 G0 X0 Y0 F10000 SOFT G64	
N1100 G1 X20 RNDM=5 ACC[X]=20 ACC[Y]=30	
N1200 G1 Y20 VELOLIM[X]=5	; Der Achsschlitten in X-Richtung soll nur mit max. 5% der für die Achse zulässigen Geschwindigkeit verfahren werden.
JERKLIM[Y]=200	; Der Achsschlitten in Y-Richtung kann mit max. 200% des für die Achse zulässigen Rucks beschleunigt/verzögert werden.
N1300 G1 X0 JERKLIM[X]=2	; Der Achsschlitten in X-Richtung soll nur mit max. 2% des für die Achse zulässigen Rucks beschleunigt/verzögert werden.
N1400 G1 Y0	
M30	

8.11 Programmierbare Kontur-/Orientierungstoleranz (CTOL, OTOL, ATOL)

Funktion

Mit den Befehlen `CTOL`, `OTOL` und `ATOL` können die über Maschinen- und Settingdaten festgelegten Bearbeitungstoleranzen für die Kompressor-Funktionen (COMPON, COMPCURV, COMPCAD), die Überschleifarten G642, G643, G645, OST und die Orientierungsglättung ORISON im NC-Programm angepasst werden.

Die programmierten Werte gelten, bis sie neu programmiert oder durch Zuweisung eines negativen Werts gelöscht werden. Sie werden ferner gelöscht bei Programmende, Kanal-Reset, BAG-Reset, NCK-Reset (Warmstart) und Power On (Kaltstart). Nach dem Löschen gelten wieder die Werte aus den Maschinen- und Settingdaten.

Syntax

```
CTOL=<Wert>
OTOL=<Wert>
ATOL [<Achse>] =<Wert>
```

Bedeutung

`CTOL` **Befehl zum Programmieren der Konturtoleranz**

`CTOL` ist gültig für:

- alle Kompressor-Funktionen
- alle Überschleifarten außer G641 und G644

<Wert>: Der Wert für die Konturtoleranz ist eine Längenangabe.

Typ: REAL

Einheit: Inch/mm (abhängig von der aktuellen Einstellung der Maßangabe)

`OTOL` **Befehl zum Programmieren der Orientierungstoleranz**

`OTOL` ist gültig für:

- alle Kompressor-Funktionen
- Orientierungsglättung ORISON
- alle Überschleifarten außer G641, G644, OSD

<Wert>: Der Wert für die Orientierungstoleranz ist eine Winkelangabe.

Typ: REAL

Einheit: Grad

ATOL **Befehl zum Programmieren einer achsspezifischen Toleranz**
 ATOL ist gültig für:

- alle Kompressor-Funktionen
- Orientierungsglättung ORISON
- alle Überschleifarten außer G641, G644, OSD

<Achse>: Name der Achse, für die eine Achstoleranz programmiert werden soll
 <Wert>: Der Wert für die Achstoleranz ist je nach Achstyp (Linear- oder Rundachse) eine Längen- oder Winkelangabe.

Typ: REAL
 Einheit: für Linearachsen: Inch/mm (abhängig von der aktuellen Einstellung der Maßangabe)
 für Rundachsen: Grad

Hinweis

CTOL und OTOL haben Vorrang vor ATOL.

Randbedingungen

Skalierende Frames

Skalierende Frames wirken auf die programmierten Toleranzen in gleicher Weise wie auf die Achspositionen, d. h. die relative Toleranz bleibt gleich.

Beispiel

Programmcode	Kommentar
COMPCAD G645 G1 F10000	; Kompressor-Funktion COMPCAD aktivieren.
X... Y... Z...	; Hier wirken die Maschinen- und Settingdaten.
X... Y... Z...	
X... Y... Z...	
CTOL=0.02	; Ab hier wirkt eine Konturtoleranz von 0,02 mm.
X... Y... Z...	
X... Y... Z...	
X... Y... Z...	
ASCALE X0.25 Y0.25 Z0.25	; Ab hier wirkt eine Konturtoleranz von 0,005 mm.
X... Y... Z...	
X... Y... Z...	
X... Y... Z...	
CTOL=-1	; Ab hier wirken wieder Maschinen- und Settingdaten.
X... Y... Z...	
X... Y... Z...	
X... Y... Z...	

Weitere Informationen

Toleranzwerte lesen

Für weitergehende Anwendungsfälle oder zur Diagnose sind die aktuell gültigen Toleranzen für die Kompressor-Funktionen (COMPON, COMPCURV, COMPCAD), die Überschleifarten G642, G643, G645, OST und die Orientierungsglättung ORISON unabhängig von der Art des Zustandekommens über Systemvariablen lesbar.

- In Synchronaktionen oder mit Vorlauf-Stopp im Teileprogramm über die Systemvariablen:

\$AC_CTOL	Konturtoleranz, die bei der Aufbereitung des aktuellen Hauptlaufsatzes wirksam war Falls keine Konturtoleranz wirksam ist, liefert \$AC_CTOL die Wurzel aus der Summe der Quadrate der Toleranzen der Geometrieachsen.
\$AC_OTOL	Orientierungstoleranz, die bei der Aufbereitung des aktuellen Hauptlaufsatzes wirksam war Falls keine Orientierungstoleranz wirksam ist, liefert \$AC_OTOL während einer aktiven Orientierungstransformation die Wurzel aus der Summe der Quadrate der Toleranzen der Orientierungsachsen, ansonsten den Wert "-1".
\$AA_ATOL [<Achse>]	Achstoleranz, die bei der Aufbereitung des aktuellen Hauptlaufsatzes wirksam war Falls eine Konturtoleranz aktiv ist, liefert \$AA_ATOL[<Geometrieachse>] die Konturtoleranz geteilt durch die Wurzel aus der Anzahl der Geometrieachsen. Falls eine Orientierungstoleranz und eine Orientierungstransformation aktiv sind, liefert \$AA_ATOL[<Orientierungsachse>] die Orientierungstoleranz geteilt durch die Wurzel aus der Anzahl der Orientierungsachsen.

Hinweis

Wenn keine Toleranzwerte programmiert wurden, dann sind die \$A-Variablen nicht differenziert genug, um die möglicherweise verschiedenen Toleranzen der einzelnen Funktionen zu unterscheiden, da sie ja nur einen Wert nennen können. Solche Fälle können auftreten, wenn die Maschinen- und Settingdaten unterschiedliche Toleranzen für Kompressor-Funktionen, Überschleifen und Orientierungsglättung einstellen. Die Variablen liefern dann den größten Wert, der bei den gerade aktiven Funktionen auftritt. Ist z. B. eine Kompressor-Funktion mit Orientierungstoleranz 0,1° und eine Orientierungsglättung ORISON mit 1° aktiv, liefert die Variable \$AC_OTOL den Wert "1". Wird die Orientierungsglättung ausgeschaltet, liest man nur noch den Wert "0.1".

- Ohne Vorlauf-Stopp im Teileprogramm über die Systemvariablen:

\$P_CTOL	Programmierte Konturtoleranz
\$P_OTOL	Programmierte Orientierungstoleranz
\$PA_ATOL	Programmierte Achstoleranz

Hinweis

Wenn keine Toleranzwerte programmiert sind, dann liefern die \$P-Variablen den Wert "-1".

8.12 Toleranz bei G0-Bewegungen (STOLF)

G0-Toleranzfaktor

G0-Bewegungen (Eilgang, Zustellbewegungen) können im Unterschied zur Werkstückbearbeitung mit größerer Toleranz verfahren werden. Dies hat den Vorteil, dass sich die Abfahrzeiten für G0-Bewegungen verkürzen.

Die Einstellung der Toleranzen bei G0-Bewegungen erfolgt durch Projektierung des G0-Toleranzfaktors (MD20560 \$MC_G0_TOLERANCE_FACTOR).

Der G0-Toleranzfaktor wird nur wirksam, wenn:

- eine der folgenden Funktionen aktiv ist:
 - Kompressorfunktionen: COMPON, COMPCURV und COMPCAD
 - Überschleiffunktionen: G642 und G645
 - Orientierungsüberschleifen: OST
 - Orientierungsglättung: ORISON
 - Glättung bei bahnrelativer Orientierung: ORIPATH
- mehrere (≥ 2) G0-Sätze aufeinanderfolgen.

Bei einem einzelnen G0-Satz wird der G0-Toleranzfaktor nicht wirksam, da **beim Übergang** von einer Nicht-G0-Bewegung zu einer G0-Bewegung (und umgekehrt) grundsätzlich die "**kleinere Toleranz**" (Werkstückbearbeitungstoleranz) wirkt!

Funktion

Durch Programmierung von `STOLF` im Teileprogramm kann der projektierte G0-Toleranzfaktor (MD20560) temporär überschrieben werden. Der Wert im MD20560 wird dabei nicht verändert. Nach Reset bzw. Teileprogrammende wird der projektierte Toleranzfaktor wieder wirksam.

Syntax

`STOLF=<Toleranzfaktor>`

Bedeutung

<code>STOLF:</code>	Befehl zur Programmierung des G0-Toleranzfaktors
<code><Toleranzfaktor>:</code>	G0-Toleranzfaktor
	Der Faktor kann sowohl größer 1 als auch kleiner 1 sein. Normalerweise werden jedoch für G0-Bewegungen größere Toleranzen einstellbar sein.
	Bei <code>STOLF=1.0</code> (entspricht dem projektierten Standardwert) sind für G0-Bewegungen dieselben Toleranzen wirksam wie für Nicht-G0-Bewegungen.

Systemvariablen

Der im Teileprogramm bzw. im aktuellen IPO-Satz wirksame G0-Toleranzfaktor ist über Systemvariablen lesbar.

- In Synchronaktionen oder mit Vorlauf-Stopp im Teileprogramm über die Systemvariable:

\$AC_STOLF Aktiver G0-Toleranzfaktor
 G0-Toleranzfaktor, der bei der Aufbereitung des aktuellen Hauptlaufsatzes wirksam war.

- Ohne Vorlauf-Stopp im Teileprogramm über die Systemvariable:

\$P_STOLF Programmierter G0-Toleranzfaktor

Ist im aktiven Teileprogramm kein Wert mit `STOLF` programmiert, dann liefern diese beiden Systemvariablen den durch MD20560 `$MC_G0_TOLERANCE_FACTOR` eingestellten Wert.

Ist in einem Satz kein Eilgang (G0) aktiv, dann liefern diese Systemvariablen immer den Wert 1.

Beispiel

Programmcode	Kommentar
COMPCAD G645 G1 F10000	; Kompressor-Funktion COMPCAD
X... Y... Z...	; Hier wirken die Maschinen- und Settingdaten.
X... Y... Z...	
X... Y... Z...	
G0 X... Y... Z...	
G0 X... Y... Z...	; Hier wirkt das Maschinendatum <code>\$MC_G0_TOLERANCE_FACTOR</code> (z.B. =3), also eine Überschleiftoleranz von <code>\$MC_G0_TOLERANCE_FACTOR*\$MA_COMPRESS_POS_TOL</code> .
CTOL=0.02	
STOLF=4	
G1 X... Y... Z...	; Ab hier wirkt eine Konturtoleranz von 0,02mm.
X... Y... Z...	
X... Y... Z...	
G0 X... Y... Z...	
X... Y... Z...	; Ab hier wirkt ein G0-Toleranzfaktor von 4, also eine Konturtoleranz von 0,08mm.

8.13 Satzwechselverhalten bei aktiver Kopplung (CPBC)

Funktion

Mit dem Befehl CPBC wird das Satzwechselkriterium vorgegeben das erfüllt sein muss, damit im Teileprogramm bei aktiver Kopplung ein Satzwechsel durchgeführt wird.

Syntax

CPBC [<Folgeachse>] = <Kriterium>

Bedeutung

CPBC:	Satzwechselkriterium bei aktiver Kopplung
<Folgeachse>:	Achsbezeichner der Folgeachse
<Kriterium>:	Satzwechselkriterium
Typ:	STRING
Wert	Bedeutung: Satzwechsel erfolgt
"NOC"	unabhängig vom Kopplungszustand
"IPOSTOP"	bei sollwertseitigem Synchronlauf
"COARSE"	bei istwertseitigem Synchronlauf "Grob"
"FINE"	bei istwertseitigem Synchronlauf "Fein"

Beispiel

Programmcode

```
; Satzwechsel erfolgt bei:  
; - Kopplung zu Folgeachse X2 == aktiv  
; - sollwertseitiger Synchronlauf == aktiv  
CPBC[X2]="IPOSTOP"
```

Achskopplungen

9.1 Mitschleppen (TRAILON, TRAILOF)

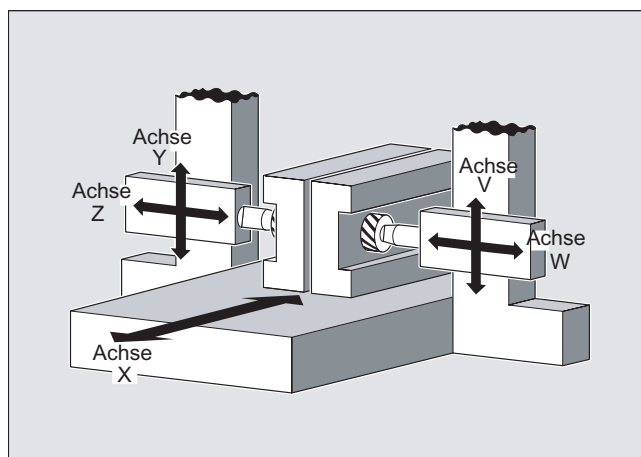
Funktion

Beim Bewegen einer definierten Leitachse fahren ihr zugeordnete Mitschleppachsen (= Folgeachsen) unter Berücksichtigung eines Koppelfaktors die von der Leitachse abgeleiteten Verfahrswege ab.

Leitachse und Folgeachsen bilden zusammen einen Mitschleppverband.

Anwendungsbereiche

- Verfahren einer Achse durch eine simulierte Achse. Die Leitachse ist eine simulierte Achse und die Mitschleppachse eine reale Achse. Damit kann die reale Achse mit Berücksichtigung des Koppelfaktors verfahren werden.
- Zweiseitenbearbeitung mit 2 Mitschleppverbänden:
 1. Leitachse Y, Mitschleppachse V
 2. Leitachse Z, Mitschleppachse W



Syntax

```
TRAILON(<Folgeachse>, <Leitachse>, <Koppelfaktor>)
TRAILOF(<Folgeachse>, <Leitachse>, <Leitachse 2>)
TRAILOF(<Folgeachse>)
```

Bedeutung

TRAILON	Befehl zum Einschalten und Definieren eines Mitschleppverbandes
	Wirksamkeit: modal
<Folgeachse>	Parameter 1: Achsbezeichnung der Mitschleppachse
	Hinweis: Eine Mitschleppachse kann auch Leitachse für weitere Mitschleppachsen sein. Auf diese Weise können unterschiedliche Mitschleppverbände aufgebaut werden.
<Leitachse>	Parameter 2: Achsbezeichnung der Leitachse
<Koppelfaktor>	Parameter 3: Koppelfaktor
	Der Koppelfaktor gibt das gewünschte Verhältnis der Wege von Mitschleppachse und Leitachse an: <Koppelfaktor> = Weg der Mitschleppachse / Weg der Leitachse Typ: REAL Voreinstellung: 1 Die Eingabe eines negativen Wertes bewirkt eine entgegengesetzte Verfahrbewegung der Leit- und Mitschleppachse. Wird der Koppelfaktor bei der Programmierung nicht angegeben, so gilt automatisch der Koppelfaktor 1.
TRAILOF	Befehl zum Ausschalten eines Mitschleppverbandes
	Wirksamkeit: modal
	TRAILOF mit 2 Parametern schaltet nur die Kopplung zur angegebenen Leitachse aus: TRAILOF (<Folgeachse>, <Leitachse>)
	Besitzt eine Mitschleppachse 2 Leitachsen, kann zum Ausschalten der beiden Kopplungen TRAILOF mit 3 Parametern aufgerufen werden: TRAILOF (<Folgeachse>, <Leitachse>, <Leitachse 2>)
	Das gleiche Ergebnis liefert die Programmierung von TRAILOF ohne Angabe einer Leitachse: TRAILOF (<Folgeachse>)

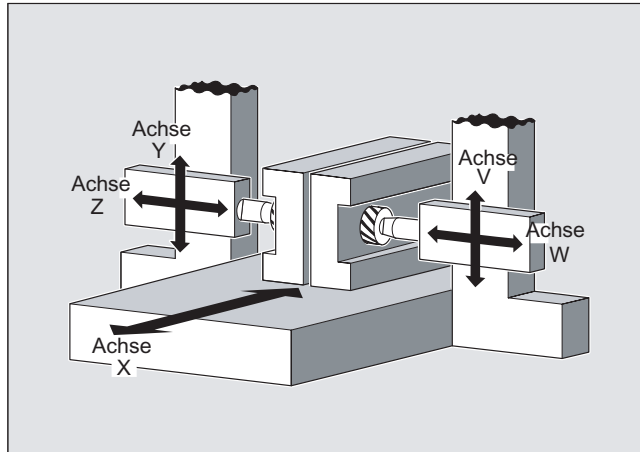
Hinweis

Das Mitschleppen erfolgt immer im Basiskoordinatensystem (BKS).

Die Anzahl der gleichzeitig aktivierbaren Mitschleppverbände wird nur begrenzt durch die Kombinationsmöglichkeiten der an der Maschine vorhandenen Achsen.

Beispiel

Das Werkstück soll zweiseitig mit der dargestellten Achskonstellation bearbeitet werden. Dazu bilden Sie 2 Mitschleppverbände.



Programmcode	Kommentar
...	
N100 TRAILON(V,Y)	; Einschalten des 1. Mitschleppverbandes
N110 TRAILON(W,Z,-1)	; Einschalten des 2. Mitschleppverbandes. Koppelfaktor negativ: Mitschleppachse fährt jeweils in entgegengesetzter Richtung wie Leitachse.
N120 G0 Z10	; Zustellung der Z- und W-Achse in entgegengesetzter Achsrichtung.
N130 G0 Y20	; Zustellung der Y- und V-Achse in gleicher Achsrichtung.
...	
N200 G1 Y22 V25 F200	; Überlagerung einer abhängigen und unabhängigen Bewegung der Mitschleppachse V.
...	
TRAILOF(V,Y)	; Ausschalten des 1. Mitschleppverbandes.
TRAILOF(W,Z)	; Ausschalten des 2. Mitschleppverbandes.

Weitere Informationen

Achstypen

Ein Mitschleppverband kann aus beliebigen Kombinationen von Linear- und Rundachsen bestehen. Als Leitachse kann dabei auch eine simulierte Achse definiert werden.

Mitschleppachsen

Einer Mitschleppachse können gleichzeitig maximal 2 Leitachsen zugeordnet werden. Die Zuordnung erfolgt in unterschiedlichen Mitschleppverbänden.

Eine Mitschleppachse kann mit allen zur Verfügung stehenden Bewegungsbefehlen programmiert werden ($G_0, G_1, G_2, G_3, \dots$). Zusätzlich zu den unabhängig definierten Wegen fährt die Mitschleppachse die mit den Koppelfaktoren aus ihren Leitachsen abgeleiteten Wege.

Dynamikbegrenzung

Die Dynamikbegrenzung ist abhängig von der Art der Aktivierung des Mitschleppverbandes:

- Aktivierung im Teileprogramm

Erfolgt die Aktivierung im Teileprogramm und sind alle Leitachsen als Programmachsen im aktivierenden Kanal, wird beim Verfahren der Leitachsen die Dynamik aller Mitschleppachsen so berücksichtigt, dass keine Mitschleppachse überlastet wird.

Erfolgt die Aktivierung im Teileprogramm mit Leitachsen, die nicht als Programmachsen im aktivierenden Kanal aktiv sind ($\$AA_TYP \neq 1$), wird beim Verfahren der Leitachsen die Dynamik der Mitschleppachse nicht berücksichtigt. Dadurch kann es bei Mitschleppachsen mit einer geringeren als der für die Kopplung benötigten Dynamik zu einer Überlastung kommen.

- Aktivierung in Synchronaktion

Erfolgt die Aktivierung in einer Synchronaktion, wird beim Verfahren der Leitachsen die Dynamik der Mitschleppachsen nicht berücksichtigt. Dadurch kann es bei Mitschleppachsen mit einer geringeren als der für die Kopplung benötigten Dynamik zu einer Überlastung kommen.

VORSICHT

Wird ein Mitschleppverband

- in Synchronaktionen
- im Teileprogramm mit Leitachsen, die nicht Programmachsen im Kanal der Mitschleppachse sind,

aktiviert, dann liegt es in der besonderen Verantwortung des Anwenders/Maschinenherstellers, geeignete Maßnahmen vorzusehen, damit es durch die Verfahrbewegungen der Leitachse nicht zu einer Überlastung der Mitschleppachsen kommt.

Kopplungsstatus

Der Kopplungsstatus einer Achse kann im Teileprogramm abgefragt werden mit der Systemvariablen:

\$AA_COUP_ACT[<Achse>]

Wert	Bedeutung
0	Keine Kopplung aktiv
8	Mitschleppen aktiv

Restweganzeige der Mitschleppachse bei Modulo-Rundachsen

Sind Leit- und Mitschleppachse Modulo-Rundachsen, summieren sich Verfahrbewegungen in der Leitachse von $n * 360^\circ$ mit $n = 1, 2, 3...$, in der Restweganzeige der Mitschleppachse bis zum Ausschalten der Kopplung auf.

Beispiel: Programmabschnitt mit TRAILON und Leitachse B und Folgeachse C

Programmcode	Kommentar
TRAILON(C,B,1)	; Kopplung einschalten
G0 B0	; Ausgangsposition
	; Restweganzeige am Satzanfang:
G91 B360	; B=360, C=360
G91 B720	; B=720, C=1080
G91 B360	; B=360, C=1440

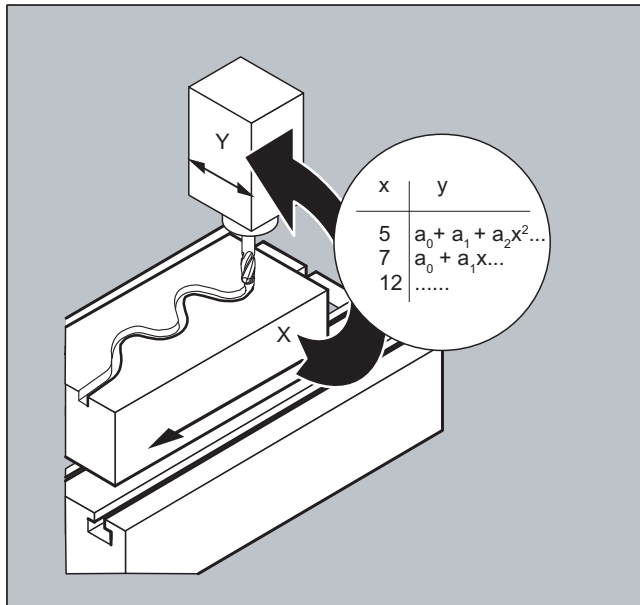
9.2 Kurventabellen (CTAB)

Funktion

Mit Hilfe von Kurventabellen können Positions- und Geschwindigkeitsbeziehungen zwischen zwei Achsen (Leit- und Folgeachse) programmiert werden. Die Kurventabellendefinition erfolgt im Teileprogramm.

Anwendung

Kurventabellen ersetzen mechanische Kurvenscheiben. Die Kurventabelle bildet dabei die Grundlage für die axiale Leitwertkopplung, indem sie den funktionellen Zusammenhang zwischen Leit- und Folgewert schafft: Die Steuerung berechnet bei entsprechender Programmierung aus einander zugeordneten Positionen von Leit- und Folgeachse ein Polynom, das der Kurvenscheibe entspricht.

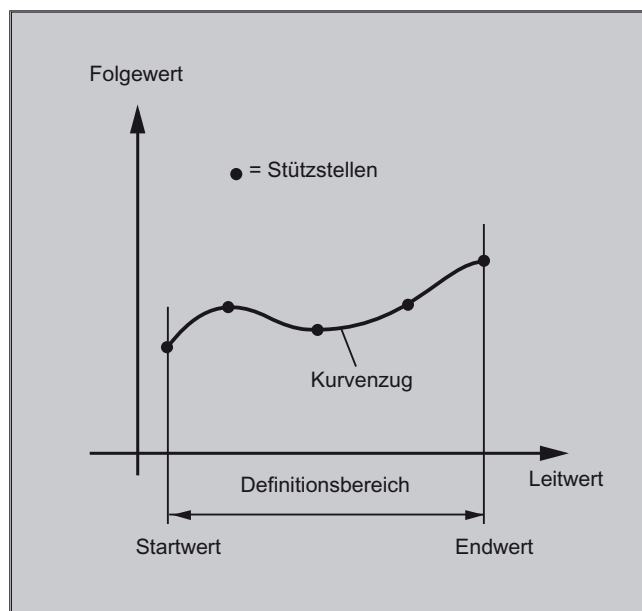


9.2.1 Kurventabellen definieren (CTABDEF, CTABEND)

Funktion

Eine Kurventabelle stellt ein Teileprogramm oder einen Teileprogrammabschnitt dar, welcher durch Voranstellen von `CTABDEF` und den abschließenden Befehl `CTABEND` gekennzeichnet ist.

Innerhalb dieses Teileprogrammabschnitts werden durch Bewegungsanweisungen einzelnen Positionen der Leitachse eindeutige Folgeachspositionen zugeordnet, die als Stützstellen für die Berechnung eines Kurvenzugs in Form eines Polynoms bis zu maximal 5. Grades dienen.



Voraussetzung

Für die Definition von Kurventabellen muss durch entsprechende MD-Projektierung Speicherplatz reserviert sein (→ Maschinenhersteller!).

Syntax

```
CTABDEF (<Folgeachse>, <Leitachse>, <n>, <Periodizität> [, <Speicherort>])
...
CTABEND
```

Bedeutung

<code>CTABDEF ()</code>	Beginn der Kurventabellendefinition
<code>CTABEND</code>	Ende der Kurventabellendefinition
<code><Folgeachse></code>	Achse, deren Bewegung über die Kurventabelle berechnet werden soll
<code><Leitachse></code>	Achse, die die Leitwerte zur Berechnung der Folgeachsbewegung liefert

<code><n></code>	<p>Nummer (ID) der Kurventabelle</p> <p>Die Nummer einer Kurventabelle ist eindeutig und unabhängig vom Speicherort. Es können keine Tabellen mit der gleichen Nummer im statischen und dynamischen NC-Speicher liegen.</p>
<code><Periodizität></code>	<p>Tabellenperiodizität</p> <p>0 Tabelle ist nicht periodisch (wird nur einmal abgearbeitet, auch bei Rundachsen)</p> <p>1 Tabelle ist periodisch bezüglich Leitachse</p> <p>2 Tabelle ist periodisch bezüglich Leitachse und Folgeachse</p>
<code><Speicherort></code>	<p>Angabe des Speicherorts (optional)</p> <p>"SRAM" Die Kurventabelle wird im statischen NC-Speicher angelegt.</p> <p>"DRAM" Die Kurventabelle wird im dynamischen NC-Speicher angelegt.</p> <p>Hinweis: Wenn für diesen Parameter kein Wert programmiert wird, dann wird der mit MD20905 \$MC_CTAB_DEFAULT_MEMORY_TYPE eingestellte Standard-Speicherort verwendet.</p>

Hinweis

Überschreiben

Eine Kurventabelle wird überschrieben, sobald bei einer neuen Tabellendefinition deren Nummer (`<n>`) benutzt wird (Ausnahme: die Kurventabelle ist in einer Achskopplung aktiv oder mit `CTABLOCK` gesperrt). **Beim Überschreiben wird keine entsprechende Warnung ausgegeben!**

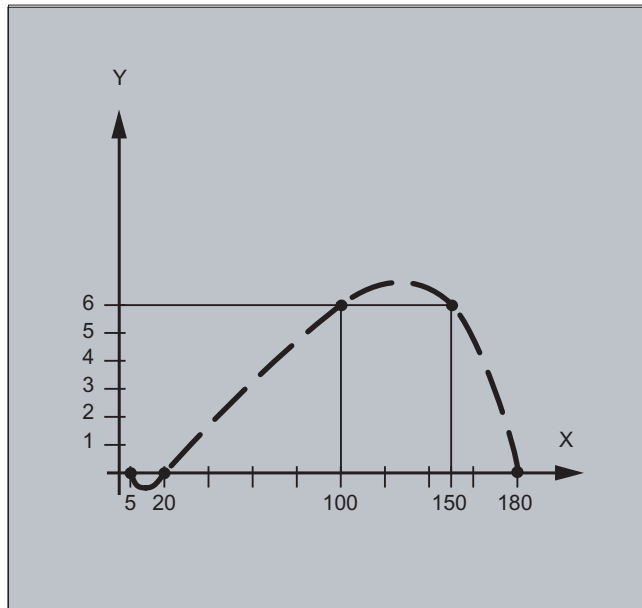
Beispiele

Beispiel 1: Programmabschnitt als Kurventabellendefinition

Ein Programmabschnitt soll unverändert zur Definition einer Kurventabelle benutzt werden. Der darin auftretende Befehl zum Vorlaufstopp `STOPRE` kann stehen bleiben und wird sofort wieder aktiv, sobald der Programmabschnitt nicht mehr zur Tabellendefinition benutzt wird und `CTABDEF` und `CTABEND` entfernt wurden.

Programmcode	Kommentar
...	
<code>CTABDEF (Y,X,1,1)</code>	<code>; Definition einer Kurventabelle.</code>
...	
<code>IF NOT (\$P_CTABDEF)</code>	
<code>STOPRE</code>	
<code>ENDIF</code>	
...	
<code>CTABEND</code>	

Beispiel 2: Definition einer nichtperiodischen Kurventabelle



Programmcode	Kommentar
N100 CTABDEF(Y,X,3,0)	; Beginn der Definition einer nichtperiodischen Kurventabelle mit der Nummer 3.
N110 X0 Y0	; 1.Bewegungsanweisung, legt Startwerte und 1. Stützstelle fest: Leitwert: 0, Folgewert: 0
N120 X20 Y0	; 2.Stützstelle: Leitwert: 0...20, Folgewert: Startwert...0
N130 X100 Y6	; 3.Stützstelle: Leitwert: 20...100, Folgewert: 0...6
N140 X150 Y6	; 4.Stützstelle: Leitwert: 100...150, Folgewert: 6...6
N150 X180 Y0	; 5.Stützstelle: Leitwert: 150...180, Folgewert: 6...0
N200 CTABEND	; Ende der Definition. Die Kurventabelle wird in ihrer internen Darstellung als Polynom maximal 5.Grades erzeugt. Die Berechnung des Kurvenzugs mit den angegebenen Stützstellen ist abhängig von der modal gewählten Interpolationsart (Kreis-, Linear-, Spline-Interpolation). Der Teileprogrammzustand vor Beginn der Definition wird wiederhergestellt.

Beispiel 3: Definition einer periodischen Kurventabelle

Definition einer periodischen Kurventabelle mit Nummer 2, Leitwertbereich von 0 bis 360, Folgeachsbewegung von 0 nach 45 und zurück nach 0:

Programmcode	Kommentar
N10 DEF REAL DEPPPOS	
N20 DEF REAL GRADIENT	
N30 CTABDEF(Y,X,2,1)	; Beginn der Definition.
N40 G1 X=0 Y=0	
N50 POLY	
N60 PO[X]=(45.0)	
N70 PO[X]=(90.0) PO[Y]=(45.0,135.0,-90)	
N80 PO[X]=(270.0)	
N90 PO[X]=(315.0) PO[Y]=(0.0,-135.0,90)	
N100 PO[X]=(360.0)	
N110 CTABEND	; Ende der Definition.
;Test der Kurve durch Kopplung von Y an X:	
N120 G1 F1000 X0	
N130 LEADON(Y,X,2)	
N140 X360	
N150 X0	
N160 LEADOF(Y,X)	
N170 DEPPPOS=CTAB(75.0,2,GRADIENT)	; Lesen der Tabellenfunktion beim Leitwert 75.0.
N180 G0 X75 Y=DEPPPOS	; Positionieren von Leit- und Folgeachse.
;Nach Einschalten der Kopplung ist kein Synchronisieren der Folgeachse nötig.	
N190 LEADON(Y,X,2)	
N200 G1 X110 F1000	
N210 LEADOF(Y,X)	
N220 M30	

Weitere Informationen**Start- und Endwert der Kurventabelle**

Als Startwert für den Beginn des Definitionsbereichs der Kurventabelle gilt die erste Angabe von zusammengehörigen Achspositionen (die erste Bewegungsanweisung) innerhalb der Kurventabellendefinition. Der Endwert des Definitionsbereichs der Kurventabelle wird entsprechend durch den letzten Verfahrbefehl bestimmt.

Verfügbarer Sprachumfang

Innerhalb der Definition der Kurventabelle steht der gesamte NC-Sprachumfang zur Verfügung.

Hinweis

Folgende Angaben sind in Kurventabellendefinitionen nicht zulässig:

- Vorlaufstopp
 - Sprünge in der Leitachsenbewegung (z. B. beim Wechsel von Transformationen)
 - Bewegungsanweisung allein für die Folgeachse
 - Bewegungsumkehr der Leitachse, d. h. Position der Leitachse muss immer eindeutig sein
 - CTABDEF- und CTABEND-Anweisung in unterschiedlichen Programmebenen.
-

Wirksamkeit von modalen Anweisungen

Sämtliche modal wirksamen Anweisungen, die innerhalb der Kurventabellendefinition getroffen werden, sind mit Abschluss der Tabellendefinition ungültig. Das Teileprogramm, in dem die Tabellendefinition erfolgt, befindet sich damit vor und nach der Tabellendefinition im gleichen Zustand.

Zuweisungen an R-Parameter

Zuweisungen an R-Parameter innerhalb der Tabellendefinition werden nach CTABEND zurückgesetzt.

Beispiel:

Programmcode	Kommentar
...	
R10=5 R11=20	; R10=5
...	
CTABDEF	
G1 X=10 Y=20 F1000	
R10=R11+5	; R10=25
X=R10	
CTABEND	
...	; R10=5

Aktivierung von ASPLINE, BSPLINE, CSPLINE

Wird innerhalb einer Kurventabellendefinition `CTABDEF ... CTABEND` ein `ASPLINE`, `BSPLINE` oder `CSPLINE` aktiviert, so sollte vor dieser Spline-Aktivierung mindestens ein Startpunkt programmiert werden. Eine sofortige Aktivierung nach `CTABDEF` sollte vermieden werden, da sonst der Spline von der aktuellen Achsposition vor der Kurventabellendefinition abhängt.

Beispiel:

Programmcode

```
...
CTABDEF(Y,X,1,0)
X0 Y0
ASPLINE
X=5 Y=10
X10 Y40
...
CTABEND
```

Wiederholte Verwendung von Kurventabellen

Der über die Kurventabelle berechnete funktionelle Zusammenhang von Leit- und Folgeachse bleibt unter der gewählten Tabellenummer über das Teileprogrammende und über POWER OFF hinaus erhalten, falls die Tabelle im statischen NC-Speicher (SRAM) abgelegt ist.

Eine Tabelle, die im dynamischen Speicher (DRAM) angelegt wurde, wird bei POWER ON gelöscht und muss eventuell noch einmal erzeugt werden.

Die einmal erstellte Kurventabelle lässt sich auf beliebige Achskombinationen von Leit- und Folgeachse anwenden und ist unabhängig davon, welche Achsen zur Erstellung der Kurventabelle benutzt wurden.

Überschreiben von Kurventabellen

Eine Kurventabelle wird überschrieben, sobald bei einer erneuten Tabellendefinition deren Nummer benutzt wird.

Ausnahme: Eine Kurventabelle ist in einer Achskopplung aktiv oder mit `CTABLOCK` gesperrt.

Hinweis

Beim Überschreiben von Kurventabellen wird keine entsprechende Warnung ausgegeben!

Kurventabellendefinition aktiv?

Mit der Systemvariablen `$P_CTABDEF` kann aus dem Teileprogramm heraus jederzeit abgefragt werden, ob eine Kurventabellendefinition aktiv ist.

Aufheben der Kurventabellendefinition

Der Teileprogrammabschnitt ist nach Ausklammern der Anweisungen zur Kurventabellendefinition wieder als reales Teileprogramm verwendbar.

Laden von Kurventabellen über "Abarbeiten von Extern"

Beim externen Abarbeiten von Kurventabellen muss die Größe des Nachladebuffers (DRAM) über MD18360 \$MN_MM_EXT_PROG_BUFFER_SIZE so gewählt werden, dass die gesamte Kurventabellendefinition gleichzeitig im Nachladebuffer abgelegt werden kann. Die Teileprogrammbearbeitung wird anderenfalls mit einem Alarm abgebrochen.

Sprünge der Folgeachse

Abhängig von der Einstellung im Maschinendatum:
MD20900 \$MC_CTAB_ENABLE_NO_LEADMOTION
können Sprünge der Folgeachse bei fehlender Bewegung der Leitachse toleriert werden.

9.2.2 Vorhandensein einer Kurventabelle prüfen (CTABEXISTS)

Funktion

Mit dem Befehl `CTABEXISTS` kann geprüft werden, ob eine bestimmte Kurventabellennummer im NC-Speicher vorhanden ist.

Syntax

`CTABEXISTS (<n>)`

Bedeutung

<code>CTABEXISTS</code>	Prüft, ob die Kurventabelle mit Nummer <code><n></code> im statischen oder dynamischen NC-Speicher vorhanden ist
	0 Tabelle existiert nicht
	1 Tabelle existiert
<code><n></code>	Nummer (ID) der Kurventabelle

9.2.3 Kurventabellen löschen (CTABDEL)

Funktion

Mit `CTABDEL` können Kurventabellen gelöscht werden.

Hinweis

Kurventabellen, die in einer Achskopplung aktiv sind, können nicht gelöscht werden.

Syntax

```
CTABDEL (<n>)
CTABDEL (<n>, <m>)
CTABDEL (<n>, <m>, <Speicherort>)
CTABDEL ()
CTABDEL (, , <Speicherort>)
```

Bedeutung

<code>CTABDEL</code>	Befehl zum Löschen von Kurventabellen
<code><n></code>	Nummer (ID) der zu löschenden Kurventabelle Beim Löschen eines Kurventabellenbereichs <code>CTABDEL (<n>, <m>)</code> wird mit <code><n></code> die Nummer der ersten Kurventabelle des Bereichs angegeben.
<code><m></code>	Beim Löschen eines Kurventabellenbereichs <code>CTABDEL (<n>, <m>)</code> wird mit <code><m></code> die Nummer der letzten Kurventabelle des Bereichs angegeben. <code><m></code> muss größer <code><n></code> sein!
<code><Speicherort></code>	Angabe des Speicherorts (optional) Beim Löschen ohne Speicherort-Angabe werden die angegebenen Kurventabellen im statischen und dynamischen NC-Speicher gelöscht. Beim Löschen mit Speicherort-Angabe werden von den angegebenen Kurventabellen nur diejenigen gelöscht, die im angegebenen Speicher liegen. Die übrigen bleiben bestehen. "SRAM" Löschen im statischen NC-Speicher "DRAM" Löschen im dynamischen NC-Speicher

Wird `CTABDEL` ohne Angabe der zu löschenden Kurventabelle programmiert, dann werden **alle** Kurventabellen bzw. alle Kurventabellen im angegebenen Speicher gelöscht:

<code>CTABDEL ()</code>	Löscht alle Kurventabellen im statischen und dynamischen NC-Speicher
<code>CTABDEL (, , "SRAM")</code>	Löscht alle Kurventabellen im statischen NC-Speicher
<code>CTABDEL (, , "DRAM")</code>	Löscht alle Kurventabellen im dynamischen NC-Speicher

Hinweis

Wenn beim Mehrfachlöschen `CTABDEL (<n>, <m>)` oder `CTABDEL ()` wenigstens eine der zu löschenden Kurventabellen in einer Kopplung aktiv ist, dann wird der Löschbefehl nicht ausgeführt, d. h. **keine** der adressierten Kurventabellen wird gelöscht.

9.2.4 Kurventabellen gegen Löschen und Überschreiben sperren (CTABLOCK, CTABUNLOCK)

Funktion

Kurventabellen können durch Setzen von Sperren vor unbeabsichtigtem Löschen und Überschreiben geschützt werden. Eine gesetzte Sperre kann jederzeit auch wieder aufgehoben werden.

Syntax

Sperre setzen:

```
CTABLOCK (<n>)
CTABLOCK (<n>, <m>)
CTABLOCK (<n>, <m>, <Speicherort>)
CTABLOCK ()
CTABLOCK (, , <Speicherort>)
```

Sperre aufheben:

```
CTABUNLOCK (<n>)
CTABUNLOCK (<n>, <m>)
CTABUNLOCK (<n>, <m>, <Speicherort>)
CTABUNLOCK ()
CTABUNLOCK (, , <Speicherort>)
```

Bedeutung

CTABLOCK	Befehl zum Setzen einer Sperre gegen Löschen/Überschreiben
CTABUNLOCK	Befehl zum Aufheben einer Sperre gegen Löschen/Überschreiben
	CTABUNLOCK gibt die mit CTABLOCK gesperrten Kurventabellen wieder frei. Kurventabellen, die in einer aktiven Kopplung wirken, bleiben weiterhin gesperrt und können nicht gelöscht werden. Die Sperre mit CTABLOCK ist aufgehoben, sobald die Sperrung durch die aktive Kopplung mit Deaktivierung der Kopplung aufgehoben wird. Damit kann diese Tabelle gelöscht werden. Ein nochmaliger CTABUNLOCK-Aufruf ist nicht notwendig.
<n>	Nummer (ID) der zu sperrenden/entsperrenden Kurventabelle
	Beim Sperren/Entsperren eines Kurventabellenbereichs CTABLOCK (<n>, <m>)/CTABUNLOCK (<n>, <m>) wird mit <n> die Nummer der ersten Kurventabelle des Bereichs angegeben.
<m>	Beim Sperren/Entsperren eines Kurventabellenbereichs CTABLOCK (<n>, <m>)/CTABUNLOCK (<n>, <m>) wird mit <m> die Nummer der letzten Kurventabelle des Bereichs angegeben.
	<m> muss größer <n> sein!

<Speicherort> Angabe des Speicherorts (optional)

Beim Setzen/Aufheben einer Sperre **ohne** Speicherort-Angabe werden die angegebenen Kurventabellen im statischen und dynamischen NC-Speicher gesperrt/entsperrt.

Beim Setzen/Aufheben einer Sperre **mit** Speicherort-Angabe werden von den angegebenen Kurventabellen nur diejenigen gesperrt/entsperrt, die im angegebenen Speicher liegen. Die übrigen werden nicht gesperrt/entsperrt.

"SRAM" Sperre setzen/aufheben im **statischen** NC-Speicher

"DRAM" Sperre setzen/aufheben im **dynamischen** NC-Speicher

Wird CTABLOCK/CTABUNLOCK ohne Angabe der zu sperrenden/entsperrenden Kurventabelle programmiert, dann werden **alle** Kurventabellen bzw. alle Kurventabellen im angegebenen Speicher gesperrt/entsperrt:

CTABLOCK ()	Sperrt alle Kurventabellen im statischen und dynamischen NC-Speicher
CTABLOCK (, , "SRAM")	Sperrt alle Kurventabellen im statischen NC-Speicher
CTABLOCK (, , "DRAM")	Sperrt alle Kurventabellen im dynamischen NC-Speicher
CTABUNLOCK ()	Entsperrt alle Kurventabellen im statischen und dynamischen NC-Speicher
CTABUNLOCK (, , "SRAM")	Entsperrt alle Kurventabellen im statischen NC-Speicher
CTABUNLOCK (, , "DRAM")	Entsperrt alle Kurventabellen im dynamischen NC-Speicher

9.2.5 Kurventabellen: Tabelleneigenschaften ermitteln (CTABID, CTABISLOCK, CTABMEMTYP, CTABPERIOD)

Funktion

Mit diesen Befehlen können wichtige Eigenschaften einer Kurventabelle (Tabellenummer, Sperrzustand, Speicherort, Periodizität) abgefragt werden.

Syntax

```
CTABID (<p>)
CTABID (<p>, <Speicherort>)
CTABISLOCK (<n>)
CTABMEMTYP (<n>)
TABPERIOD (<n>)
```

Bedeutung

CTABID	<p>Liefert die Tabellennummer, die im angegebenen Speicher als die <p>-te Kurventabelle eingetragen ist.</p> <p>Beispiel:</p> <p>CTABID(1, "SRAM") liefert die Nummer der ersten Kurventabelle im statischen NC-Speicher. Die erste Kurventabelle entspricht dabei der Kurventabelle mit der höchsten Tabellennummer.</p> <p>Hinweis:</p> <p>Wird zwischen aufeinander folgenden Aufrufen von CTABID die Reihenfolge der Kurventabellen im Speicher geändert, z. B. durch Löschen von Kurventabellen mit CTABDEL, kann CTABID(<p>, ...) mit derselben Nummer <p> eine andere Kurventabelle liefern als vorher.</p>
CTABISLOCK	<p>Gibt den Sperrzustand der Kurventabelle mit Nummer <n> zurück:</p> <ul style="list-style-type: none"> 0 Tabelle ist nicht gesperrt 1 Tabelle ist gesperrt durch CTABLOCK 2 Tabelle ist gesperrt durch aktive Kopplung 3 Tabelle ist gesperrt durch CTABLOCK und aktive Kopplung -1 Tabelle existiert nicht
CTABMEMTYP	<p>Liefert den Speicherort der Kurventabelle mit Nummer <n>:</p> <ul style="list-style-type: none"> 0 Tabelle im statischen NC-Speicher 1 Tabelle im dynamischen NC-Speicher -1 Tabelle existiert nicht
CTABPERIOD	<p>Liefert die Periodizität der Kurventabelle mit Nummer <n>:</p> <ul style="list-style-type: none"> 0 Tabelle ist nicht periodisch 1 Tabelle ist periodisch in der Leitachse 2 Tabelle ist periodisch in der Leit- und Folgeachse -1 Tabelle existiert nicht
<p>	Eintragsnummer im Speicher
<n>	Nummer (ID) der Kurventabelle
<Speicherort>	<p>Angabe des Speicherorts (optional)</p> <p>"SRAM" Statischer NC-Speicher</p> <p>"DRAM" Dynamischer NC-Speicher</p> <p>Hinweis:</p> <p>Wenn für diesen Parameter kein Wert programmiert wird, dann wird der mit MD20905 \$MC_CTAB_DEFAULT_MEMORY_TYPE eingestellte Standard-Speicherort verwendet.</p>

9.2.6 Kurventabellenwerte lesen (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX)

Funktion

Folgende Kurventabellenwerte können im Teileprogramm gelesen werden:

- Folgeachs- und Leitachswerte am Anfang und Ende einer Kurventabelle
- Folgeachswerte am Anfang und Ende eines Kurvensegments
- Folgeachswert zu einem Leitachswert
- Leitachswert zu einem Folgeachswert
- Minimal- und Maximalwert der Folgeachse
 - im gesamten Definitionsbereich der Kurventabelle
 - oder
 - in einem definierten Kurventabellenintervall

Syntax

```
CTABTSV (<n>, <Gradient> [, <Folgeachse>])
CTABTEV (<n>, <Gradient> [, <Folgeachse>])
CTABTSP (<n>, <Gradient> [, <Leitachse>])
CTABTEP (<n>, <Gradient> [, <Leitachse>])
CTABSSV (<Leitwert>, <n>, <Gradient> [, <Folgeachse>])
CTABSEV (<Leitwert>, <n>, <Gradient> [, <Folgeachse>])
CTAB (<Leitwert>, <n>, <Gradient> [, <Folgeachse>, <Leitachse>])
CTABINV (<Folgewert>, <Näherungswert>, <n>, <Gradient> [, <Folgeachse>,
<Leitachse>])
CTABTMIN (<n> [, <Folgeachse>])
CTABTMAX (<n> [, <Folgeachse>])
CTABTMIN (<n>, <a>, <b> [, <Folgeachse>, <Leitachse>])
CTABTMAX (<n>, <a>, <b> [, <Folgeachse>, <Leitachse>])
```

Bedeutung

CTABTSV:	Folgeachswert am Anfang der Kurventabelle Nr. <n> lesen
CTABTEV:	Folgeachswert am Ende der Kurventabelle Nr. <n> lesen
CTABTSP:	Leitachswert am Anfang der Kurventabelle Nr. <n> lesen
CTABTEP:	Leitachswert am Ende der Kurventabelle Nr. <n> lesen
CTABSSV:	Folgeachswert am Anfang des zum angegebenen Leitachswert (<Leitwert>) gehörenden Kurvensegments lesen
CTABSEV:	Folgeachswert am Ende des zum angegebenen Leitachswert (<Leitwert>) gehörenden Kurvensegments lesen
CTAB:	Folgeachswert zum angegebenen Leitachswert (<Leitwert>) lesen
CTABINV:	Leitachswert zum angegebenen Folgeachswert (<Folgewert>) lesen

CTABTMIN:	<p>Minimalwert der Folgeachse bestimmen:</p> <ul style="list-style-type: none"> im gesamten Definitionsbereich der Kurventabelle oder in einem definierten Intervall $\langle a \rangle \dots \langle b \rangle$
CTABTMAX:	<p>Maximalwert der Folgeachse bestimmen:</p> <ul style="list-style-type: none"> im gesamten Definitionsbereich der Kurventabelle oder in einem definierten Intervall $\langle a \rangle \dots \langle b \rangle$
$\langle n \rangle$:	Nummer (ID) der Kurventabelle
$\langle \text{Gradient} \rangle$:	Im Parameter $\langle \text{Gradient} \rangle$ wird die Steigung der Kurventabellenfunktion an der ermittelten Position zurückgegeben
$\langle \text{Folgeachse} \rangle$:	Achse, deren Bewegung über die Kurventabelle berechnet werden soll (optional)
$\langle \text{Leitachse} \rangle$:	Achse, die die Leitwerte zur Berechnung der Folgeachsbewegung liefert (optional)
$\langle \text{Folgewert} \rangle$:	Folgeachswert zum Lesen des zugehörigen Leitachswerts bei CTABINV
$\langle \text{Leitwert} \rangle$:	<p>Leitachswert:</p> <ul style="list-style-type: none"> zum Lesen des zugehörigen Folgeachswerts bei CTAB oder für die Auswahl des Kurvensegments bei CTABSSV/CTABSEV
$\langle \text{Näherungswert} \rangle$:	Die Zuordnung eines Leitachswerts zu einem Folgeachswert bei CTABINV muss nicht immer eindeutig sein. CTABINV benötigt daher als Parameter einen Näherungswert für den erwarteten Leitachswert.
$\langle a \rangle$:	Untere Grenze des Leitwertintervalls bei CTABTMIN/CTABTMAX
$\langle b \rangle$:	Obere Grenze des Leitwertintervalls bei CTABTMIN/CTABTMAX
	<p>Hinweis:</p> <p>Das Leitwertintervall $\langle a \rangle \dots \langle b \rangle$ muss innerhalb des Definitionsbereichs der Kurventabelle liegen.</p>

Beispiele

Beispiel 1:

Folgeachs- und Leitachswerte am Anfang und Ende der Kurventabelle sowie Minimal- und Maximalwert der Folgeachse im gesamten Definitionsbereich der Kurventabelle bestimmen.

Programmcode	Kommentar
N10 DEF REAL STARTPOS	
N20 DEF REAL ENDPOS	
N30 DEF REAL STARTPARA	
N40 DEF REAL ENDPARA	
N50 DEF REAL MINVAL	
N60 DEF REAL MAXVAL	
N70 DEF REAL GRADIENT	
...	
N100 CTABDEF(Y,X,1,0)	; Beginn der Tabellendefinition
N110 X0 Y10	; Startposition 1.Tabellensegment
N120 X30 Y40	; Endposition 1.Tabellensegment = Startposition 2.Tabellensegment
N130 X60 Y5	; Endposition 2.Tabellensegment = ...
N140 X70 Y30	
N150 X80 Y20	
N160 CTABEND	; Ende der Tabellendefinition.
...	
N200 STARTPOS=CTABTSV(1,GRADIENT)	; Folgeachswert am Kurventabellenanfang = 10
N210 ENDPOS=CTABTEV(1,GRADIENT)	; Folgeachswert am Kurventabellenende = 20
N220 STARTPARA=CTABTSP(1,GRADIENT)	; Leitachswert am Kurventabellenanfang = 0
N230 ENDPARA=CTABTEP(1,GRADIENT)	; Leitachswert am Kurventabellenende = 80
N240 MINVAL=CTABTMIN(1)	; Minimalwert der Folgeachse bei Y=5
N250 MAXVAL=CTABTMAX(1)	; Maximalwert der Folgeachse bei Y=40

Beispiel 2:

Bestimmung der Folgeachswerte am Anfang und Ende des zum Leitachswert X=30 gehörenden Kurvensegments.

Programmcode	Kommentar
N10 DEF REAL STARTPOS	
N20 DEF REAL ENDPOS	
N30 DEF REAL GRADIENT	
...	
N100 CTABDEF(Y,X,1,0)	; Beginn der Tabellendefinition.
N110 X0 Y0	; Startposition 1.Tabellensegment
N120 X20 Y10	; Endposition 1.Tabellensegment = Startposition 2.Tabellensegment
N130 X40 Y40	Endposition 2.Tabellensegment = ...
N140 X60 Y10	
N150 X80 Y0	
N160 CTABEND	; Ende der Tabellendefinition.
...	
N200 STARTPOS=CTABSSV(30.0,1,GRADIENT)	; Startposition Y im 2.Segment = 10
N210 ENDPOS=CTABSEV(30.0,1,GRADIENT)	; Endposition Y im 2.Segment = 40

Weitere Informationen

Verwendung in Synchronaktionen

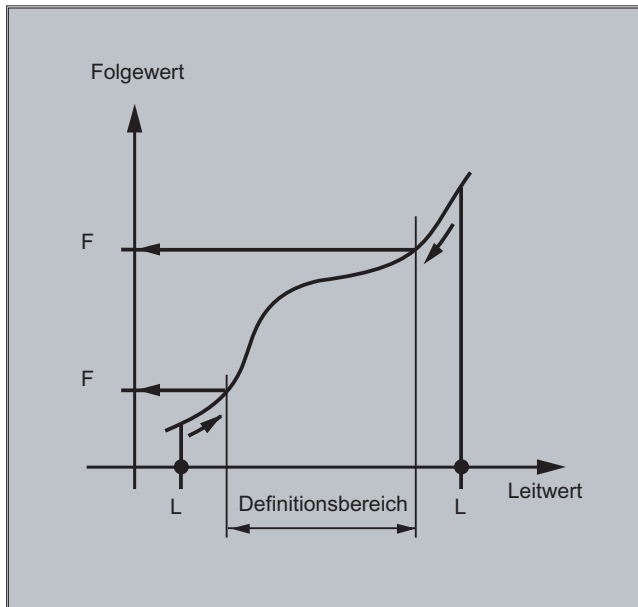
Alle Befehle zum Lesen von Kurventabellenwerten können auch in Synchronaktionen verwendet werden (siehe auch Kapitel "Bewegungssynchronaktionen").

Bei Verwendung der Befehle CTABINV, CTABTMIN und CTABTMAX ist darauf zu achten, dass:

- zum Ausführungszeitpunkt ausreichend NC-Leistung verfügbar ist
- oder
- vor dem Aufruf die Anzahl der Segmente der Kurventabelle abgefragt wird, um gegebenenfalls die betreffende Tabelle unterteilen zu können

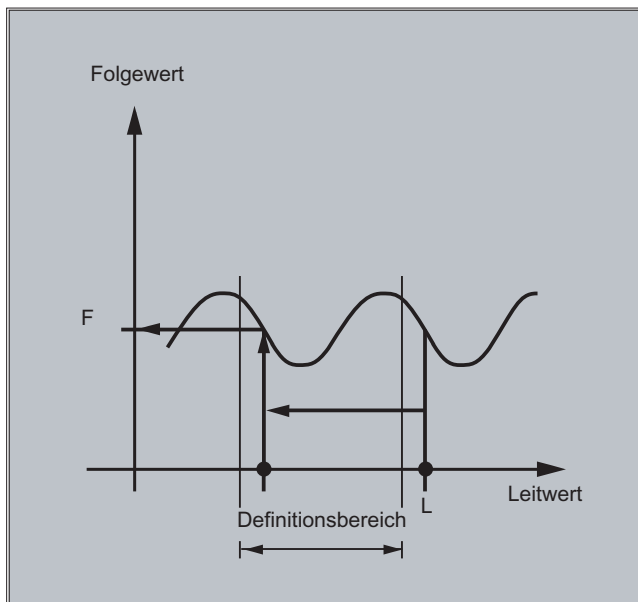
CTAB bei nichtperiodischen Kurventabellen

Liegt der angegebene <Leitwert> außerhalb des Definitionsbereichs, wird als Folgewert die obere bzw. untere Grenze ausgegeben:



CTAB bei periodischen Kurventabellen

Liegt der angegebene <Leitwert> außerhalb des Definitionsbereichs, wird der Leitwert Modulo des Definitionsbereichs bewertet und der entsprechende Folgewert ausgegeben:



Näherungswert für CTABINV

Der Befehl `CTABINV` benötigt einen Näherungswert für den erwarteten Leitwert. `CTABINV` gibt den Leitwert zurück, der dem Näherungswert am nächsten liegt. Der Näherungswert kann z. B. der Leitwert aus dem vorherigen Interpolationstakt sein.

Steigung der Kurventabellenfunktion

Die Ausgabe der Steigung (`<Gradient>`) ermöglicht es, die Geschwindigkeit der Leit- oder Folgeachse an der entsprechenden Position zu berechnen.

Angabe der Leit- oder Folgeachse

Die optionale Angabe der Leit- und/oder Folgeachse ist wichtig, falls Leit- und Folgeachse in verschiedenen Längeneinheiten projiziert sind.

CTABSSV, CTABSEV

Die Befehle `CTABSSV` und `CTABSEV` sind in folgenden Fällen **nicht** dazu geeignet, programmierte Segmente abzufragen:

- Kreise oder Evolventen sind programmiert.
- Fasen bzw. Runden mit `CHF/RND` ist aktiv.
- Überschleifen mit `G643` ist aktiv.
- NC-Satz-Kompression mit `COMPON/COMPCURV/COMPCAD` ist aktiv.

9.2.7 Kurventabellen: Ressourcennutzung prüfen (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL)

Funktion

Mit diesen Befehlen hat der Programmierer die Möglichkeit, sich aktuell über die Belegung der Ressourcen für Kurventabellen, Tabellensegmente und Polynome zu informieren.

Syntax

```
CTABNO  
CTABNOMEM(<Speicherort>)  
CTABFNO(<Speicherort>)  
CTABSEGID(<n>, <Speicherort>)  
CTABSEG(<Speicherort>, <Segmentart>)  
CTABFSEG(<Speicherort>, <Segmentart>)  
CTABMSEG(<Speicherort>, <Segmentart>)  
CTABPOLID(<n>)  
CTABPOL(<Speicherort>)  
CTABFPOL(<Speicherort>)  
CTABMPOL(<Speicherort>)
```

Bedeutung

CTABNO	Gesamtanzahl der definierten Kurventabellen bestimmen (im statischen und dynamischen NC-Speicher)
CTABNOMEM	Anzahl der definierten Kurventabellen im angegebenen <Speicherort> bestimmen
CTABFNO	Anzahl der noch möglichen Kurventabellen im angegebenen <Speicherort> bestimmen
CTABSEGID	Anzahl der Kurvensegmente der angegebenen <Segmentart> bestimmen, die von der Kurventabelle mit Nummer <n> verwendet werden
CTABSEG	Anzahl der verwendeten Kurvensegmente der angegebenen <Segmentart> im angegebenen <Speicherort> bestimmen
CTABFSEG	Anzahl der noch möglichen Kurvensegmente der angegebenen <Segmentart> im angegebenen <Speicherort> bestimmen
CTABMSEG	Anzahl der maximal möglichen Kurvensegmente der angegebenen <Segmentart> im angegebenen <Speicherort> bestimmen
CTABPOLID	Anzahl der Kurvenpolynome bestimmen, die von der Kurventabelle mit Nummer <n> verwendet werden
CTABPOL	Anzahl der verwendeten Kurvenpolynome im angegebenen <Speicherort> bestimmen
CTABFPOL	Anzahl der noch möglichen Kurvenpolynome im angegebenen <Speicherort> bestimmen
CTABMPOL	Anzahl der maximal möglichen Kurvenpolynome im angegebenen <Speicherort> bestimmen
<n>	Nummer (ID) der Kurventabelle
<Speicherort>	Angabe des Speicherorts (optional) "SRAM" Statischer NC-Speicher "DRAM" Dynamischer NC-Speicher
	Hinweis: Wenn für diesen Parameter kein Wert programmiert wird, dann wird der mit MD20905 \$MC_CTAB_DEFAULT_MEMORY_TYPE eingestellte Standard-Speicherort verwendet.
<Segmentart>	Angabe der Segmentart (optional) "L" Lineare Segmente "P" Polynomsegmente
	Hinweis: Wenn für diesen Parameter kein Wert programmiert wird, dann wird die Summe aus Linear- und Polynom-Segmenten ausgegeben.

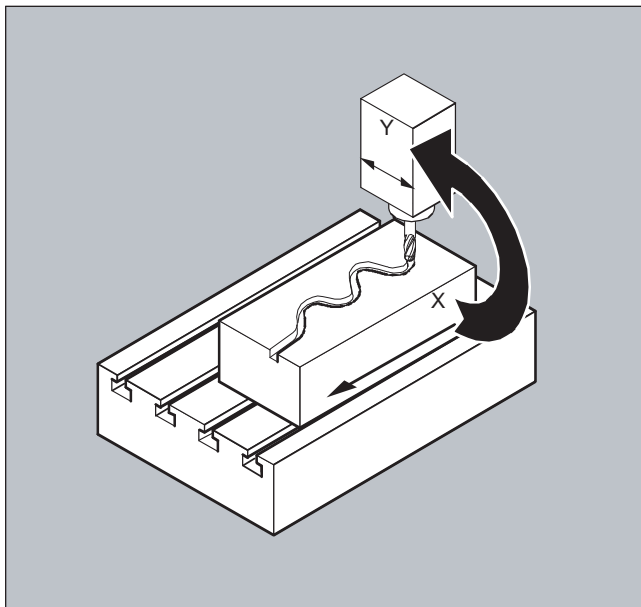
9.3 Axiale Leitwertkopplung (LEADON, LEADOF)

Hinweis

Diese Funktion steht für SINUMERIK 828D nicht zur Verfügung!

Funktion

Bei der axialen Leitwertkopplung werden eine Leit- und eine Folgeachse synchron verfahren. Dabei ist die jeweilige Position der Folgeachse über eine Kurventabelle bzw. ein daraus berechnetes Polynom eindeutig einer - ggf. simulierten - Position der Leitachse zugeordnet.



Leitachse heißt diejenige Achse, die die Eingangswerte für die Kurventabelle liefert. **Folgeachse** heißt die Achse, die die über die Kurventabelle errechneten Positionen einnimmt.

Ist- und Sollwertkopplung

Als Leitwerte, also Ausgangswerte zur Positionsermittlung der Folgeachse können verwendet werden:

- Istwerte der Leitachsposition: Istwertkopplung
- Sollwerte der Leitachsposition: Sollwertkopplung

Die Leitwertkopplung gilt immer im Basiskoordinatensystem.

Zur Erstellung von Kurventabellen siehe Kapitel "Kurventabellen".

Syntax

LEADON (<Folgeachse>, <Leitachse>, <n>)
LEADOF (<Folgeachse>, <Leitachse>)

oder Ausschalten ohne Angabe der Leitachse:

LEADOF (<Folgeachse>)

Die Leitwertkopplung kann sowohl vom Teileprogramm als auch während der Bewegung aus Synchronaktionen heraus ein- und ausgeschaltet werden.

Bedeutung

LEADON:	Leitwertkopplung einschalten
LEADOF:	Leitwertkopplung ausschalten
<Folgeachse>:	Folgeachse
<Leitachse>:	Leitachse
<n>:	Kurventabellen-Nummer
\$SA_LEAD_TYPE:	Umschaltung zwischen Soll- und Istwertkopplung

Leitwertkopplung ausschalten, LEADOF

Mit dem Ausschalten der Leitwertkopplung wird die Folgeachse wieder zur normalen Kommandoachse!

Axiale Leitwertkopplung und verschiedene Betriebszustände, RESET

Abhängig von der Einstellung im Maschinendatum werden Leitwertkopplungen mit RESET ausgeschaltet.

Beispiel Leitwertkopplung aus Synchronaktion

Bei einer Pressenanlage soll eine herkömmliche mechanische Kopplung zwischen einer Leitachse (Stempelwelle) und Achsen eines Transfersystems aus Transferachsen und Hilfsachsen durch ein elektronisches Koppelsystem ersetzt werden.

Es demonstriert, wie bei einer Pressenanlage ein mechanisches Transfersystem durch ein elektronisches Transfersystem ersetzt wird. Die Kopplungs- und Entkopplungsvorgänge sind als **statische Synchronaktionen** realisiert.

Von der Leitachse LW (Stempelwelle) werden Transferachsen und Hilfsachsen als Folgeachsen über Kurventabellen definiert gesteuert.

Folgeachsen

X Vorschub- bzw. Längsachse
YL Schließ- bzw. Querachse
ZL Hubachse
U Walzenvorschub, Hilfsachse
V Richtkopf, Hilfsachse
W Befettung, Hilfsachse

Aktionen

Als Aktionen treten in den Synchronaktionen z. B. auf:

- Einkoppeln, LEADON (<Folgeachse>, <Leitachse>, <Kurventabellen-Nummer>)
- Auskoppeln, LEADOF (<Folgeachse>, <Leitachse>)
- Istwertsetzen, PRESETON (<Achse>, <Wert>)
- Merker setzen, \$AC_MARKER[i]=<Wert>
- Kopplungsart: reeller/virtueller Leitwert
- Anfahren von Achspositionen, POS [<Achse>]=<Wert>

Bedingungen

Als Bedingungen werden digitale schnelle Eingänge, Echtzeitvariablen \$AC_MARKER und Positionsvergleiche, mit dem logischen Operator AND verknüpft, ausgewertet.

Hinweis

Im folgenden Beispiel wurden Zeilenwechsel, Einrückungen und **Fettsatz** ausschließlich dafür verwendet, die Lesbarkeit der Programmierung zu erhöhen. Für die Steuerung ist alles unter einer Zeilennummer stehende einzeilig.

Kommentar

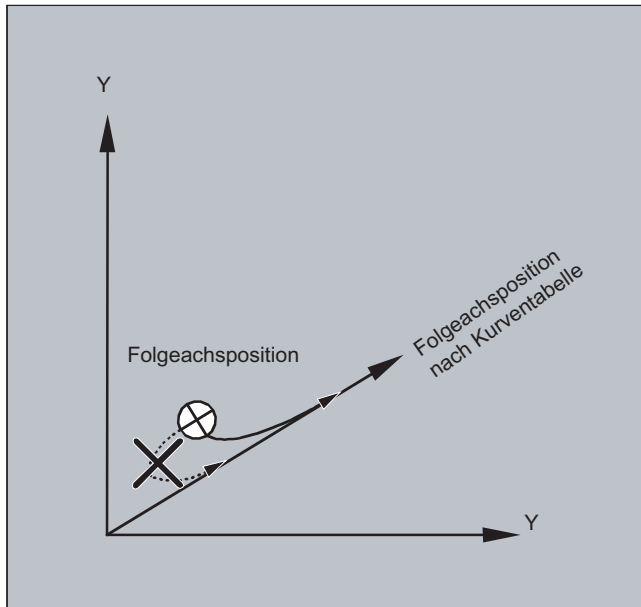
Programmcode	Kommentar
	; Definiert sämtliche statische Synchronaktionen.
	; ****Marker rücksetzen
N2 \$AC_MARKER[0]=0 \$AC_MARKER[1]=0 \$AC_MARKER[2]=0 \$AC_MARKER[3]=0 \$AC_MARKER[4]=0 \$AC_MARKER[5]=0 \$AC_MARKER[6]=0 \$AC_MARKER[7]=0	
	; **** E1 0=>1 Kopplung Transfer EIN
N10 IDS=1 EVERY (\$A_IN[1]==1) AND (\$A_IN[16]==1) AND (\$AC_MARKER[0]==0) DO LEADON(X,LW,1) LEADON(YL,LW,2) LEADON(ZL,LW,3) \$AC_MARKER[0]=1	
	; **** E1 0=>1 Kopplung Walzenvorschub EIN
N20 IDS=11 EVERY (\$A_IN[1]==1) AND (\$A_IN[5]==0) AND (\$AC_MARKER[5]==0) DO LEADON(U,LW,4) PRESETON(U,0) \$AC_MARKER[5]=1	
	; **** E1 0->1 Kopplung Richtkopf EIN
N21 IDS=12 EVERY (\$A_IN[1]==1) AND (\$A_IN[5]==0) AND (\$AC_MARKER[6]==0) DO LEADON(V,LW,4) PRESETON(V,0) \$AC_MARKER[6]=1	
	; **** E1 0->1 Kopplung Befettung EIN
N22 IDS=13 EVERY (\$A_IN[1]==1) AND (\$A_IN[5]==0) AND (\$AC_MARKER[7]==0) DO LEADON(W,LW,4) PRESETON(W,0) \$AC_MARKER[7]=1	
	; **** E2 0=>1 Kopplung AUS
N30 IDS=3 EVERY (\$A_IN[2]==1) DO LEADOF(X,LW) LEADOF(YL,LW) LEADOF(ZL,LW) LEADOF(U,LW) LEADOF(V,LW) LEADOF(W,LW) \$AC_MARKER[0]=0 \$AC_MARKER[1]=0 \$AC_MARKER[3]=0 \$AC_MARKER[4]=0 \$AC_MARKER[5]=0 \$AC_MARKER[6]=0 \$AC_MARKER[7]=0	
....	
N110 G04 F01	
N120 M30	

Beschreibung

Die Leitwertkopplung erfordert die Synchronisation von Leit- und Folgeachse. Diese Synchronisation kann nur erreicht werden, wenn die Folgeachse bei Einschalten der Leitwertkopplung innerhalb des Toleranzbereiches des aus der Kurventabelle berechneten Kurvenzugs steht.

Der Toleranzbereich für die Stellung der Folgeachse ist über Maschinendatum MD 37200: COUPLE_POS_POL_COARSE A_LEAD_TYPE definiert.

Befindet sich die Folgeachse mit dem Einschalten der Leitwertkopplung noch nicht an der entsprechenden Position, wird der Synchronlauf automatisch hergestellt, sobald sich der berechnete Positionssollwert für die Folgeachse der tatsächlichen Folgeachseposition nähert. Die Folgeachse wird dabei während des Synchronisationsvorganges in die Richtung verfahren, die durch die Sollgeschwindigkeit der Folgeachse (berechnet aus Leitachsgeschwindigkeit und nach Kurventabelle CTAB) definiert ist.

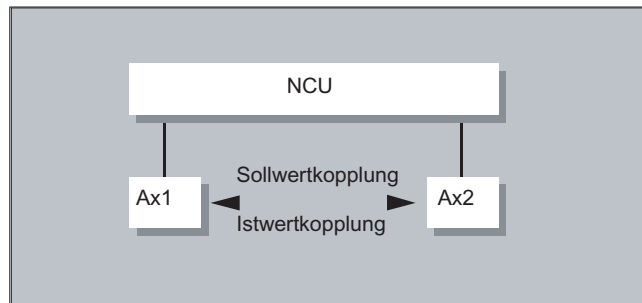


Kein Synchronlauf

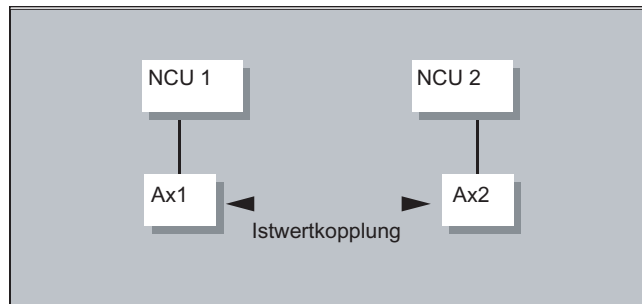
Entfernt sich die berechnete Folgeachssollposition mit Einschalten der Leitwertkopplung von der aktuellen Folgeachseposition, wird kein Synchronlauf hergestellt.

Ist- und Sollwertkopplung

Die Sollwertkopplung liefert im Vergleich zur Istwertkopplung einen besseren Synchronlauf zwischen Leit- und Folgeachse und ist deshalb standardmäßig voreingestellt.



Sollwertkopplung ist nur möglich, wenn Leit- und Folgeachse von derselben NCU interpoliert werden. Bei einer externen Leitachse kann die Folgeachse nur über Istwerte an die Leitachse gekoppelt werden.



Eine **Umschaltung** ist über das Settingdatum \$SA_LEAD_TYPE möglich.

Das Umschalten zwischen Ist- und Sollwertkopplung sollte immer bei Stillstand der Folgeachse erfolgen. Denn nur im Stillstand wird nach dem Umschalten neu synchronisiert.

Anwendungsbeispiel

Das Lesen der Istwerte kann bei großen Maschinenerschütterungen nicht fehlerfrei erfolgen. Beim Einsatz der Leitwertkopplung im Pressentferner kann es daher in den Arbeitsschritten mit größten Erschütterungen notwendig werden, von Istwertkopplung auf Sollwertkopplung umzuschalten.

Leitwertsimulation bei Sollwertkopplung

Über Maschinendatum lässt sich der Interpolator für die Leitachse vom Servo trennen. Damit können bei Sollwertkopplung Sollwerte ohne tatsächliche Bewegung der Leitachse erzeugt werden.

Die über Sollwertkopplung erzeugten Leitwerte sind zur Benutzung z. B. in Synchronaktionen aus folgenden Variablen lesbar:

- | | |
|---------------|--------------------------|
| - \$AA_LEAD_P | Leitwert Position |
| - \$AA_LEAD_V | Leitwert Geschwindigkeit |

Leitwerte erzeugen

Leitwerte können wahlweise mit anderen selbst programmierten Verfahren erzeugt werden. Die so erzeugten Leitwerte werden in die Variable

- \$AA_LEAD_SP	Leitwert Position
- \$AA_LEAD_SV	Leitwert Geschwindigkeit

geschrieben und aus ihnen gelesen. Zur Benutzung dieser Variablen muss das Settingdatum \$SA_LEAD_TYPE = 2 gesetzt werden.

Status der Kopplung

Im NC-Teileprogramm können Sie den Kopplungsstatus mit folgender Systemvariablen abfragen:

\$AA_COUP_ACT[Achse]
0: Keine Kopplung aktiv
16: Leitwertkopplung aktiv

Status-Verwaltung bei Synchronaktionen

Schalt- und Koppelvorgänge werden über Echtzeitvariablen:

\$AC_MARKER[i] = n
verwaltet mit:
i Merker-Nummer
n Statuswert

9.4 Elektronisches Getriebe (EG)

Funktion

Mit Hilfe der Funktion "Elektronisches Getriebe" ist es möglich, die Bewegung einer **Folgeachse** nach linearem Bewegungssatz abhängig von bis zu fünf **Leitachsen** zu steuern. Die Zusammenhänge zwischen den Leitachsen und der Folgeachse sind je Leitachse durch den Koppelfaktor definiert.

Der berechnete Folgeachs-Bewegungsanteil wird aus den einzelnen Leitachsen-Bewegungsanteilen multipliziert mit den jeweiligen Koppelfaktoren durch Addition gebildet. Bei der Aktivierung eines EG-Achsverbundes kann die Synchronisation der Folgeachse auf eine definierte Position veranlasst werden. Ein Getriebeverband kann aus dem Teileprogramm:

- definiert,
- eingeschaltet,
- ausgeschaltet,
- gelöscht

werden.

Die Folgeachsbewegung kann wahlweise abgeleitet werden aus den

- Sollwerten der Leitachsen sowie den
- Istwerten der Leitachsen.

Als Erweiterung können auch nichtlineare Zusammenhänge zwischen den Leitachsen und der Folgeachse über **Kurventabellen** (siehe Kapitel Bahnverhalten) realisiert werden. Elektronische Getriebe können kaskadiert werden, d. h. die Folgeachse eines Elektronischen Getriebes kann Leitachse für ein weiteres Elektronisches Getriebe sein.

9.4.1 Elektronisches Getriebe definieren (EGDEF)

Funktion

Ein EG-Achsverband wird durch die Angabe der Folgeachse und mindestens einer, jedoch höchstens fünf Leitachsen mit dem jeweiligen Kopplungstyp festgelegt.

Voraussetzung

Voraussetzung für eine EG-Achsverband-Definition:

Für die Folgeachse darf noch keine Achskopplung definiert sein (ggf. muss eine bestehende vorher mit `EGDEL` gelöscht werden).

Syntax

EGDEF (Folgeachse, Leitachse1, Kopplungstyp1, Leitachse2, Kopplungstyp2, . . .)

Bedeutung

EGDEF	Definition eines elektronischen Getriebes
Folgeachse	Achse, die von Leitachsen beeinflusst wird
Leitachse1	Achsen, die die Folgeachse beeinflussen
Leitachse5	
Kopplungstyp1	Kopplungstyp
Kopplungstyp5	Der Kopplungstyp muss nicht für alle Leitachsen gleich sein und ist daher für jede Leitachse einzeln anzugeben.
Wert:	Bedeutung:
0	Die Folgeachse wird beeinflusst vom Istwert der entsprechenden Leitachse.
1	Die Folgeachse wird beeinflusst vom Sollwert der entsprechenden Leitachse.

Hinweis

Die Koppelfaktoren werden bei der Definition des EG-Kopplungsverbandes mit Null vorbesetzt.

Hinweis

EGDEF löst Vorlaufstopp aus. Die Getriebedefinition mit EGDEF ist auch dann unverändert zu verwenden, wenn bei Systemen eine oder mehrere Leitachsen über **Kurventabelle** auf die Folgeachse einwirken.

Beispiel

Programmcode	Kommentar
EGDEF(C,B,1,Z,1,Y,1)	; Definition eines EG-Achsverbandes. Die Leitachsen B, Z, Y beeinflussen die Folgeachse C über den Sollwert.

9.4.2 Elektronisches Getriebe einschalten (EGON, EGONSYN, EGONSYNE)

Funktion

Für das Einschalten eines EG-Achsverbandes existieren 3 Varianten.

Syntax

Variante 1:

Der EG-Achsverband wird ohne Synchronisation selektiv eingeschaltet mit:

`EGON (FA, "Satzwechselmodus", LA1, Z1, N1, LA2, Z2, N2, . . . , LA5, Z5, N5)`

Variante 2:

Der EG-Achsverband wird mit Synchronisation selektiv eingeschaltet mit:

`EGONSYN (FA, "Satzwechselmodus", SynPosFA, [, LAi, SynPosLAI, Zi, Ni])`

Variante 3:

Der EG-Achsverband wird mit Synchronisation selektiv eingeschaltet und der Anfahrmodus vorgegeben mit:

`EGONSYNE (FA, "Satzwechselmodus", SynPosFA, Anfahrmodus [, LAi, SynPosLAI, Zi, Ni])`

Bedeutung

Variante 1:

FA
Satzwechselmodus

Folgeachse

Folgende Modi können benutzt werden:

"NOC"	Satzwechsel erfolgt sofort
"FINE"	Satzwechsel erfolgt bei "Synchronlauf fein"
"COARSE"	Satzwechsel erfolgt bei "Synchronlauf grob"
"IPOSTOP"	Satzwechsel erfolgt bei sollwertseitigem Synchronlauf

LA1, . . . LA5
Z1, . . . Z5
N1, . . . N5

Leitachsen

Zähler für den Koppelfaktor i

Nenner für den Koppelfaktor i

Koppelfaktor $i = \text{Zähler } i / \text{Nenner } i$

Es dürfen nur die Leitachsen programmiert werden, die zuvor mit `EGDEF` spezifiziert worden sind. Es muss mindestens eine Leitachse programmiert werden.

Variante 2:

FA	Folgeachse
Satzwechselmodus	Folgende Modi können benutzt werden:
	"NOC" Satzwechsel erfolgt sofort
	"FINE" Satzwechsel erfolgt bei "Synchronlauf fein"
	"COARSE" Satzwechsel erfolgt bei "Synchronlauf grob"
	"IPOSTOP" Satzwechsel erfolgt bei sollwertseitigem Synchronlauf
[, LA _i , SynPosLA _i , Z _i , N _i]	(Eckige Klammern nicht schreiben)
	Mind. 1, max. 5 Folgen von:
LA ₁ , ... LA ₅	Leitachsen
SynPosLA _i	Synchronposition für die i. Leitachse
Z ₁ , ... Z ₅	Zähler für den Koppelfaktor i
N ₁ , ... N ₅	Nenner für den Koppelfaktor i
	Koppelfaktor i = Zähler i / Nenner i

Es dürfen nur Leitachsen programmiert werden, die zuvor mit `EGDEF` spezifiziert worden sind. Durch die programmierten "Synchronpositionen" für die Folgeachse (`SynPosFA`) und für die Leitachsen (`SynPosLA`) werden Positionen definiert, in denen der Koppelverband als *synchron* gilt. Sofern sich das elektronische Getriebe beim Einschalten nicht in synchronem Zustand befindet, fährt die Folgeachse auf ihre definierte Synchronposition.

Variante 3:

Die Parameter entsprechen denen der Variante 2 zuzüglich:

Anfahrmodus	Folgende Modi können benutzt werden:
	"NTGT" Nächste Zahnücke zeitoptimiert anfahren
	"NTGP" Nächste Zahnücke wegoptimiert anfahren
	"ACN" Rundachse in negativer Drehrichtung verfahren absolut
	"ACP" Rundachse in positiver Drehrichtung verfahren absolut
	"DCT" Zeitoptimiert zur programmierten Synchronposition
	"DCP" Wegoptimiert zur programmierten Synchronposition

Die Variante 3 hat nur Auswirkungen auf Modulo-Folgeachsen, die an Modulo-Leitachsen gekoppelt sind. Zeitoptimierung berücksichtigt die Geschwindigkeitsgrenzen der Folgeachse.

Weitere Informationen

Beschreibung der Einschaltvarianten

Variante 1:

Die Positionen der Leitachsen sowie der Folgeachse zum Zeitpunkt des Einschaltens werden gespeichert als "Synchronpositionen". Die "Synchronpositionen" können mit den Systemvariablen `$AA_EG_SYN` gelesen werden.

Variante 2:

Wenn Moduloachsen im Koppelverband sind, werden ihre Positionswerte modulo reduziert. Damit ist gewährleistet, dass die nächstmögliche Synchronposition angefahren wird (sog. *relative Synchronisation*: z. B. die nächste Zahnücke). Wenn für die Folgeachse nicht "Freigabe Folgeachsüberlagerung" Nahtstellensignal DB(30 +Achsnnummer), DBX 26 Bit 4 gegeben ist, wird nicht auf die Synchronposition gefahren. Stattdessen wird das Programm beim EGONSYN-Satz angehalten und es wird der selbstlöschende Alarm 16771 gemeldet, solange bis das o.g. Signal gesetzt wird.

Variante 3:

Der Zahnabstand (Grad) ergibt sich aus: $360 * Zi/Ni$. Für den Fall, dass die Folgeachse zum Aufrufzeitpunkt steht, liefert wegoptimiert das gleiche Verhalten wie zeitoptimiert.

Bei bereits fahrender Folgeachse wird mit `NTGP` unabhängig von der aktuellen Geschwindigkeit der Folgeachse auf die nächste Zahnücke synchronisiert. Bei bereits fahrender Folgeachse wird mit `NTGT` abhängig von der aktuellen Geschwindigkeit der Folgeachse auf die nächste Zahnücke synchronisiert. Die Achse wird dazu ggf. auch abgebremst.

Kurventabellen

Wird für eine der Leitachsen eine **Kurventabelle** verwendet, so muss:

- | | |
|-----|---|
| Ni | der Nenner des Koppelfaktors linearer Kopplungen auf 0 gesetzt werden. (Nenner 0 wäre für lineare Kopplungen unzulässig). Nenner Null ist für die Steuerung das Kennzeichen, dass |
| Zi | als Nummer der zu verwendenden Kurventabelle interpretiert werden soll. Die Kurventabelle mit der angegebenen Nummer muss zum Einschaltzeitpunkt bereits definiert sein. |
| LAI | Die Angabe der Leitachse entspricht der Leitachsangabe bei Kopplung über Koppelfaktor (lineare Kopplung). |

Weitere Hinweise über die Nutzung von Kurventabellen und das Kaskadieren von Elektronischen Getrieben und deren Synchronisierung finden Sie in:

Literatur:

Funktionshandbuch Sonderfunktionen; Achskopplungen und ESR (M3), Kapitel "Mitschleppen und Leitwertkopplung".

Verhalten des Elektronischen Getriebes bei Power On, RESET, Betriebsartenwechsel, Suchlauf

- Nach Power On ist **keine** Kopplung aktiv.
- Aktive Kopplungen bleiben über RESET und Betriebsartenwechsel erhalten.
- Bei Satzsuchlauf werden Befehle zum Schalten, Löschen, Definieren des Elektronischen Getriebes nicht ausgeführt und nicht aufgesammelt, sondern übergangen.

Systemvariablen des Elektronischen Getriebes

Mit Hilfe der Systemvariablen des Elektronischen Getriebes kann das Teileprogramm aktuelle Zustände eines EG-Achsverbandes ermitteln und ggf. darauf reagieren.

Die Systemvariablen des Elektronischen Getriebes sind wie folgt gekennzeichnet:

\$AA_EG_ ...

oder

\$VA_EG_ ...

Literatur:

Handbuch der Systemvariablen

9.4.3 Elektronisches Getriebe ausschalten (EGOFS, EGOFC)

Funktion

Für das Ausschalten eines aktiven EG-Achsverbandes existieren 3 Varianten.

Programmierung

Variante 1:

Syntax

EGOFS (Folgeachse)

Bedeutung

Das elektronische Getriebe wird ausgeschaltet. Die Folgeachse wird zum Stillstand abgebremst. Der Aufruf löst Vorlaufstopp aus.

Variante 2:

Syntax

EGOFS (Folgeachse, Leitachse1, ..., Leitachse5)

Bedeutung

Diese Parametrierung des Befehls erlaubt **selektiv** den Einfluss einzelner Leitachsen auf die Bewegung der Folgeachse zu beseitigen.

Es muss wenigstens eine Leitachse angegeben werden. Der Einfluss der angegebenen Leitachsen auf die Folgeachse wird gezielt ausgeschaltet. Der Aufruf löst Vorlaufstopp aus. Verbleiben noch aktive Leitachsen, so läuft die Folgeachse unter deren Einfluss weiter. Sind alle Leitachsen einflüsse auf diese Weise ausgeschaltet, so wird die Folgeachse zum Stillstand abgebremst.

Variante 3:

Syntax

EGOFC (Folgespindel)

Bedeutung

Das elektronische Getriebe wird ausgeschaltet. Die Folgespindel läuft mit der zum Ausschaltzeitpunkt aktuellen Drehzahl/Geschwindigkeit weiter. Der Aufruf löst Vorlaufstopp aus.

Hinweis

Diese Variante ist nur für Spindeln erlaubt.

9.4.4 Definition eines Elektronischen Getriebes löschen (EGDEL)

Funktion

Ein EG-Achsverband muss ausgeschaltet sein, bevor seine Definition gelöscht werden kann.

Programmierung

Syntax

EGDEL (Folgeachse)

Bedeutung

Die Kopplungsdefinition des Achsverbandes wird gelöscht. Es wird bis zum Erreichen der maximalen Anzahl gleichzeitig aktivierter Achsverbände wieder möglich, weitere Achsverbände mit EGDEF neu zu definieren. Der Aufruf löst Vorlaufstopp aus.

9.4.5 Umdrehungsvorschub (G95) / Elektronisches Getriebe (FPR)

Funktion

Mit dem FPR-Befehl kann auch die Folgeachse eines Elektronischen Getriebes als vorschubbestimmende Achse des Umdrehungsvorschubes angegeben werden. Für diesen Fall gilt folgendes Verhalten:

- Der Vorschub ist abhängig von der Sollgeschwindigkeit der Folgeachse des Elektronischen Getriebes.
- Die Sollgeschwindigkeit wird berechnet aus den Geschwindigkeiten der Leitspindeln und Modulo-Leitachsen (die nicht Bahnachsen sind) und deren zugeordneten Koppelfaktoren.
- Geschwindigkeitsanteile von linearen bzw. nicht Modulo-Leitachsen und überlagerte Bewegungen der Folgeachse werden nicht berücksichtigt.

9.5 Synchronspindel

Funktion

Im Synchronbetrieb gibt es eine Leitspindel (LS) und eine Folgespindel (FS), das sog. **Synchronspindel**paar. Die Folgespindel folgt bei aktiver Kopplung (Synchronbetrieb) den Bewegungen der Leitspindel entsprechend dem festgelegten Funktionszusammenhang.

Die Synchronspindelpaare lassen sich für jede Maschine sowohl mit Hilfe von kanalspezifischen Maschinendaten fest projektieren oder über das CNC-Teileprogramm anwendungsspezifisch definieren. Je NC-Kanal sind bis zu 2 Synchronspindelpaare gleichzeitig betreibbar.

Die Kopplung kann aus dem Teileprogramm

- definiert bzw. geändert
- eingeschaltet
- ausgeschaltet
- gelöscht

werden.

Darüber hinaus kann abhängig vom Softwarestand

- auf die Synchronlaufbedingung gewartet
- das Satzwechselerhalten verändert
- die Kopplungsart entweder Sollwertkopplung oder Istwertkopplung ausgewählt oder der Winkelversatz zwischen Leit- und Folgespindel vorgegeben
- beim Einschalten der Kopplung eine vorhergehende Programmierung der Folgespindel übernommen
- entweder eine gemessene oder eine bereits bekannte Synchronlaufabweichung korrigiert werden.

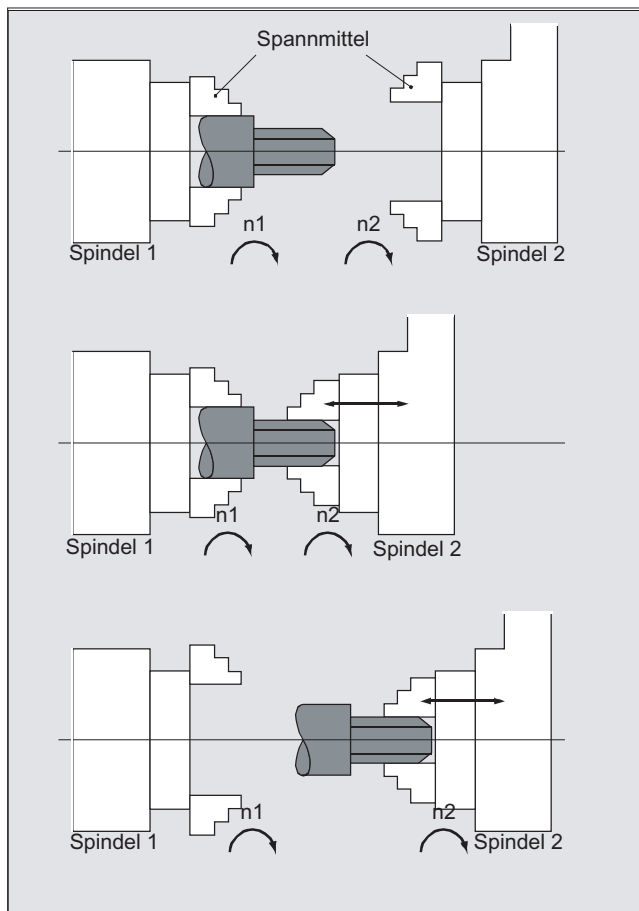
9.5.1 Synchronspindel: Programmierung (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC)

Funktion

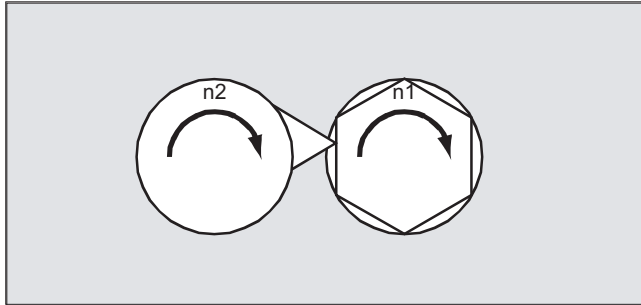
Die Funktion Synchronspindel ermöglicht ein synchrones Verfahren zweier Spindeln (Folgespindel FS und Leitspindel LS), z. B. zur fliegenden Werkstückübergabe.

Die Funktion bietet folgende Modi:

- Drehzahlsynchronität ($n_{FS} = n_{LS}$)
- Lagesynchronität ($\phi_{FS} = \phi_{LS}$)
- Lagesynchronität mit Winkelversatz ($\phi_{FS} = \phi_{LS} + \Delta\phi$)



Durch Vorgabe eines Übersetzungsverhältnisses ungleich 1 zwischen Leit- und Folgespindel ist auch eine Mehrkantbearbeitung (Polygondrehen) möglich.



Syntax

```
COUPDEF (<FS>, <LS>, <ÜFS>, <ÜLS>, <Satzwechsel>, <Koppelart>)  
COUPON (<FS>, <LS>, <POSFS>)  
COUPONC (<FS>, <LS>)  
COUPOF (<FS>, <LS>, <POSFS>, <POSLS>)  
COUPOFS (<FS>, <LS>)  
COUPOFS (<FS>, <LS>, <POSFS>)  
COUPRES (<FS>, <LS>)  
COUPDEL (<FS>, <LS>)  
WAITC (<FS>, <Satzwechsel>, <LS>, <Satzwechsel>)
```

Hinweis

Verkürzte Schreibweise

Bei den Anweisungen `COUPOF`, `COUPOFS`, `COUPRES` und `COUPDEL` ist eine verkürzte Schreibweise ohne Angabe der Leitspindel möglich.

Bedeutung

COUPDEF:	Kopplung anwenderspezifisch definieren/ändern
COUPON:	Kopplung einschalten. Ausgehend von der aktuellen Drehzahl synchronisiert sich die Folgespindel auf die Leitspindel
COUPONC:	Kopplung beim Einschalten mit vorhergehender Programmierung von M3 S... oder M4 S... übernehmen.
COUPOF:	Eine Differenzdrehzahl der Folgespindel wird sofort übernommen. Kopplung ausschalten. <ul style="list-style-type: none">mit sofortigem Satzwechsel: COUPOF (<S2>, <S1>)Satzwechsel erst nach Überfahren der Ausschaltposition(en) <POSFS> bzw. <POSLS>: COUPOF (<S2>, <S1>, <POSFS>) COUPOF (<S2>, <S1>, <POSFS>, <POSLS>)

COUPOFS:	Ausschalten einer Kopplung mit Stopp der Folgespindel. Satzwechsel schnellstmöglich mit sofortigen Satzwechsel: COUPOFS (<S2>, <S1>) Satzwechsel erst nach Überfahren der Ausschaltposition: COUPOFS (<S2>, <S1>, <POSFS>)
COUPRES:	Kopplungsparameter zurücksetzen auf projektierte MD und SD
COUPDEL:	Anwenderdefinierte Kopplung löschen
WAITC:	Warten Synchronlaufbedingung (NOC werden auf IPO bei Satzwechsel aufgehoben)
<FS>:	Bezeichnung der Folgespindel

Optionale Parameter:

<LS>:	Bezeichnung der Leitspindel Angabe mit Spindelnummer: z. B. s2, s1
<ÜFS>, <ÜLS>:	Übersetzungsverhältnis zwischen FS und LS. <ÜFS> = Zähler, <ÜLS> = Nenner Voreinstellung: <ÜFS> / <ÜLS> = 1.0 ; Angabe des Nenners optional
<Satzwechsel>:	Satzwechselerhalten Der Satzwechsel erfolgt: "NOC" sofort "FINE" mit Erreichen von "Synchronlauf fein" "COARSE" mit Erreichen von "Synchronlauf grob" "IPOSTOP" mit Erreichen von IPOSTOP, d. h. nach sollwertseitigem Synchronlauf (Voreinstellung) Das Satzwechselerhalten ist modal wirksam.
<Koppelart>:	Kopplungsart: Kopplung zwischen FS und LS "DV" Sollwertkopplung (Voreinstellung) "AV" Istwertkopplung "VV" Geschwindigkeitskopplung Die Kopplungsart ist modal wirksam.
<POSFS>:	Winkelversatz zwischen Leit- und Folgespindel Wertebereich: 0°... 359,999°
<POSFS>, <POSLS>:	Ausschaltpositionen von Folge- und Leitspindel "Der Satzwechsel wird nach überfahren der POS _{FS} , POS _{LS} freigegeben" Wertebereich: 0°... 359,999°

Beispiele

Beispiel 1: Arbeiten mit Leit- und Folgespindel

Programmcode	Kommentar
	; Leitspindel = Masterspindel = Spindel 1
	; Folgespindel = Spindel 2
N05 M3 S3000 M2=4 S2=500	; Leitspindel dreht mit 3000 U/min, Folgespindel dreht mit 500 U/min.
N10 COUPDEF(S2,S1,1,1,"NOC","Dv")	; Definition der Kopplung (kann auch projiziert werden).
...	
N70 SPCON	; Leitspindel in Lageregelung nehmen (Sollwertk.).
N75 SPCON(2)	; Folgespindel in Lageregelung nehmen.
N80 COUPON(S2,S1,45)	; Fliegend auf Offsetposition = 45 Grad einkoppeln.
...	
N200 FA[S2]=100	; Positioniergeschwindigkeit = 100 grd/min
N205 SPOS[2]=IC(-90)	; 90 Grad überlagert in negative Richtung fahren.
N210 WAITC(S2,"Fine")	; Warten auf Synchronlauf "fein".
N212 G1 X... Y... F...	; Bearbeitung
...	
N215 SPOS[2]=IC(180)	; 180 Grad überlagert in positive Richtung fahren.
N220 G4 S50	; Verweilzeit = 50 Umdrehungen der Masterspindel
N225 FA[S2]=0	; Projektierte Geschw. (MD) aktivieren.
N230 SPOS[2]=IC(-7200)	; 20 Umdrehungen. Mit projektierter Geschwindigkeit in negative Richtung fahren.
...	
N350 COUPOF(S2,S1)	; Fliegend auskoppeln, S=S2=3000
N355 SPOSA[2]=0	; FS bei Null Grad stoppen.
N360 G0 X0 Y0	
N365 WAITS(2)	; Warten auf Spindel 2.
N370 M5	; FS stoppen.
N375 M30	

Beispiel 2: Programmierung einer Differenzdrehzahl

Programmcode	Kommentar
	; Leitspindel = Masterspindel = Spindel 1
	; Folgespindel = Spindel 2
N01 M3 S500	; Leitspindel dreht mit 500 U/min.
N02 M2=3 S2=300	; Folgespindel dreht mit 300 U/min.
...	
N10 G4 F1	; Verweilzeit der Masterspindel.
N15 COUPDEF(S2,S1,-1)	; Koppelfaktor mit Übersetzungsverhältnis -1:1
N20 COUPON(S2,S1)	; Kopplung aktivieren. Die Drehzahl der Folgespindel ergibt sich aus der Drehzahl der Leitspindel und dem Koppelfaktor.
...	
N26 M2=3 S2=100	; Programmierung einer Differenzdrehzahl.

Beispiel 3: Beispiele der Übernahme einer Bewegung zur Differenzdrehzahl

1. Kopplung bei vorhergehender Programmierung der Folgespindel mit COUPON einschalten

Programmcode	Kommentar
	; Leitspindel = Masterspindel = Spindel 1
	; Folgespindel = Spindel 2
N05 M3 S100 M2=3 S2=200	; Leitspindel dreht mit 100 U/min, Folgespindel mit 200 U/min.
N10 G4 F5	; Verweilzeit = 5 Sekunden der Masterspindel
N15 COUPDEF(S2,S1,1)	; Übersetzungsverhältnis FS zu LS ist 1,0 (Voreinstellung).
N20 COUPON(S2,S1)	; Fliegend auf Leitspindel einkoppeln.
N10 G4 F5	; Folgespindel dreht mit 100 U/min.

2. Kopplung bei vorhergehender Programmierung der Folgespindel mit COUPONC einschalten

Programmcode	Kommentar
	; Leitspindel = Masterspindel = Spindel 1
	; Folgespindel = Spindel 2
N05 M3 S100 M2=3 S2=200	; Leitspindel dreht mit 100 U/min, Folgespindel mit 200 U/min.
N10 G4 F5	; Verweilzeit = 5 Sekunden der Masterspindel
N15 COUPDEF(S2,S1,1)	; Übersetzungsverhältnis FS zu LS ist 1,0 (Voreinstellung).
N20 COUPONC(S2,S1)	; Fliegend auf Leitspindel einkoppeln und vorhergehende Drehzahl zu S2 übernehmen.
N10 G4 F5	; S2 dreht mit 100U/min + 200U/min = 300U/min

3. Kopplung bei stehender Folgespindel mit COUPON einschalten

Programmcode	Kommentar
	; Leitspindel = Masterspindel = Spindel 1
	; Folgespindel = Spindel 2
N05 SPOS=10 SPOS[2]=20	; Folgespindel S2 im Positionierbetrieb.
N15 COUPDEF(S2,S1,1)	; Übersetzungsverhältnis FS zu LS ist 1,0 (Voreinstellung).
N20 COUPON(S2,S1)	; Fliegend auf Leitspindel einkoppeln.
N10 G4 F1	; Kopplung wird geschlossen, S2 bleibt auf 20 Grad stehen.

4. Kopplung bei stehender Folgespindel mit COUPONC einschalten

Hinweis

Positionier- oder Achsbetrieb

Befindet sich die Folgespindel vor dem Einkoppeln im Positionier- oder Achsbetrieb, dann verhält sich die Folgespindel bei COUPON (<FS>, <LS>) und COUPONC (<FS>, <LS>) gleich.

ACHTUNG
Leitspindel und Achsbetrieb Befindet sich die Leitspindel vor der Definition der Kopplung im Achsbetrieb, wirkt auch nach dem Einschalten der Kopplung der Geschwindigkeitsgrenzwert aus Maschinendatum: MD32000 \$MA_MAX_AX_VELO (maximale Achsgeschwindigkeit) Zur Vermeidung dieses Verhaltens muss die Achse vor der Definition der Kopplung in den Spindelbetrieb (M3 S... oder M4 S...) geschaltet werden.

Weitere Informationen

Synchronspindelpaar festlegen

Projektierte Kopplung:

Bei der projektierten Kopplung werden Leit- und Folgespindel über Maschinendatum festgelegt. Die projektierten Spindeln können im Teileprogramm nicht verändert werden. Die Parametrierung der Kopplung kann mit `COUPDEF` im Teileprogramm erfolgen (Voraussetzung: kein Schreibschutz festgelegt).

Anwenderdefinierte Kopplung:

Mit `COUPDEF` kann eine Kopplung im Teileprogramm neu definiert oder verändert werden. Ist bereits eine Kopplung aktiv, muss diese vor der Definition einer neuen Kopplung zuerst mit `COUPDEL` gelöscht werden.

Kopplung definieren: COUPDEF

Eine Kopplung wird vollständig definiert durch:

```
COUPDEF (<FS>, <LS>, <ÜFS>, <ÜLS>, Satzwechselverhalten, Koppelart)
```

Folgespindel (FS) und Leitspindel (LS)

Mit den Achsnamen für die Folge- (FS) und Leitspindel (LS) wird die Kopplung eindeutig bestimmt. Die Achsnamen müssen mit jeder Anweisung `COUPDEF` programmiert werden. Die anderen Kopplungsparameter sind modal wirksam und müssen nur programmiert werden, wenn sie geändert werden.

Beispiel:

```
COUPDEF (S2, S1)
```

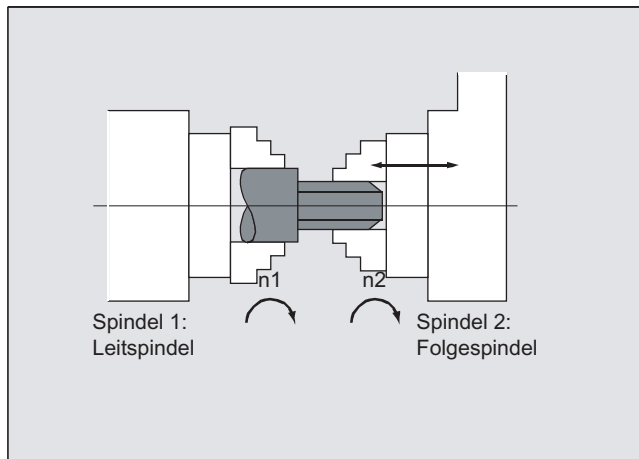
Übersetzungsverhältnis ÜFS / ÜLS

Das Übersetzungsverhältnis wird als Drehzahlverhältnis zwischen Folgespindel (Zähler) und Leitspindel (Nenner) angegeben. Der Zähler muss programmiert werden. Wird kein Nenner programmiert, wird Nenner = 1.0 gesetzt.

Beispiel:

Folgespindel S2 und Leitspindel S1, Übersetzungsverhältnis = $1 / 4 = 0.25$.

COUPDEF (S2,S1,1.0, 4.0)



Hinweis

Das Übersetzungsverhältnis kann auch bei eingeschalteter Kopplung und drehenden Spindeln verändert werden.

Satzwechselerhalten NOC, FINE, COARSE, IPOSTOP

Bei der Programmierung des Satzwechselerhaltens ist folgende verkürzte Schreibweise möglich:

- "NO": sofort (Voreinstellung)
- "FI": mit Erreichen von "Synchronlauf fein"
- "CO": mit Erreichen von "Synchronlauf grob"
- "IP": mit Erreichen von IPOSTOP, d. h. nach sollwertseitigem Synchronlauf

Kopplungsart DV, AV



Die Kopplungsart darf nur bei ausgeschalteter Kopplung verändert werden!

Synchronbetrieb einschalten COUPON, POSFS

- Einschalten der Kopplung mit beliebigem Winkelbezug zwischen LS und FS:

- COUPON (S2, S1)
- COUPON (S2, S1, <POSFS>)
- COUPON (S2)

- Einschalten der Kopplung mit Winkelversatz <POSFS>

Zur positionssynchronen Kopplung für profilierte Werkstücke.

<POSFS> bezieht sich auf die 0°-Position der Leitspindel in positiver Drehrichtung

Wertebereich <POSFS>: 0°... 359,999°

- COUPON (S2, S1, 30)

Auf diese Weise können Sie auch bei schon aktivierter Kopplung den Winkelversatz ändern.

Positionieren der Folgespindel

Bei eingeschalteter Synchronspindelkopplung lassen sich auch Folgespindeln unabhängig von der durch die Leitspindel ausgelösten Bewegung im Bereich $\pm 180^\circ$ positionieren.

Positionierung SPOS

Die Folgespindel kann mit `SPOS=...` interpoliert werden.

Beispiel:

```
SPOS [2] = IC (-90)
```

Weitere Informationen zu `SPOS` finden sich in:

Literatur:

Programmierhandbuch Grundlagen

Differenzdrehzahl M3 S... oder M4 S...

Eine Differenzdrehzahl entsteht durch vorzeichenbehaftete Überlagerung zweier Drehzahlquellen und wird zur Folgespindel z. B. mit `S<n>=...` oder `M<n>=3, M<n>=4` im Drehzahlsteuerbetrieb während einer aktiven Synchronspindelkopplung erneut programmiert. Dabei wird dieser Drehzahlanteil über den Koppelfaktor von der Leitspindel abgeleitet und der Folgespindel vorzeichenrichtig dazu addiert.

Hinweis

Mit der Drehrichtung `M3` oder `M4` muss auch die Drehzahl `S...` neu programmiert werden, weil sonst die fehlende Programmierung durch einen Alarm gemeldet wird.

Weitere Informationen zur Differenzdrehzahl siehe:

Literatur:

Funktionshandbuch Erweiterungsfunktionen; Synchronspindel (S3)

Differenzdrehzahl bei COUPONC

Übernahme einer Bewegung zur Differenzdrehzahl

Durch das Einschalten einer Synchronspindelkopplung mit COUPONC wird eine aktuell wirksame Drehzahl der Folgespindel (M3 S... oder M4 S...) überlagert.

Hinweis

Freigabe der Überlagerung

Eine Überlagern einer Spindeldrehzahl (M3 S... oder M4 S...) durch Synchronspindelkopplung COUPONC wird nur wirksam, wenn die Überlagerung freigegeben ist.

Dynamikeinschränkung der Leitspindel

Die Dynamik der Leitspindel muss so weit eingeschränkt werden, dass bei einer Überlagerung der Folgespindel deren Dynamikgrenzwerte nicht überschritten werden.

Geschwindigkeit, Beschleunigung: FA, ACC, OVRA, VELOLIMA

Axiale Geschwindigkeit und Beschleunigung einer Folgespindeln sind programmierbar mit:

- FA[SPI(S<n>)] bzw. FA[S<n>] (axiale Geschwindigkeit)
- ACC[SPI(S<n>)] bzw. ACC[S<n>] (axiale Beschleunigung)
- OVRA[SPI(S<n>)] bzw. OVRA[S<n>] (axialer Override)
- VELOLIMA[SPI(S<n>)] bzw. VELOLIMA[S<n>] (axiale Geschwindigkeitsüberhöhung bzw. -reduktion)

Mit <n> = 1, 2, 3, ... (Spindelnummer der Folgespindeln)

Literatur:

Programmierhandbuch Grundlagen

Hinweis

Beschleunigungsanteil JERKLIMA[S<n>]

Die Programmierung einer axialen Geschwindigkeitsüberhöhung bzw. -reduktion ist bei Spindeln aktuell nicht wirksam.

Weitere Informationen zur Projektierung der axialen Dynamik finden sich in:

Literatur:

Funktionshandbuch Erweiterungsfunktionen; Rundachsen (R2)

Programmierbares Satzwechselverhalten WAITC

Mit `WAITC` kann das Satzwechselverhalten, z. B. nach Änderung von Kopplungsparametern oder Positioniervorgängen, mit unterschiedlichen Synchronlaufbedingungen (grob, fein, `IPOSTOP`) vorgegeben werden. Sind keine Synchronlaufbedingungen angegeben, gilt das bei der Definition `COUPDEF` angegebene Satzwechselverhalten.

Beispiel:

Warten auf das Erreichen der Synchronlaufbedingung entsprechend `COUPDEF`

```
WAITC ( )
```

Warten auf das Erreichen der Synchronlaufbedingung `FINE` bei Folgespindel `S2` und `COARSE` bei Folgespindel `S4`:

```
WAITC (S2, "FINE", S4, "COARSE")
```

Kopplung ausschalten COUPOF

Mit `COUPOF` kann das Ausschaltverhalten der Kopplung vorgegeben werden:

- Ausschalten der Kopplung mit sofortigem Satzwechsel:
 - `COUPOF (S2, S1)` (mit Angabe der Leitspindel)
 - `COUPOF (S2)` (ohne Angabe der Leitspindel)
- Ausschalten der Kopplung nach Überfahren von Ausschaltpositionen. Der Satzwechsel erfolgt nach dem Überfahren der Ausschaltpositionen.
 - `COUPOF (S2, S1, 150)` (Ausschaltposition FS: 150°)
 - `COUPOF (S2, S1, 150, 30)` (Ausschaltposition FS: 150°, LS: 30°)

Kopplung ausschalten mit Stopp der Folgespindel COUPOFS

Mit `COUPOFS` kann das Ausschaltverhalten der Kopplung mit Stopp der Folgespindel vorgegeben werden:

- Ausschalten der Kopplung mit Stopp der Folgespindel und sofortigem Satzwechsel:
 - `COUPOFS (S2, S1)` (mit Angabe der Leitspindel)
 - `COUPOFS (S2)` (ohne Angabe der Leitspindel)
- Ausschalten der Kopplung nach Überfahren von Ausschaltpositionen mit Stopp der Folgespindel. Der Satzwechsel erfolgt nach dem Überfahren der Ausschaltpositionen.
 - `COUPOFS (S2, S1, 150)` (Ausschaltposition FS: 150°)

Kopplungen löschen COUPDEL

Mit `COUPDEL` wird die Kopplung gelöscht:

- `COUPDEL (S2, S1)` (mit Angabe der Leitspindel)
- `COUPDEL (S2)` (ohne Angabe der Leitspindel)

Kopplungsparameter zurücksetzen COUNPRES

Mit `COUNPRES` werden die in den Maschinen- und Settingdaten parametrisierten Werte der Kopplung aktiviert:

- `COUNPRES (S2, S1)` (mit Angabe der Leitspindel)
- `COUNPRES (S2)` (ohne Angabe der Leitspindel)

Systemvariablen

Aktueller Kopplungszustand der Folgespindel

Der aktuelle Kopplungszustand einer Folgespindel kann über folgende Systemvariable gelesen werden:

\$AA_COUP_ACT [<FS>]

Wert	Bedeutung
0	keine Kopplung aktiv
4	Synchronspindelkopplung aktiv
Hinweis Andere Werte der Systemvariablen beziehen sich auf den Achsbetrieb	
Literatur: Listenhandbuch Systemvariablen	

Aktueller Winkelversatz

Der aktuelle Winkelversatz einer Folgespindel bezüglich der Leitspindel kann über folgende Systemvariable gelesen werden:

- \$AA_COUP_OFFS [<FS>] (Sollwertseitiger Winkelversatz)
- \$VA_COUP_OFFS [<FS>] (Istwertseitiger Winkelversatz)

Hinweis

Nach Wegnahme der Reglerfreigabe bei eingeschalteter Kopplung und Nachführbetrieb stellt sich nach erneuter Erteilung der Reglerfreigabe ein anderer Positionsoffset ein als der ursprünglich programmierte Wert. In diesem Fall kann der veränderte Positionsoffset gelesen werden und ggf. im Teileprogramm korrigiert werden.

9.6 Master/Slave-Kopplung (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS)

Funktion

Die "Master/Slave-Kopplung" ermöglicht:

- das Einkoppeln der Slave-Achsen auf ihre Master-Achse im Stillstand der beteiligten Achsen.
- das Koppeln und Trennen von **drehenden**, drehzahlgesteuerten Spindeln.
- die dynamische Projektierung.

Hinweis

Bei Achsen und Spindeln im Positionierbetrieb wird die Kopplung nur im Stillstand geschlossen und getrennt.

Syntax

```
MASLON(<Slave_1>, <Slave_2>, ...)  
MASLOF(<Slave_1>, <Slave_2>, ...)  
MASLOFS(<Slave_1>, <Slave_2>, ...)
```

Dynamische Projektierung:

```
MASLDEF(<Slave_1>, <Slave_2>, ..., <Master>)  
MASLDEL(<Slave_1>, <Slave_2>, ...)
```

Bedeutung

MASLON:	Eine temporäre Master/Slave-Kopplung einschalten
<Slave_1>, ...:	Slave-Achsen Achsen, die in einem Master/Slave-Verband von einer Master-Achse geführt werden sollen
MASLOF:	Eine aktive Master/Slave-Kopplung trennen
<Slave_1>, ...:	Slave-Achsen
MASLOFS:	Master/Slave-Kopplung trennen und Slave-Spindeln automatisch abbremsten (siehe Hinweis "Koppelverhalten bei Spindeln im Drehzahlsteuerbetrieb!")
<Slave_1>, ...:	Slave-Achsen
MASLDEF:	Master/Slave-Verband aus dem Teileprogramm heraus anlegen/ändern
<Slave_1>, ...:	Slave-Achsen
<Master>:	Master-Achse Achse, die in einem Master/Slave-Verband definierte Slave-Achsen führen soll

MASLDEL: Master/Slave-Kopplung trennen und Definition des Verbandes löschen
<Slave_1>, ...: Slave-Achsen
Hinweis:
Die in den Maschinendaten projektierten Master-/Slave-Definitionen bleiben erhalten.

Hinweis

Koppelverhalten bei Spindeln im Drehzahlsteuerbetrieb

Bei Spindeln im Drehzahlsteuerbetrieb wird das Koppelverhalten von MASLON, MASLOF, MASLOFS und MASLDEL explizit über das folgende Maschinendatum festgelegt:

MD37263 \$MA_MS_SPIND_COUPLING_MODE

In der Standardeinstellung mit MD37263 = 0 erfolgt das Einkoppeln und Trennen der Slave-Achsen ausschließlich im Stillstand der beteiligten Achsen. MASLOFS entspricht dem MASLOF.

Bei MD37263 = 1 wird die Koppelanweisung unmittelbar, und damit auch in der Bewegung ausgeführt. Die Kopplung wird bei MASLON sofort geschlossen und bei MASLOFS oder MASLOF sofort getrennt. Die zu diesem Zeitpunkt drehenden Slave-Spindeln behalten bei MASLOF ihre Drehzahlen bis zur erneuten Drehzahlprogrammierung bei. Bei MASLOFS werden sie dagegen automatisch abgebremst.

Hinweis

Bei MASLOF/MASLOFS entfällt der implizite Vorlaufstopp. Bedingt durch den fehlenden Vorlaufstopp liefern die \$P-Systemvariablen für die Slave-Achsen bis zum Zeitpunkt erneuter Programmierung keine aktualisierten Werte.

Hinweis

Für die Slave-Achse kann der Istwert durch PRESETON auf den gleichen Wert der Master-Achse synchronisiert werden. Dazu muss die dauerhafte Master/Slave-Kopplung kurzfristig ausgeschaltet werden, um den Istwert der nicht referierten Slave-Achse mit POWER ON auf den Wert der Master-Achse zu setzen. Danach wird die dauerhafte Kopplung wieder hergestellt.

Die dauerhafte Master/Slave-Kopplung wird mit folgender MD-Einstellung aktiviert:

MD37262 \$MA_MS_COUPLING_ALWAYS_ACTIVE = 1

Sie hat keine Auswirkung auf die Sprachbefehle der temporären Kopplung.

Beispiele

Beispiel 1: Dynamische Projektierung einer Master/Slave-Kopplung

Dynamische Projektierung einer Master/Slave-Kopplung aus dem Teileprogramm heraus. Die nach einer Achscontainerdrehung relevante Achse soll zur Master-Achse werden.

Programmcode	Kommentar
MASLDEF(AUX,S3)	; S3 Master für AUX
MASLON(AUX)	; Kopplung ein für AUX
M3=3 S3=4000	; Drehrichtung rechts
MASLDEL(AUX)	; Projektierung löschen und Trennen der Kopplung
AXCTSWE(CT1)	; Containerdrehung

Beispiel 2: Istwertkopplung einer Slave-Achse

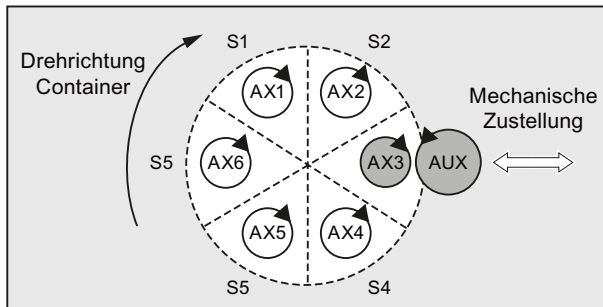
Istwertkopplung einer Slave-Achse auf den gleichen Wert der Master-Achse durch PRESETON. Bei einer permanenten Master/Slave-Kopplung soll an der Slave-Achse der Istwert durch PRESETON verändert werden.

Programmcode	Kommentar
N37262 \$MA_MS_COUPLING_ALWAYS_ACTIVE[AX2]=0	; Permanente Kopplung kurz ausschalten.
N37263 NEWCONF	
N37264 STOPRE	
MASLOF(Y1)	; Temporäre Kopplung aus.
N5 PRESETON(Y1,0,Z1,0,B1,0,C1,0,U1,0)	; Istwert setzen der nicht referierten Slave-Achsen, da diese mit POWER ON aktiviert sind.
N37262 \$MA_MS_COUPLING_ALWAYS_ACTIVE[AX2]=1	; Permanente Kopplung aktivieren.
N37263 NEWCONF	

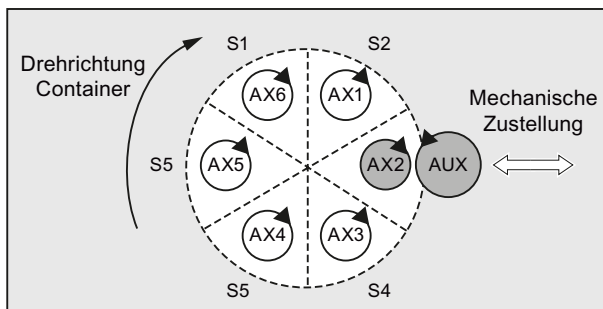
Beispiel 3: Kopplungssequenz Lage 3 / Container CT1

Damit die Kopplung nach der Containerdrehung mit einer anderen Spindel geschlossen werden kann, muss vorher die alte Kopplung getrennt, die Projektierung gelöscht und die neue Kopplung projiziert werden.

Ausgangssituation:



Nach Drehung um einen Slot:



Synchronaktionen

10.1 Definition einer Synchronaktion

Eine Synchronaktion wird in einem Satz eines Teileprogramms definiert. Innerhalb dieses Satzes dürfen keine weiteren Befehle programmiert werden, die nicht Bestandteil der Synchronaktion sind.

Eine Synchronaktion besteht aus folgenden Komponenten:

Optional Gültigkeit, Ident-Nr.	Bedingungsteil			Aktionsteil		
	Häufigkeit	Optional G-Funktion	Bedingung	Schlüsselwort	Optional: G-Funktion	Aktionen
k. A.	k. A.	G...	Logischer Ausdruck	DO	G...	Aktion 1 ... Aktion n
ID=<Nr>	WHENEVER					
IDS=<Nr>	FROM					
	WHEN					
	EVERY					

k. A. = "keine Angabe"

Syntax

```
DO <Aktion 1> ... <Aktion n>
<Häufigkeit> [<G-Funktion>] <Bedingung> DO <Aktion 1> ... <Aktion n>
ID=<Nr> <Häufigkeit> [<G-Funktion>] <Bedingung> DO <Aktion 1> ...
<Aktion n>
IDS=<Nr> <Häufigkeit> [<G-Funktion>] <Bedingung> DO <Aktion 1> ...
<Aktion n>
```

Literatur

Eine ausführliche Beschreibung der Funktionalität von Synchronaktionen findet sich in:
Funktionshandbuch Synchronaktionen

Pendeln

11.1 Asynchrones Pendeln (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB)

Funktion

Eine Pendelachse fährt zwischen den zwei Umkehrpunkten 1 und 2 mit gegebenem Vorschub hin und her, bis die Pendelbewegung abgeschaltet wird.

Andere Achsen können während der Pendelbewegung beliebig interpoliert werden. Über eine Bahnbewegung oder mit einer Positionierachse kann eine kontinuierliche Zustellung erreicht werden. Dabei besteht jedoch **kein Zusammenhang** zwischen der Pendel- und der Zustellbewegung.

Eigenschaften des asynchronen Pendelns

- Das asynchrone Pendeln ist achsspezifisch über Satzgrenzen hinweg wirksam.
- Über das Teileprogramm ist ein satzsynchrones Einschalten der Pendelbewegung gewährleistet.
- Eine gemeinsame Interpolation von mehreren Achsen und eine Überlagerung von Pendelstrecken sind nicht möglich.

Programmierung

Über die folgenden Befehle ist ein der Abarbeitung des NC-Programms entsprechendes Einschalten und Beeinflussen des asynchronen Pendelns vom Teileprogramm her möglich.

Die programmierten Werte werden satzsynchron im Hauptlauf in die entsprechenden Settingdaten eingetragen und bleiben bis zur nächsten Änderung wirksam.

Syntax

```
OSP1 [<Achse>] = <Wert>  OSP2 [<Achse>] = <Wert>
OST1 [<Achse>] = <Wert>  OST2 [<Achse>] = <Wert>
FA [<Achse>] = <Wert>
OSCTRL [<Achse>] = (<Setzoption>, <Rücksetzoption>)
OSNSC [<Achse>] = <Wert>
OSE [<Achse>] = <Wert>
OSB [<Achse>] = <Wert>
OS [<Achse>] = 1
OS [<Achse>] = 0
```

Bedeutung

<Achse>	Name der Pendelachse
OS	Pendeln ein-/ausschalten Wert: 1 Pendeln e inschalten 0 Pendeln a usschalten
OSP1	Position von Umkehrpunkt 1 festlegen
OSP2	Position von Umkehrpunkt 2 festlegen Hinweis: Falls ein inkrementelles Verfahren aktiv ist, so wird die Position inkrementell zur letzten im NC-Programm programmierten entsprechenden Umkehrposition berechnet.
OST1	Haltezeit im Umkehrpunkt 1 in [s] festlegen
OST2	Haltezeit im Umkehrpunkt 2 in [s] festlegen <Wert>: -2 Interpolation wird ohne Warten auf Genauhalt fortgesetzt -1 Warten auf Genauhalt grob 0 Warten auf Genauhalt fein >0 Warten auf Genauhalt fein und anschließend Abwarten der angegebenen Haltezeit Hinweis: Die Einheit für die Haltezeit ist identisch mit der über G4 programmierten Haltezeit.
FA	Vorschubgeschwindigkeit festlegen Als Vorschubgeschwindigkeit gilt die definierte Vorschubgeschwindigkeit der Positionierachse. Ist keine Vorschubgeschwindigkeit definiert, gilt der im Maschinendatum hinterlegte Wert.
OSCTRL	Setz- und Rücksetzoptionen angeben Die Optionswerte 0 - 3 verschlüsseln das Verhalten an den Umkehrpunkten beim Ausschalten. Es kann eine der Varianten 0 - 3 ausgewählt werden. Die übrigen Einstellungen sind nach Bedarf kombinierbar mit der gewählten Variante. Mehrere Optionen werden durch Pluszeichen (+) aneinandergefügt. <Wert>: 0 Beim Abschalten der Pendelbewegung im nächsten Umkehrpunkt stoppen (Voreinstellung) Hinweis: Nur durch Rücksetzen der Werte 1 und 2 möglich. 1 Beim Abschalten der Pendelbewegung in Umkehrpunkt 1 stoppen 2 Beim Abschalten der Pendelbewegung in Umkehrpunkt 2 stoppen 3 Beim Abschalten der Pendelbewegung keinen Umkehrpunkt anfahren, falls keine Ausfeuerungshübe programmiert sind 4 Nach dem Ausfeuern Endposition anfahren 8 Wird die Pendelbewegung durch Restweglöschen abgebrochen, sollen anschließend Ausfeuerungshübe abgearbeitet und ggf. die Endposition angefahren werden.

11.1 Asynchrones Pendeln (OS, OSP1, OSP2, OST1, OST2, OSCCTRL, OSNSC, OSE, OSB)

	16	Wird die Pendelbewegung durch Restweglöschen abgebrochen, soll wie beim Abschalten die entsprechende Umkehrposition angefahren werden.
	32	Geänderter Vorschub ist erst ab dem nächsten Umkehrpunkt aktiv
	64	FA gleich 0, FA = 0: Wegüberlagerung ist aktiv FA ungleich 0, FA <> 0: Geschwindigkeitsüberlagerung ist aktiv
	128	Bei Rundachse DC (kürzester Weg)
	256	Ausfeuerhub wird als Doppelhub ausgeführt.(Standard) 1=Ausfeuerhub wird als Einzelhub ausgeführt.
	512	Zuerst Startposition anfahren
OSNSC		Anzahl der Ausfeuerungshübe festlegen
OSE		Endposition (im WKS) festlegen, die nach Ausschalten des Pendelns angefahren werden soll Hinweis: Bei Programmierung von OSE wird für OSCCTRL implizit Option 4 wirksam.
OSB		Startposition (im WKS) festlegen, die vor Einschalten des Pendelns angefahren werden soll Die Startposition wird vor Umkehrpunkt 1 angefahren. Stimmt die Startposition mit der Umkehrposition 1 überein, so wird als nächstes die Umkehrposition 2 angefahren. Beim Erreichen der Startposition wirkt keine Haltezeit, auch wenn die Startposition mit der Umkehrposition 1 übereinstimmt, stattdessen wird auf Genauhalt fein gewartet. Eine eingestellte Genauhaltbedingung wird eingehalten. Hinweis: Damit die Startposition angefahren wird, muss im Settingdatum SD43770 \$SA_OSCILL_CTRL_MASK Bit 9 gesetzt sein.

Beispiele

Beispiel 1: Pendelachse soll zwischen zwei Umkehrpunkten pendeln

Die Pendelachse Z soll zwischen Position 10 und 100 pendeln. Umkehrpunkt 1 soll mit Genauhalt fein, Umkehrpunkt 2 mit Genauhalt grob angefahren werden. Der Vorschub für die Pendelachse soll 250 betragen. Am Ende der Bearbeitung sollen 3 Ausfeuerungshübe erfolgen und die Pendelachse soll die Endposition 200 ansteuern. Der Vorschub für die Zustellachse soll 1 betragen, das Ende der Zustellung in X-Richtung soll bei Position 15 erreicht sein.

Programmcode	Kommentar
WAITP(X, Y, Z)	; Ausgangsstellung.
G0 X100 Y100 Z100	; Umschalten in Positionierachsbetrieb.
WAITP(X, Z)	
OSP1[Z]=10 OSP2[Z]=100	; Umkehrpunkt 1, Umkehrpunkt 2.
OSE[Z]=200	; Endposition.
OST1[Z]=0 OST2[Z]=-1	; Haltezeit an U1: Genauhalt fein ; Haltezeit an U2: Genauhalt grob
FA[Z]=250 FA[X]=1	; Vorschub Pendelachse, Vorschub Zustellachse.
OSCTRL[Z]=(4, 0)	; Setzoptionen.
OSNSC[Z]=3	; 3 Ausfeuerungshübe.
OS[Z]=1	; Pendeln starten.
WHEN \$A_IN[3]==TRUE DO DELDTG(X)	; Restweglöschen.
POS[X]=15	; Ausgangsstellung X-Achse
POS[X]=50	Endstellung X-Achse.
OS[Z]=0	; Pendeln stoppen.
M30	

Hinweis

Der Befehlsfolge `OSP1[Z]=...` bis `OSNSC[Z]=...` kann auch in einem Satz programmiert werden.

Beispiel 2: Pendeln mit Online-Änderung der Umkehrposition

Die für das asynchrone Pendeln erforderlichen Settingdaten können im Teileprogramm eingestellt werden.

Werden im Teileprogramm die Settingdaten direkt beschrieben, so wird die Änderung schon zum Vorlaufzeitpunkt wirksam. Synchrones Verhalten kann über einen Vorlaufstopp (STOPRE) erreicht werden.

Programmcode	Kommentar
\$SA_OSCILL_REVERSE_POS1[Z]=-10	
\$SA_OSCILL_REVERSE_POS2[Z]=10	
G0 X0 Z0	
WAITP(Z)	
ID=1 WHENEVER \$AA_IM[Z] < \$\$AA_OSCILL_REVERSE_POS1[Z] DO \$AA_OVR[X]=0	; Wenn der Istwert der
ID=2 WHENEVER \$AA_IM[Z] < \$\$AA_OSCILL_REVERSE_POS2[Z] DO \$AA_OVR[X]=0	Pendelachse den
	Umkehrpunkt überschritten
	hat, wird die
	Zustellachse angehalten.
OS[Z]=1 FA[X]=1000 POS[X]=40	; Pendeln einschalten.
OS[Z]=0	; Pendeln ausschalten.
M30	

Weitere Informationen**Pendelachse**

Für die Pendelachse gilt:

- Jede Achse kann als Pendelachse benutzt werden.
- Gleichzeitig können mehrere Pendelachsen aktiv sein (maximal: Anzahl der Positionierachsen).
- Für die Pendelachse ist immer - unabhängig vom im Programm aktuell gültigen G-Befehl - Linearinterpolation G1 aktiv.

Die Pendelachse kann:

- Eingangsachse für die dynamische Transformation sein
- Führungsachse bei Gantry- und Mitschleppachsen sein
- verfahren werden:
 - ohne Ruckbegrenzung (BRISK)
 - oder
 - mit Ruckbegrenzung (SOFT)
 - oder
 - mit geknickter Beschleunigungskennlinie (wie Positionierachsen)

Pendelumkehrpunkte

Bei der Festlegung der Pendelpositionen sind die aktuellen Verschiebungen zu beachten:

- Absolute Angabe

OSP1 [Z] = <Wert>

Position Umkehrpunkt = Summe der Verschiebungen + programmierter Wert

- Relative Angabe

OSP1 [Z] = IC(<Wert>)

Position Umkehrpunkt = Umkehrpunkt 1 + programmierter Wert

Beispiel:

Programmcode
N10 OSP1 [Z] = 100 OSP2 [Z] = 110
...
...
N40 OSP1 [Z] = IC(3)

WAITP

Soll mit einer Geometrieachse gependelt werden, so muss diese mit `WAITP` zum Pendeln freigegeben werden.

Nach beendetem Pendeln wird mit `WAITP` die Pendelachse wieder als Positionierachse eingetragen und kann wieder normal verwendet werden.

Pendeln mit Bewegungssynchronaktionen und Haltezeiten

Nach Ablauf der eingestellten Haltezeiten findet beim Pendeln der interne Satzwechsel statt (sichtbar an den neuen Restwegen der Achsen). Beim Satzwechsel wird die Ausschaltfunktion überprüft. Dabei wird nach der eingestellten Steuereinstellung für den Bewegungsablauf (`OSCTRL`) die Ausschaltfunktion festgelegt. *Dieses Zeitverhalten ist durch den Vorschuboverride beeinflussbar.*

Unter Umständen wird danach noch ein Pendelhub ausgeführt, bevor die Ausfeuerungs hübe gestartet oder die Endposition angefahren wird. *Es entsteht dabei der Eindruck, es verändert sich das Ausschaltverhalten. Dies ist aber nicht der Fall.*

11.2 Über Synchronaktionen gesteuertes Pendeln (OSCILL)

Funktion

Bei dieser Art des Pendelns ist nur an den Umkehrpunkten bzw. innerhalb definierter Umkehrbereiche eine Zustellbewegung zugelassen.

Je nach Anforderung kann die Pendelbewegung während der Zustellung

- fortgeführt oder
- angehalten werden, bis die Zustellung vollständig ausgeführt ist.

Syntax

1. Parameter für das Pendeln festlegen
2. Bewegungssynchronaktionen definieren
3. Achsen zuordnen, Zustellung festlegen

Bedeutung

OSP1 [<Pendelachse>] =	Position des Umkehrpunkts 1
OSP2 [<Pendelachse>] =	Position des Umkehrpunkts 2
OST1 [<Pendelachse>] =	Haltezeit in Umkehrpunkt 1 in Sekunden
OST2 [<Pendelachse>] =	Haltezeit in Umkehrpunkt 2 in Sekunden
FA [<Pendelachse>] =	Vorschub der Pendelachse
OSCTRL [<Pendelachse>] =	Setz- bzw. Rücksetzoptionen
OSNSC [<Pendelachse>] =	Anzahl der Ausfeuerungsstöße
OSE [<Pendelachse>] =	Endposition
WAITP (<Pendelachse>)	Achse für das Pendeln freigeben

Achszuordnung, Zustellung

OSCILL [<Pendelachse>] = (<Zustellachse 1>, <Zustellachse 2>, <Zustellachse 3>)
 POSP [<Zustellachse>] = (<Endpos>, <Teillänge>, <Modus>)

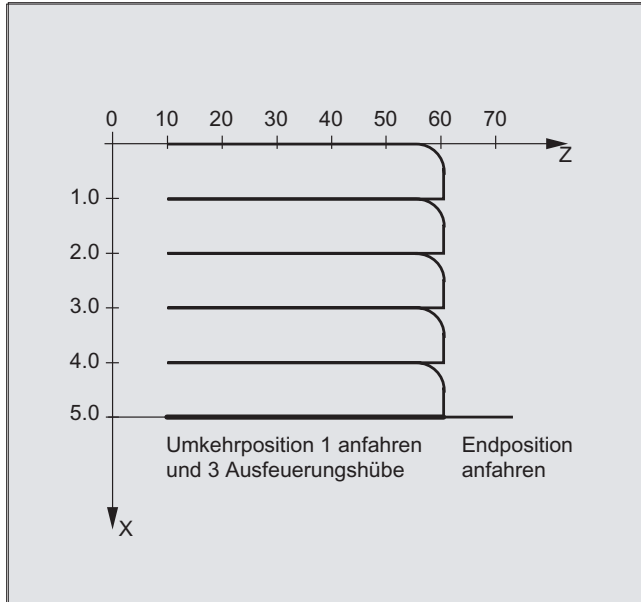
OSCILL:	Zustellachse(n) der Pendelachse zuordnen
POSP:	Gesamt- und Teilzustellungen festlegen (siehe Kapitel Datei- und Programmverwaltung)
Endpos:	Endposition für die Zustellachse, nachdem alle Teilzustellungen abgefahren sind.
Teillänge:	Größe der Teilzustellung am Umkehrpunkt/Umkehrbereich
Modus:	Aufteilung der Gesamtzustellung in Teilzustellungen = zwei gleich große Restschritte (Voreinstellung); = alle Teilzustellungen gleich groß

Bewegungssynchronaktionen

WHEN... .. DO	wenn..., dann...
WHENEVER ... DO	immer wenn..., dann...

Beispiel

Im Umkehrpunkt 1 soll keine Zustellung erfolgen. Beim Umkehrpunkt 2 soll die Zustellung bereits im Abstand ii2 vor dem Umkehrpunkt 2 erfolgen und die Pendelachse im Umkehrpunkt nicht auf das Beenden der Teilzustellung warten. Die Achse Z ist Pendelachse und die Achse X Zustellachse.



1. Parameter für das Pendeln

Programmcode	Kommentar
DEF INT ii2	; Variable für Umkehrbereich 2 definieren
OSP1[Z]=10 OSP2[Z]=60	; Umkehrpunkt 1 und 2 definieren
OST1[Z]=0 OST2[Z]=0	; Umkehrpunkt 1: Genauhalt fein Umkehrpunkt 2: Genauhalt fein
FA[Z]=150 FA[X]=0.5	; Vorschub Pendelachse Z, Vorschub Zustellachse X
OSCTRL[Z]=(2+8+16,1)	; Pendelbewegung abschalten im Umkehrpunkt 2; nach RWL Ausfeuern und Endposition anfahren; nach RWL entsprechende Umkehrposition anfahren
OSNC[Z]=3	; Ausfeuerungshübe
OSE[Z]=70	; Endposition = 70
ii2=2	; Umkehrbereich einstellen
WAITP(Z)	; Erlaube Pendeln für Z-Achse

2. Bewegungssynchronaktion

Programmcode	Kommentar
WHENEVER \$AA_IM[Z] < \$SA_OSCILL_REVERSE_POS2[Z] DO \$AA_OVR[X]=0 \$AC_MARKER[0]=0	; Immer wenn die aktuelle Position der Pendelachse Z im MKS kleiner als der Beginn des Umkehrbereichs 2 ist, dann setze den axialen Override der Zustellachse X auf 0% und den Merker mit dem Index 0 auf den Wert 0.
WHENEVER \$AA_IM[Z] >= \$SA_OSCILL_REVERSE_POS2[Z] DO \$AA_OVR[Z]=0	; Immer wenn die aktuelle Position der Pendelachse Z im MKS größer gleich der Umkehrposition 2 ist, dann setze den axialen Override der Pendelachse Z auf 0%.
WHENEVER \$AA_DTEPW[X]==0 DO \$AC_MARKER[0]=1	; Immer wenn der Restweg der Teilzustellung gleich 0 ist, dann setze den Merker mit dem Index 0 auf den Wert 1.
WHENEVER \$AC_MARKER[0]==1 DO \$AA_OVR[X]=0 \$AA_OVR[Z]=100	; Immer wenn der Merker mit dem Index 0 gleich 1 ist, dann setze den axialen Override der Zustellachse X auf 0%. Damit wird eine zu frühe Zustellung verhindert (Pendelachse Z hat den Umkehrbereich 2 noch nicht wieder verlassen, die Zustellachse X ist aber bereit für eine erneute Zustellung). Setze den axialen Override der Pendelachse Z von 0% (Aktion der 2. Synchronaktion) zum Verfahren wieder auf 100%.

-> muss in einem Satz programmiert werden

3. Pendeln starten

Programmcode	Kommentar
OSCILL[Z]=(X) POSP[X]=(5,1,1)	; Starten der Achsen Der Pendelachse Z wird die Achse X als Zustellachse zugewiesen. Die Achse X soll bis Endposition 5 in Schritten von 1 fahren.
M30	; Programmende

Beschreibung

1. Pendelparameter festlegen

Vor dem Bewegungssatz, der die Zuordnung von Zustell- und Pendelachse sowie die Festlegung der Zustellung enthält, sind die Parameter für das Pendeln festzulegen (siehe "Asynchrones Pendeln").

2. Bewegungssynchronaktionen festlegen

Über Synchronbedingungen erfolgt:

Zustellung unterdrücken, bis sich die Pendelachse innerhalb eines Umkehrbereichs (ii1, ii2) oder an einem Umkehrpunkt (U1, U2) befindet.

Pendelbewegung während der Zustellung im Umkehrpunkt **anhalten**.

Pendelbewegung nach beendeter Teilzustellung wieder **starten**.

Start der nächsten Teilzustellung festlegen.

3. Pendeln- und Zustellachse zuordnen sowie Gesamt- und Teilzustellung festlegen.

Pendelparameter festlegen

Zuordnung von Pendel- und Zustellachse: OSCILL

```
OSCILL[Pendelachse] = (Zustellachse1, Zustellachse2, Zustellachse3)
```

Mit dem Befehl `OSCILL` erfolgen die Achszuordnungen und der Start der Pendelbewegung.

Maximal können einer Pendelachse 3 Zustellachsen zugewiesen werden.

Hinweis

Vor dem Start des Pendelns müssen die Synchronbedingungen für das Verhalten der Achsen festgelegt sein.

Zustellungen festlegen: POSP

```
POSP[Zustellachse] = (Endpos, Teillänge, Modus)
```

Mit dem Befehl `POSP` werden der Steuerung mitgeteilt:

- Gesamtzustellung (über die Endposition)
- Die Größe der jeweiligen Teilzustellung am Umkehrpunkt bzw. im Umkehrbereich
- Das Teilzustellverhalten bei Erreichen der Endposition (über den Modus)

Modus = 0	Für die beiden letzten Teilzustellungen erfolgt eine Aufteilung des verbleibenden Weges bis zum Zielpunkt auf 2 gleich große Restschritte (Voreinstellung).
Modus = 1	Alle Teilzustellungen sind gleich groß. Sie werden aus der Gesamtzustellung berechnet.

Bewegungssynchronaktionen festlegen

Die im folgenden ausgeführten Bewegungssynchronaktionen werden ganz allgemein zum Pendeln verwendet.

Sie finden Beispiellösungen für die Lösung von einzelnen Anforderungen, die Ihnen als Bausteine für die Erstellung von anwenderspezifischen Pendelbewegungen dienen.

Hinweis

Im Einzelfall können die Synchronbedingungen auch anders programmiert werden.

Schlüsselwörter

WHEN ... DO ...	wenn..., dann...
WHENEVER ... DO	immer wenn..., dann...

Funktionen

Mit den im folgenden detailliert beschriebenen Sprachmitteln können Sie folgende Funktionen

realisieren:

1. Zustellung im Umkehrpunkt.
2. Zustellung im Umkehrbereich.
3. Zustellung in beiden Umkehrpunkten.
4. Anhalten der Pendelbewegung im Umkehrpunkt.
5. Pendelbewegung wieder starten.
6. Teilzustellung nicht zu früh starten.

Für alle hier beispielhaft dargestellten Synchronaktionen gelten die Annahmen:

- Umkehrpunkt 1 < Umkehrpunkt 2
- Z = Pendelachse
- X = Zustellachse

Hinweis

Für nähere Erläuterungen, siehe Kapitel Bewegungssynchronaktionen.

Pendeln- und Zustellachse zuordnen sowie Gesamt- und Teilzustellung festlegen

Zustellung im Umkehrbereich

Die Zustellbewegung soll innerhalb eines Umkehrbereichs beginnen, bevor der Umkehrpunkt erreicht ist.

Diese Synchronaktionen verhindern die Zustellbewegung, bis sich die Pendelachse in einem Umkehrbereich befindet.

Unter den gegebenen Annahmen (siehe oben) ergeben sich folgende Anweisungen:

Umkehrbereich 1:

WHENEVER \$AA_IM[Z]>\$SA_OSCILL_RESERVE_POS1[Z]+ii1 DO \$AA_OVR[X]=0 ; Immer wenn die aktuelle Position der Pendelachse im MKS größer als der Beginn des Umkehrbereichs 1 ist, dann setze den axialen Override der Zustellachse auf 0%.

Umkehrbereich 2:

WHENEVER \$AA_IM[Z]<\$SA_OSCILL_RESERVE_POS2[Z]+ii2 DO \$AA_OVR[X]=0 ; Immer wenn die aktuelle Position der Pendelachse im KS kleiner als der Beginn des Umkehrbereichs 2 ist, dann setze den axialen Override der Zustellachse auf 0%.

Zustellung im Umkehrpunkt

Solange die Pendelachse den Umkehrpunkt nicht erreicht hat, findet keine Bewegung der Zustellachse statt.

Unter den gegebenen Annahmen (siehe oben) ergeben sich folgende Anweisungen:

Umkehrbereich 1:

WHENEVER \$AA_IM[Z]<>\$SA_OSCILL_RESERVE_POS1[Z] DO \$AA_OVR[X]=0 \$AA_OVR[Z]=100 ; Immer wenn die aktuelle Position der Pendelachse Z im MKS größer oder kleiner als die Position des Umkehrpunkts 1 ist, dann setze den axialen Override der Zustellachse X auf 0% und den axialen Override der Pendelachse Z auf 100%.

Umkehrbereich 2:

Für Umkehrpunkt 2:

WHENEVER \$AA_IM[Z] <=>\$SA_OSCILL_RESERVE_POS2[Z] DO \$AA_OVR[X]=0 \$AA_OVR[Z]=100 ; Immer wenn die aktuelle Position der Pendelachse Zu im MKS größer oder kleiner als die Position des Umkehrpunkts 2 ist, dann setze den axialen Override der Zustellachse X auf 0% und den axialen Override der Pendelachse Z auf 100%.

Anhalten der Pendelbewegung im Umkehrpunkt

Die Pendelachse wird am Umkehrpunkt angehalten, gleichzeitig beginnt die Zustellbewegung. Die Pendelbewegung wird fortgesetzt, wenn die Zustellbewegung vollständig ausgeführt ist. Gleichzeitig kann diese Synchronaktion dazu benutzt werden, die Zustellbewegung zu starten, falls diese durch eine vorhergehende Synchronaktion, die noch wirksam ist, gestoppt wurde.

Unter den gegebenen Annahmen (siehe oben) ergeben sich folgende Anweisungen:

Umkehrbereich 1:

WHENEVER \$SA_IM[Z] == \$SA_OSCILL_RESERVE_POS1[Z] DO \$AA_OVR[X]=0 \$AA_OVR[Z]=100 ; Immer wenn die aktuelle Position der Pendelachse im MKS gleich der Umkehrposition 1 ist, dann setze den axialen Override der Pendelachse auf 0% und den axialen Override der Zustellachse auf 100%.

Umkehrbereich 2:

WHENEVER \$SA_IM[Z] == \$SA_OSCILL_RESERVE_POS2[Z] DO \$AA_OVR[X]=0 \$AA_OVR[Z]=100 ; Immer wenn die aktuelle Position der Pendelachse Zu im MKS gleich der Umkehrposition 2 ist, dann setze den axialen Override der Pendelachse X auf 0% und den axialen Override der Zustellachse auf 100%.

Online-Auswertung des Umkehrpunktes

Steht auf der rechten Seite des Vergleichs eine mit \$\$ gekennzeichnete Hauptlaufvariable, so werden die beiden Variablen im IPO-Takt laufend ausgewertet und miteinander verglichen.

Hinweis

Mehr Informationen hierzu siehe Kapitel "Bewegungssynchronaktionen".

Pendelbewegung wieder starten

Diese Synchronaktion wird dazu benutzt, die Bewegung der Pendelachse fortzusetzen, wenn die Teilzustellbewegung abgeschlossen ist.

Unter den gegebenen Annahmen (siehe oben) ergeben sich folgende Anweisungen:

```
WHENEVER $AA_DTEPW[X]==0 DO $AA_OVR[Z]=100 ;
```

Immer wenn der REStweg für die Teilzustellung der Zustellachse X im WKS gleich Null ist, dann setze den axialen Override der Pendelachse auf 100%.

Nächste Teilzustellung

Nach erfolgter Zustellung muss ein zu frühes Starten der nächsten Teilzustellung verhindert werden.

Dazu wird ein kanalspezifischer Merker (\$AC_MARKER[Index]) verwendet, der am Ende der Teilzustellung (Teilrestweg = 0) gesetzt wird und beim Verlassen des Umkehrbereichs gelöscht wird. Dann wird mit einer Synchronaktion die nächste Zustellbewegung verhindert.

Unter den gegebenen Annahmen (siehe oben) ergeben sich z. B. für Umkehrpunkt 1 folgende Anweisungen:

1. Marker setzen:

```
WHENEVER $AA_DTEPW[X]==0 DO $AC_MARKER[1]=1 ;
```

Immer wenn der Restweg für die Teilzustellung der Zustellachse X im WKS gleich Null ist, dann setze den Merker mit Index 1 auf 1.

2. Marker löschen

```
WHENEVER $AA_IM[Z]<> $SA_OSCILL_RESERVE_POS1[Z] DO $AC_MARKER[1]=0 ;
```

Immer wenn die aktuelle Position der Pendelachse Z ium MKS größer oder kleiner als die Position des Umkehrpunkts 1 ist, dann setze den Merker 1 auf 0.

3. Zustellung verhindern

```
WHENEVER $AC_MARKER[1]==1 DO $AA_OVR[X]=0 ;
```

Immer wenn der Merker 1 gleich ist, dann setze den axialen Override der Zustellachse X auf 0%.

Stanzen und Nibbeln

12.1 Aktivierung, Deaktivierung

12.1.1 Stanzen und Nibbeln ein oder aus (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC)

Funktion

Stanzen bzw. Nibbeln aktivieren/deaktivieren

Mit `PON` und `SON` wird die Stanz- bzw. Nibbelfunktion aktiviert. `SPOF` beendet alle stanz- und nibbelspezifischen Funktionen. Die modal wirksamen Befehle `PON` und `SON` schließen sich gegenseitig aus, d. h. `PON` deaktiviert `SON` und umgekehrt.

Stanzen/Nibbeln mit Vorspann

Die Funktionen `SONS` und `PONS` schalten ebenfalls die Stanz- bzw. Nibbelfunktionen ein.

Im Gegensatz zu der bei `SON/PON` wirksamen Hubsteuerung auf Interpolationsebene erfolgt bei diesen Funktionen die signaltechnische Steuerung der Hubauslösung auf Servoebene. Hierdurch kann mit höheren Hubfrequenzen und damit höherer Stanzleistung gearbeitet werden.

Während der Signalauswertung im Vorspann sind alle Funktionen verriegelt, die zur Positionsänderung der Nibbel- oder Stanzachsen führen (z. B. Handradfahren, Änderungen von Frames über PLC, Messfunktionen).

Stanzen mit Verzögerung

`PDELAYON` bewirkt eine verzögerte Ausgabe des Stanzhubs. Der modal wirksame Befehl hat vorbereitende Funktion und steht damit in der Regel vor `PON`. Nach `PDELAYOF` wird normal weitergestanzt.

Hinweis

Die Verzögerungszeit wird eingestellt im Settingdatum
SD42400 \$SC_PUNCH_DWELLTIME.

Wegabhängige Beschleunigung

Mit `PUNCHACC` kann eine Beschleunigungskennlinie festgelegt werden, die je nach Lochabstand unterschiedliche Beschleunigungen definiert.

Zweites Stanz-Interface

Maschinen, die abwechselnd ein zweites Stand-Interface (zweite Stanzeinheit oder ein vergleichbares Medium) nutzen sollen, können auf ein zweites Paar der schnellen digitalen Ein- und Ausgänge der Steuerung (I/O-Paar) umgeschaltet werden. Für beide Stand-Interfaces ist die volle Stanz-/Nibbel-Funktionalität nutzbar. Die Umschaltung zwischen erstem und zweitem Stanz-Interface erfolgt über die Befehle SPIF1 und SPIF2.

Hinweis

Voraussetzung: Über Maschinendaten muss ein zweites I/O-Paar für die Stanzfunktionalität definiert sein (→ siehe Angaben des Maschinenherstellers!).

Syntax

```
PON G... X... Y... Z...
SON G... X... Y... Z...
SONS G... X... Y... Z...
PONS G... X... Y... Z...
PDELAYON
PDELAYOF
PUNCHACC (<Smin>, <Amin>, <Smax>, <Amax>)
SPIF1/SPIF2
SPOF
```

Bedeutung

PON	Stanzen aktivieren
SON	Nibbeln aktivieren
PONS	Stanzen mit Vorspann aktivieren
SONS	Nibbeln mit Vorspann aktivieren
SPOF	Stanzen/Nibbeln deaktivieren
PDELAYON	Stanzen mit Verzögerung aktivieren
PDELAYOF	Stanzen mit Verzögerung deaktivieren
PUNCHACC	Wegabhängige Beschleunigung aktivieren

Parameter:

<Smin>	Kleinster Lochabstand
<Amin>	Anfangsbeschleunigung
	<Amin> kann größer als <Amax> sein.
<Smax>	Größter Lochabstand
<Amax>	Endbeschleunigung
	<Amax> kann größer als <Amin> sein.

SPIF1 **Erstes** Stanz-Interface aktivieren
Die Hubsteuerung erfolgt über das erste Paar der schnellen I/O.

SPIF2 **Zweites** Stanz-Interface aktivieren
Die Hubsteuerung erfolgt über das zweite Paar der schnellen I/O.

Hinweis:
Nach RESET oder Steuerungshochlauf ist immer das erste Stanz-Interface aktiv. Wird nur ein Stanz-Interface benutzt, so muss dieses nicht programmiert werden.

Beispiele

Beispiel 1: Nibbeln aktivieren

Programmcode	Kommentar
...	
N70 X50 SPOF	; Positionieren ohne Stanzauslösung.
N80 X100 SON	; Nibbeln aktivieren, Auslösung eines Hubs vor der Bewegung (X=50) und am Ende der programmierten Bewegung (X=100).
...	

Beispiel 2: Stanzen mit Verzögerung

Programmcode	Kommentar
...	
N170 PDELAYON X100 SPOF	; Positionieren ohne Stanzauslösung, Aktivierung der verzögerten Stanzauslösung.
N180 X800 PON	; Stanzen aktivieren. Nach Erreichen der Endposition wird Stanzhub verzögert ausgegeben.
N190 PDELAYOF X700	; Stanzen mit Verzögerung deaktivieren, normale Stanzauslösung am Ende der programmierten Bewegung.
...	

Beispiel 3: Stanzen mit zwei Stand-Interfaces

Programmcode	Kommentar
...	
N170 SPIF1 X100 PON	; Am Ende des Satzes erfolgt eine Hubauslösung auf dem ersten schnellen Ausgang. Das Signal "Hub aktiv" wird auf dem ersten Eingang überwacht.
N180 X800 SPIF2	; Die zweite Hubauslösung erfolgt auf dem zweiten schnellen Ausgang. Das Signal "Hub aktiv" wird auf dem zweiten Eingang überwacht.
N190 SPIF1 X700	; Die Hubsteuerung für alle weiteren Hübe erfolgt mit dem ersten Interface.
...	

Weitere Informationen

Stanz- und Nibbeln mit Vorspann (PONS/SONS)

Stanz- und Nibbeln mit Vorspann ist nicht gleichzeitig in mehreren Kanälen möglich. PONS bzw. SONS kann nur jeweils in einem Kanal aktiviert werden.

Wegabhängige Beschleunigung (PUNCHACC)

Beispiel:

PUNCHACC (2, 50, 10, 100)

Lochabstände unter 2mm:

Es wird mit einer Beschleunigung von 50% der Maximalbeschleunigung verfahren.

Lochabstände von 2mm bis 10mm:

Die Beschleunigung wird proportional zum Abstand auf 100% gesteigert.

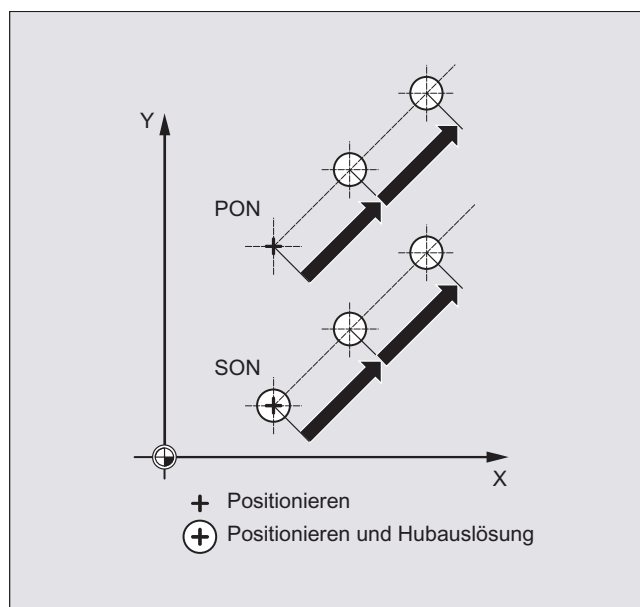
Lochabstände größer als 10mm:

Verfahren mit einer Beschleunigung von 100%.

Auslösung des ersten Hubs

Die Auslösung des ersten Hubs nach Aktivierung der Funktion erfolgt beim Nibbeln und Stanzen zeitlich unterschiedlich:

- PON/PONS:
 - Alle Hübe - auch der des ersten Satzes nach Aktivierung - erfolgen im Satzende.
- SON/SONS:
 - Der erste Hub nach Aktivierung des Nibbelns erfolgt bereits im Satzanfang.
 - Alle weiteren Hübe werden jeweils im Satzende ausgelöst.



Stanzen und Nibbeln auf der Stelle

Eine Hubauslösung erfolgt nur dann, wenn der Satz eine Verfahrinformation für die Stanz- oder Nibbelachsen (Achsen der aktiven Ebene) enthält.

Um dennoch einen Hub an gleicher Stelle auszulösen, wird eine der Stanz-/Nibbelachsen mit Verfahrweg 0 programmiert.

Arbeiten mit drehbaren Werkzeugen

Hinweis

Um drehbare Werkzeuge tangential an die programmierte Bahn anzustellen, verwenden Sie die Tangentialsteuerung.

Verwendung von M-Befehlen

Mit Hilfe der Makrotechnik ist es nach wie vor möglich, spezielle M-Funktionen statt der Sprachbefehle zu benutzen (Kompatibilität). Dabei gelten die folgenden Entsprechungen zu älteren Systemen:

M20, M23	\triangle	SPOF
M22	\triangle	SON
M25	\triangle	PON
M26	\triangle	PDELAYON

Beispiel für Makrodatei:

Programmcode	Kommentar
DEFINE M25 AS PON	; Stanzen ein
DEFINE M125 AS PONS	; Stanzen mit Vorspann ein
DEFINE M22 AS SON	; Nibbeln ein
DEFINE M122 AS SONS	; Nibbeln mit Vorspann ein
DEFINE M26 AS PDELAYON	; Stanzen mit Verzögerung ein
DEFINE M20 AS SPOF	; Stanzen, Nibbeln aus
DEFINE M23 AS SPOF	; Stanzen, Nibbeln aus

Programmierbeispiel:

Programmcode	Kommentar
...	
N100 X100 M20	; Positionieren ohne Stanzauslösung.
N110 X120 M22	; Nibbeln aktivieren, vor und nach Bewegung Hubauslösung.
N120 X150 Y150 M25	; Stanzen aktivieren, Hubauslösung am Ende der Bewegung.
...	

12.2 Automatische Wegaufteilung

Funktion

Unterteilung in Teilstrecken

Bei aktiviertem Stanzen bzw. Nibbeln bewirken sowohl SPP als auch SPN eine Aufteilung der für die Bahnachsen programmierten Gesamtverfahrstrecke in eine Anzahl von gleichlangen Teilstrecken (äquidistante Wegaufteilung). Intern entspricht jede Teilstrecke einem Satz.

Anzahl der Hübe

Beim Stanzen erfolgt der erste Hub am Endpunkt der ersten Teilstrecke, beim Nibbeln dagegen am Startpunkt der ersten Teilstrecke. Über die Gesamtfahrstrecke ergeben sich damit folgende Zahlen:

Stanzen: Anzahl der Hübe = Anzahl der Teilstrecken

Nibbeln: Anzahl der Hübe = Anzahl der Teilstrecken + 1

Hilfsfunktionen

Hilfsfunktionen werden im ersten der erzeugten Sätze ausgeführt.

Syntax

SPP=

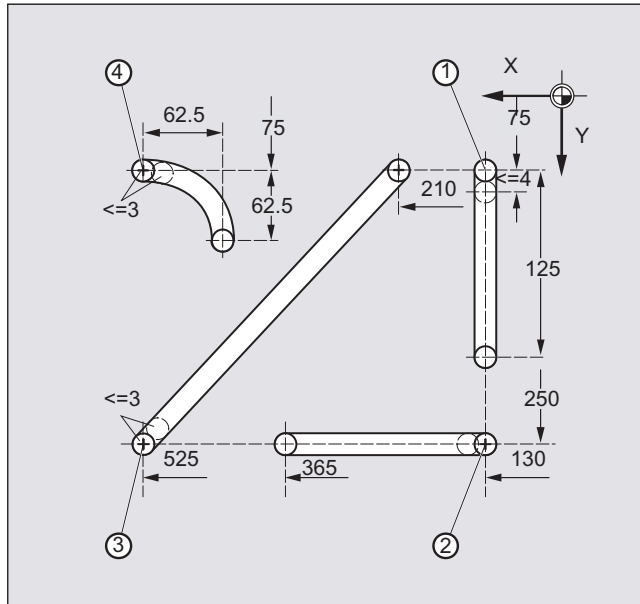
SPN=

Bedeutung

SPP	Größe der Teilstrecke (maximaler Hubabstand); modal wirksam
SPN	Anzahl der Teilstrecken pro Satz; satzweise wirksam

Beispiel 1

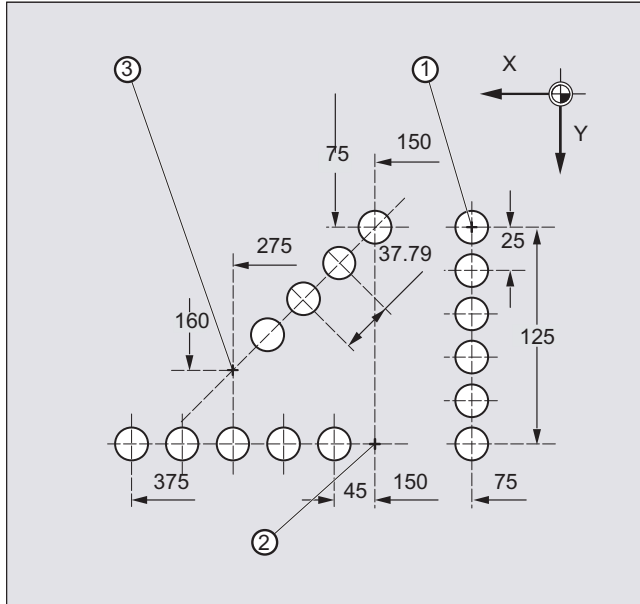
Die programmierten Nibbelstrecken sollen automatisch in gleichgroße Teilstrecken aufgeteilt werden.



Programmcode	Kommentar
N100 G90 X130 Y75 F60 SPOF	; Positionieren auf Startpunkt 1
N110 G91 Y125 SPP=4 SON	; Nibbeln ein; maximale Teilstreckenlänge für automatische Wegaufteilung: 4 mm
N120 G90 Y250 SPOF	; Nibbeln aus; Positionieren auf Startpunkt 2
N130 X365 SON	; Nibbeln ein; maximale Teilstreckenlänge für automatische Wegaufteilung: 4 mm
N140 X525 SPOF	; Nibbeln aus; Positionieren auf Startpunkt 3
N150 X210 Y75 SPP=3 SON	; Nibbeln ein; maximale Teilstreckenlänge für automatische Wegaufteilung: 3 mm
N160 X525 SPOF	; Nibbeln aus; Positionieren auf Startpunkt 4
N170 G02 X-62.5 Y62.5 I J62.5 SPP=3 SON	; Nibbeln ein; maximale Teilstreckenlänge für automatische Wegaufteilung: 3 mm
N180 G00 G90 Y300 SPOF	; Nibbeln aus

Beispiel 2

Für die einzelnen Lochreihen soll eine automatische Wegaufteilung erfolgen. Für die Aufteilung wird jeweils die maximale Teilstreckenlänge (SPP-Wert) angegeben.



Programmcode	Kommentar
N100 G90 X75 Y75 F60 PON	; Positionieren auf Startpunkt 1; Stanz ein Einzelloch stanzen
N110 G91 Y125 SPP=25	; Maximale Teilstreckenlänge für automatische Wegaufteilung: 25 mm
N120 G90 X150 SPOF	; Stanzen aus; Positionieren auf Startpunkt 2
N130 X375 SPP=45 PON	; Stanzen ein; maximale Teilstreckenlänge für automatische Wegaufteilung: 45 mm
N140 X275 Y160 SPOF	; Stanzen aus; Positionieren auf Startpunkt 3
N150 X150 Y75 SPP=40 PON	; Stanzen ein; anstelle der programmierten Teilstreckenlänge von 40 mm wird die ;berechnete Teilstreckenlänge von 37,79 mm verwendet.
N160 G00 Y300 SPOF	; Stanzen aus; Positionieren

12.2.1 Wegaufteilung bei Bahnachsen

Länge der Teilstrecke SPP

Mit SPP geben Sie den maximalen Hubabstand und damit die maximale Länge der Teilstrecken an, in die die Gesamtverfahrstrecke aufgeteilt werden soll. Das Ausschalten des Befehls erfolgt mit $SPOF$ oder $SPP=0$.

Beispiel:

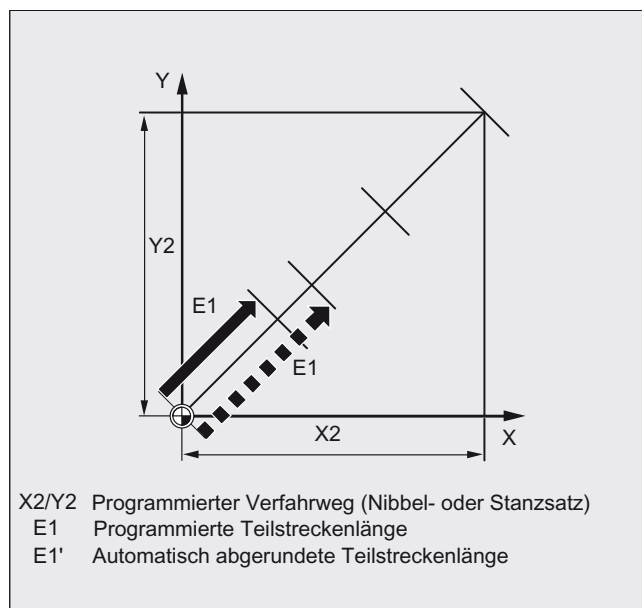
```
N10 SON X0 Y0
```

```
N20 SPP=2 X10
```

Die Gesamtverfahrstrecke von 10 mm wird in 5 Teilstrecken von je 2 mm ($SPP=2$) aufgeteilt.

Hinweis

Die Wegaufteilung mit SPP erfolgt immer äquidistant: alle Teilstrecken sind gleich lang. Das heißt, die programmierte Teilstreckengröße (Wert von SPP) ist nur dann gültig, wenn der Quotient aus Gesamtverfahrstrecke und SPP -Wert ganzzahlig ist. Ist das nicht der Fall, so wird die Größe der Teilstrecke intern so reduziert, dass sich ein ganzzahliger Quotient ergibt.



Beispiel:

```
N10 G1 G91 SON X10 Y10
```

```
N20 SPP=3.5 X15 Y15
```

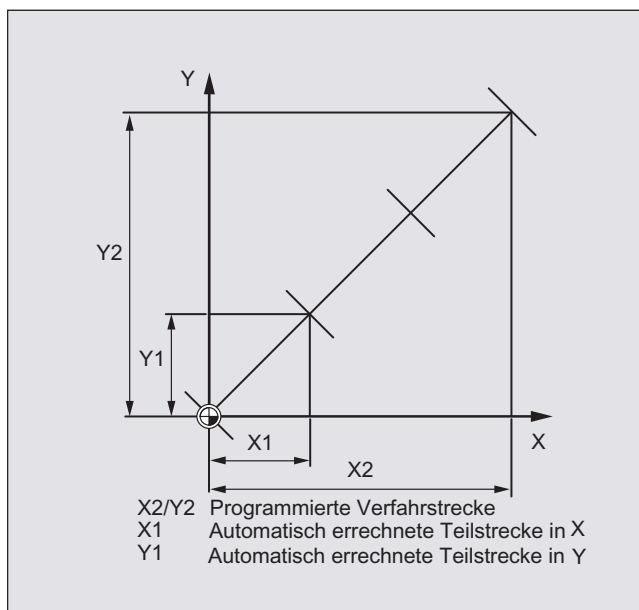
Bei der Gesamtverfahrstrecke von 15 mm und einer Teilstreckenlänge von 3,5 mm ergibt sich ein nicht ganzzahliger Quotient (4.28). Somit erfolgt eine Reduktion des SPP -Werts bis zum nächstmöglichen ganzzahligen Quotienten. In diesem Fall ergibt sich eine Teilstreckenlänge von 3 mm.

Anzahl der Teilstrecken SPN

Mit SPN definieren Sie die Anzahl der Teilstrecken, die aus der Gesamtverfahrstrecke erzeugt werden soll. Die Länge der Teilstrecken wird automatisch berechnet. Da SPN satzweise wirksam ist, muss vorher Stanzen oder Nibbeln mit PON oder SON aktiviert werden.

SPP und SPN im gleichen Satz

Programmieren Sie in einem Satz sowohl die Teilstreckenlänge (SPP) als auch Anzahl der Teilstrecken (SPN), so gilt für diesen Satz SPN , für alle weiteren SPP . Wurde SPP schon vor SPN aktiviert, so wird es nach dem Satz mit SPN wieder wirksam.



Hinweis

Sofern Stanzen/Nibbeln grundsätzlich in der Steuerung verfügbar ist, ist die Programmierung der automatischen Wegaufteilung mit SPN bzw. SPP auch unabhängig von dieser Technologie aktivierbar.

12.2.2 Wegaufteilung bei Einzelachsen

Sind neben den Bahnachsen auch Einzelachsen als Stanz-Nibbel-Achse definiert, so können auch sie der automatischen Wegaufteilung unterliegen.

Verhalten der Einzelachse bei SPP

Die programmierte Länge der Teilstrecke (SPP) bezieht sich grundsätzlich auf die Bahnachsen. Daher wird in einem Satz, in dem neben der Einzelachs-bewegung und dem SPP-Wert keine Bahnachse programmiert ist, der SPP-Wert ignoriert.

Sind sowohl Einzel- als auch Bahnachse im Satz programmiert, so richtet sich das Verhalten der Einzelachse nach der Einstellung des entsprechenden Maschinendatums.

1. Standardeinstellung

Der Weg der Einzelachse wird gleichmäßig auf die durch SPP erzeugten Zwischensätze verteilt.

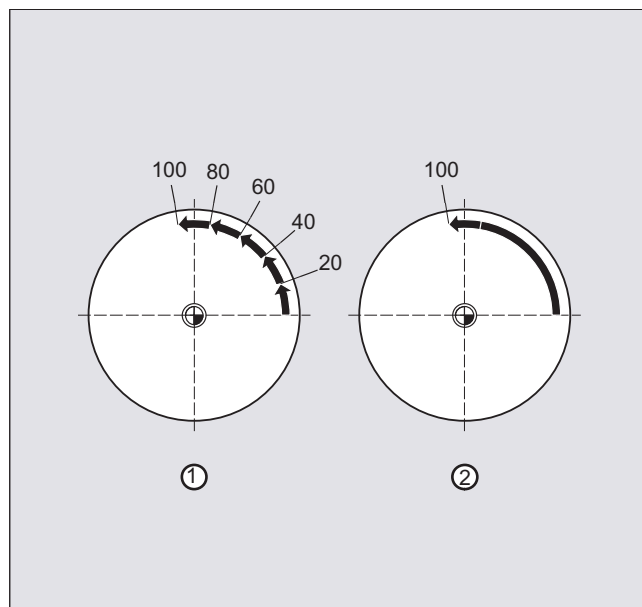
Beispiel:

```
N10 G1 SON X10 A0
```

```
N20 SPP=3 X25 A100
```

Durch die Hubstrecke von 3 mm werden bei der Gesamtverfahrstrecke der X-Achse (Bahnachse) von 15 mm 5 Sätze erzeugt.

Die A-Achse dreht sich damit in jedem Satz um 20° .



1. Einzelachse ohne Wegaufteilung

Die Einzelachse verfährt ihren Gesamtweg im ersten der erzeugten Sätze.

2. Unterschiedliche Wegaufteilung

Das Verhalten der Einzelachse ist abhängig von der Interpolation der Bahnachsen:

- Kreisinterpolation: Wegaufteilung
- Linearinterpolation: keine Wegaufteilung

Verhalten bei SPN

Die programmierte Anzahl von Teilstrecken gilt auch, wenn nicht gleichzeitig eine Bahnachse programmiert ist.

Voraussetzung: Einzelachse ist als Stanz-Nibbel-Achse definiert.

Schleifen

13.1 Schleifenspezifische Werkzeugüberwachung im Teileprogramm (TMON, TMOF)

Funktion

Mit dem Befehl `TMON` können Sie für Schleifwerkzeuge (Typ 400 - 499) die Geometrie- und Drehzahlüberwachung im NC-Teileprogramm aktivieren. Die Überwachung bleibt aktiv, bis sie im Teileprogramm durch den Befehl `TMOF` abgeschaltet wird.

Hinweis

Bitte beachten Sie die Angaben des Maschinenherstellers!

Voraussetzung

Die schleifspezifischen Werkzeug-Parameter `$TC_TPG1` bis `$TC_TPG9` müssen gesetzt sein.

Syntax

`TMON (<T-Nr. >)`
`TMOF (<T-Nr. >)`

Bedeutung

<code>TMON</code>	Befehl zum Einschalten der schleifspezifischen Werkzeugüberwachung
<code>TMOF</code>	Befehl zum Ausschalten der schleifspezifischen Werkzeugüberwachung
<code><T-Nr. ></code>	Angabe der T-Nummer
	Hinweis:
	Nur notwendig, wenn das Werkzeug mit dieser T-Nummer nicht aktiv ist.
<code>TMOF (0)</code>	Überwachung für alle Werkzeuge ausschalten

Weitere Informationen

Schleifenspezifische Werkzeug-Parameter

Parameter	Bedeutung	Datentyp
\$TC_TPG1	Spindelnummer	INT
\$TC_TPG2	Verkettungsvorschrift Die Parameter werden automatisch für die linke und rechte Scheibenseite identisch gehalten.	INT
\$TC_TPG3	Minimaler Scheibenradius	REAL
\$TC_TPG4	Minimale Scheibenbreite	REAL
\$TC_TPG5	Aktuelle Scheibenbreite	REAL
\$TC_TPG6	Maximale Drehzahl	REAL
\$TC_TPG7	Maximale Umfangsgeschwindigkeit	REAL
\$TC_TPG8	Winkel der schrägen Scheibe	REAL
\$TC_TPG9	Parameter-Nummer für Radiusberechnung	INT

Literatur:

Funktionshandbuch Grundfunktionen; Werkzeugkorrektur (W1)

Werkzeugüberwachung einschalten über Werkzeugawahl

In Abhängigkeit von einem Maschinendatum kann für die Schleifwerkzeuge (Typ 400 - 499) die Werkzeugüberwachung implizit mit der Werkzeugawahl eingeschaltet werden.

Zu jedem Zeitpunkt kann für jede Spindel nur **eine** Überwachung aktiv sein.

Geometrieüberwachung

Überwacht werden der aktuelle Scheibenradius und die aktuelle Breite.

Die Überwachung des Drehzahlsollwerts auf den Drehzahlgrenzwert erfolgt zyklisch unter Berücksichtigung des Spindel-Overrides.

Als Drehzahlgrenzwert gilt der kleinere Wert, der sich bei Vergleich von maximaler Drehzahl mit der berechneten Drehzahl aus maximaler Scheibenumfangsgeschwindigkeit und aktuellem Scheibenradius ergibt.

Arbeiten ohne T- und D-Nummer

Per Maschinendatum kann eine Standard-T-Nummer und Standard-D-Nummer eingestellt werden,

die nicht mehr programmiert werden muss und nach Power On / Reset wirksam wird.

Beispiel: Arbeiten mit derselben Schleifscheibe

Über das Maschinendatum kann eingestellt werden, dass das aktive Werkzeug bei Reset erhalten bleibt (siehe " Freie D-Nummernvergabe, Schneidenummer (Seite 443) ").

Weitere Funktionen

14.1 Achsfunktionen (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL)

Funktion

`AXNAME` wird z. B. bei der Erstellung allgemeingültiger Zyklen verwendet, wenn die Namen der Achsen nicht bekannt sind.

`AX` wird für die indirekte Programmierung von Geometrie- und Synchronachsen verwendet. Der Achsbezeichner wird dabei in einer Variablen vom Typ `AXIS` hinterlegt oder von einem Befehl wie `AXNAME` oder `SPI` geliefert.

`SPI` wird verwendet, wenn Achsfunktionen für eine Spindel, z. B. Synchronspindel, programmiert werden.

`AXTOSPI` wird verwendet, um einen Achsbezeichner in einen Spindelindex zu wandeln (Umkehrfunktion zu `SPI`).

`AXSTRING` wird verwendet, um einen Achsbezeichner (Datentyp `AXIS`) in einen String zu wandeln (Umkehrfunktion zu `AXNAME`).

`ISAXIS` wird in allgemeingültigen Zyklen verwendet, um sicherzustellen, dass eine bestimmte Geometrieachse vorhanden ist und damit ein nachfolgender Aufruf von `$P_AXNX` nicht mit Fehler abgebrochen wird.

`MODAXVAL` wird verwendet, um bei Modulo-Rundachsen die Modulo-Position zu ermitteln.

Syntax

```
AXNAME("String")
AX[AXNAME("String")]
SPI(n)

AXTOSPI(A) oder AXTOSPI(B) oder AXTOSPI(C)
AXSTRING(SPI(n))
ISAXIS(<Geometrieachsnummer>)
<Modulo-Position>=MODAXVAL(<Achse>,<Achspannung>)
```

Bedeutung

<code>AXNAME</code>	Konvertiert einen Eingangsstring in Achsbezeichner; der Eingangsstring muss gültigen Achsnamen enthalten.
<code>AX</code>	Variabler Achsbezeichner
<code>SPI</code>	Konvertiert Spindelnummer in Achsbezeichner; der Übergabeparameter muss eine gültige Spindelnummer enthalten.
<code>n</code>	Spindelnummer

<p>AXTOSPI</p> <p>X, Y, Z</p> <p>AXSTRING</p> <p>ISAXIS</p> <p>MODAXVAL</p>	<p>Wandelt einen Achsbezeichner in einen Spindelindex vom Typ Integer um. AXTOSPI entspricht der Umkehrfunktion zu SPI.</p> <p>Achsbezeichner vom Typ AXIS als Variable oder Konstante</p> <p>Es wird der String mit zugeordneter Spindelnummer ausgegeben.</p> <p>Prüft, ob die angegebene Geometrieachse vorhanden ist.</p> <p>Ermittelt bei Modulo-Rundachsen die Modulo-Position; diese entspricht dem Modulo-Rest bezogen auf den parametrisierten Modulo-Bereich (beträgt in der Standardeinstellung 0 bis 360 Grad; über MD30340 MODULO_RANGE_START und MD30330 \$MA_MODULO_RANGE können Beginn und Größe des Modulo-Bereichs verändert werden).</p>
---	---

Hinweis

SPI-Erweiterungen

Die Achsfunktion SPI(n) ist auch für das Lesen und Schreiben von Framekomponenten einsetzbar. Damit können Frames z. B. mit der Syntax \$P_PFRAME[SPI(1),TR]=2.22 geschrieben werden.

Durch die zusätzliche Programmierung von Achspositionen über die Adresse AX[SPI(1)]=<Achsposition> kann eine Achse verfahren werden. Voraussetzung dafür ist, dass sich die Spindel im Positionier- oder Achsbetrieb befindet.

Beispiele

Beispiel 1: AXNAME, AX, ISAXIS

Programmcode	Kommentar
OVRA[AXNAME("Planachse")]=10	; Override für Planachse
AX[AXNAME("Planachse")]=50.2	; Endposition für Planachse
OVRA[SPI(1)]=70	; Override für Spindel 1
AX[SPI(1)]=180	; Endposition für Spindel 1
IF ISAXIS(1)==FALSE GOTOF WEITER	; Abszisse vorhanden?
AX[\$P_AXN1]=100	; Abszisse verfahren
WEITER:	

Beispiel 2: AXSTRING

Bei der Programmierung mit AXSTRING[SPI(n)] wird nicht mehr der Achsindex der Achse, der die Spindel zugeordnet ist, als Spindelnummer ausgegeben, sondern es wird der String "Sn" ausgegeben.

Programmcode	Kommentar
AXSTRING[SPI(2)]	; Es wird der String "S2" ausgegeben.

Beispiel 3: MODAXVAL

Die Modulo-Position der Modulo-Rundachse A soll ermittelt werden.

Ausgangswert für die Berechnung ist die Achsposition 372.55.

Der parametrisierte Modulo-Bereich beträgt 0 bis 360 Grad:

MD30340 MODULO_RANGE_START = 0

MD30330 \$MA_MODULO_RANGE = 360

Programmcode	Kommentar
R10=MODAXVAL(A,372.55)	; Berechnete Modulo-Position R10 = 12.55.

Beispiel 4: MODAXVAL

Wenn sich der programmierte Achsbezeichner nicht auf eine Modulo-Rundachse bezieht, dann wird der zu wandelnde Wert (<Achsposition>) unverändert zurückgegeben.

Programmcode	Kommentar
R11=MODAXVAL(X,372.55)	; X ist Linearachse; R11 = 372.55.

14.2 Umschaltbare Geometrieachsen (GEOAX)

Funktion

Mit der Funktion "Umschaltbare Geometrieachsen" lässt sich der über Maschinendaten konfigurierte Geometrieachsverbund vom Teileprogramm aus verändern. Dabei kann eine als synchrone Zusatzachse definierte Kanalachse eine beliebige Geometrieachse ersetzen.

Syntax

GEOAX (<n>, <Kanalachse>, <n>, <Kanalachse>, <n>, <Kanalachse>)
GEOAX ()

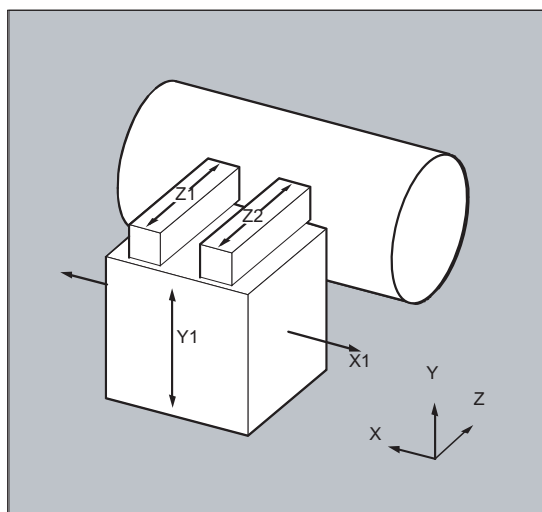
Bedeutung

GEOAX (...)	Befehl zum Umschalten der Geometrieachsen Hinweis: GEOAX () ohne Parameterangabe ruft die Grundkonfiguration der Geometrieachsen auf.
<n>	Mit diesem Parameter wird die Nummer der Geometrieachse angegeben, der die nachfolgend angegebene Kanalachse zugeordnet werden soll. Wertebereich: 1, 2 oder 3 Hinweis: Mit <n>=0 kann die nachfolgend angegebene Kanalachse aus dem Geometrieachsverbund ersatzlos entfernt werden.
<Kanalachse>	Mit diesem Parameter wird der Name der Kanalachse angegeben, die in den Geometrieachsverbund aufgenommen werden soll.

Beispiele

Beispiel 1: Zwei Achsen wechselweise als Geometrieachse schalten

Ein Werkzeugschlitten kann über die Kanalachsen X1, Y1, Z1, Z2 verfahren werden:



Die Geometrieachsen sind so projektiert, dass nach dem Einschalten zunächst Z1 als 3. Geometrieachse unter dem Geometrieachsamen "Z" wirksam ist und zusammen mit X1 und Y1 den Geometrieachsverbund bildet.

Im Teileprogramm sollen nun die Achsen Z1 und Z2 wechselweise als Geometrieachse Z zum Einsatz kommen:

Programmcode	Kommentar
...	
N100 GEOAX(3,Z2)	; Als 3. Geometrieachse (Z) fungiert Kanalachse Z2.
N110 G1 ...	
N120 GEOAX(3,Z1)	; Als 3. Geometrieachse (Z) fungiert Kanalachse Z1.
...	

Beispiel 2: Umschalten der Geometrieachsen bei 6 Kanalachsen

Eine Maschine besitzt 6 Kanalachsen mit den Namen XX, YY, ZZ, U, V, W.

Die Grundeinstellung der Geometrieachskonfiguration über Maschinendaten ist:

Kanalachse XX = 1. Geometrieachse (X-Achse)

Kanalachse YY = 2. Geometrieachse (Y-Achse)

Kanalachse ZZ = 3. Geometrieachse (Z-Achse)

Programmcode	Kommentar
N10 GEOAX()	; Grundkonfiguration der Geometrieachsen ist wirksam.
N20 G0 X0 Y0 Z0 U0 V0 W0	; Alle Achsen im Eilgang auf Position 0.
N30 GEOAX(1,U,2,V,3,W)	; Kanalachse U wird zur ersten (X), V zur zweiten (Y) und W zur dritten Geometrieachse (Z).
N40 GEOAX(1,XX,3,ZZ)	; Kanalachse XX wird zur ersten (X), ZZ zur dritten Geometrieachse (Z). Kanalachse V bleibt zweite Geometrieachse (Y).
N50 G17 G2 X20 I10 F1000	; Vollkreis in der X/Y-Ebene. Es fahren die Kanalachsen XX und V.
N60 GEOAX(2,W)	; Kanalachse W wird zweite Geometrieachse (Y).
N80 G17 G2 X20 I10 F1000	; Vollkreis in der X/Y-Ebene. Es fahren die Kanalachsen XX und W.
N90 GEOAX()	; Zurücksetzen auf Grundzustand.
N100 GEOAX(1,U,2,V,3,W)	; Kanalachse U wird zur ersten (X), V zur zweiten (Y) und W zur dritten Geometrieachse (Z).
N110 G1 X10 Y10 Z10 XX=25	; Kanalachsen U, V, W fahren jeweils auf die Position 10. XX als Zusatzachse fährt auf Position 25.
N120 GEOAX(0,V)	; V wird aus Geometrieachsverbund herausgenommen. U und W sind weiterhin erste (X) und dritte Geometrieachse (Z). Die zweite Geometrieachse (Y) bleibt unbelegt.
N130 GEOAX(1,U,2,V,3,W)	; Kanalachse U bleibt erste (X), V wird zur zweiten (Y), W bleibt dritte Geometrieachse (Z).
N140 GEOAX(3,V)	; V wird zur dritten Geometrieachse (Z), wobei W überschrieben und damit aus dem Geometrieachsverbund herausgenommen wird. Die zweite Geometrieachse (Y) ist nach wie vor unbelegt.

Hinweis

Achskonfiguration

Die Zuordnung zwischen den Geometrieachsen, Zusatzachsen, Kanalachsen und Maschinenachsen, sowie die Festlegung der Namen der einzelnen Achstypen wird über folgende Maschinendaten getroffen:

MD20050 \$MC_AXCONF_GEOAX_ASSIGN_TAB (Zuordnung Geometrieachse zu Kanalachse)

MD20060 \$MC_AXCONF_GEOAX_NAME_TAB (Geometrieachsname im Kanal)

MD20070 \$MC_AXCONF_MACHAX_USED (Maschinenachsnummer gültig im Kanal)

MD20080 \$MC_AXCONF_CHANAX_NAME_TAB (Kanalachsname im Kanal)

MD10000 \$MN_AXCONF_MACHAX_NAME_TAB (Maschinenachsname)

MD35000 \$MA_SPIND_ASSIGN_TO_MACHAX (Zuordnung Spindel zu Maschinenachse)

Literatur:

Funktionshandbuch Grundfunktionen; Achsen, Koordinatensysteme Frames (K2)

Einschränkungen

- Die Umschaltung der Geometrieachsen ist nicht möglich bei:
 - aktiver Transformation
 - aktiver Spline-Interpolation
 - aktiver Werkzeugradiuskorrektur
 - aktiver Werkzeugfeinkorrektur
- Weisen Geometrieachse und Kanalachse gleiche Namen auf, ist kein Wechsel der jeweiligen Geometrieachse möglich.
- Keine der an der Umschaltung beteiligten Achsen darf an einer Aktion beteiligt sein, die über die Satzgrenzen hinweg andauern kann, wie es z. B. bei Positionierachsen vom Typ A oder bei Folgeachsen möglich ist.
- Mit dem Befehl `GEOAX` können nur bereits beim Einschalten vorhandene Geometrieachsen ersetzt werden (also keine neuen definiert werden).
- Ein Achstausch mit `GEOAX` während der Aufbereitung der Konturtablelle (`CONTPRON`, `CONTDCON`) führt zum Alarm.

Randbedingungen

Achszustand nach dem Ersetzen

Eine durch die Umschaltung im Geometrieachsverbund ersetzte Achse ist nach dem Umschaltvorgang über ihren Kanalachsamen als Zusatzachse programmierbar.

Frames, Schutzbereiche, Arbeitsfeldbegrenzungen

Mit dem Umschalten der Geometrieachsen werden alle Frames, Schutzbereiche und Arbeitsfeldbegrenzungen gelöscht.

Polarkoordinaten

Ein Tausch der Geometrieachsen mit `GEOAX` setzt analog einem Ebenenwechsel mit `G17-G19` die modalen Polarkoordinaten auf den Wert 0.

DRF, NPV

Eine eventuelle Handrad-Verschiebung (DRF) oder eine externe Nullpunktverschiebung (NPV) bleibt nach der Umschaltung wirksam.

Grundkonfiguration der Geometrieachsen

Der Befehl `GEOAX ()` ruft die Grundkonfiguration des Geometrieachsverbunds auf.

Nach POWER ON und bei Umschalten in die Betriebsart "Referenzpunktfahren" wird automatisch auf die Grundkonfiguration zurückgeschaltet.

Werkzeuglängenkorrektur

Eine aktive Werkzeuglängenkorrektur ist auch nach dem Umschaltvorgang wirksam. Sie gilt jedoch für die neu aufgenommenen bzw. positionsgetauschten Geometrieachsen als noch nicht herausgefahren. Beim ersten Bewegungsbefehl für diese Geometrieachsen besteht der resultierende Verfahrweg dementsprechend aus der Summe von Werkzeuglängenkorrektur und programmiertem Verfahrweg.

Geometrieachsen, die bei einer Umschaltung ihre Position im Achsverband beibehalten, behalten auch ihren Status bezüglich der Werkzeuglängenkorrektur.

Geometrieachskonfiguration bei aktiver Transformation

Die in einer aktiven Transformation geltende Geometrieachskonfiguration (festgelegt über Maschinendaten) ist über die Funktion "Umschaltbare Geometrieachsen" nicht veränderbar.

Besteht die Notwendigkeit, im Zusammenhang mit Transformationen die Geometrieachskonfiguration zu ändern, so ist dies nur über eine weitere Transformation möglich.

Eine über `GEOAX` veränderte Geometrieachskonfiguration wird durch Aktivierung einer Transformation gelöscht.

Widersprechen sich Einstellungen der Maschinendaten für die Transformation und für die Umschaltung von Geometrieachsen, so haben die Einstellungen in der Transformation Vorrang.

Beispiel:

Eine Transformation sei aktiv. Laut Maschinendaten soll die Transformation bei einem RESET erhalten bleiben, gleichzeitig soll bei einem RESET jedoch die Grundkonfiguration der Geometrieachsen hergestellt werden. In diesem Fall bleibt die Geometrieachskonfiguration erhalten, die mit der Transformation festgelegt wurde.

14.3 Achscontainer (AXCTSWE, AXCTSWED, AXCTSWEC)

Funktion

Über die Befehle AXCTSWE bzw. AXCTSWED wird die Drehung des angegebenen Achscontainers freigegeben.

Über den Befehl AXCTSWEC wird eine bereits gegebene Freigabe zur Achscontainer-Drehung wieder zurückgenommen.

Syntax

AXCTSWE (<ID>)
AXCTSWED (<ID>)
AXCTSWEC (<ID>)

Bedeutung

AXCTSWE: Freigabe zum Drehen des Achscontainers
Die Programmverarbeitung wird durch AXCTSWE nicht angehalten.
Die Drehung erfolgt, sobald alle am Achscontainer beteiligten Kanäle die Freigabe erteilt haben.

AXCTSWED: Freigabe zum Drehen des Achscontainers ohne Berücksichtigung der anderen am Achscontainer beteiligten Kanäle

Hinweis

- Befehlsvariante zur Vereinfachung der Inbetriebnahme des Teileprogramms bzw. Synchronaktion.
- Das Verhalten bezüglich der anderen am Achscontainer beteiligten Kanäle, kann vorgegeben werden über:
MD12760 \$MN_AXCT_FUNCTION_MASK, Bit 0

AXCTSWEC: Rücknahme der Freigabe zum Drehen des Achscontainers

Hinweis

Die Freigabe zum Drehen des Achscontainers kann nur zurück genommen werden, wenn die Drehung noch nicht begonnen wurde:

\$AN_AXCTSWA[<Achscontainer>] == 0

Zu Systemvariable siehe "Achscontainer (AXCTSWE, AXCTSWED, AXCTSWEC) (Seite 606)"

<ID>: Bezeichner des Achscontainers oder einer Containerachse:

CT<Nummer>: Defaultbezeichner eines Achscontainers:

MD12750 \$MN_AXCT_NAME_TAB

Beispiel: CT1

<Container>: Anwenderspezifischer Bezeichner eines Achscontainers:

MD12750 \$MN_AXCT_NAME_TAB

Beispiel: CONTAINER_1

<Achse>: Bezeichner einer im Kanal bekannten Containerachse

Hinweis

Schrittweite

Die Schrittweite einer Achscontainer-Drehung wird eingestellt über das Settingdatum:

SD41700 \$SN_AXCT_SWWIDTH

Weitere Informationen

Diagnose

Der aktuelle Status eines Achscontainers kann über folgende Systemvariable gelesen werden:

Systemvariable	Typ	Beschreibung
\$AC_AXCTSWA[<Name>]	BOOL	Kanal-spezifischer Status des Achscontainers
\$AN_AXCTSWA[<Achscontainer>]	BOOL	NCU-spezifischer Status des Achscontainer
\$AN_AXCTSWE[<Achscontainer>]	INT	Slot-spezifischer Status der Achscontainer-Drehung Die Systemvariable liefert Bit-weise den Status der Slots des Achscontainers. Jedes Bit entspricht einem Slot.
\$AN_AXCTAS[<Achscontainer>]	INT	Anzahl der Plätze (Slots) um die der Achscontainer aktuell weitergeschaltet wurde.

Achscontainer-Drehung mit implizitem GET / GETD

Über nachfolgendes Maschinendatum kann eingestellt werden, dass mit dem Befehl AXCTSWE alle Container-Achsen des Kanals mittels impliziten GET / GETD in den Kanal geholt werden. Ein Achstausch ist erst nach erfolgter Container-Drehung wieder möglich.

MD10722 \$MN_AXCHANGE_MASK, Bit 1 = 1

Hinweis

Die Achscontainer-Drehung mit impliziten GET / GETD wird **nicht** für eine Achse im Zustand "Hauptlaufachse" (z. B. PLC-Achse) durchgeführt, da die Achse für die Achscontainer-Drehung den Zustand verlassen müsste.

14.4 Warten auf gültige Achsposition (WAITENC)

Funktion

Mit dem Sprachbefehl `WAITENC` kann im NC-Programm gewartet werden, bis für die mit MD34800 `$MA_WAIT_ENC_VALID = 1` projizierten Achsen synchronisierte bzw. restaurierte Achspositionen zur Verfügung stehen.

Im Wartezustand kann eine Unterbrechung erfolgen, z. B. durch Start eines ASUPs oder durch Betriebsartenwechsel nach JOG. Mit der Programmfortsetzung wird der Wartezustand ggf. wieder eingenommen.

Hinweis

Der Wartezustand wird in der Bedienoberfläche durch den Haltezustand "Warten auf Mess-System" angezeigt.

Syntax

`WAITENC` kann im Programmteil eines beliebigen NC-Programms programmiert werden.

Die Programmierung muss in einem eigenen Satz erfolgen:

```
...  
WAITENC  
...
```

Beispiel

`WAITENC` wird z. B. im ereignisgesteuerten Anwenderprogramm `.../_N_CMA_DIR/_N_PROG_EVENT_SPF` verwendet, wie das folgende Anwendungsbeispiel zeigt.

Anwendungsbeispiel: Werkzeugrückzug nach POWER OFF mit Orientierungstransformation

Eine Bearbeitung mit Werkzeugorientierung wurde durch Spannungsausfall abgebrochen. Beim anschließenden Hochlauf wird das ereignisgesteuerte Anwenderprogramm `.../_N_CMA_DIR/_N_PROG_EVENT_SPF` aufgerufen.

Im ereignisgesteuerten Anwenderprogramm wird mit `WAITENC` auf synchronisierte bzw. restaurierte Achspositionen gewartet, um danach einen Frame berechnen zu können, der das WKS in Werkzeugrichtung ausrichtet.

Programmcode	Kommentar
...	
IF \$P_PROG_EVENT == 4	; Hochlauf.
IF \$P_TRAFO <> 0	; Transformation wurde angewählt.
WAITENC	; Warten auf gültige Achspositionen der Orientierungsachsen.
TOROTZ	; Z-Achse des WKS in Richtung der Werkzeugachse drehen.
ENDIF	
M17	
ENDIF	
...	

Danach kann das Werkzeug in der Betriebsart JOG durch eine Rückzugsbewegung in Richtung der Werkzeugachse freigefahren werden.

14.5 Programmierbare Parametersatzumschaltung (SCPARA)

Funktion

Mit dem Befehl `SCPARA` kann für eine Achse die Umschaltung auf einen bestimmten Parametersatz angefordert werden.

Hinweis

Keine Parametersatzumschaltung während Gewindebearbeitung

Bei Gewindeschneiden `G33` und Gewindebohren `G331 / G332` wird der Parametersatz von der Steuerung ausgewählt und kann nicht verändert werden.

Gesperrte Parametersatzumschaltung

Eine Parametersatzumschaltung kann auch über die NC/PLC-Nahstelle angefordert werden. Um Umschaltkonflikte zu vermeiden, kann über die NC/PLC-Nahstelle die Parametersatzumschaltung der NC (`SCPARA`) gesperrt werden:

DB31, ... DBX9.3 (Parametersatzvorgabe durch NC gesperrt)

Hinweis

Wird eine Parametersatzumschaltung durch `SCPARA` angefordert, während die Parametersatzumschaltung über die NC/PLC-Nahstelle gesperrt ist, wird die Umschaltung ohne Fehlermeldung abgewiesen.

Syntax

```
SCPARA [<Achse>] = <Wert>
```

Bedeutung

<code>SCPARA:</code>	Befehl: Parametersatz umschalten
<code><Achse>:</code>	Achsbezeichner (Kanalachse)
	Typ: AXIS
<code><Wert>:</code>	Parametersatznummer: 1, 2, 3, ... max. Parametersatznummer

Beispiel

Programmcode	Kommentar
...	
N110 SCPARA[X] = 3	; Anwahl: Achse X, 3. Parametersatz
...	

Weitere Informationen

Freigabe der Parametersatzumschaltung

Die Parametersatzumschaltung der Achse muss explizit freigegeben werden:

MD35590 \$MA_PARAMSET_CHANGE_ENABLE[<Achse>]

Parametersatznummer lesen

Die Nummer des angewählten Parametersatzes (Soll-Parametersatz) kann über die Systemvariable \$AA_SCPAR gelesen werden.

Literatur

Ausführliche Informationen zu den Parametersätzen finden sich in:

Funktionshandbuch Grundfunktionen; Kapitel "Geschwindigkeiten, Soll-/Istwertsysteme, Regelung (G2)" > "Regelung" > "Parametersätze des Lagereglers"

14.6 Vorhandenen NC-Sprachumfang prüfen (STRINGIS)

Funktion

Mit der Funktion `STRINGIS (...)` kann geprüft werden, ob der angegebene String als Element der NC-Programmiersprache im aktuellen Sprachumfang zur Verfügung steht.

Definition

```
INT STRINGIS (STRING <Name>)
```

Syntax

```
STRINGIS (<Name>)
```

Bedeutung

STRINGIS: Funktion mit Rückgabewert
<Name>: Name des zu prüfenden Elementes der NC-Programmiersprache
Rückgabewert: Das Format des Rückgabewertes ist yxx (Dezimal).

Elemente der NC-Programmiersprache

Folgende Elemente der NC-Programmiersprache können geprüft werden:

- G-Codes aller existierenden G-Funktionsgruppen z.B. `G0`, `INVCW`, `POLY`, `ROT`, `KONT`, `SOFT`, `CUT2D`, `CDON`, `RMB`, `SPATH`
- DIN- oder NC-Adressen wie z.B. `ADIS`, `RNDM`, `SPN`, `SR`, `MEAS`
- Funktionen z.B. `TANG (...)` oder `GETMDACT`
- Prozeduren z.B. `SBLOF`.
- Schlüsselworte z.B. `ACN`, `DEFINE` oder `SETMS`
- Systemdaten z.B. Maschinendaten `$M...`, Settingdaten `$S...` oder Optionsdaten `$O...`
- Systemvariable `$A...`, `$V...`, `$P...`
- Rechenparameter `R...`
- Zyklennamen von aktivierten Zyklen
- GUD- und LUD-Variablen
- Makro-Namen
- Label-Namen

Rückgabewert

Der Rückgabewert ist nur in den erst 3 Dezimalstellen relevant. Das Format des Rückgabewertes ist yxx, mit y = Basisinformation und xx = Detailinformation.

Rückgabewert	Bedeutung
000	Der String 'name' ist im vorliegenden System nicht bekannt ¹⁾
100	Der String 'name' ist ein Element der NC-Programmiersprache, aber aktuell nicht programmierbar (Option/Funktion ist inaktiv)
2xx	Der String 'name' ist ein programmierbares Element der NC-Programmiersprache (Option/Funktion ist aktiv). Die Detailinformation xx enthält weitere Informationen über die Art des Elements:
	xx Bedeutung
	01 DIN-Adresse oder NC-Adresse ²⁾
	02 G-Code (z.B. G04, INVCW)
	03 Funktion mit Rückgabewert
	04 Funktion ohne Rückgabewert
	05 Schlüsselwort (z.B. DEFINE)
	06 Maschinen- (\$M...), Setting- (\$S...) oder Optionsdatum (\$O...)
	07 Systemparameter, z.B. Systemvariable (\$...) oder Rechenparameter (R...)
	08 Zyklus (Der Zyklus muss im NCK geladen und die Zyklenprogramme aktiv sein ³⁾)
	09 GUD-Variable (Die GUD-Variable muss der in GUD-Definitionsdateien definierten und die GUD-Variablen aktiviert sein)
	10 Makroname (Das Makro muss in der Makro-Definitionsdateien definierten und Makros aktivierten sein) ⁴⁾
	11 LUD-Variable des aktuellen Teileprogramms
	12 ISO G-Code (ISO Sprachmodus muss aktiv ist)
400	Der String 'name' ist eine NC-Adresse, die nicht als xx == 01 oder xx == 10 erkannt wurde und die nicht G oder R ist ²⁾
y00	Keine spezifische Zuordnung möglich
<p>1) Steuerungs-abhängig ist unter Umständen nur eine Untermenge der Siemens NC-Sprachbefehle bekannt, z.B. SINUMERIK 802D sl. Auf diesen Steuerungen wird für Strings, die prinzipiell Siemens NC-Sprachbefehle sind, der Wert 0 zurückgegeben. Dieses Verhalten kann über MD10711 \$MN_NC_LANGUAGE_CONFIGURATION verändert werden. Bei MD10711 = 1 wird dann für Siemens NC-Sprachbefehle immer der Wert 100 zurückgegeben.</p> <p>2) NC-Adressen sind folgende Buchstaben: A, B, C, E, I, J, K, Q, U, V, W, X, Y, Z. Diese NC-Adressen können auch mit einer Adresserweiterung programmiert werden. Die Adresserweiterung kann bei der Prüfung mit STRINGIS angegeben werden. Beispiel: 201 == STRINGIS("A1"). Die Buchstaben: D, F, H, L, M, N, O, P, S, T sind NC-Adressen oder Hilfsfunktionen die anwenderdefiniert verwendet werden. Für sie wird immer der Wert 400 zurückgegeben. Beispiel: 400 == STRINGIS("D"). Diese NC-Adressen können bei der Prüfung mit STRINGIS nicht mit Adresserweiterung angegeben werden. Beispiel: 000 == STRINGIS("M02"), aber 400 == STRINGIS("M").</p> <p>3) Namen von Zyklenparametern können mit STRINGIS nicht geprüft werden.</p> <p>4) Als Makro definierte Adress z.B. G, H, M, L werden als Makro identifiziert</p>	

Beispiele

In den folgenden Beispielen wird angenommen, dass die als String angegebenen NC-Sprachelemente, sofern nicht besonders vermerkt, in der Steuerung prinzipiell programmierbar sind.

1. Der String "T" ist als Hilfsfunktion definiert:

```
400 == STRINGIS("T")  
000 == STRINGIS("T3")
```

2. Der String "X" ist als Achse definiert:

```
201 == STRINGIS("X")  
201 == STRINGIS("X1")
```

3. Der String "A2" ist als NC-Adresse mit Erweiterung definiert:

```
201 == STRINGIS("A")  
201 == STRINGIS("A2")
```

4. Der String "INVCW" ist als benannter G-Code definiert:

```
202 == STRINGIS("INVCW")
```

5. Der String "\$MC_GCODE_RESET_VALUES" ist als Maschinendatum definiert:

```
206 == STRINGIS("$MC_GCODE_RESET_VALUES")
```

6. Der String "GETMDACT" ist eine NC-Sprachfunktion:

```
203 == STRINGIS("GETMDACT ")
```

7. Der String "DEFINE" ist ein Schlüsselwort:

```
205 == STRINGIS("DEFINE")
```

8. Der String "\$TC_DP3" ist ein Systemparameter (Werkzeuglängenkomponente):

```
207 == STRINGIS("$TC_DP3")
```

9. Der String "\$TC_TP4" ist ein Systemparameter (Werkzeuggröße):

```
207 == STRINGIS("$TC_TP4")
```

10. Der String "\$TC_MPP4" ist ein Systemparameter (Magazinplazzustand):

- Die Werkzeugmagazin-Verwaltung ist aktiv: 207 == STRINGIS("\$TC_MPP4") ;
- Die Werkzeugmagazin-Verwaltung ist nicht aktiv: 000 == STRINGIS("\$TC_MPP4")

Siehe auch unten Absatz: Werkzeugmagazin-Verwaltung.

11. Der String "MACHINERY_NAME" ist als GUD-Variable definiert:

```
209 == STRINGIS("MACHINERY_NAME")
```

12. Der String "LONGMACRO" ist als Makro definiert:

```
210 == STRINGIS("LONGMACRO")
```

13. Der String "MYVAR" ist als LUD-Variable definiert:

```
211 == STRINGIS("MYVAR")
```

14. Der String "XYZ" ist kein im NCK bekannter Befehl, GUD-Variable, Makro- oder Zyklus-Name:

```
000 == STRINGIS("XYZ")
```

Werkzeugmagazin-Verwaltung

Ist die Funktion Werkzeugmagazin-Verwaltung nicht aktiv, liefert STRINGIS für die Systemparameter der Werkzeugmagazin-Verwaltung, unabhängig vom Maschinendatum

- MD10711 \$MN_NC_LANGUAGE_CONFIGURATION

immer den Wert 000.

ISO Modus

Ist die Funktion "ISO Modus" aktiv:

- MD18800 \$MN_MM_EXTERN_LANGUAGE (Aktivierung externer NC-Sprachen)
- MD10880 \$MN_MM_EXTERN_CNC_SYSTEM (zu adaptierendes Steuerungssystem)

überprüft STRINGIS den angegebenen String zuerst als SINUMERIK G-Code. Ist der String kein SINUMERIK G-Code wird er anschließend als ISO G-Code überprüft.

Programmierte Umschaltungen (G290 (SINUMERIK Mode), G291 (ISO Mode)) haben auf STRINGIS keine Auswirkung.

Beispiel

Die für die Funktion STRINGIS(...) relevanten Maschinendaten haben folgende Werte:

- MD10711 \$MN_NC_LANGUAGE_CONFIGURATION = 2 (Es werden nur die NC-Sprachbefehle als bekannt angesehen, deren Optionen gesetzt sind)
- MD19410 \$ON_TRAFO_TYPE_MASK = 'H0' (Option: Transformationen)
- MD10700 \$MN_PREPROCESSING_LEVEL='H43' (Vorverarbeitung für Zyklen aktiv)

Das folgende Beispielprogramm wird ohne Fehlermeldung abgearbeitet:

Programmcode	Kommentar
N1 R1=STRINGIS("TRACYL")	; R1 == 0, da TRACYL wegen der fehlenden ; Transformations-Option als "nicht bekannt" ; erkannt wird
N2 IF STRINGIS("TRACYL") == 204	;
N3 TRACYL(1,2,3)	; N3 wird übersprungen
N4 ELSE	
N5 G00	; und stattdessen N5 ausgeführt
N6 ENDIF	
N7 M30	

14.7 Fenster aus dem Teileprogramm interaktiv aufrufen (MMC)

Funktion

Über den Befehl `MMC` können aus dem Teileprogramm auf dem HMI anwenderdefinierte Dialogfenster (Dialogbilder) angezeigt werden.

Das Aussehen der Dialogfenster wird durch rein textuelle Projektierung festgelegt (COM-Datei im Zyklenverzeichnis), die HMI -System-Software bleibt dabei unverändert.

Anwenderdefinierte Dialogfenster können nicht zeitgleich in verschiedenen Kanälen aufgerufen werden.

Syntax

`MMC (<Kommando>, <Quittungsmodus>)`

Bedeutung

MMC:	Unterprogrammbezeichner
<Kommando>:	Parameter vom Typ STRING
	Enthält das MMC-Kommando, z. B. in folgender Form: "CYCLES, PICTURE_ON, T_SK.COM, BILD, MGUD.DEF, BILD_3.AWB, TEST_1, A1"
	CYCLES: Bedienbereich, in dem die projektierten Anwenderdialoge ausgeführt werden.
	PICTURE_ON bzw. PICTURE_OFF: Befehl: Bildanwahl bzw. Bildabwahl
	T_SK.COM: Com-Datei: Name der Dialogbild-Datei (Anwenderzyklen). Hier ist das Aussehen der Dialogbilder festgelegt. Im Dialogbild können Anwendervariablen und/oder Kommentartexte angezeigt werden.
	BILD: Dialogbildname: Die einzelnen Bilder werden durch den Dialogbildnamen ausgewählt.
	MGUD.DEF: Anwenderdatendefinitionsdatei, auf die beim Lesen/Schreiben von Variablen zugegriffen wird.
	BILD_3.AWB: Grafikdatei
	TEST_1: Anzeigezeit oder Quittungsvariable
	A1: Textvariablen..."

14.7 Fenster aus dem Teileprogramm interaktiv aufrufen (MMC)

<Quittungsmodus>:	Parameter vom Typ CHAR						
Wert:	<table border="0" style="width: 100%;"> <tr> <td style="padding-right: 20px;">"N":</td> <td>ohne Quittung Die Programmbearbeitung wird nach Absenden des Kommandos fortgeführt. Der Absender wird nicht benachrichtigt, wenn das Kommando nicht erfolgreich ausgeführt werden kann.</td> </tr> <tr> <td style="padding-right: 20px;">"S":</td> <td>synchrone Quittung Die Programmabarbeitung wird solange angehalten, bis die Empfängerkomponente das Kommando quittiert hat. Bei positiver Quittung wird der nächste Befehl abgearbeitet. Bei negativer Quittung wird eine Fehlermeldung ausgegeben.</td> </tr> <tr> <td style="padding-right: 20px;">"A":</td> <td>asynchrone Quittung Die Programmabarbeitung wird nach Absetzen des Befehls fortgeführt. Die Quittung wird in einer Quittungsvariablen (vordefinierte Systemvariable) abgelegt und ist explizit aus dem Programm abzufragen. Der auf den Quittungsmodus folgende Parameter ist die Nummer der Quittungsvariablen.</td> </tr> </table>	"N":	ohne Quittung Die Programmbearbeitung wird nach Absenden des Kommandos fortgeführt. Der Absender wird nicht benachrichtigt, wenn das Kommando nicht erfolgreich ausgeführt werden kann.	"S":	synchrone Quittung Die Programmabarbeitung wird solange angehalten, bis die Empfängerkomponente das Kommando quittiert hat. Bei positiver Quittung wird der nächste Befehl abgearbeitet. Bei negativer Quittung wird eine Fehlermeldung ausgegeben.	"A":	asynchrone Quittung Die Programmabarbeitung wird nach Absetzen des Befehls fortgeführt. Die Quittung wird in einer Quittungsvariablen (vordefinierte Systemvariable) abgelegt und ist explizit aus dem Programm abzufragen. Der auf den Quittungsmodus folgende Parameter ist die Nummer der Quittungsvariablen.
"N":	ohne Quittung Die Programmbearbeitung wird nach Absenden des Kommandos fortgeführt. Der Absender wird nicht benachrichtigt, wenn das Kommando nicht erfolgreich ausgeführt werden kann.						
"S":	synchrone Quittung Die Programmabarbeitung wird solange angehalten, bis die Empfängerkomponente das Kommando quittiert hat. Bei positiver Quittung wird der nächste Befehl abgearbeitet. Bei negativer Quittung wird eine Fehlermeldung ausgegeben.						
"A":	asynchrone Quittung Die Programmabarbeitung wird nach Absetzen des Befehls fortgeführt. Die Quittung wird in einer Quittungsvariablen (vordefinierte Systemvariable) abgelegt und ist explizit aus dem Programm abzufragen. Der auf den Quittungsmodus folgende Parameter ist die Nummer der Quittungsvariablen.						

Wird kein <Quittungsmodus> programmiert, wird als Standardeinstellung die synchrone Quittung verwendet.

14.8 Programmlaufzeit / Werkstückzähler

14.8.1 Programmlaufzeit / Werkstückzähler (Übersicht)

Zur Unterstützung des Werkzeugmaschinenbedieners werden Informationen zur Programmlaufzeit und Werkstückzahl bereitgestellt.

Diese Informationen können als Systemvariablen im NC- und/oder PLC-Programm bearbeitet werden. Gleichzeitig stehen diese Informationen für die Anzeige auf der Bedienoberfläche zur Verfügung.

14.8.2 Programmlaufzeit

Funktion

Die Funktion "Programmlaufzeit" stellt NC-interne Timer zur Überwachung technologischer Prozesse zur Verfügung, die über NC- und Kanal-spezifische Systemvariablen im Teileprogramm und in Synchronaktionen gelesen werden können.

Der Trigger zur Laufzeitmessung (\$AC_PROG_NET_TIME_TRIGGER) ist die einzige schreibbare Systemvariable der Funktion und dient zur selektiven Messung von Programmabschnitten. D. h. durch Beschreiben des Triggers im NC-Programm kann die Zeitmessung ein- und wieder ausgeschaltet werden.

Systemvariable	Bedeutung	Aktivität
NC-spezifisch		
\$AN_SETUP_TIME	Zeit seit dem letzten Steuerungshochlauf mit Standardwerten ("Kaltstart") in Minuten Wird bei jedem Steuerungshochlauf mit Standardwerten automatisch auf "0" zurückgesetzt.	<ul style="list-style-type: none"> immer aktiv
\$AN_POWERON_TIME	Zeit seit dem letzten Normalhochlauf der Steuerung ("Warmstart") in Minuten Wird bei jedem Normalhochlauf der Steuerung automatisch auf "0" zurückgesetzt.	
Kanal-spezifisch		
\$AC_OPERATING_TIME	Gesamtlaufzeit von NC-Programmen in der Betriebsart Automatik in Sekunden Der Wert wird mit jedem Steuerungshochlauf automatisch auf "0" zurückgesetzt.	<ul style="list-style-type: none"> Aktivierung über MD27860 nur Betriebsart AUTOMATIK
\$AC_CYCLE_TIME	Laufzeit des angewählten NC-Programms in Sekunden Der Wert wird mit dem Start eines neuen NC-Programms automatisch auf "0" zurückgesetzt.	

Systemvariable	Bedeutung	Aktivität	
\$AC_CUTTING_TIME	Bearbeitungszeit in Sekunden Gemessen wird die Laufzeit der Bahnachsen (mindestens eine ist aktiv) ohne aktiven Eilgang in allen NC-Programmen zwischen NC-Start und Programmende / NC-Reset. Die Messung wird zusätzlich bei aktiver Verweilzeit unterbrochen. Der Wert wird bei jedem Steuerungshochlauf mit Standardwerten automatisch auf "0" zurückgesetzt.		
\$AC_ACT_PROG_NET_TIME	Aktuelle Netto-Laufzeit des aktuellen NC-Programms in Sekunden Wird mit dem Start eines NC-Programms automatisch auf "0" zurückgesetzt.	<ul style="list-style-type: none"> • immer aktiv • nur Betriebsart AUTOMATIK 	
\$AC_OLD_PROG_NET_TIME	Netto-Laufzeit des gerade korrekt mit M30 beendeten Programms in Sekunden		
\$AC_OLD_PROG_NET_TIME_COUNT	Änderungen auf \$AC_OLD_PROG_NET_TIME Nach POWER ON steht \$AC_OLD_PROG_NET_TIME_COUNT auf "0". \$AC_OLD_PROG_NET_TIME_COUNT wird immer dann erhöht, wenn die Steuerung \$AC_OLD_PROG_NET_TIME neu geschrieben hat.		
\$AC_PROG_NET_TIME_TRIGGER	Trigger zur Laufzeitmessung:		<ul style="list-style-type: none"> • nur Betriebsart AUTOMATIK
	0	Neutraler Zustand Der Trigger ist nicht aktiv.	
	1	Beenden Beendet die Messung und kopiert den Wert aus \$AC_ACT_PROG_NET_TIME in \$AC_OLD_PROG_NET_TIME. \$AC_ACT_PROG_NET_TIME wird auf "0" gesetzt und läuft danach weiter.	
	2	Start Startet die Messung und setzt dabei \$AC_ACT_PROG_NET_TIME auf "0". \$AC_OLD_PROG_NET_TIME wird nicht verändert.	
	3	Stopp Stoppt die Messung. Verändert \$AC_OLD_PROG_NET_TIME nicht und hält \$AC_ACT_PROG_NET_TIME bis zum Fortsetzen konstant.	
4	Fortsetzen Fortsetzen der Messung, d. h. eine vorher gestoppte Messung wird wieder aufgenommen. \$AC_ACT_PROG_NET_TIME läuft weiter. \$AC_OLD_PROG_NET_TIME wird nicht verändert.		

Systemvariable	Bedeutung	Aktivität
Durch POWER ON werden alle Systemvariablen auf "0" zurückgesetzt!		
Literatur: Eine ausführliche Beschreibung der aufgelisteten Systemvariablen findet sich in: Funktionshandbuch Grundfunktionen; BAG, Kanal, Programmbetrieb, Reset-Verhalten (K1), Kapitel: Programmlaufzeit		

Hinweis

Maschinenhersteller

Das Einschalten der aktivierbaren Timer erfolgt über das Maschinendatum MD27860 \$MC_PROCESSTIMER_MODE.

Das Verhalten der aktiven Zeitmessungen bei bestimmten Funktionen (z. B. GOTOS, Override = 0%, aktiver Probelaufvorschub, Programmtest, ASUP, PROG_EVENT, ...) wird konfiguriert über die Maschinendaten MD27850 \$MC_PROG_NET_TIMER_MODE und MD27860 \$MC_PROCESSTIMER_MODE.

Literatur:

Funktionshandbuch Grundfunktionen; BAG, Kanal, Programmbetrieb, Reset-Verhalten (K1), Kapitel: Programmlaufzeit

Hinweis

Restzeit für ein Werkstück

Wenn nacheinander gleiche Werkstücke produziert werden, kann aus den Timerwerten:

- Bearbeitungszeit für das zuletzt produzierte Werkstück (siehe \$AC_OLD_PROG_NET_TIME)

und

- aktuelle Bearbeitungszeit (siehe \$AC_ACT_PROG_NET_TIME)

die verbleibende Restzeit für ein Werkstück ermittelt werden.

Die Restzeit wird zusätzlich zur aktuellen Bearbeitungszeit auf der Bedienoberfläche angezeigt.

ACHTUNG

Verwendung von STOPRE

Die Systemvariablen \$AC_OLD_PROG_NET_TIME und \$AC_OLD_PROG_NET_TIME_COUNT erzeugen keinen impliziten Vorlaufstopp. Bei der Verwendung im Teileprogramm ist das unkritisch, wenn der Wert der Systemvariablen aus dem vorangegangenen Programmlauf stammt. Wenn aber der Trigger zur Laufzeitmessung (\$AC_PROG_NET_TIME_TRIGGER) hochfrequent geschrieben wird und sich dadurch \$AC_OLD_PROG_NET_TIME sehr oft ändert, dann sollte im Teileprogramm ein explizites STOPRE verwendet werden.

Randbedingungen

- **Satzsuchlauf**
Bei Satzsuchlauf werden keine Programmlaufzeiten ermittelt.
- **REPOS**
Die Zeitdauer eines REPOS-Vorgangs wird der aktuellen Bearbeitungszeit (\$AC_ACT_PROG_NET_TIME) angerechnet.

Beispiele

Beispiel 1: Zeitdauer von "mySubProgrammA" messen

Programmcode

```
...
N50 DO $AC_PROG_NET_TIME_TRIGGER=2
N60 FOR ii= 0 TO 300
N70 mySubProgrammA
N80 DO $AC_PROG_NET_TIME_TRIGGER=1
N95 ENDFOR
N97 mySubProgrammB
N98 M30
```

Nachdem das Programm die Zeile `N80` verarbeitet hat, steht in `$AC_OLD_PROG_NET_TIME` die Nettolaufzeit von "mySubProgrammA".

Der Wert von `$AC_OLD_PROG_NET_TIME`:

- bleibt über `M30` hinaus erhalten.
- wird nach jedem Schleifendurchlauf aktualisiert.

Beispiel 2: Zeitdauer von "mySubProgrammA" und "mySubProgrammC" messen

Programmcode

```
...
N10 DO $AC_PROG_NET_TIME_TRIGGER=2
N20 mySubProgrammA
N30 DO $AC_PROG_NET_TIME_TRIGGER=3
N40 mySubProgrammB
N50 DO $AC_PROG_NET_TIME_TRIGGER=4
N60 mySubProgrammC
N70 DO $AC_PROG_NET_TIME_TRIGGER=1
N80 mySubProgrammD
N90 M30
```

14.8.3 Werkstückzähler

Funktion

Die Funktion "Werkstückzähler" stellt diverse Zähler zur Verfügung, die insbesondere für die steuerungsinterne Zählung von Werkstücken verwendet werden können. Die Zähler existieren als kanalspezifische Systemvariablen mit Schreib- und Lese-Zugriff im Wertebereich von 0 bis 999 999 999.

Systemvariable	Bedeutung
\$AC_REQUIRED_PARTS	Anzahl der zu fertigenden Werkstücke (Soll-Werkstückzahl) In diesem Zähler kann die Anzahl der Werkstücke definiert werden, bei dessen Erreichen die Ist-Werkstückzahl (\$AC_ACTUAL_PARTS) auf "0" zurückgesetzt wird.
\$AC_TOTAL_PARTS	Anzahl der insgesamt gefertigten Werkstücke (Ist-Werkstückzahl Gesamt) Dieser Zähler gibt die Anzahl aller ab Startzeitpunkt gefertigten Werkstücke an. Der Wert wird nur bei einem Steuerungshochlauf mit Standardwerten automatisch auf "0" zurückgesetzt.
\$AC_ACTUAL_PARTS	Anzahl der gefertigten Werkstücke (Ist-Werkstückzahl) In diesem Zähler wird die Anzahl aller ab Startzeitpunkt gefertigten Werkstücke registriert. Bei einem Erreichen der Soll-Werkstückzahl (\$AC_REQUIRED_PARTS) wird der Zähler automatisch auf "0" zurückgesetzt (\$AC_REQUIRED_PARTS > 0 vorausgesetzt).
\$AC_SPECIAL_PARTS	Anzahl der vom Anwender gezählten Werkstücke Dieser Zähler erlaubt dem Anwender eine Werkstückzählung nach eigener Definition. Definiert werden kann eine Alarmausgabe bei Erreichen der Soll-Werkstückzahl (\$AC_REQUIRED_PARTS). Eine Nullung des Zählers muss der Anwender selbst vornehmen.

Hinweis

Alle Werkstückzähler werden bei einem Steuerungshochlauf mit Standardwerten auf "0" gesetzt und können unabhängig von ihrer Aktivierung gelesen und geschrieben werden.

Hinweis

Über kanalspezifische Maschinendaten kann auf die Zähler-Aktivierung, den Zeitpunkt der Nullung und den Zählalgorithmus Einfluss genommen werden.

Hinweis

Werkstückzählung mit anwenderdefiniertem M-Befehl

Über Maschinendaten kann eingestellt werden, dass die Zählimpulse für die verschiedenen Werkstückzähler statt über das Programmende M2/M30 über anwenderdefinierte M-Befehle ausgelöst werden.

Literatur

Weitere Informationen zur Funktion "Werkstückzähler" siehe:
Funktionshandbuch Grundfunktionen; BAG, Kanal, Programmbetrieb, Reset-Verhalten (K1),
Kapitel: Werkstückzähler

14.9 Process DataShare - Ausgabe auf ein externes Gerät/Datei (EXTOPEN, WRITE, EXTCLOSE)

Funktion

Mit der Funktion "Process DataShare" ist es möglich, Daten aus einem Teileprogramm heraus auf ein externes Gerät / eine externe Datei zu schreiben, z. B. zum Zweck der Protokollierung von Produktionsdaten oder zur Steuerung von Zusatzaggregaten an einer Steuerung.

Die Ausgabe auf ein externes Gerät/Datei erfolgt in drei Schritten:

1. Externes Gerät/Datei öffnen

Mit dem Befehl `EXTOPEN` wird das externe Gerät/Datei für den Kanal zum Schreiben geöffnet.

2. Daten schreiben

Das Ausgabedatum kann mit den Stringfunktionen der NC-Sprache ("Stringoperationen (Seite 83)"), z. B. `SPRINT`, aufbereitet werden. Das Schreiben selbst erfolgt über den `WRITE`-Befehl.

3. Externes Gerät/Datei schließen

Mit dem Befehl `EXTCLOSE` oder durch Erreichen des Programmendes (`M30`) sowie bei Kanal-Reset wird das im Kanal belegte externe Gerät/Datei wieder freigegeben.

Hinweis

In einem Teileprogramm/Kanal kann auch mehr als ein externes Gerät/Datei belegt werden.

Verfügbarkeit

Die Funktion ist verfügbar:

- nur in Teileprogrammen (**nicht** in Synchronaktionen).
- parallel in allen Bearbeitungskanälen des NCK für alle verfügbaren (projektierten) Ausgabegeräte.

Pro Ausgabegerät ist beim Öffnen des Geräts vorgebar, ob das Gerät exklusiv nur von einem Kanal aus genutzt werden darf oder ob es geteilt unter den Kanälen, die darauf ausgeben wollen, verwendet werden kann ("Shared"-Modus).

Syntax

```
DEF INT <Fehler>
DEF STRING [<n>] <Ausgabe>
...
EXTOPEN (<Fehler>, "<ExtG>", <Bearbeitungsmodus>, <Benutzungsmodus>, <Schreibmodus>)
...
<Ausgabe>="Ausgabe Daten"
WRITE (<Fehler>, "<ExtG>", <Ausgabe>)
...
EXTCLOSE (<Fehler>, "<ExtG>")
```

Bedeutung

EXTOPEN: **Befehl zum Öffnen eines externen Geräts/Datei**

<Fehler>:

Parameter 1: Variable für die Rückgabe des Fehlerwerts

Anhand des Fehlerwerts kann im Programm das Gelingen der Operation ausgewertet werden und entsprechend fortgefahren werden.

Typ: INT

Werte: 0 kein Fehler

- 1 externes Gerät kann nicht geöffnet werden
- 2 externes Gerät ist nicht projiziert
- 3 externes Gerät mit ungültigem Pfad projiziert
- 4 keine Zugriffsrechte für externes Gerät
- 5 Benutzungsmodus: externes Gerät bereits "exklusiv" belegt
- 6 Benutzungsmodus: externes Gerät bereits "shared" belegt
- 7 Dateilänge größer als LOCAL_DRIVE_MAX_FILESIZE
- 8 maximale Anzahl externer Geräte überschritten
- 9 Option für LOCAL_DRIVE nicht gesetzt
- 11 V.24-Schnittstelle bereits durch Easy-Message-Funktion belegt (nur 828D)
- 12 Schreibmodus: Angabe widersprüchlich zu extdev.ini
- 16 ungültiger externer Pfad programmiert
- 22 externes Gerät nicht gemountet

<ExtG>:

Parameter 2: Symbolischer Bezeichner für das zu öffnende externe Gerät/Datei

Typ: STRING

Der symbolische Bezeichner besteht aus:

1. dem logischen Gerätenamen
2. ggf. gefolgt von einem Dateipfad (angehängt mit "/").

Folgende **logische Gerätenamen** sind definiert:

"LOCAL_DRIVE":	Lokale CompactFlash Card (vordefiniert)
"CYC_DRIVE":	reservierte Laufwerksangabe zur Verwendung in SIEMENS-Zyklen (vordefiniert)
"/dev/ext/1",...	Verfügbare Netzlaufwerke
"/dev/ext/9":	Hinweis: Projektierung in der Datei extdev.ini erforderlich!
"/dev/cyc/1",	reservierte Laufwerksangaben zur Verwendung in SIEMENS-Zyklen
"/dev/cyc/2":	Hinweis: Projektierung in der Datei extdev.ini erforderlich!
"/dev/v24":	V.24-Schnittstelle Hinweis: Projektierung in der Datei extdev.ini erforderlich!

Dateipfad:

- Zu "LOCAL_DRIVE" und "CYC_DRIVE" muss ein Dateipfad angegeben werden, z. B.:
"LOCAL_DRIVE/my_dir/my_file.txt"
- Die logischen Gerätenamen "/dev/ext/1...9" und "/dev/cyc/1...2" können per Projektierung:
 - schon auf eine Datei verweisen, dann darf nur der logische Gerätenamen angegeben werden, z. B.:
"/dev/ext/4"
 - oder auf ein Verzeichnis, dann muss ein Dateipfad angegeben werden, z. B.:
"/dev/ext/5/my_dir/my_file.txt"
- Zu "/dev/v24" darf kein Dateipfad angehängt sein.

	<p>Hinweis: Für die logischen Gerätenamen <code>"/dev/ext/1...9"</code>, <code>"/dev/v24"</code> und <code>"/dev/cyc/1...2"</code> wird Groß-/Kleinschreibung ignoriert, bei der Pfadangabe zu einer Datei ist Groß-/Kleinschreibung signifikant. Für <code>"LOCAL_DRIVE"</code> und <code>"CYC_DRIVE"</code> ist nur Großschreibung zulässig.</p>
<code><Bearbeitungsmodus></code> :	<p>Parameter 3: Bearbeitungsmodus für die <code>WRITE</code>-Befehle zu diesem Gerät/Datei</p> <p>Typ: <code>STRING</code></p> <p>Werte: <code>"SYN"</code>: Synchrones Schreiben Die Programmausführung wird angehalten, bis der Schreibvorgang abgeschlossen ist. Die erfolgreiche Beendigung des synchronen Schreibens kann durch Auswerten der Fehlervariablen des <code>WRITE</code>-Befehls überprüft werden.</p> <p><code>"ASYN"</code>: Asynchrones Schreiben Die Programmausführung wird durch den <code>WRITE</code>-Befehl nicht unterbrochen.</p> <p>Hinweis: Die Fehlervariable des <code>WRITE</code>-Befehls ist in diesem Modus nicht aussagekräftig und hat immer den Wert 0 (kein Fehler). Es gibt in diesem Modus keine Sicherheit, dass der <code>WRITE</code>-Befehl erfolgreich war.</p>
<code><Benutzungsmodus></code> :	<p>Parameter 4: Benutzungsmodus für dieses Gerät/Datei</p> <p>Typ: <code>STRING</code></p> <p>Werte: <code>"SHARED"</code>: Gerät/Datei wird im "Shared"-Modus angefordert. Andere Kanäle können das Gerät mitverwenden, d. h. ebenfalls in diesem Modus öffnen.</p> <p><code>"EXCL"</code>: Gerät/Datei wird in dem Kanal exklusiv verwendet, kein anderer Kanal kann das Gerät mitverwenden.</p>

14.9 Process DataShare - Ausgabe auf ein externes Gerät/Datei (EXTOPEN, WRITE, EXTCLOSE)

<Schreibmodus>:

Parameter 5: Schreibmodus für die WRITE-Befehle zu diesem Gerät/Datei (optional)

Typ: STRING

Werte: "APP": Anhängen
Die Datei bleibt in ihrem Inhalt erhalten, Schreibaufrufe fügen an das Ende an.

"OVR": Überschreiben
Der Inhalt der Datei wird gelöscht und durch nachfolgende Schreibaufrufe neu erstellt.

Hinweis:

Mit diesem Parameter kann der in der Datei extdev.ini projektierte Schreibmodus **nicht** überschrieben werden. Im Konfliktfall wird der EXTOPEN-Aufruf mit Fehler quittiert.

WRITE: Befehl zum Schreiben der Ausgabedaten

Beschreibung siehe "Datei schreiben (WRITE) (Seite 151)!"

EXTCLOSE: Befehl zum Schließen eines geöffneten externen Geräts/Datei

<Fehler>: **Parameter 1:** Variable für die Rückgabe des Fehlerwerts

Typ: INT

Werte: 0 kein Fehler
16 ungültiger externer Pfad programmiert
21 Fehler beim Schließen des externen Geräts

<ExtG>: **Parameter 2:** Symbolischer Bezeichner für das zu schließende externe Gerät/Datei

Beschreibung siehe unter EXTOPEN!

Hinweis:

Der Bezeichner muss identisch zu dem im EXTOPEN-Befehl angegebenen Bezeichner sein!

Beispiel

```
Programmcode
-----
N10  DEF INT RESULT
N20  DEF BOOL EXTDEVICE
N30  DEF STRING[80] AUSGABE
N40  DEF INT PHASE
N50  EXTOPEN(RESULT, "LOCAL_DRIVE/my_file.txt", "SYN", "SHARED")
N60  IF RESULT > 0
N70      MSG("Fehler bei EXTOPEN:" << RESULT)
N80  ELSE
N90      EXTDEVICE=TRUE
N100 ENDIF
...
N200 PHASE=4
N210 IF EXTDEVICE
N220     AUSGABE=SPRINT("Ende Phase: %D", PHASE)
N230     WRITE(RESULT, "LOCAL_DRIVE/my_file.txt", AUSGABE)
N240 ENDIF
...
```

Weitere Informationen

Einfluss auf Bahnsteuerbetrieb

Die Befehle `EXTOPEN`, `WRITE` und `EXTCLOSE` lösen jeweils Vorlaufstopp aus und unterbrechen damit den Bahnsteuerbetrieb.

Verhalten bei Satzsuchlauf

Während "Satzsuchlauf mit Berechnung" erfolgt mit `WRITE` keine Ausgabe. Es werden jedoch die `EXTOPEN`- und `EXTCLOSE`-Befehle aufgesammelt und -- nachdem das Suchziel erreicht wurde -- mit NC-Start wirksam gesetzt. Nachfolgende `WRITE`-Befehle finden damit dieselbe Umgebung wie bei der normalen Programmbearbeitung vor.

Bei Suchlauf mit Berechnung im Modus "Programmtest" (SERUPRO) werden `EXTOPEN`, `WRITE` und `EXTCLOSE` wie bei der normalen Programmbearbeitung ausgeführt.

Reset-Verhalten

Mit Teileprogramm-Ende und Kanal-Reset werden alle in diesem Kanal geöffneten externen Geräte/Dateien geschlossen.

Verfügbare externe Geräte

Als externe Geräte/Dateien können zur Verfügung stehen:

- Dateien auf der lokalen CompactFlash Card

Mit lokaler CompactFlash Card ist der Speicher gemeint, auf den mit dem symbolischen Bezeichner LOCAL_DRIVE aus HMI heraus verwiesen wird. Bei SINUMERIK 840D sl ist dies das lokale Laufwerk, bei SINUMERIK 828D die Anwender CompactFlash Card.

Hinweis

Für die Ausgabe auf das Gerät LOCAL_DRIVE ist bei SINUMERIK 840D sl die Option "Zusätzlich xxx MB HMI-Anwenderspeicher auf CF-Card der NCU" erforderlich. Für SINUMERIK 828D muss eine Anwender CompactFlash Card vorhanden sein, eine Option ist hier nicht erforderlich.

- Dateien auf einem Netzlaufwerk
- V.24-Schnittstelle

Hinweis

Für die Ausgabe auf die V.24-Schnittstelle ist bei SINUMERIK 840D sl das NCU-Optionsmodul RS232-Schnittstelle erforderlich. Bei SINUMERIK 828D erfolgt die Ausgabe auf die integrierte V.24-Schnittstelle (Voraussetzung: MD51233 \$MNS_ENABLE_GSM_MODEM = 0).

Projektierung

Die Projektierung der zu verwendenden externen Geräte erfolgt in der Datei /oem/sinumerik/nck/extdev.ini bzw. /user/sinumerik/nck/extdev.ini. Sind beide Dateien vorhanden, haben die Einträge im User-Bereich Vorrang. Die Datei kann im Bedienbereich INBETRIEBNAHME unter SYSTEMDATEN/CF-Karte gepflegt werden.

Hinweis

Für die Verwendung von LOCAL_DRIVE und CYC_DRIVE ist keine Projektierung in der Datei extdev.ini erforderlich. Die beiden Geräte sind immer verfügbar, sobald die entsprechende Option gesetzt bzw. die Anwender CompactFlash Card vorhanden ist.

Im Abschnitt [ExternalDevices] der Datei extdev.ini werden die zu verwendenden externen Geräte definiert/aufgezählt. Als Geräte können ein serielles Device (/dev/v24) und bis zu neun Dateien oder Verzeichnisse (/dev/ext/1...9) angegeben werden. Die Notation für die Angabe ist Linux-Schreibweise. Zeilen, die mit ";" beginnen, sind Kommentar und werden überlesen.

Mit Ausnahme von /dev/v24 können die Geräte als Verzeichnispfad – abgeschlossen mit angehängtem "/" – oder als Dateipfad – also mit angehängtem voll qualifizierten Pfad, endend mit einem Dateinamen (ohne endendes "/") – vereinbart werden. Zu einem Gerät mit Verzeichnispfad muss bei Verwendung im Teileprogramm ein Dateiname (Pfad) mit angegeben werden.

Außer für /dev/v24 erfolgt die Definition eines Geräts durch die drei mit Komma getrennten Angaben für "Server", "Pfad" und dem optionalen "Schreibmodus".

Zu Dateien bzw. Verzeichnissen (wirkt dann auf alle Dateien in dem Verzeichnis) kann angegeben werden, ob die Datei nach dem Öffnen überschrieben werden soll ("O" = Overwrite) oder ob die Ausgaben an die Datei angehängt werden sollen ("A" = Append). Standardwert ist "A". Eine nichtexistierende Datei/Verzeichnis wird beim Öffnen neu angelegt.

Für das Gerät V.24-Schnittstelle werden nur die Einstellungen für Baudrate, Daten-Bits, Stopp-Bits, Parität, Protokoll und ggf. Ende in dieser Reihenfolge angegeben.

Für die Dateien, die auf LOCAL_DRIVE erzeugt/abgelegt werden, wird über das Datum LOCAL_DRIVE_MAX_FILESIZE eine maximale Dateigröße in Bytes – gültig einheitlich für alle Dateien – eingestellt. Die Dateigröße wird bei Ausführung eines EXTOPEN-Befehls im Append-Modus überprüft. Optional kann mit dem Datum LOCAL_DRIVE_FILE_MODE der Schreibmodus ("O" = Overwrite, "A" = Append) festgelegt werden. Standardwert ist "A".

Hinweis

Eine Kopiervorlage für die Projektierungsdatei extdev.ini steht im Verzeichnis /siemens/sinumerik/nck zur Verfügung.

Hinweis

Änderungen an der Datei extdev.ini werden erst nach Neustart/Hochlauf des NCK wirksam.

Hinweis

USB-Geräte

Für SINUMERIK 828D kann als Ziel eines der Geräte auch "usb" (ohne Partitionsangabe!) für ein frontseitiges USB-Gerät definiert werden. Das Gerät an USB kann aus dem Teileprogramm heraus nur mittelbar über einen symbolischen Gerätebezeichner "/dev/ext/x" angesprochen werden.

Bei SINUMERIK 840D sl können als USB-Geräte nur statisch verbundene USB-Schnittstellen einer TCU projektiert werden. Die Projektierung erfolgt über die Art SERVER:/PATH als Spezifikation für "Server" im obigen Sinne, wobei SERVER der TCU-Name ist, und /PATH die USB-Schnittstelle bezeichnet. Die jeweiligen USB-Schnittstellen einer TCU werden mit "dev0-0", "dev0-1", "dev1-0" angesprochen. Die Pfadangabe beginnt immer mit "/Partition", wobei die Partition durch ihre zweistellige Partitionsnummer oder ihren Partitionsnamen angegeben sein kann und ggf. mit einem Dateipfad bis zum gewünschten Ziel verlängert wird, also z. B.:

```
/dev/ext/8 = "TCU4:/dev0-0, /01/, A"
```

```
/dev/ext/8 = "TCU4:/dev0-0, /01/mydir.dir/"
```

```
/dev/ext/8 = "TCU4:/dev0-0, /myfirstpartition/Mydir.dir/myfile.txt, O"
```

14.9 Process DataShare - Ausgabe auf ein externes Gerät/Datei (EXTOPEN, WRITE, EXTCLOSE)

Beispiele:

```
[ExternalDevices]
; Kommentarzeile
; example for V24
; /dev/v24 = "9600, 8, 1, none, rts [, etx]"
; examples for network drives
; /dev/ext/1 = "[USERNAME[/DOMAIN][%PASSWORD]@]SERVER/SHARE/, /, A"
; /dev/ext/2 = "[USERNAME[/DOMAIN][%PASSWORD]@]SERVER/SHARE, /myfile.txt, O"
; /dev/ext/3 = "[USERNAME[/DOMAIN][%PASSWORD]@]SERVER/SHARE, /mydir/, A"
; /dev/ext/4 = "SERVER:/dev0-0, /01/, A"
; ...
; SINUMERIK 828 only (USB)
; /dev/ext/9 = "usb, / [, O]"
; default: Partitionsnummer = 1

; SIEMENS only
; /dev/cyc/1= "[USERNAME[/DOMAIN][%PASSWORD]@]SERVER/SHARE, /mydir/, A"
; /dev/cyc/2= "[USERNAME[/DOMAIN][%PASSWORD]@]SERVER/SHARE/mydir, /, A"

LOCAL_DRIVE_MAX_FILESIZE = 50000
LOCAL_DRIVE_FILE_MODE = "O"
```

Wirksamkeit des EXTOPEN-Parameters <Schreibmodus>

Durch Angabe des Schreibmodus sowohl bei der Projektierung in der Datei extdev.ini als auch beim EXTOPEN-Aufruf kann es zu Rechtkonflikten kommen, die beim EXTOPEN-Aufruf ggf. mit Fehler quittiert werden:

Wert aus extdev.ini	Wert des EXTOPEN-Parameters		
	"OVR"	"APP"	-
"O"	O	Fehler	O
"A"	Fehler	A	A
-	O	A	A
	Erläuterung: O: Modus "Überschreiben" ist wirksam. A: Modus "Anhängen" ist wirksam. Fehler: EXTOPEN-Aufruf wird mit Fehler quittiert.		

LOCAL_DRIVE: Dateiattribute

Die mit `EXTOPEN` auf `LOCAL_DRIVE` angelegten Dateien erhalten folgende Dateiattribute:

- Eigentümer: "user" Schreib-/Leserecht gesetzt
- Gruppe: "operator" Schreib-/Leserecht gesetzt

Maximale Anzahl geöffneter externer Geräte

Über alle NC-Kanäle hinweg können gleichzeitig maximal 10 Ausgabegeräte geöffnet sein. Zusätzlich gibt es noch zwei reservierte Einträge für Siemens-Zyklen.

Zu diesen Geräten können gleichzeitig maximal 5 Aufträge aktiv sein.

14.10 Alarme (SETAL)

Funktion

In einem NC-Programm können Alarme gesetzt werden. Diese werden in der Bedienoberfläche in einem besonderen Feld dargestellt. Mit einem Alarm ist jeweils eine Reaktion der Steuerung entsprechend der Alarmkategorie verbunden.

Literatur:

Weiterführende Informationen zu den Alarmreaktionen siehe Inbetriebnahmehandbuch.

Syntax

```
SETAL (<Alarmnummer> [, <Zeichenkette>])
```

Bedeutung

SETAL:	Schlüsselwort zur Programmierung eines Alarms. SETAL muss in einem eigenen NC-Satz programmiert werden.										
<Alarmnummer>:	Variable vom Typ INT. Enthält die Alarmnummer. Der gültige Bereich für Alarmnummern liegt zwischen 60000 und 69999, wovon 60000 bis 64999 für SIEMENS-Zyklen reserviert sind und 65000 bis 69999 für den Anwender zur Verfügung stehen.										
<Zeichenkette>:	Bei der Programmierung von Anwenderzyklenalarmen kann zusätzlich eine Zeichenkette mit bis zu 4 Parametern angegeben werden. In diesen Parametern können variable Anwendertexte definiert werden. Es stehen aber auch folgende vordefinierte Parameter zur Verfügung:										
	<table><thead><tr><th>Parameter</th><th>Bedeutung</th></tr></thead><tbody><tr><td>%1</td><td>Kanalnummer</td></tr><tr><td>%2</td><td>Satznummer, Label</td></tr><tr><td>%3</td><td>Textindex für Zyklenalarme</td></tr><tr><td>%4</td><td>zusätzlicher Alarmparameter</td></tr></tbody></table>	Parameter	Bedeutung	%1	Kanalnummer	%2	Satznummer, Label	%3	Textindex für Zyklenalarme	%4	zusätzlicher Alarmparameter
Parameter	Bedeutung										
%1	Kanalnummer										
%2	Satznummer, Label										
%3	Textindex für Zyklenalarme										
%4	zusätzlicher Alarmparameter										

Hinweis

Alarmtexte müssen in der Bedienoberfläche projiziert werden.

Hinweis

Soll eine Alarmausgabe in der auf der Bedienoberfläche aktiven Sprache erfolgen, benötigt der Anwender Informationen über die aktuell auf dem HMI eingestellte Sprache. Diese Information kann im Teileprogramm und in Synchronaktionen über die Systemvariable \$AN_LANGUAGE_ON_HMI abgefragt werden (siehe "Aktuelle Sprache im HMI (Seite 827)").

Beispiel

Programmcode	Kommentar
...	
N100 SETAL(65000)	; Alarm Nr. 65000 setzen
...	

14.11 Erweitertes Stillsetzen und Rückziehen (ESR)

Funktion

Die Funktion "Erweitertes Stillsetzen und Rückziehen", im weiteren Verlauf mit ESR bezeichnet, bietet die Möglichkeit, in Fehlersituationen prozessabhängig flexibel zu reagieren:

- **Erweitertes Stillsetzen**

Soweit es die spezifische Fehlersituation erlaubt, werden alle für das Erweiterte Stillsetzen freigegebenen Achsen geordnet stillgesetzt.

- **Rückziehen**

Das sich im Eingriff befindliche Werkzeug wird schnellstmöglich vom Werkstück zurückgezogen.

- **Generatorbetrieb (SINAMICS-Antriebsfunktion "Vdc-Regelung")**

Bei Unterschreitung eines parametrierbaren Werts der Zwischenkreisspannung, z. B. bei Ausfall der Netzspannung, wird die für das Rückziehen benötigte elektrische Energie durch Rückspeisen der Bremsenergie eines dafür vorgesehenen Antriebs (Generatorbetrieb) erzeugt.

Auslösequellen

Allgemeine Quellen (NC-extern/global oder BAG-/kanalspezifisch)

- Digitale Eingänge (z. B. auf NCU-Baugruppe) bzw. das steuerungsinterne, rücklesbare Abbild digitaler Ausgänge (\$A_IN, \$A_OUT)
- Kanalzustand (\$AC_STAT)
- VDI-Signale (\$A_DBB)
- Sammelmeldungen einer Anzahl von Alarmen (\$AC_ALARM_STAT)

Axiale Quellen

- Notrückzugsschwelle der Folgeachse (Synchronlauf der elektronischen Kopplung, \$VC_EG_SYNCDIFF[<Folgeachse>])
- Antrieb: Zwischenkreis-Warnschwelle (drohende Unterspannung), \$AA_ESR_STAT[<Achse>]
- Antrieb: Generator-Minimaldrehzahl-Schwelle (keine rückspeisbare Rotationsenergie mehr vorhanden), \$AA_ESR_STAT[<Achse>].

Verknüpfungslogik der statischen Synchronaktionen: Quellen-/Reaktions-Verknüpfung

Die flexiblen Verknüpfungsmöglichkeiten der statischen Synchronaktionen werden genutzt, um aufgrund von Quellen relativ zeitnah bestimmte Reaktionen auszulösen.

Die Verknüpfung aller relevanten Quellen mit Hilfe statischer Synchronaktionen liegt in den Händen des Anwenders. Er kann die Quellen-Systemvariablen als Ganzes oder mit Hilfe von Bit-Masken auch selektiv auswerten und hieran seine gewünschten Reaktionen knüpfen. Die statischen Synchronaktionen sind in allen Betriebsarten wirksam.

Literatur:

Funktionshandbuch Synchronaktionen

Aktivierung

Funktionsfreigabe

Die Funktionen Generatorbetrieb, Stillsetzen, Rückziehen werden durch Setzen des zugehörigen Steuersignals \$AA_ESR_ENABLE freigegeben. Dieses Steuersignal kann von Synchronaktionen verändert werden.

Funktionsauslösung

ESR wird gemeinsam für alle freigegebenen Achsen ausgelöst durch Setzen der Systemvariablen \$AC_ESR_TRIGGER.

Der Generatorbetrieb wird "automatisch" im Antrieb bei Erkennung drohender Zwischenkreis-Unterspannung aktiv.

Antriebsautarkes Stillsetzen und/oder Rückziehen werden aktiv bei Erkennung eines Kommunikationsausfalls (zwischen NC und Antrieb) sowie bei Erkennung einer Zwischenkreis-Unterspannung im Antrieb (Konfiguration und Freigabe vorausgesetzt).

Antriebsautarkes Stillsetzen und/oder Rückziehen können zusätzlich auch von der NC-Seite her ausgelöst werden durch das Setzen des entsprechenden Steuersignals \$AN_ESR_TRIGGER (Broadcast-Kommando an alle Antriebe).

Literatur

Ausführliche Informationen zu ESR siehe:

Funktionshandbuch Sonderfunktionen; Erweitertes Stillsetzen und Rückziehen (R3)

14.11.1 NC-geführtes ESR

14.11.1.1 NC-geführtes Rückziehen (POLF, POLFA, POLFMASK, POLFMLIN)

Funktion

Für NC-geführtes Rückziehen sind bestimmte Ausgangsbedingungen erforderlich (siehe "NC-geführtes Rückziehen (POLF, POLFA, POLFMASK, POLFMLIN) (Seite 637)"). Sind diese Voraussetzungen erfüllt, dann wird für die im Kanal konfigurierte(n) Rückzugsachse(n) durch Setzen der Systemvariablen \$AC_ESR_TRIGGER (bzw. \$AA_ESR_TRIGGER für Einzelachsen) Schnellabheben (LIFTFAST) aktiviert.

Syntax

```
POLF(<Achse>)=<Position>
POLFA(<Achse>,<Typ>,<Position>)
POLFMASK(<Achse_1>,<Achse_2>,...)
POLFMLIN(<Achse_1>,<Achse_2>,...)
```

Für POLFA sind folgende Kurzformen erlaubt:

```
POLFA(<Achse>,<Typ>) ; Kurzform für Einzelachs-Rückziehen
POLFA(Achse,0/1/2) ; Schnelle Deaktivierung oder Aktivierung
POLFA(Achse,0,$AA_POLFA[Achse]) ; Bewirkt einen Vorlaufstopp
POLFA(Achse,0) ; Bewirkt keinen Vorlaufstopp
```

Bedeutung

POLF:	Adresse zur Angabe der Zielposition der Rückzugsachse POLF ist modal wirksam.
<Achse>:	Name der Geometrie- oder Kanal-/Maschinenachse, die zurückzieht
<Position>:	Rückzugsposition Typ: REAL Für Geometrieachse gilt WKS, sonst MKS. Bei gleichen Bezeichnern für Geometrie- und Kanal-/Maschinenachse wird im WKS zurückgezogen.
POLFA:	Vordefinierter Unterprogrammaufruf zur Angabe der Rückzugsposition von Einzelachsen
<Achse>:	Kanalachsbezeichner
<Typ>:	Positionsangabemodus Typ: INT Wert: 0: Positionswert als ungültig markieren 1: Positionswert ist absolut 2: Positionswert ist inkrementell (Distanz)
	Hinweis: Ist eine Achse keine Einzelachse oder fehlt der Typ bzw. Typ=0, dann wird ein entsprechender Alarm gemeldet.

	<Position>:	Rückzugposition (s. o.)
		Hinweis: Der Positionswert wird auch mit Typ=0 übernommen. Nur ist dieser Wert dann als ungültig markiert und muss für das Rückziehen neu programmiert werden.
POLFMASK:		Vordefinierter Unterprogrammaufruf zur Auswahl der Achsen, die nach Auslösung des Schnellabhebens unabhängig voneinander zurückgezogen werden sollen.
	<Achse_1>, ...:	Namen der Achsen, die beim Schnellabheben auf ihre mit POLF definierten Positionen fahren sollen. Alle angegebenen Achsen müssen sich im gleichen Koordinatensystem befinden.
		POLFMASK() ohne Angabe einer Achse deaktiviert das Schnellabheben für alle Achsen, die unabhängig voneinander zurückgezogen wurden.
POLFMLIN:		Vordefinierter Unterprogrammaufruf zur Auswahl der Achsen, die nach Auslösung des Schnellabhebens im linearen Zusammenhang zurückgezogen werden sollen.
	<Achse_1>, ...:	S. O.
		POLFMLIN() ohne Angabe einer Achse deaktiviert das Schnellabheben für alle Achsen, die im linearen Zusammenhang zurückgezogen wurden.

Hinweis

Bevor über POLFMASK oder POLFMLIN das Schnellabheben auf eine feste Position freigegeben werden kann, muss für die ausgewählten Achsen eine Position mit POLF programmiert worden sein.

Hinweis

Werden Achsen nacheinander mit POLFMASK, POLFMLIN oder POLFMLIN, POLFMASK freigegeben, gilt für die jeweilige Achse immer die letzte Festlegung.

 VORSICHT

Die mit POLF programmierten Positionen und die Aktivierung durch POLFMASK oder POLFMLIN werden bei Teileprogrammstart gelöscht. Das heißt, der Anwender muss in jedem Teileprogramm die Werte für POLF und die selektierten Achsen in POLFMASK bzw. POLFMLIN neu programmieren.

ACHTUNG

Wird bei Verwendung der Kurzformen POLFA nur der Typ geändert, dann muss der Anwender sicherstellen, dass entweder die Rückzugposition oder der Rückzugsweg einen sinnvollen Wert enthält. Insbesondere sind die Rückzugposition und der Rückzugsweg nach POWER ON neu zu setzen.

Beispiel

Rückziehen einer Einzelachse:

Programmcode	Kommentar
MD37500 \$MA_ESR_REACTION[AX1]=21	; NC-geführtes Rückziehen.
...	
\$AA_ESR_ENABLE[AX1]=1	
POLFA (AX1,1,20.0)	; AX1 wird die axiale Rückzugsposition 20.0 (absolut) zugeteilt.
\$AA_ESR_TRIGGER[AX1]=1	; Ab hier beginnt der Rückzug.

Weitere Informationen

Voraussetzungen für NC-geführtes Rückziehen

- Im Kanal ist eine Rückzugsachse für das NC-geführte Rückziehen projektiert:
MD37500 \$MA_ESR_REACTION = 21
- ESR muss für diese Achse freigegeben sein:
\$AA_ESR_ENABLE = 1
- Verzögerungszeiten sind definiert:
MD21380 \$MC_ESR_DELAY_TIME1
MD21381 \$MC_ESR_DELAY_TIME2
- Im Teileprogramm sind die achsspezifischen Rückzugspositionen mit `POLF` programmiert.
- Die Achsen sind mit `POLFMASK/POLFMLIN` für das NC-geführte Rückziehen ausgewählt.
- Für die Rückzugbewegung müssen die Freigabesignale gesetzt sein und gesetzt bleiben.

NC-geführtes Rückziehen freigeben und starten

Wenn die Systemvariable `$AC_ESR_TRIGGER = 1` gesetzt wird, und wenn in diesem Kanal eine Rückzugsachse konfiguriert ist (d. h. MD37500 \$MA_ESR_REACTION = 21) und für diese Achse `$AA_ESR_ENABLE = 1` gesetzt ist, dann wird in diesem Kanal Schnellabheben (LIFTFAST) aktiviert.

Die mit `POLF` (bzw. `LFPOS`) konfigurierte Abhebebewegung der mit `POLFMASK` oder `POLFMLIN` selektierte(n) Achse(n) ersetzt die für diese Achse(n) im Teileprogramm festgelegte Bahnbewegung.

14.11 Erweitertes Stillsetzen und Rückziehen (ESR)

Für den Rückzug steht maximal die Summe der Zeiten MD21380 \$MC_ESR_DELAY_TIME1 und MD21381 \$MC_ESR_DELAY_TIME2 zur Verfügung. Nach Ablauf dieser Zeitspanne wird auch für die Rückzugachse Schnellbremsen eingeleitet mit anschließendem Nachführen.

Hinweis

Die Erweiterte Rückzugbewegung (d. h. durch \$AC_ESR_TRIGGER ausgelöstes LIFTFAST/LFPOS) ist **nicht unterbrechbar** und kann nur durch Not-Halt vorzeitig beendet werden.

Hinweis

Durch \$AC_ESR_TRIGGER ausgelöstes Rückziehen ist gegen Mehrfach-Rückzug verriegelt.

Einzelachs-Rückziehen

Beim Einzelachs-Rückziehen muss mit *POLFA* die Rückzugsposition der Einzelachse programmiert worden sein und die folgenden Bedingungen müssen eingehalten werden:

- \$AA_ESR_ENABLE = 1
- <Achse> muss zum Triggerzeitpunkt (\$AA_ESR_TRIGGER = 1) eine Einzelachse sein.
- <Typ> muss entweder 1 oder 2 sein.

Rückzugsrichtung beim Schnellabheben

Zum Zeitpunkt der Aktivierung des Schnellabhebens wird der gültige Frame berücksichtigt.

Hinweis

Frames mit Drehung beeinflussen über *POLF* auch die Richtung, in welche abgehoben wird.

Achstausch

Rückzugsachsen sind immer genau einem NC-Kanal zuzuordnen und dürfen nicht zwischen den Kanälen getauscht werden. Beim Versuch, eine Rückzugsachse in einen anderen Kanal zu tauschen, wird ein Alarm gemeldet. Erst nachdem diese Achse mit \$AA_ESR_ENABLE[AX] = 0 wieder deaktiviert wurde, kann die Achse in einen neuen Kanal getauscht werden. Nach dem erfolgten Achstausch können Achsen wieder mit \$AA_ESR_ENABLE[AX] = 1 beaufschlagt werden.

Neutrale Achsen

Neutrale Achsen können kein NC-geführtes ESR ausführen.

14.11.1.2 NC-geführtes Stillsetzen

Funktion

Für die im Kanal konfigurierte(n) Stillsetzachse(n) wird durch Setzen der Systemvariablen \$AC_ESR_TRIGGER (bzw. \$AA_ESR_TRIGGER für Einzelachsen) das NC-geführte Stillsetzen aktiviert.

Voraussetzungen

- Im Kanal ist eine Stillsetzachse für das NC-geführte Stillsetzen projektiert:
MD37500 \$MA_ESR_REACTION = 22
- ESR muss für diese Achse freigegeben sein:
\$AA_ESR_ENABLE = 1
- Verzögerungszeiten sind definiert:
MD21380 \$MC_ESR_DELAY_TIME1 (Verzögerungszeit ESR-Achsen)
MD21381 \$MC_ESR_DELAY_TIME2 (ESR-Zeit für interpolatorisches Bremsen)

Ablauf

Für die Dauer der Zeitspanne in MD21380 interpoliert die Achse ungestört weiter wie programmiert. Nach Ablauf der Zeitspanne in MD21380 wird interpolatorisch geführtes Bremsen (Rampenstopp) eingeleitet. Für das interpolatorisch geführte Bremsen steht dann maximal die Zeitspanne in MD21381 zur Verfügung. Nach Ablauf dieser Zeitspanne wird Schnellbremsen mit anschließendem Nachführen eingeleitet.

Beispiel

Stillsetzen einer Einzelachse:

Programmcode	Kommentar
MD37500 \$MC_ESR_REACTION[AX1]=22	; NC-geführtes Stillsetzen.
MD21380 \$MC_ESR_DELAY_TIME1 [AX1]=0.3	
MD21381 \$MC_ESR_DELAY_TIME2 [AX1]=0.06	
...	
\$AA_ESR_ENABLE [AX1]=1	
\$AA_ESR_TRIGGER [AX1]=1	; Ab hier beginnt das Stillsetzen.

14.11.2 Antriebsautarkes ESR

14.11.2.1 Antriebsautarkes Stillsetzen projektieren (ESRS)

Funktion

Mit der Funktion `ESRS(...)` werden die Antriebsparameter für das "Stillsetzen" der antriebsautarken ESR-Funktion projiziert.

Syntax

```
ESRS (<Achse_1>, <Stillsetzzeit_1> [, ..., <Achse_n>, <Stillsetzzeit_n>])
```

Bedeutung

`ESRS(...)`:

Funktion zum Schreiben der Antriebsparameter für die ESR-Funktion "Stillsetzen"

Die Funktion:

- muss alleine im Satz stehen.
- löst Vorlaufstopp aus.
- kann nicht in Synchronaktionen verwendet werden.

`<Achse_1>`,
...,
`<Achse_n>`:

Achse, für die antriebsautarkes Stillsetzen projiziert werden soll

Im Antrieb wird für diese Achse der Antriebsparameter `p0888` (Konfiguration) geschrieben:

`p0888 = 1`

Typ: `AXIS`

Wertebereich: `Kanalachsbezeichner`

`<Stillsetzzeit_1>`,
...,
`<Stillsetzzeit_n>`:

Zeitdauer, über die der Antrieb nach dem Auftreten eines Fehlers mit dem aktuellen Drehzahlsollwert konstant weiter fährt

Im Antrieb wird für die angegebene Achse der Antriebsparameter `p0892` (Zeitsstufe) geschrieben:

`p0892 = <Stillsetzzeit>`

Einheit: `s`

Typ: `REAL`

Wertebereich: `0.00 - 20.00`

In einem Funktionsaufruf können maximal 5 Achsen programmiert werden; `n = 5`

14.11.2.2 Antriebsautarkes Rückziehen projektieren (ESRR)

Funktion

Mit der Funktion `ESRR(...)` werden die Antriebsparameter für das "Rückziehen" der antriebsautarken ESR-Funktion projiziert.

Syntax

```
ESRR(<Achse_1>, <Rückzugsweg_1>, <Rückzugsgeschwindigkeit_1> [, ... , <Achse_n>,
<Rückzugsweg_n>, <Rückzugsgeschwindigkeit_n>])
```

Bedeutung

<code>ESRR(...):</code>	<p>Funktion zum Schreiben der Antriebsparameter für die ESR-Funktion "Rückziehen"</p> <p>Die Funktion:</p> <ul style="list-style-type: none"> • muss alleine im Satz stehen. • löst Vorlaufstopp aus. • kann nicht in Synchronaktionen verwendet werden.
<code><Achse_1>, ... , <Achse_n>:</code>	<p>Achse, für die antriebsautarkes Rückziehen projiziert werden soll</p> <p>Im Antrieb wird für diese Achse der Antriebsparameter p0888 (Konfiguration) geschrieben:</p> <p>p0888 = 2</p> <p>Typ: AXIS</p> <p>Wertebereich: Kanalachsbezeichner</p>
<code><Rückzugsweg_1>, ... , <Rückzugsweg_n>:</code>	<p>Der Rückzugsweg wird für den Antrieb in eine Rückzugsdrehzahl umgerechnet. Der Wert wird für die angegebene Achse in den Antriebsparameter p0893 (Drehzahl) geschrieben:</p> <p>p0893 = (<Rückzugsweg_n> umgerechnet in Rückzugsdrehzahl)</p> <p>Einheit: mm/min, inch/min, Grad/min (abhängig von der Einheit der Achse)</p> <p>Typ: REAL</p> <p>Wertebereich: MIN - MAX</p>
<code><Rückzugs geschwindigkeit_1>, ... , <Rückzugs geschwindigkeit_n>:</code>	<p>Die Rückzugsgeschwindigkeit wird für den Antrieb in eine Zeitdauer umgerechnet. Der Wert wird für die angegebene Achse in den Antriebsparameter p0892 (Zeitstufe) [s] geschrieben:</p> <p>p0892 = <Rückzugsweg_n> / <Rückzugsgeschwindigkeit_n></p> <p>Einheit: mm/min, inch/min, Grad/min (abhängig von der Einheit der Achse)</p> <p>Typ: REAL</p> <p>Wertebereich: 0.00 - MAX</p>

In einem Funktionsaufruf können maximal 5 Achsen programmiert werden; n = 5

Eigene Abspanprogramme

15.1 Unterstützende Funktionen für das Abspannen

Funktionen

Für das Abspannen werden Ihnen fertige Bearbeitungszyklen angeboten. Darüber hinaus haben Sie die Möglichkeit, mit den nachfolgend aufgeführten Funktionen eigene Abspanprogramme zu erstellen:

- Konturtabelle erstellen (CONTPRON)
- Codierte Konturtabelle erstellen (CONTDCON)
- Konturaufbereitung ausschalten (EXECUTE)
- Schnittpunkt zwischen zwei Konturelementen ermitteln (INTERSEC)
(Nur für Tabellen, die durch CONTPRON erstellt wurden.)
- Konturelemente einer Tabelle satzweise abarbeiten (EXECTAB)
(Nur für Tabellen, die durch CONTPRON erstellt wurden.)
- Kreisdaten berechnen (CALCDAT)

Hinweis

Sie können diese Funktionen nicht nur zum Abspannen, sondern universell einsetzen.

Voraussetzungen

Vor dem Aufruf der Funktionen CONTPRON oder CONTDCON müssen:

- ein Startpunkt angefahren werden, der eine kollisionsfreie Bearbeitung erlaubt.
- die Schneidenradiuskorrektur mit G40 ausgeschaltet sein.

15.2 Konturtabelle erstellen (CONTPRON)

Funktion

Mit dem Befehl `CONTPRON` wird die Konturaufbereitung eingeschaltet. Die nachfolgend aufgerufenen NC-Sätze werden nicht abgearbeitet, sondern in einzelne Bewegungen aufgeteilt und in der Konturtabelle abgelegt. Jedem Konturelement entspricht eine Tabellenzeile im zweidimensionalen Feld der Konturtabelle. Die Anzahl der ermittelten Hinterschnitte wird zurückgeliefert.

Syntax

Konturaufbereitung einschalten:

```
CONTPRON (<Konturtabelle>, <Bearbeitungsart>, <Hinterschnitte>, <Bearbeitungsrichtung>)
```

Konturaufbereitung ausschalten und in den normalen Abarbeitungsmodus zurückkehren:

```
EXECUTE (<FEHLER>)
```

Siehe " Konturaufbereitung ausschalten (EXECUTE) (Seite 661) "

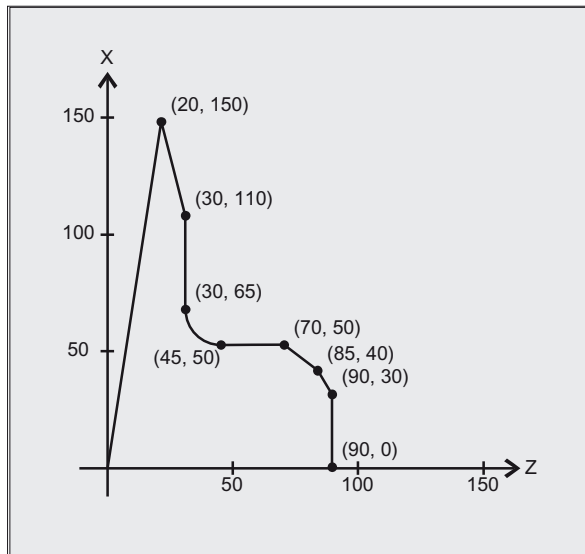
Bedeutung

<code>CONTPRON</code>	Befehl zum Einschalten der Konturaufbereitung zur Erstellung einer Konturtabelle
<code><Konturtabelle></code>	Name der Konturtabelle
<code><Bearbeitungsart></code>	Parameter für die Bearbeitungsart
	Typ: CHAR
	Wert: "G" Längsdrehen: Innenbearbeitung
	"L" Längsdrehen: Außenbearbeitung
	"N" Plandrehen: Innenbearbeitung
	"P" Plandrehen: Außenbearbeitung
<code><Hinterschnitte></code>	Ergebnisvariable für die Anzahl auftretender Hinterschnittelemente
	Typ: INT
<code><Bearbeitungsrichtung></code>	Parameter für die Bearbeitungsrichtung
	Typ: INT
	Wert: 0 Konturaufbereitung vorwärts (Standardwert)
	1 Konturaufbereitung in beiden Richtungen

Beispiel 1

Erstellen einer Konturtabelle mit:

- Namen "KTAB"
- max. 30 Konturelementen (Kreise, Geraden)
- einer Variablen für die Anzahl auftretender Hinterschnittelemente
- einer Variablen für Fehlermeldungen



NC-Programm:

Programmcode	Kommentar
N10 DEF REAL KTAB[30,11]	; Konturtabelle mit Namen KTAB und max. 30 Konturelementen, Parameterwert 11 (Spaltenzahl der Tabelle) ist eine feste Größe.
N20 DEF INT ANZHINT	; Variable für die Anzahl der Hinterschnittelemente mit Namen ANZHINT.
N30 DEF INT FEHLER	; Variable für die Fehlerrückmeldung (0=kein Fehler, 1=Fehler).
N40 G18	
N50 CONTPRON(KTAB,"G",ANZHINT)	; Konturaufbereitung einschalten.
N60 G1 X150 Z20	; N60 bis N120: Konturbeschreibung
N70 X110 Z30	
N80 X50 RND=15	
N90 Z70	
N100 X40 Z85	
N110 X30 Z90	
N120 X0	
N130 EXECUTE(FEHLER)	; Füllen der Konturtabelle beenden, Umschalten auf normalen Programmbetrieb.
N140 ...	; Weitere Bearbeitung der Tabelle.

Konturtabelle KTAB:

Index Zeile	Spalte									
(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
7	7	11	0	0	20	150	0	82.40535663	0	0
0	2	11	20	150	30	110	-1111	104.0362435	0	0
1	3	11	30	110	30	65	0	90	0	0
2	4	13	30	65	45	50	0	180	45	65
3	5	11	45	50	70	50	0	0	0	0
4	6	11	70	50	85	40	0	146.3099325	0	0
5	7	11	85	40	90	30	0	116.5650512	0	0
6	0	11	90	30	90	0	0	90	0	0
0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0

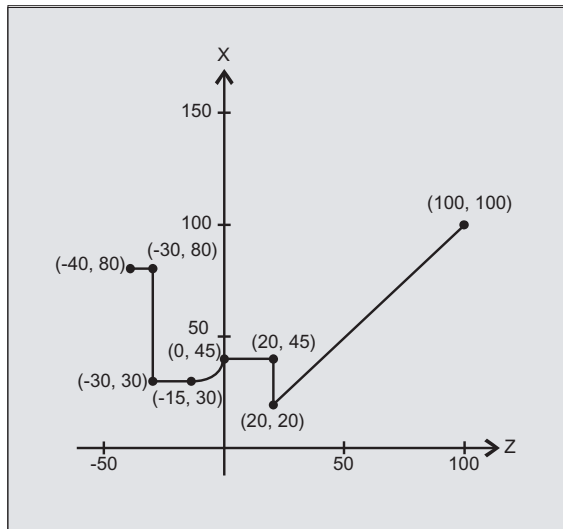
Erläuterung der Spalteninhalte:

- (0) Zeiger auf nächstes Konturelement (auf die Zeilennummer desselben)
- (1) Zeiger auf vorhergehendes Konturelement
- (2) Codierung des Konturmodus für die Bewegung
Mögliche Werte für X = abc
 $a = 10^2 \quad G90 = 0 \quad G91 = 1$
 $b = 10^1 \quad G70 = 0 \quad G71 = 1$
 $c = 10^0 \quad G0 = 0 \quad G1 = 1 \quad G2 = 2 \quad G3 = 3$
- (3), (4) Anfangspunkt der Konturelemente
(3) = Abszisse, (4) = Ordinate in der aktuellen Ebene
- (5), (6) Endpunkt der Konturelemente
(5) = Abszisse, (6) = Ordinate in der aktuellen Ebene
- (7) Max-/min-Anzeiger: kennzeichnet lokale Maxima und Minima in der Kontur
- (8) Maximaler Wert zwischen Konturelement und Abszisse (bei Längsbearbeitung) bzw. Ordinate (bei Planbearbeitung). Der Winkel ist abhängig von der programmierten Bearbeitungsart.
- (9), (10) Mittelpunktskoordinaten des Konturelements, wenn es ein Kreissatz ist.
(9) = Abszisse, (10) = Ordinate

Beispiel 2

Erstellen einer Konturtabelle mit

- Namen KTAB
- max. 92 Konturelementen (Kreise, Geraden)
- Betriebsart: Längsdrehen, Außenbearbeitung
- Aufbereitung vorwärts und rückwärts



NC-Programm:

Programmcode	Kommentar
N10 DEF REAL KTAB[92,11]	; Konturtabelle mit Namen KTAB und max. 92 Konturelementen, Parameterwert 11 ist eine feste Größe.
N20 DEF CHAR BT="L"	; Betriebsart für CONTPRON: Längsdrehen, Außenbearbeitung
N30 DEF INT HE=0	; Anzahl der Hinterschnittelemente=0
N40 DEF INT MODE=1	; Aufbereitung vorwärts und rückwärts
N50 DEF INT ERR=0	; Fehlerrückmeldung
...	
N100 G18 X100 Z100 F1000	
N105 CONTPRON(KTAB,BT,HE,MODE)	; Konturaufbereitung einschalten.
N110 G1 G90 Z20 X20	
N120 X45	
N130 Z0	
N140 G2 Z-15 X30 K=AC(-15) I=AC(45)	
N150 G1 Z-30	
N160 X80	
N170 Z-40	
N180 EXECUTE(ERR)	; Füllen der Konturtabelle beenden, Umschalten auf normalen Programmbetrieb.
...	

Konturtabelle KTAB:

Nach Ende der Konturaufbereitung steht die Kontur in beiden Richtungen zur Verfügung.

Index	Spalte										
Zeile	(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)
0	6 ¹⁾	7 ²⁾	11	100	100	20	20	0	45	0	0
1	0 ³⁾	2	11	20	20	20	45	-3	90	0	0
2	1	3	11	20	45	0	45	0	0	0	0
3	2	4	12	0	45	-15	30	5	90	-15	45
4	3	5	11	-15	30	-30	30	0	0	0	0
5	4	7	11	-30	30	-30	45	-1111	90	0	0
6	7	0 ⁴⁾	11	-30	80	-40	80	0	0	0	0
7	5	6	11	-30	45	-30	80	0	90	0	0
8	1 ⁵⁾	2 ⁶⁾	0	0	0	0	0	0	0	0	0
	...										
83	84	0 ⁷⁾	11	20	45	20	80	0	90	0	0
84	90	83	11	20	20	20	45	-1111	90	0	0
85	0 ⁸⁾	86	11	-40	80	-30	80	0	0	0	0
86	85	87	11	-30	80	-30	30	88	90	0	0
87	86	88	11	-30	30	-15	30	0	0	0	0
88	87	89	13	-15	30	0	45	-90	90	-15	45
89	88	90	11	0	45	20	45	0	0	0	0
90	89	84	11	20	45	20	20	84	90	0	0
91	83 ⁹⁾	85 ¹⁰⁾	11	20	20	100	100	0	45	0	0

Erläuterung der Spalteninhalte und der Anmerkungen zu den Zeilen 0, 1, 6, 8, 83, 85 und 91

Es gelten die im Beispiel 1 genannten Erläuterungen der Spalteninhalte.

Immer in Tabellen-Zeile 0:

- 1) Vorgänger: Zeile n enthält das Konturende vorwärts
- 2) Nachfolger: Zeile n ist das Konturtabellenende vorwärts

Je einmal innerhalb der Konturelemente vorwärts:

- 3) Vorgänger: Konturbeginn (vorwärts)
- 4) Nachfolger: Konturende (vorwärts)

Immer auf Zeile Konturtabellenende (vorwärts) +1:

- 5) Vorgänger: Anzahl der Hinterschnitte vorwärts
- 6) Nachfolger: Anzahl der Hinterschnitte rückwärts

Je einmal innerhalb der Konturelemente rückwärts:

- 7) Nachfolger: Konturende (rückwärts)
- 8) Vorgänger: Konturbeginn (rückwärts)

Immer in letzter Tabellen-Zeile:

- 9) Vorgänger: Zeile n ist der Konturtabellenanfang (rückwärts)
- 10) Nachfolger: Zeile n enthält den Konturanfang (rückwärts)

Weitere Informationen

Erlaubte Verfahrbefehle, Koordinatensystem

Für die Konturprogrammierung sind folgende G-Befehle zulässig:

- G-Gruppe 1: G0, G1, G2, G3

Zusätzlich möglich sind:

- Rundung und Fase
- Kreisprogrammierung über CIP und CT

Die Funktionen Spline, Polynom und Gewinde führen zu Fehlern.

Änderungen des Koordinatensystems durch Einschalten eines Frames sind zwischen CONTPRON und EXECUTE nicht zulässig. Gleiches gilt für einen Wechsel zwischen G70 und G71 bzw. G700 und G710.

Ein Tausch der Geometrieachsen mit GEOAX während der Aufbereitung der Konturtablelle führt zu einem Alarm.

Hinterschnittlelemente

Die Konturbeschreibung der einzelnen Hinterschnittlelemente kann wahlweise in einem Unterprogramm oder in einzelnen Sätzen erfolgen.

Abspannen unabhängig von der programmierten Konturrichtung

Die Konturaufbereitung mit CONTPRON wurde so erweitert, dass nach ihrem Aufruf die Konturtablelle unabhängig von der programmierten Richtung zur Verfügung steht.

15.3 Codierte Konturtable erstellen (CONTDCON)

Funktion

Bei der mit `CONTDCON` eingeschalteten Konturaufbereitung werden die nachfolgend aufgerufenen NC-Sätze in einer 6-spaltigen Konturtable speichergünstig codiert abgelegt. Jedem Konturelement entspricht eine Tabellenzeile in der Konturtable. Aus Kenntnis der unten angegebenen Codierungsregeln können Sie z. B. für Zyklen aus den Tabellenzeilen DIN-Code-Programme zusammenstellen. In der Tabellenzeile mit der Nummer 0 werden die Daten des Ausgangspunkts gespeichert.

Syntax

Konturaufbereitung einschalten:

`CONTDCON (<Konturtable>, <Bearbeitungsrichtung>)`

Konturaufbereitung ausschalten und in den normalen Abarbeitungsmodus zurückkehren:

`EXECUTE (<FEHLER>)`

Siehe " Konturaufbereitung ausschalten (EXECUTE) (Seite 661) "

Bedeutung

`CONTDCON`

Befehl zum Einschalten der Konturaufbereitung zur Erstellung einer codierten Konturtable

`<Konturtable>`

Name der Konturtable

`<Bearbeitungsrichtung>`

Parameter für Bearbeitungsrichtung

Typ: INT

Wert: 0 Konturaufbereitung gemäß der Folge der Kontursätze (Standardwert)

1 unzulässig

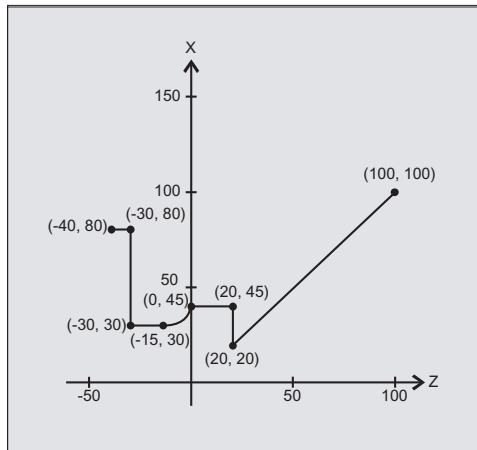
Hinweis

Die für `CONTDCON` zugelassenen G-Codes im zu tabellierenden Programmstück sind umfangreicher als bei `CONTPRON`. Darüber hinaus werden Vorschübe und Vorschubtyp pro Konturstück mitgespeichert.

Beispiel

Erstellen einer Konturtabelle mit:

- Namen "KTAB"
- Konturelementen (Kreise, Geraden)
- Betriebsart: Drehen
- Bearbeitungsrichtung: vorwärts

**NC-Programm:**

Programmcode	Kommentar
N10 DEF REAL KTAB [9,6]	; Konturtabelle mit Namen KTAB und 9 Tabellenzeilen. Diese erlauben 8 Kontursätze. Der Parameterwert 6 (Spaltenzahl der Tabelle) ist eine feste Größe.
N20 DEF INT MODE = 0	; Variable für die Bearbeitungsrichtung. Standardwert 0: nur in programmierter Richtung der Kontur.
N30 DEF INT ERROR = 0	; Variable für die Fehlerrückmeldung.
...	
N100 G18 G64 G90 G94 G710	
N101 G1 Z100 X100 F1000	
N105 CONTDCON (KTAB, MODE)	; Aufruf Konturaufbereitung (MODE darf weggelassen werden).
N110 G1 Z20 X20 F200	; Konturbeschreibung.
N120 G9 X45 F300	
N130 Z0 F400	
N140 G2 Z-15 X30 K=AC(-15) I=AC(45) F100	
N150 G64 Z-30 F600	
N160 X80 F700	
N170 Z-40 F800	
N180 EXECUTE(ERROR)	; Füllen der Konturtabelle beenden, Umschalten auf normalen Programmbetrieb.
...	

Konturtabelle KTAB:

Zeilenindex	Spaltenindex					
	0	1	2	3	4	5
	Konturmodus	Endpunkt Abszisse	Endpunkt Ordinate	Mittelpunkt Abszisse	Mittelpunkt Ordinate	Vorschub
0	30	100	100	0	0	7
1	11031	20	20	0	0	200
2	111031	20	45	0	0	300
3	11031	0	45	0	0	400
4	11032	-15	30	-15	45	100
5	11031	-30	30	0	0	600
6	11031	-30	80	0	0	700
7	11031	-40	80	0	0	800
8	0	0	0	0	0	0

Erläuterung der Spalteninhalte:

Zeile 0: Codierungen für den **Startpunkt**:

- Spalte 0: 10^0 (Einerstelle): G0 = 0
 10^1 (Zehnerstelle): G70 = 0, G71 = 1, G700 = 2, G710 = 3
- Spalte 1: Startpunkt Abszisse
- Spalte 2: Startpunkt Ordinate
- Spalte 3-4: 0
- Spalte 5: Zeilenindex des letzten Konturstückes in der Tabelle

Zeilen 1-n: Einträge der **Konturstücke**

- Spalte 0: 10^0 (Einerstelle): G0 = 0, G1 = 1, G2 = 2, G3 = 3
 10^1 (Zehnerstelle): G70 = 0, G71 = 1, G700 = 2, G710 = 3
 10^2 (Hunderterstelle): G90 = 0, G91 = 1
 10^3 (Tausenderstelle): G93 = 0, G94 = 1, G95 = 2, G96 = 3
 10^4 (Zehntausenderstelle): G60 = 0, G44 = 1, G641 = 2, G642 = 3
 10^5 (Hunderttausender Stelle): G9 = 1
- Spalte 1: Endpunkt Abszisse
- Spalte 2: Endpunkt Ordinate
- Spalte 3: Mittelpunkt Abszisse bei Kreisinterpolation
- Spalte 4: Mittelpunkt Ordinate bei Kreisinterpolation
- Spalte 5: Vorschub

Weitere Informationen

Erlaubte Verfahrbefehle, Koordinatensystem

Für die Konturprogrammierung sind folgende G-Gruppen und G-Befehle zulässig:

G-Gruppe 1:	G0, G1, G2, G3
G-Gruppe 10:	G60, G64, G641, G642
G-Gruppe 11:	G9
G-Gruppe 13:	G70, G71, G700, G710
G-Gruppe 14:	G90, G91
G-Gruppe 15:	G93, G94, G95, G96, G961

Zusätzlich möglich sind:

- Rundung und Fase
- Kreisprogrammierung über `CIP` und `CT`

Die Funktionen Spline, Polynom und Gewinde führen zu Fehlern.

Änderungen des Koordinatensystems durch Einschalten eines Frames sind zwischen `CONTDCON` und `EXECUTE` nicht zulässig. Gleiches gilt für einen Wechsel zwischen `G70` und `G71` bzw. `G700` und `G710`.

Ein Tausch der Geometrieachsen mit `GEOAX` während der Aufbereitung der Konturtabelle führt zu einem Alarm.

Bearbeitungsrichtung

Die mit `CONTDCON` erzeugte Konturtabelle ist zum Abspannen in der programmierten Richtung der Kontur vorgesehen.

15.4 Schnittpunkt zwischen zwei Konturelementen ermitteln (INTERSEC)

Funktion

INTERSEC ermittelt den Schnittpunkt von zwei normierten Konturelementen aus mit `CONTPRON` erzeugten Konturtabellen.

Syntax

```
<Status>=INTERSEC (<Konturtable_1> [<Konturelement_1>],  
<Konturtable_2> [<Konturelement_2>], <Schnittpunkt>, <Bearbeitungsart>)
```

Bedeutung

INTERSEC	Schlüsselwort zur Ermittlung des Schnittpunkts zweier Konturelemente aus mit <code>CONTPRON</code> erzeugten Konturtabellen
<Status>	Variable für den Schnittpunktstatus Typ: <code>BOOL</code> Wert: <code>TRUE</code> Schnittpunkt gefunden <code>FALSE</code> kein Schnittpunkt gefunden
<Konturtable_1>	Name der ersten Konturtable
<Konturelement_1>	Nummer des Konturelements der ersten Konturtable
<Konturtable_2>	Name der zweiten Konturtable
<Konturelement_2>	Nummer des Konturelements der zweiten Konturtable
<Schnittpunkt>	Schnittpunkt-Koordinaten in der aktiven Ebene (<code>G17 / G18 / G19</code>) Typ: <code>REAL</code>
<Bearbeitungsart>	Parameter für die Bearbeitungsart Typ: <code>INT</code> Wert: <code>0</code> Schnittpunktberechnung in der mit Parameter 2 aktiven Ebene (Standardwert) <code>1</code> Schnittpunktberechnung unabhängig der übergebenen Ebene

Hinweis

Beachten Sie, dass die Variablen vor ihrer Verwendung definiert sein müssen.

Die Übergabe der Konturen erfordert die Einhaltung der mit `CONTPRON` definierten Werte:

Parameter	Bedeutung
2	Codierung des Kontur-Mode für die Bewegung
3	Kontur-Anfangpunkt Abszisse
4	Kontur-Anfangpunkt Ordinate
5	Kontur-Endpunkt Abszisse
6	Kontur-Endpunkt Ordinate
9	Mittelpunktskoordinate für die Abszisse (nur bei Kreis-Kontur)
10	Mittelpunktskoordinate für die Ordinate (nur bei Kreis-Kontur)

Beispiel

Schnittpunkt von Konturelement 3 der Tabelle `TABNAME1` und Konturelement 7 der Tabelle `TABNAME2` ermitteln. Die Schnittpunkt-Koordinaten in der aktiven Ebene werden in der Variablen `ISCOORD` (1. Element = Abszisse, 2. Element = Ordinate) abgelegt. Existiert kein Schnittpunkt, erfolgt ein Sprung zu `KEINSCH` (kein Schnittpunkt gefunden).

Programmcode	Kommentar
<code>DEF REAL TABNAME1 [12,11]</code>	<code>; Konturtabelle 1</code>
<code>DEF REAL TABNAME2 [10,11]</code>	<code>; Konturtabelle 2</code>
<code>DEF REAL ISCOORD [2]</code>	<code>; Variable für Schnittpunkt-Koordinaten.</code>
<code>DEF BOOL ISPOINT</code>	<code>; Variable für Schnittpunktstatus.</code>
<code>DEF INT MODE</code>	<code>; Variable für Bearbeitungsart.</code>
<code>...</code>	
<code>MODE=1</code>	<code>; Berechnung unabhängig von der aktiven Ebene.</code>
<code>N10 ISPOINT=INTERSEC (TABNAME1 [3], TABNAME2 [7], ISCOORD, MODE)</code>	<code>; Aufruf Schnittpunkt der Konturelemente.</code>
<code>N20 IF ISPOINT==FALSE GOTOF KEINSCH</code>	<code>; Sprung zu KEINSCH.</code>
<code>...</code>	

15.5 Konturelemente einer Tabelle satzweise abfahren (EXECTAB)

Funktion

Mit dem Befehl `EXECTAB` können Sie Konturelemente einer Tabelle, die z. B. mit dem Befehl `CONTPRON` erzeugt wurde, satzweise abfahren.

Syntax

```
EXECTAB (<Konturtabelle> [<Konturelement>])
```

Bedeutung

<code>EXECTAB</code>	Befehl zum Abfahren eines Konturelements
<code><Konturtabelle></code>	Name der Konturtabelle
<code><Konturelement></code>	Nummer des Konturelements

Beispiel

Die Konturelemente 0 bis 2 der Tabelle `KTAB` sollen satzweise abgefahren werden.

Programmcode	Kommentar
<code>N10 EXECTAB(KTAB[0])</code>	<code>; Element 0 der Tabelle KTAB verfahren.</code>
<code>N20 EXECTAB(KTAB[1])</code>	<code>; Element 1 der Tabelle KTAB verfahren.</code>
<code>N30 EXECTAB(KTAB[2])</code>	<code>; Element 2 der Tabelle KTAB verfahren.</code>

15.6 Kreisdaten berechnen (CALCDAT)

Funktion

Mit dem Befehl `CALCDAT` können Sie aus drei oder vier bekannten Kreispunkten den Radius und die Kreismittelpunkt-Koordinaten berechnen. Die angegebenen Punkte müssen unterschiedlich sein. Bei 4 Punkten, die nicht exakt auf dem Kreis liegen, wird für Kreismittelpunkt und Radius ein Mittelwert gewählt.

Syntax

```
<Status>=CALCDAT(<Kreispunkte> [<Anzahl>, <Art>] , <Anzahl>, <Ergebnis>)
```

Bedeutung

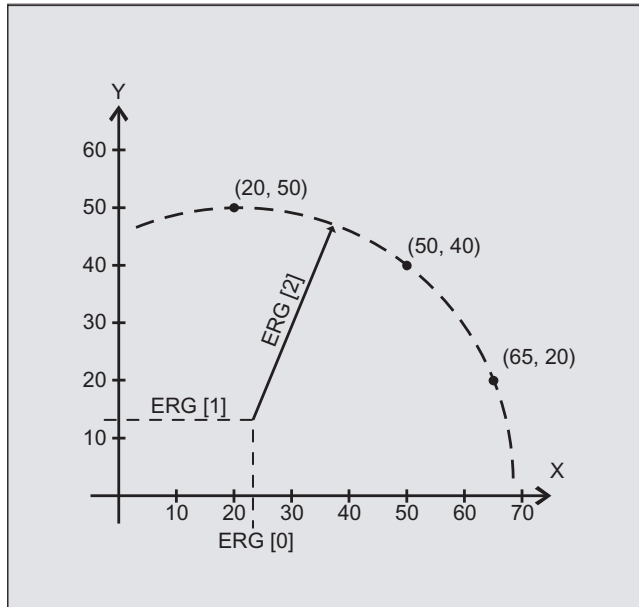
<code>CALCDAT</code>	Befehl zur Berechnung von Radius und Mittelpunkt-Koordinaten eines Kreises aus 3 oder 4 Punkten
<code><Status></code>	Variable für den Kreisberechnungsstatus Typ: BOOL Wert: TRUE Die angegebenen Punkte liegen auf einem Kreis. FALSE Die angegebenen Punkte liegen nicht auf einem Kreis.
<code><Kreispunkte> []</code>	Variable zur Angabe der Kreispunkte mit den Parametern: <code><Anzahl></code> Anzahl der Kreispunkte (3 oder 4) <code><Art></code> Art der Koordinatenangabe, z. B. 2 für 2-Punkt-Koordinaten
<code><Anzahl></code>	Parameter für die Anzahl der zur Berechnung verwendeten Punkte (3 oder 4)
<code><Ergebnis> [3]</code>	Variable für Ergebnis: Angabe von Kreismittelpunkt-Koordinaten und Radius 0 Kreismittelpunkt-Koordinate: Abszissenwert 1 Kreismittelpunkt-Koordinate: Ordinatenwert 2 Radius

Hinweis

Beachten Sie, dass die Variablen vor ihrer Verwendung definiert sein müssen.

Beispiel

Von drei Punkten soll ermittelt werden, ob sie auf einem Kreisabschnitt liegen.



Programmcode	Kommentar
N10 DEF REAL PKT[3,2] = (20,50,50,40,65,20)	; Variable zur Angabe der Kreispunkte
N20 DEF REAL ERG[3]	; Variable für Ergebnis
N30 DEF BOOL STATUS	; Variable für Status
N40 STATUS=CALCDAT(PKT,3,ERG)	; Aufruf der ermittelten Kreisdaten.
N50 IF STATUS == FALSE GOTOF ERROR	; Sprung zu Fehler

15.7 Konturaufbereitung ausschalten (EXECUTE)

Funktion

Mit dem Befehl `EXECUTE` wird die Konturaufbereitung abgeschaltet und gleichzeitig in den normalen Abarbeitungsmodus zurückgeschaltet.

Syntax

```
EXECUTE (<FEHLER>)
```

Bedeutung

<code>EXECUTE</code>	Befehl zum Beenden der Konturaufbereitung
<code><FEHLER></code>	Variable für Fehlerrückmeldung
	Typ: INT
	Der Wert der Variablen zeigt an, ob die Kontur fehlerfrei aufbereitet werden konnte:
0	Fehler
1	kein Fehler

Beispiel

```
Programcode
...
N30 CONTPRON(...)
N40 G1 X... Z...
...
N100 EXECUTE(...)
...
```


Zyklen extern programmieren

16.1 Technologische Zyklen

16.1.1 Einleitung

Inhalt

In diesem Kapitel sind die Technologischen Zyklen ab Version 2.6 für die Erstellung externer NC-Programme dokumentiert.

Aufbau

Die Dokumentation ist wie folgt strukturiert:

- **Programmierung**
Zyklusname und Aufrufreihenfolge der Übergabeparameter
- **Parameter**
Tabelle zur Erläuterung der einzelnen Parameter

Parameterbeschreibung

In der Tabelle befinden sich die Namen der intern verwendeten Parameter und die Erklärung zur Bedeutung und dem möglichen Wertebereich. Außerdem sind Abhängigkeiten zwischen den Parametern erklärt. Die Spalte zum Verweis auf den Parameter in der Maske soll dazu dienen, beim Rückübersetzen von extern generierten Zyklenuufrufen an der Steuerung programmierte Werte wieder zu finden.

Parameter "nur für die Oberfläche"

In der Tabelle sind Parameter mit "nur für die Oberfläche" gekennzeichnet. Diese sind für die Funktion des Zyklus nicht von Bedeutung. Sie werden nur benötigt, um Zyklenuufrufe vollständig rückübersetzen zu können. Sind sie nicht programmiert, kann der Zyklus trotzdem rückübersetzt werden, die Felder sind dann entsprechend farblich gekennzeichnet und müssen in der Maske ausgefüllt werden.

Parameter "reserviert"

Parameter, die mit "reserviert" beschrieben sind, müssen mit Wert 0 oder Leerkomma programmiert werden, damit die Zuordnung der folgenden Aufrufparameter zu den internen Zyklusparametern stimmt. Ausnahme: bei Stringparametern Wert "" oder Leerkomma.

Kompatibilität

Die Technologischen Zyklen ab Version 2.6 stellen eine Weiterentwicklung der Zyklenpakete für SINUMERIK 840D sl bis GIV 1.5 (Zyklen bis Version 7.5) dar. NC-Programme mit Zyklenaufrufen dieser früheren SW-Versionen sind weiterhin lauffähig.

Die meisten Zyklen wurden um neue Übergabeparameter erweitert oder vorhandene Parameter wurden in ihrem Wertebereich erweitert, um neue Funktionen programmieren zu können (wie z. B. der häufig verwendete Parameter `_VARI` für die Bearbeitungsart).

Der Begriff "Kompatibilität" in dieser Dokumentation verweist auf Eingabewerte, die früher nicht programmiert wurden. Werden Werte dementsprechend versorgt, läuft der Zyklus funktionell so ab, wie bis Version 7.5.

Zyklen auf Positionsmuster wiederholen

Bohr- und Fräszyklen können auf Positionsmuster wiederholt werden (modale Aufrufe). Vor dem Zyklus ist dann in derselben Zeile `MCALL` zu schreiben, z. B. `MCALL CYCLE83(...)`.

Hinweis

Sind bestimmte Übergabeparameter (z. B. `_VARI`, `_GMODE`, `_DMODE`, `_AMODE`) indirekt als Parameter programmiert, wird die Eingabemaske beim Rückübersetzen geöffnet, kann aber nicht abgespeichert werden, weil es zu bestimmtem Auswahlfeldern keine eindeutige Zuordnung gibt.

16.1.2 Bohren, Zentrieren - CYCLE81

Programmierung

CYCLE81 (REAL RTP, REAL RFP, REAL SDIS, REAL DP, REAL DPR, REAL _DTB,
INT _GMODE, INT _DMODE, INT _AMODE)

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	RP	RTP	Rückzugsebene (abs)
2	Z0	RFP	Bezugspunkt (abs)
3	SC	_SDIS	Sicherheitsabstand (additiv auf Bezugspunkt, ohne Vorzeichen einzugeben)
4	Z1/Ø	_DP	Bohrtiefe (abs)/Durchmesser der Zentrierung (abs), siehe _GMODE
5	Z1	-DPR	Bohrtiefe (ink)
6	DT	_DTB	Verweilzeit auf Endbohrtiefe, siehe _AMODE
7		_GMODE	Geometriemode (Bewertung der programmierten Geometriewerte) EINER: reserviert ZEHNER: Zentrierung bezogen auf die Tiefe/den Durchmesser 0 = Kompatibilität, Tiefe 1 = Durchmesser
8		_DMODE	Displaymode EINER: Bearbeitungsebene G17/G18/G19 0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv)
9		_AMODE	Alternativmode EINER: Bohrtiefe Z1 (abs/ink) 0 = Kompatibilität, aus Programmierung DP/DPR 1 = inkrementell 2 = absolut ZEHNER: Verweilzeit auf Endbohrtiefe DT in Sekunden/Umdrehungen 0 = Kompatibilität, aus Vorzeichen von DTB (> 0 Sekunden oder < 0 Umdrehungen) 1 = in Sekunden 2 = in Umdrehungen

16.1.3 Bohren, Plansenken - CYCLE82

Programmierung

CYCLE82 (REAL RTP, REAL RFP, REAL SDIS, REAL DP, REAL DPR, REAL DTB,
INT _GMODE, INT _DMODE, INT _AMODE)

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	RP	RTP	Rückzugsebene (abs)
2	Z0	RFP	Bezugspunkt (abs)
3	SC	SDIS	Sicherheitsabstand (additiv auf Bezugspunkt, ohne Vorzeichen einzugeben)
4	Z1	DP	Bohrtiefe (abs), siehe _AMODE
5	Z1	DPR	Bohrtiefe (ink), siehe _AMODE
6	DT	DTB	Verweilzeit auf Endbohrtiefe, siehe _AMODE
7		_GMODE	Geometriemode (Bewertung der programmierten Geometriewerte) EINER: reserviert ZEHNER: Bohrtiefe bezogen auf Spitze/Schaft 0 = Kompatibilität, Spitze 1 = Schaft
8		_DMODE	Displaymode EINER: Bearbeitungsebene G17/G18/G19 0 = Kompatibilität, es bleibt die vor Zyklusaufruf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv)
9		_AMODE	Alternativmode EINER: Bohrtiefe Z1 (abs/ink) 0 = Kompatibilität aus Programmierung DP/DPR 1 = inkrementell 2 = absolut ZEHNER: Verweilzeit DT auf Endbohrtiefe in Sekunden/Umdrehungen 0 = Kompatibilität, aus Vorzeichen von DT (> 0 Sekunden/< 0 Umdrehungen) 1 = in Sekunden 2 = in Umdrehungen

16.1.4 Reiben - CYCLE85

Programmierung

CYCLE85 (REAL RTP, REAL RFP, REAL SDIS, REAL DP, REAL DPR, REAL DTB, REAL FFR, REAL RFF, INT _GMODE, INT _DMODE, INT _AMODE)

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	RP	RTP	Rückzugsebene (abs)
2	Z0	RFP	Bezugspunkt (abs)
3	SC	SDIS	Sicherheitsabstand (additiv auf Bezugspunkt, ohne Vorzeichen einzugeben)
4	Z1	DP	Bohrtiefe (abs), siehe _AMODE
5	Z1	DPR	Bohrtiefe (ink), siehe _AMODE
6	DT	DTB	Verweilzeit auf Endbohrtiefe, siehe _AMODE
7	F	FFR	Vorschub
8	FR	RFF	Vorschub beim Rückzug
9		_GMODE	reserviert
10		_DMODE	Displaymode EINER: Bearbeitungsebene G17/G18/G19 0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv)
11		_AMODE	Alternativmode (Bohren) EINER: Bohrtiefe Z1 (abs/ink) 0 = Kompatibilität, aus Programmierung DP/DPR 1 = inkrementell 2 = absolut ZEHNER: Verweilzeit DT auf Endbohrtiefe in Sekunden/Umdrehungen 0 = Kompatibilität, aus Vorzeichen von DT (> 0 Sekunden oder < 0 Umdrehungen) 1 = in Sekunden 2 = in Umdrehungen

16.1.5 Tieflochbohren - CYCLE83

Programmierung

```
CYCLE83 (REAL RTP, REAL RFP, REAL SDIS, REAL DP, REAL DPR, REAL FDEP,
REAL FDPR, REAL _DAM, REAL DTB, REAL DTS, REAL FRF, INT VARI, INT
_AXN, REAL _MDEP, REAL _VRT, REAL _DTD, REAL _DIS1, INT _GMODE, INT
_DMODE, INT _AMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	RP	RTP	Rückzugsebene (abs)
2	Z0	RFP	Bezugspunkt (abs)
3	SC	SDIS	Sicherheitsabstand (additiv auf Bezugspunkt, ohne Vorzeichen einzugeben)
4	Z1	DP	Endbohrtiefe (abs), siehe _AMODE
5	Z1	DPR	Endbohrtiefe (ink), siehe _AMODE
6	D	FDEP	1. Bohrtiefe (abs), siehe _AMODE
7	D	FDPR	1. Bohrtiefe (ink), siehe _AMODE
8	DF	_DAM	Betrag/Prozentsatz für jede weitere Zustellung (Degressionsbetrag/Prozentsatz), siehe _AMODE
9	DTB	DTB	Verweilzeit auf Bohrtiefe, siehe _AMODE
10	DTS	DTS	Verweilzeit am Anfangspunkt (nur bei Entspanen), siehe _AMODE
11	FD1	FRF	Prozentsatz für den Vorschub bei der ersten Zustellung, siehe _AMODE
12		VARI	Bearbeitungsart EINER: Spänebrechen/Entspanen 0 = Spänebrechen 1 = Entspanen
13		_AXN	Werkzeugachse: 0 = 3. Geometrieachse 1 = 1. Geometrieachse 2 = 2. Geometrieachse > 2 = 3. Geometrieachse
14	V1	_MDEP	minimale Zustellung (nur bei Prozentsatz für Degression)
15	V2	_VRT	Rückzugsbetrag nach jeder Bearbeitung (nur bei Spänebrechen) > 0 = variabler Rückzugsbetrag 0 = Standardwert 1 mm
16	DT	_DTD	Verweilzeit auf Endbohrtiefe, siehe _AMODE
17	V3	_DIS1	Vorhalteabstand (nur bei Entspanen), siehe _AMODE
18		_GMODE	Geometriemode (Bewertung der programmierten Geometriewerte) EINER: reserviert ZEHNER: Bohrtiefe bezogen auf Spitze/Schaft 0 = Spitze 1 = Schaft

Nr.	Param Maske	Param intern	Erklärung
19		_DMODE	<p>Displaymode</p> <hr/> <p>EINER: Bearbeitungsebene G17/G18/G19</p> <hr/> <p>0 = Kompatibilität, es bleibt die vor Zyklusauf Ruf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv)</p>
20		_AMODE	<p>Alternativmode</p> <hr/> <p>EINER: Bohrtiefe = Endbohrtiefe Z1 (abs/ink)</p> <hr/> <p>0 = Kompatibilität, aus Programmierung DP/DPR 1 = inkrementell 2 = absolut</p> <hr/> <p>ZEHNER: Verweilzeit auf Bohrtiefe DTB in Sekunden/Umdrehungen</p> <hr/> <p>0 = Kompatibilität aus Vorzeichen von DTB (> 0 Sekunden oder < 0 Umdrehungen) 1 = in Sekunden 2 = in Umdrehungen</p> <hr/> <p>HUNDERTER: Verweilzeit am Anfangspunkt von DTS in Sekunden/Umdrehungen</p> <hr/> <p>0 = Kompatibilität aus Vorzeichen von DTS (> 0 Sekunden oder < 0 Umdrehungen) 1 = in Sekunden 2 = in Umdrehungen</p> <hr/> <p>TAUSENDER: Verweilzeit auf Endbohrtiefe DT in Sekunden/Umdrehungen</p> <hr/> <p>0 = Kompatibilität aus Vorzeichen von _DTD (> 0 Sekunden oder < 0 Umdrehungen) 1 = in Sekunden 2 = in Umdrehungen</p> <hr/> <p>ZEHNTAUSENDER: 1. Bohrtiefe D (abs/ink)</p> <hr/> <p>0 = Kompatibilität, aus Programmierung FDEP/FDPR 1 = inkrementell 2 = absolut</p> <hr/> <p>HUNDERTTAUSENDER: Betrag/Prozentsatz DAM für jede weitere Zustellung (Degression)</p> <hr/> <p>0 = Kompatibilität, aus Vorzeichen von DAM (>0 Betrag oder <0 Faktor 0.001 bis 1.0) 1 = Betrag 2 = Prozentsatz (0.001 bis 100 %)</p> <hr/> <p>EINEMILLION: Vorhalteabstand V3 automatisch/manuell</p> <hr/> <p>0 = Kompatibilität aus Vorzeichen von _DIS1 (= 0 automatisch oder > 0 manuell) 1 = automatisch (wird im Zyklus berechnet) 2 = manuell (programmierter Wert)</p> <hr/> <p>ZEHNMILLION: Vorschubfaktor für erste Zustellung FRF als Faktor/Prozentsatz</p> <hr/> <p>0 = Kompatibilität, als Faktor (0.001 bis 1.0, FRF = 0 bedeutet 100%) 1 = Prozentsatz (0.001 bis 999,999 %)</p>

16.1.6 Ausdrehen - CYCLE86

Programmierung

```
CYCLE86 (REAL RTP, REAL RFP, REAL SDIS, REAL DP, REAL DPR, REAL DTB,
INT SDIR, REAL RPA, REAL RPO, REAL RPAP, REAL POSS, INT _GMODE, INT
_DMODE, INT _AMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	RP	RTP	Rückzugsebene (abs)
2	Z0	RFP	Bezugspunkt (abs)
3	SC	SDIS	Sicherheitsabstand (additiv auf Bezugspunkt, ohne Vorzeichen einzugeben)
4	Z1	DP	Bohrtiefe (abs), siehe _AMODE
5	Z1	DPR	Bohrtiefe (ink), siehe _AMODE
6	DT	DTB	Verweilzeit auf Endbohrtiefe, siehe _AMODE
7	DIR	SDIR	Spindeldrehrichtung 3 = M3 4 = M4
8	DX	RPA	Abhebebetrag in X-Richtung
9	DY	RPO	Abhebebetrag in Y-Richtung
10	DZ	RPAP	Abhebebetrag in Z-Richtung
11	SPOS	POSS	Spindelposition zum Abheben (für orientierten Spindelhalt, in Grad)
12		_GMODE	Geometriemode EINER: Abhebemodus 0 = abheben, Kompatibilität 1 = nicht abheben
13		_DMODE	Displaymode EINER: Bearbeitungsebene G17/G18/G19 0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv)
14		_AMODE	Alternativmode EINER: Bohrtiefe Z1 (abs/ink) 0 = Kompatibilität, aus Programmierung DP/DPR 1 = inkrementell 2 = absolut ZEHNER: Verweilzeit auf Endbohrtiefe DT in Sekunden/Umdrehungen 0 = Kompatibilität, aus Vorzeichen von DT (> 0 Sekunden oder < 0 Umdrehungen) 1 = in Sekunden 2 = in Umdrehungen

16.1.7 Gewindebohren ohne Ausgleichsfutter - CYCLE84

Programmierung

```
CYCLE84 (REAL RTP, REAL RFP, REAL SDIS, REAL DP, REAL DPR, REAL DTB,
INT SDAC, REAL MPIT, REAL PIT, REAL POSS, REAL SST, REAL SST1, INT
_AXN, INT _PITA, INT _TECHNO, INT _VARI, REAL _DAM, REAL _VRT,
STRING[15] _PITM, STRING[5] _PTAB, STRING[20] _PTABA, INT _GMODE,
INT _DMODE, INT _AMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	RP	RTP	Rückzugsebene (abs)
2	Z0	RFP	Bezugspunkt (abs)
3	SC	SDIS	Sicherheitsabstand (additiv auf Bezugspunkt, ohne Vorzeichen einzugeben)
4	Z1	DP	Bohrtiefe = Endbohrtiefe (abs), siehe _AMODE
5	Z1	DPR	Bohrtiefe = Endbohrtiefe (ink), siehe _AMODE
6	DT	DTB	Verweilzeit auf Bohrtiefe in Sekunden
7	SDE	SDAC	Drehrichtung nach Zyklusende
8		MPIT	Gewindegröße nur für "ISO metrisch" (Steigung wird intern während der Laufzeit berechnet)
9	P	PIT	Gewindesteigung als Wert, Maßeinheit siehe _PITA
10	αS^1	POSS	Spindelposition für orientierten Spindelstop
11	S	SST	Spindeldrehzahl für Gewindebohren
12	SR	SST1	Spindeldrehzahl für Rückzug
13		_AXN	Bohrachse: 0 = 3. Geometrieachse 1 = 1. Geometrieachse 2 = 2. Geometrieachse ≥ 3 = 3. Geometrieachse
14		_PITA	Maßeinheit der Gewindesteigung (Auswertung PIT und MPIT) 0 = Steigung in mm - Auswertung MPIT/PIT 1 = Steigung in mm - Auswertung PIT 2 = Steigung in TPI - Auswertung PIT (Gewindegänge pro inch) 3 = Steigung in inch - Auswertung PIT 4 = MODUL - Auswertung PIT

Zyklen extern programmieren

16.1 Technologische Zyklen

Nr.	Param Maske	Param intern	Erklärung
15		_TECHNO	Technologie ¹⁾ <hr/> EINER: Genauhaltverhalten 0 = Genauhaltverhalten wie vor Zyklusaufwurf aktiv 1 = Genauhalt G601 2 = Genauhalt G602 3 = Genauhalt G603 <hr/> ZEHNER: Vorsteuerung 0 = mit/ohne Vorsteuerung wie vor Zyklusaufwurf aktiv 1 = mit Vorsteuerung FFWON 2 = ohne Vorsteuerung FFWOF <hr/> HUNDERT: Beschleunigung 0 = SOFT/BRISK/DRIVE wie vor Zyklusaufwurf aktiv 1 = mit Ruckbegrenzung SOFT 2 = ohne Ruckbegrenzung BRISK 3 = reduzierte Beschleunigung DRIVE <hr/> TAUSENDER: MCALL Spindelbetrieb 0 = bei MCALL Spindelbetrieb wieder aktivieren 1 = bei MCALL in Lageregelung bleiben
16		_VARI	Bearbeitungsart: <hr/> EINER: 0 = 1 Schnitt 1 = Spänebrechen (Tieflochgewindebohren) 2 = Entspanen (Tieflochgewindebohren) <hr/> TAUSENDER: ISO/SIEMENS Mode für Eingabemaske nicht relevant 1 = Aufruf aus ISO-Kompatibilität 0 = Aufruf aus SIEMENS-Kontext
17	D	_DAM	maximale Tiefenzustellung (nur bei Entspanen/Spänebrechen)
18	V2	_VRT	Rückzugsbetrag nach jeder Bearbeitung (nur bei Spänebrechen), siehe _AMODE
19		_PITM	String als Merker für Gewindesteigungseingabe ²⁾
20		_PTAB	String für Gewindetabelle ("", "ISO", "BSW", "BSP", "UNC") ²⁾
21		_PTABA	String für Auswahl in der Gewindetabelle (z. B. "M 10", "M 12", ...) ²⁾
22		_GMODE	Geometriemode (Bewertung der programmierten Geometriewerte) <hr/> EINER: reserviert <hr/> ZEHNER: reserviert

Nr.	Param Maske	Param intern	Erklärung
23		<code>_DMODE</code>	<p>Displaymode</p> <hr/> <p>EINER: Bearbeitungsebene G17/G18/G19</p> <hr/> <p>0 = Kompatibilität, es bleibt die vor Zyklusaufruf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv)</p> <hr/> <p>ZEHNER: reserviert</p> <hr/> <p>HUNDERTER: reserviert</p> <hr/> <p>TAUSENDER: Kompatibilitätsmode (nur für Eingabemaske zu Rückübersetzen), wenn MD 52216 Bit0 = 1¹⁾</p> <hr/> <p>0 = Technologieparameter werden angezeigt (Kompatibilität): TECHNO-Parameter wirken 1 = Technologieparameter werden nicht angezeigt: Technologie "wie vor Zyklusaufruf programmiert" wirkt</p>
24		<code>_AMODE</code>	<p>Alternativmode</p> <hr/> <p>EINER: Bohrtiefe = Endbohrtiefe Z1 (abs/ink)</p> <hr/> <p>0 = Kompatibilität, aus Programmierung DP/DPR 1 = inkrementell 2 = absolut</p> <hr/> <p>ZEHNER: reserviert</p> <hr/> <p>HUNDERTER: reserviert</p> <hr/> <p>TAUSENDER: Gewindedrehrichtung rechts/links</p> <hr/> <p>0 = Kompatibilität, aus Vorzeichen PIT/MPIT 1 = rechts 2 = links</p> <hr/> <p>ZEHNTAUSENDER: reserviert</p> <hr/> <p>HUNDERTTAUSENDER: reserviert</p> <hr/> <p>EINEMILLION: Rückzugsbetrag nach jeder Bearbeitung V2 manuell/automatisch</p> <hr/> <p>0 = Kompatibilität, aus Programmierung <code>_VRT</code> (> 0 variabler Wert oder ≤ 0 Standardwert 1 mm/0.0394 inch) 1 = automatisch (Standardwert 1mm/0.0394 inch) 2 = manuell (wie unter V2 programmiert)</p>

¹⁾ Felder Technologie können ausgeblendet sein in Abhängigkeit von Settingdatum
`SD52216 $MCS_FUNCTION_MASK_DRILL`

²⁾ Die Parameter 19, 20, und 21 werden nur bei der Gewindeauswahl in den Gewindetabellen der Eingabemaske verwendet. Ein Zugriff auf die Gewindetabellen über Zyklusdefinition bei Zykluslaufzeit ist nicht möglich.

16.1.8 Gewindebohren mit Ausgleichsfutter - CYCLE840

Programmierung

```
CYCLE840 (REAL RTP, REAL RFP, REAL SDIS, REAL DP, REAL DPR, REAL DTB,
INT SDR, INT SDAC, INT ENC, REAL MPIT, REAL PIT, INT _AXN, INT
_PITA, INT _TECHNO, STRING[15] _PITM, STRING[5] _PTAB, STRING[20]
_PTAB, INT _GMODE, INT _DMODE, INT _AMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	RP	RTP	Rückzugsebene (abs)
2	Z0	RFP	Bezugspunkt (abs)
3	SC	SDIS	Sicherheitsabstand (additiv auf Bezugspunkt, ohne Vorzeichen einzugeben)
4	Z1	DP	Bohrtiefe (abs), siehe _AMODE
5	Z1	DPR	Bohrtiefe (ink) , siehe _AMODE
6	DT	DTB	Verweilzeit auf Bohrtiefe/auf Sicherheitsabstand nach Rückzug in Sekunden, siehe ENC
7		SDR	Drehrichtung für den Rückzug
8	SDE	SDAC	Drehrichtung nach Zyklusende
9		ENC	Gewindebohren mit Spindelgeber (G33)/Gewindebohren ohne Spindelgeber(G63) 0 = mit Spindelgeber - Steigung aus MPIT/PIT - ohne DT 20 = mit Spindelgeber - Steigung aus MPIT/PIT - mit DT nach Rückzug auf Sicherheitsabstand 11 = ohne Spindelgeber - Steigung aus MPIT/PIT - mit DT auf Bohrtiefe 1 = ohne Spindelgeber - Steigung aus programmierten Vorschub - mit DT auf Bohrtiefe (Vorschub = Drehzahl · Steigung)
10		MPIT	Gewindegröße nur für "ISO metrisch" (Steigung wird intern während der Laufzeit berechnet) Wertebereich: 3 bis 48 (für M3 bis M48), alternativ zu PIT
11		PIT	Gewindesteigung als Wert, Maßeinheit siehe _PITA) Wertebereich: > 0, alternativ zu MPIT
12		_AXN	Bohrachse: 0 = 3. Geometrieachse 1 = 1. Geometrieachse 2 = 2. Geometrieachse ≥ 3 = 3. Geometrieachse
13		_PITA	Maßeinheit der Gewindesteigung (Auswertung PIT und MPIT) 0 = Steigung in mm - Auswertung MPIT/PIT 1 = Steigung in mm - Auswertung PIT 2 = Steigung in TPI - Auswertung PIT (Gewindegänge pro inch) 3 = Steigung in inch - Auswertung PIT 4 = MODUL - Auswertung PIT

Nr.	Param Maske	Param intern	Erklärung
14		<code>_TECHNO</code>	Technologie ¹⁾ EINER: Genauhaltverhalten 0 = Genauhalt wie vor Zyklusaufruf aktiv 1 = Genauhalt G601 2 = Genauhalt G602 3 = Genauhalt G603 ZEHNER: Vorsteuerung 0 = mit/ohne Vorsteuerung wie vor Zyklusaufruf aktiv 1 = mit Vorsteuerung FFWON 2 = ohne Vorsteuerung FFWOF
15		<code>_PITM</code>	String als Merker für Gewindesteigungseingabe ²⁾
16		<code>_PTAB</code>	String für Gewindetabelle ("", "ISO", "BSW", "BSP", "UNC") ²⁾
17		<code>_PTABA</code>	String für Auswahl in der Gewindetabelle (z.B. "M 10", "M 12", ...) ²⁾
18		<code>_GMODE</code>	reserviert
19		<code>_DMODE</code>	Displaymode EINER: Bearbeitungsebene G17/G18/G19 0 = Kompatibilität, es bleibt die vor Zyklusaufruf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv) ZEHNER: reserviert HUNDERTER: reserviert TAUSENDER: Kompatibilitätsmode (nur für Eingabemaske zu Rückübersetzen), wenn MD 52216 Bit0 = 1 ¹⁾ 0 = Technologieparameter werden angezeigt (Kompatibilität): TECHNO-Parameter wirken 1 = Technologieparameter werden nicht angezeigt: Technologie "wie vor Zyklusaufruf programmiert" wirkt
20		<code>_AMODE</code>	Alternativmode EINER: Bohrtiefe Z1 (abs/ink) 0 = Kompatibilität, aus Programmierung DP/DPR 1 = inkrementell 2 = absolut

¹⁾ Felder Technologie können ausgeblendet sein in Abhängigkeit von Settingdatum SD52216 MCS_FUNCTION_MASK_DRILL

²⁾ Die Parameter 15, 16, und 17 werden nur bei der Gewindeauswahl in den Gewindetabellen der Eingabemaske verwendet. Ein Zugriff auf die Gewindetabellen über Zyklusdefinition bei Zykluslaufzeit ist nicht möglich!

16.1.9 Bohrgewinde fräsen - CYCLE78

Programmierung

```
CYCLE78 (REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, REAL _ADPR, REAL
_FDPR, REAL _LDPR, REAL _DIAM, REAL _PIT, INT _PITA, REAL _DAM, REAL
_MDEP, INT _VARI, INT _CDIR, REAL _GE, REAL _FFD, REAL _FRDP, REAL
_FFR, REAL _FFP2, INT _FFA, STRING[15] _PITM, STRING[20] _PTAB,
STRING[20] _PTABA, INT _GMODE, INT _DMODE, INT _AMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	RP	_RTP	Rückzugsebene (abs)
2	Z0	_RFP	Bezugspunkt der Werkzeugachse (abs)
3	SC	_SDIS	Sicherheitsabstand (additiv auf Bezugspunkt, ohne Vorzeichen einzugeben)
4	Z1	_DP	Endbohrtiefe (abs/ink), siehe _AMODE
5		_ADPR	Anbohrtiefe mit reduziertem Bohrvorschub (ink), mit VARI ZEHNTAUSEND wirksam
6	D	_FDPR	maximale Tiefenzustellung (ink) D ≥ Z1 ⇒ eine Zustellung auf Endbohrtiefe D < Z1 ⇒ Tiefbohrzyklus mit mehreren Zustellungen und Entspanen
7	ZR	_LDPR	Restbohrtiefe beim Durchbohren (ink), mit Vorschub FR
8	Ø	_DIAM	Nenndurchmesser des Gewindes
9	P	_PIT	Gewindesteigung als Zahlenwert
10		_PITA	Bewertung der Gewindesteigung P 1 = Steigung in mm/U 2 = Steigung in Gänge/Zoll 3 = Steigung in inch/U 4 = Steigung als MODUL
11	DF	_DAM	Betrag/Prozentsatz für jede weitere Zustellung (Degression), siehe _AMODE
12	V1	_MDEP	minimale Zustellung (ink), nur bei Degression wirksam
13		_VARI	Bearbeitungsart EINER: reserviert ZEHNER: 0 = kein Entspanen vor Gewindefräsen (wirkt nur auf Endbohrtiefe) 1 = Entspanen vor Gewindefräsen (wirkt nur auf Endbohrtiefe) HUNDERTER: 0 = Rechtsgewinde 1 = Linksgewinde TAUSENDER: 0 = keine Restbohrtiefe mit Bohrvorschub FR 1 = Restbohrtiefe mit Bohrvorschub FR

Nr.	Param Maske	Param intern	Erklärung
			ZEHNTAUSENDER:
			0 = kein Anbohren mit verringertem Vorschub 1 = Anbohren mit verringertem Vorschub Anbohrvorschub = 0.3 F1, wenn F1 < 0.15 mm/U Anbohrvorschub = 0.1 mm/U, wenn F1 ≥ 0.15 mm/U
14		_CDIR	Fräsrichtung 0 = Gleichlauf 1 = Gegenlauf 4 = Gegenlauf + Gleichlauf (Kombination Schruppen + Schlichten)
15	Z2	_GE	Rückzugsbetrag vor Gewindefräsen (ink)
16	F1	_FFD	Bohrvorschub (mm/min bzw. in/min oder mm/U)
17	FR	_FRDP	Bohrvorschub für Restbohrtiefe (mm/min oder mm/U)
18	F2	-FFR	Vorschub für Gewindefräsen (mm/min oder mm/Zahn)
19	FS	_FFP2	Schlichtvorschub für CDIR=4 (mm/min oder mm/Zahn)
20		_FFA	Bewertung Vorschübe EINER: Bohrvorschub F1 ZEHNER: Bohrvorschub für Restbohrtiefe FR HUNDERTER: Vorschub für Gewindefräsen F2 TAUSENDER: Schlichtvorschub FS
21		_PITM	String als Merker für Gewindesteigungseingabe (nur für die Oberfläche) ¹⁾
22		_PTAB	String für Gewindetabelle ("", "ISO", "BSW", "BSP", "UNC") (nur für die Oberfläche) ¹⁾
23		_PTABA	String für Auswahl in der Gewindetabelle (z.B. "M 10", "M 12", ...) (nur für die Oberfläche) ¹⁾
24		_GMODE	Geometriemode, reserviert
25		_DMODE	Displaymode EINER: Bearbeitungsebene G17/18/19 0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv)
26		_AMODE	Alternativmode EINER: Bohrtiefe = Endbohrtiefe Z1 abs/ink 0 = absolut 1 = inkrementell ZEHNER: Betrag/Prozentsatz DF für jede weitere Zustellung (Degression) 0 = Betrag 1 = Prozentsatz (0.001 bis 100 %)

Hinweis

¹⁾ Die Parameter 21, 22 und 23 werden nur bei der Gewindeauswahl in den Gewindetabellen der Eingabemaske verwendet. Ein Zugriff auf die Gewindetabellen über Zyklusdefinition bei Zykluslaufzeit ist nicht möglich.

16.1.10 Beliebige Positionen- CYCLE802

Programmierung

```
CYCLE802 (INT _XA, INT _YA, REAL _X0, REAL _Y0, REAL _X1, REAL _Y1,
REAL _X2, REAL _Y2, REAL _X3, REAL _Y3, REAL _X4, REAL _Y4, REAL
_X5, REAL _Y5, REAL _X6, REAL _Y6, REAL _X7, REAL _Y7, REAL _X8,
REAL _Y8, INT _VARI, INT _UMODE, INT _DMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1		_XA	Alternative für alle X-Positionen (9-stelliger Dezimalwert) Stellenzahl: 876543210 (Die Stelle entspricht der Bohrposition Xn) Stellenwert: 1 = absolut (1. programmierte Position immer absolut!) 2 = inkrementell
2		_YA	Alternative für alle Y-Positionen (9-stelliger Dezimalwert) Stellenzahl: 876543210 (Die Stelle entspricht der Bohrposition Yn) Stellenwert: 1 = Position eingegeben (abs) 2 = Position eingegeben (ink)
3	X0	_X0	1. Position X
4	Y0	_Y0	1. Position Y
5	X1	_X1	2. Position X
6	Y1	_Y1	2. Position Y
7	X2	_X2	3. Position X
8	Y2	_Y2	3. Position Y
9	X3	_X3	4. Position X
10	Y3	_Y3	4. Position Y
11	X4	_X4	5. Position X
12	Y4	_Y4	5. Position Y
13	X5	_X5	6. Position X
14	Y5	_Y5	6. Position Y
15	X6	_X6	7. Position X
16	Y6	_Y6	7. Position Y
17	X7	_X7	8. Position X
18	Y7	_Y7	8. Position Y
19	X8	_X8	9. Position X
20	Y8	_Y8	9. Position Y

Nr.	Param Maske	Param intern	Erklärung
21		_VARI	reserviert
22		_UMODE	reserviert
23		_DMODE	Displaymode
			<u>EINER: Bearbeitungsebene G17/18/19</u>
			0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebene aktiv
			1 = G17 (nur im Zyklus aktiv)
			2 = G18 (nur im Zyklus aktiv)
			3 = G19 (nur im Zyklus aktiv)

Hinweis

Nicht benötigte Positionen der Parameter X1/Y1 bis X8/Y8 können weggelassen werden.

Die Alternativwerte für _XA, und _YA sind jedoch vollständig für alle 9 Positionen anzugeben.

16.1.11 Lochreihe - HOLES1

Programmierung

```
HOLES1 (REAL SPCA, REAL SPCO, REAL STA1, REAL FDIS, REAL DBH, INT
NUM, INT _VARI, INT _UMODE, STRING[200] _HIDE, INT _NSP, INT _DMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	X0	SPCA	Bezugspunkt für Lochreihe in der 1. Achse (abs)
2	Y0	SPCO	Bezugspunkt für Lochreihe in der 2. Achse (abs)
3	α0	STA1	Grund-Drehwinkel (Winkel zur 1. Achse)
4	L0	FDIS	Abstand der 1. Bohrung vom Bezugspunkt
5	L	DBH	Abstand zwischen den Bohrungen
6	N	NUM	Anzahl der Bohrungen
7		_VARI	reserviert
8		_UMODE	reserviert
9		_HIDE	Ausgeblendete Positionen <ul style="list-style-type: none"> • max. 198 Zeichen • Angabe der fortlaufenden Positionsnummer, z. B. "1,3" (Positionen 1 und 3 werden nicht ausgeführt)
10		_NSP	reserviert
11		_DMODE	Displaymode EINER: Bearbeitungsebene G17/18/19 0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv)

16.1.12 Gitter oder Rahmen - CYCLE801

Programmierung

```
CYCLE801 (REAL _SPCA, REAL _SPCO, REAL _STA, REAL _DIS1, REAL _DIS2,
          INT _NUM1, INT _NUM2, INT _VARI, INT _UMODE, REAL _ANG1, REAL _ANG2,
          STRING[200] _HIDE, INT _NSP, INT _DMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	X0	_SPCA	Bezugspunkt für Positionsmuster (Gitter/Rahmen) in der 1. Achse (abs)
2	Y0	_SPCO	Bezugspunkt für Positionsmuster (Gitter/Rahmen) in der 2. Achse (abs)
3	α 0	_STA	Grund-Drehwinkel (Winkel zur 1. Achse) < 0 = Drehung im Uhrzeigersinn > 0 = Drehung im Gegenuhrzeigersinn
4	L1	_DIS1	Abstand der Spalten (Positionsabstand 1. Achse, ohne Vorzeichen einzugeben)
5	L2	_DIS2	Abstand der Zeilen (Positionsabstand 2. Achse, ohne Vorzeichen einzugeben)
6	N1	_NUM1	Anzahl der Spalte
7	N2	_NUM2	Anzahl der Zeilen
8		_VARI	Bearbeitungsart EINER: Positionsmuster 0 = Gitter 1 = Rahmen ZEHNER: reserviert HUNDERTER: reserviert
9		_UMODE	reserviert
10	α X	_ANG1	Scherwinkel zur 1. Achse (Schrägstellung der Zeilen bezogen auf die 1. Achse) < 0 = Messung im Uhrzeigersinn (0 bis -90 Grad) > 0 = Messung im Gegenuhrzeigersinn (0 bis 90 Grad)
11	α Y	_ANG2	Scherwinkel zur 2. Achse (Schrägstellung der Spalten bezogen auf die 2. Achse) < 0 = Messung im Uhrzeigersinn (0 bis -90 Grad) > 0 = Messung im Gegenuhrzeigersinn (0 bis 90 Grad)
12		_HIDE	Ausgeblendete Positionen <ul style="list-style-type: none"> max. 198 Zeichen Angabe der fortlaufenden Positionsnummer, z. B. "1,3" (Positionen 1 und 3 werden nicht ausgeführt)
13		_NSP	reserviert
14		_DMODE	Displaymode EINER: Bearbeitungsebene G17/18/19 0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebenen aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv)

16.1.13 Lochkreis - HOLES2

Programmierung

```
HOLES2 (REAL CPA, REAL CPO, REAL RAD, REAL STA1, REAL INDA, INT NUM,
INT _VARI, INT _UMODE, STRING[200] _HIDE, INT _NSP, INT _DMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	X0	CPA	Mittelpunkt für Lochkreis in der 1. Achse (abs)
2	Y0	CPO	Mittelpunkt für Lochkreis in der 2. Achse (abs)
3	R	RAD	Radius des Lochkreises
4	$\alpha 0$	STA1	Anfangswinkel
5	$\alpha 1$	INDA	Fortschaltwinkel (nur bei Teilkreis) < 0 = Uhrzeigersinn > 0 = Gegenuhrzeigersinn
6	N	NUM	Anzahl der Positionen
7		_VARI	Bearbeitungsart EINER: reserviert ZEHNER: Positionierungsart 0 = Position anfahren - Linear 1 = Position anfahren - auf Kreisbahn HUNDERTER: : reserviert TAUSENDER: Kreismuster 0 = Kompatibilitätsmode, wenn INDA = 0 dann Vollkreis, INDA <> 0 dann Teilkreis 1 = Vollkreis 2 = Teilkreis
8		_UMODE	reserviert
9		_HIDE	Ausgeblendete Positionen <ul style="list-style-type: none"> max. 198 Zeichen Angabe der fortlaufenden Positionsnummer, z. B. "1,3" (Positionen 1 und 3 werden nicht ausgeführt)
10		_NSP	reserviert
11		_DMODE	Displaymode EINER: Bearbeitungsebene G17/18/19 0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv)

16.1.14 Planfräsen - CYCLE61

Programmierung

```
CYCLE61 (REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, REAL _PA, REAL
_PO, REAL _LENG, REAL _WID, REAL _MID, REAL _MIDA, REAL _FALD, REAL
_FFP1, INT _VARI, INT _LIM, INT _DMODE, INT _AMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	RP	_RTP	Rückzugsebene (abs)
2	Z0	_RFP	Bezugspunkt der Werkzeugachse, Höhe Rohteil (abs)
3	SC	_SDIS	Sicherheitsabstand (additiv auf Bezugspunkt, ohne Vorzeichen einzugeben)
4	Z1	_DP	Höhe Fertigteil (abs/ink), siehe _AMODE
5	X0	_PA	Eckpunkt 1 in der 1. Achse (abs)
6	Y0	_PO	Eckpunkt 1 in der 2. Achse (abs)
7	X1	_LENG	Eckpunkt 2 in der 1. Achse (abs/ink), siehe _AMODE
8	Y1	_WID	Eckpunkt 2 in der 2. Achse (abs/ink), siehe _AMODE
9	DZ	_MID	maximale Tiefenzustellung
10	DXY	_MIDA	maximale Ebenenzustellung (Einheit, siehe _AMODE)
11	UZ	_FALD	Schlichtaufmaß Tiefe
12	F	_FFP1	Bearbeitungsvorschub
13		_VARI	Bearbeitungsart
			EINER: Bearbeitung
			1 = Schruppen
			2 = Schlichten
			ZEHNER: Bearbeitungsrichtung
			1 = parallel zur 1. Achse, eine Richtung
			2 = parallel zur 2. Achse, eine Richtung
			3 = parallel zur 1. Achse, wechselnde Richtung
			4 = parallel zur 2. Achse, wechselnde Richtung

Nr.	Param Maske	Param intern	Erklärung
14		_LIM	<p>Eingrenzungen</p> <p>EINER: Eingrenzung 1. Achse minus</p> <p>0 = nein</p> <p>1 = ja</p> <p>ZEHNER: Eingrenzung 1. Achse plus</p> <p>0 = nein</p> <p>1 = ja</p> <p>HUNDERTER: Eingrenzung 2. Achse minus</p> <p>0 = nein</p> <p>1 = ja</p> <p>TAUSENDER: Eingrenzung 2. Achse plus</p> <p>0 = nein</p> <p>1 = ja</p>
15		_DMODE	<p>Displaymode</p> <p>EINER: Bearbeitungsebene G17/18/19</p> <p>0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebene aktiv</p> <p>1 = G17 (nur im Zyklus aktiv)</p> <p>2 = G18 (nur im Zyklus aktiv)</p> <p>3 = G19 (nur im Zyklus aktiv)</p>
16		_AMODE	<p>Alternativmode</p> <p>EINER: Endtiefe (_DP)</p> <p>0 = absolut</p> <p>1 = inkrementell</p> <p>ZEHNER: Einheit für Ebenenzustellung (_MIDA)</p> <p>0 = mm</p> <p>1 = % vom Werkzeugdurchmesser</p> <p>HUNDERTER: reserviert</p> <p>TAUSENDER: Länge der Fläche</p> <p>0 = inkrementell</p> <p>1 = absolut</p> <p>ZEHNTAUSENDER: Breite der Fläche</p> <p>0 = inkrementell</p> <p>1 = absolut</p>

16.1.15 Rechtecktasche fräsen - POCKET3

Programmierung

```
POCKET3 (REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, REAL _LENG, REAL
_WID, REAL _CRAD, REAL _PA, REAL _PO, REAL _STA, REAL _MID, REAL
_FAL, REAL _FALD, REAL _FFP1, REAL _FFD, INT _CDIR, INT _VARI, REAL
_MIDA, REAL _AP1, REAL _AP2, REAL _AD, REAL _RAD1, REAL _DP1, INT
_UMODE, REAL _FS, REAL _ZFS, INT _GMODE, INT _DMODE, INT _AMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	RP	_RTP	Rückzugsebene (abs)
2	Z0	_RFP	Bezugspunkt der Werkzeugachse (abs)
3	SC	_SDIS	Sicherheitsabstand (additiv auf Bezugspunkt, ohne Vorzeichen einzugeben)
4	Z1	_DP	Taschentiefe (abs/ink), siehe _AMODE)
5	L	_LENG	Taschenlänge (ink, mit Vorzeichen einzugeben)
6	W	_WID	Taschenbreite (ink, mit Vorzeichen einzugeben)
7	RP	_CRAD	Eckenradius der Tasche
8	X0	_PA	Bezugspunkt, 1. Achse (abs)
9	YO	_PO	Bezugspunkt, 2. Achse (abs)
10	$\alpha 0$	_STA	Drehwinkel, Winkel zwischen Längsachse (L) und 1. Achse
11	DZ	_MID	maximale Tiefenzustellung
12	UXY	_FAL	Schlichtaufmaß Ebene
13	UZ	_FALD	Schlichtaufmaß Tiefe
14	F	_FFP1	Vorschub in der Ebene
15	FZ	_FFD	Zustellvorschub Tiefe
16		_CDIR	Fräsrichtung: 0 = Gleichlauf 1 = Gegenlauf
17		_VARI	Bearbeitungsart EINER: 1 = Schruppen 2 = Schlichten 4 = Schlichten Rand 5 = Anfasen ZEHNER: 0 = vorgebohrt, Zustellung mit G0 1 = senkrecht, Zustellung mit G1 2 = helikal 3 = pendeln auf Taschenlängsachse HUNDERTER: reserviert

Zyklen extern programmieren

16.1 Technologische Zyklen

Nr.	Param Maske	Param intern	Erklärung
18	DXY	_MIDA	maximale Ebenenzustellung, Einheit siehe _AMODE
19	L1	_AP1	Länge der Vorbearbeitung (ink)
20	W1	_AP2	Breite der Vorbearbeitung (ink)
21	AZ	_AD	Tiefe der Vorbearbeitung (ink)
22	ER EW	_RAD1	Radius der Helixbahn beim Eintauchen helikal maximaler Eintauchwinkel für pendelnd
23	EP	_DP1	Helixsteigung bei Eintauchen helikal
24		_UMODE	reserviert
25	FS	_FS	Fasenbreite (ink)
26	ZFS	_ZFS	Eintauchtiefe (Werkzeugspitze) bei Anfasen (abs/ink), siehe _AMODE
27		_GMODE	Geometriemode EINER: reserviert ZEHNER: reserviert HUNDERTER: Auswahl Bearbeitung/nur Startpunktberechnung 0 = Kompatibilitätsmode 1 = normale Bearbeitung TAUSENDER: Bemaßung über Mitte/Ecke 0 = Kompatibilitätsmode 1 = Bemaßung über Mitte 2 = Bemaßung Eckpunkt, Taschenlage +LENG/+WID 3 = Bemaßung Eckpunkt, Taschenlage -LENG/+WID 4 = Bemaßung Eckpunkt, Taschenlage +LENG/-WID 5 = Bemaßung Eckpunkt, Taschenlage -LENG/-WID ZEHNTAUSENDER: Komplettbearbeitung/Nachbearbeitung 0 = Kompatibilitätsmode (_AP1, _AP2 und _AD wie bisher behandeln) 1 = Komplettbearbeitung 2 = Nachbearbeitung
28		_DMODE	Displaymode EINER: Bearbeitungsebene G17/G18/G19 0 = Kompatibilität, es bleibt die vor Zyklusauf Ruf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv) ZEHNER: Vorschubart: G-Gruppe (G94/G95) für Flächen- und Tiefenvorschub 0 = Kompatibilitätsmode 1 = G-Code wie vor Zyklusauf Ruf. G94/G95 für Flächen- und Tiefenvorschub gleich

Nr.	Param Maske	Param intern	Erklärung
29		_AMODE	Alternativmode
			EINER: Taschentiefe (Z1)
			0 = absolut (Kompatibilitätsmode)
			1 = inkrementell
			ZEHNER: Einheit für Ebenenzustellung (DXY)
			0 = mm
			1 = % vom Werkzeugdurchmesser
			HUNDERTER: Eintauchtiefe bei Anfasen (ZFS)
			0 = absolut
			1 = inkrementell

16.1.16 Kreistasche fräsen - POCKET4

Programmierung

```
POCKET4 (REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, REAL _CDIAM,
REAL _PA, REAL _PO, REAL _MID, REAL _FAL, REAL _FALD, REAL _FFP1,
REAL _FFD, INT _CDIR, INT _VARI, REAL _MIDA, REAL _AP1, REAL _AD,
REAL _RAD1, REAL _DP1, INT _UMODE, REAL _FS, REAL _ZFS, INT _GMODE,
INT _DMODE, INT _AMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	RP	_RTP	Rückzugsebene (abs)
2	Z0	_RFP	Bezugspunkt der Werkzeugachse (abs)
3	SC	_SDIS	Sicherheitsabstand (additiv auf Bezugspunkt, ohne Vorzeichen einzugeben)
4	Z1	_DP	Taschentiefe (abs/ink), siehe _AMODE
5	Ø	_DIAM	Taschendurchmesser oder Taschenradius, siehe _DMODE
6	X0	_PA	Bezugspunkt 1. Achse (abs)
7	Y0	_PO	Bezugspunkt 2. Achse (abs)
8	DZ	_MID	maximale Tiefenzustellung, siehe _VARI = ebenenweise maximale Helixsteigung, siehe _VARI = helikal
9	UXY	_FAL	Schlichtaufmaß Ebene
10	UZ	_FALD	Schlichtaufmaß Tiefe
11	F	_FFP1	Vorschub für Flächenbearbeitung
12	FZ	_FFD	Zustellvorschub Tiefe

Zyklen extern programmieren

16.1 Technologische Zyklen

Nr.	Param Maske	Param intern	Erklärung
13		_CDIR	Fräsrichtung 0 = Gleichlauf 1 = Gegenlauf
14		_VARI	Bearbeitungsart EINER: 1 = Schruppen 2 = Schlichten 4 = Schlichten Rand 5 = Anfassen ZEHNER: Zustellungsart (Schruppen und Schlichten) 0 = vorgebohrt, Zustellung mit G0 (Tasche ist vorgearbeitet) 1 = senkrecht Zustellung mit G1 2 = helikal HUNDERT: reserviert TAUSENDER: 0 = ebenenweise 1 = helikal
15	DXY	_MIDA	maximale Ebenenzustellung, siehe _AMODE, $0 = 0,8 \cdot \text{WZG-Durchmesser}$
16	Ø	_AP1	Durchmesser/Radius der Vorbearbeitung (ink)
17	AZ	_AD	Tiefe der Vorbearbeitung (ink)
18	ER	_RAD1	Radius der Helixbahn beim Eintauchen helikal
19	EP	_DP1	Helixsteigung bei Eintauchen auf Helixbahn
20		_UMODE	reserviert
21	FS	_FS	Fasenbreite (ink)
22	ZFS	_ZFS	Eintauchtiefe (Werkzeugspitze) bei Anfassen (abs/ink), siehe _AMODE
23		_GMODE	Geometriemode EINER: reserviert ZEHNER: reserviert HUNDERTER: Bearbeitung/Startpunktberechnung 0 = Kompatibilitätsmode 1 = normale Bearbeitung TAUSENDER: reserviert ZEHNTAUSENDER: Komplettbearbeitung/Nachbearbeitung 0 = Kompatibilitätsmode (_AP1 und _AD wie bisher behandeln) 1 = Komplettbearbeitung 2 = Nachbearbeitung

Nr.	Param Maske	Param intern	Erklärung
24		_DMODE	<p>Displaymode</p> <hr/> <p>EINER: Bearbeitungsebene G17/18/19</p> <hr/> <p>0 = Kompatibilität, es bleibt die vor Zyklusauf Ruf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv)</p> <hr/> <p>ZEHNER: Vorschubart: G-Gruppe (G94/G95) für Flächen- und Tiefenvorschub</p> <hr/> <p>0 = Kompatibilitätsmode 1 = G-Code wie vor Zyklusauf Ruf. G94/G95 für Flächen- und Tiefenvorschub gleich</p> <hr/> <p>HUNDERTER:</p> <hr/> <p>0 = Kompatibilitätsmode (_CDIAM/_AP1 als Radius eingegeben) 1 = _CDIAM/_AP1 als Durchmesser eingegeben</p>
25		_AMODE	<p>Alternativmode</p> <hr/> <p>EINER: Taschentiefe (Z1)</p> <hr/> <p>0 = absolut (Kompatibilitätsmode) 1 = inkrementell</p> <hr/> <p>ZEHNER: Einheit für Zustellbreite (DXY)</p> <hr/> <p>0 = mm 1 = % vom Werkzeugdurchmesser</p> <hr/> <p>HUNDERTER: Eintauchtiefe bei Anfasen (ZFS)</p> <hr/> <p>0 = absolut 1 = inkrementell</p>

16.1.17 Rechteckzapfen fräsen - CYCLE76

Programmierung

```
CYCLE76 (REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, REAL _DPR, REAL
_LENG, REAL _WID, REAL _CRAD, REAL _PA, REAL _PO, REAL _STA, REAL
_MID, REAL _FAL, REAL _FALD, REAL _FFP1, REAL _FFD, INT _CDIR, INT
_VARI, REAL _AP1, REAL _AP2, REAL _FS, REAL _ZFS, INT _GMODE, INT
_DMODE, INT _AMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	RP	_RTP	Rückzugsebene (abs)
2	Z0	_RFP	Bezugspunkt der Werkzeugachse (abs)
3	SC	_SDIS	Sicherheitsabstand (additiv auf Bezugspunkt, ohne Vorzeichen einzugeben)
4	Z1	_DP	Zapfentiefe(abs)
5		_DPR	Zapfentiefe (ink), bezogen auf Z0 (ohne Vorzeichen einzugeben)
6	L	_LENG	Zapfenlänge, siehe _GMODE(ohne Vorzeichen einzugeben)
7	W	_WID	Zapfenbreite, siehe _GMODE (ohne Vorzeichen einzugeben)
8	R	_CRAD	Eckenradius des Zapfen (ohne Vorzeichen einzugeben)
9	X0	_PA	Bezugspunkt Zapfen, 1. Achse der Ebene (abs)
10	Y0	_PO	Bezugspunkt Zapfen, 2. Achse der Ebene (abs)
11	$\alpha 0$	_STA	Drehwinkel, Winkel zwischen Längsachse (L) und 1. Achse der Ebene
12	DZ	_MID	maximale Tiefenzustellung (ink, ohne Vorzeichen einzugeben)
13	UXY	_FAL	Schlichtaufmaß Ebene (ink), Aufmaß an der Randkontur
14	UZ	_FALD	Schlichtaufmaß Tiefe (ink), Aufmaß am Grund (ohne Vorzeichen einzugeben)
15	FX	_FFP1	Vorschub an der Kontur
16	FZ	_FFD	Zustellvorschub Tiefe
17		_CDIR	Fräsrichtung (ohne Vorzeichen einzugeben)
			EINER:
			0 = Gleichlauf
			1 = Gegenlauf
18		_VARI	Bearbeitung
			EINER:
			1 = Schruppen
			2 = Schlichten
			5 = Anfasen
19	L1	_AP1	Länge des Rohteilzapfens
20	W1	_AP2	Breite des Rohteilzapfens
21	FS	_FS	Fasenbreite (ink)
22	ZFS	_ZFS	Eintauchtiefe (Werkzeugspitze) bei Anfasen (abs, ink), siehe _AMODE

Nr.	Param Maske	Param intern	Erklärung
23		<code>_GMODE</code>	<p>Mode für Bewertung der programmierten Geometriewerte</p> <hr/> <p>EINER: reserviert</p> <hr/> <p>ZEHNER: reserviert</p> <hr/> <p>HUNDERTER: Auswahl Bearbeitung oder nur Startpunktberechnung</p> <hr/> <p>0 = Kompatibilitätsmode 1 = normale Bearbeitung</p> <hr/> <p>TAUSENDER: Bemaßung des Zapfens über Mitte oder Ecke</p> <hr/> <p>0 = Kompatibilitätsmode 1 = Bemaßung über Mitte 2 = Bemaßung Eckpunkt, Zapfen +L +W 3 = Bemaßung Eckpunkt, Zapfen -L +W 4 = Bemaßung Eckpunkt, Zapfen +L -W 5 = Bemaßung Eckpunkt, Zapfen -L -W</p> <hr/> <p>ZEHNTAUSENDER: Komplettbearbeitung oder Nachbearbeitung</p> <hr/> <p>0 = Kompatibilitätsmode 1 = Komplettbearbeitung 2 = Nachbearbeitung</p>
24		<code>_DMODE</code>	<p>Displaymode</p> <hr/> <p>EINER: Bearbeitungsebene G17/G18/G19</p> <hr/> <p>0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv)</p>
25		<code>_AMODE</code>	<p>Alternativmode</p> <hr/> <p>EINER: Endtiefe Z1 (abs/ink)</p> <hr/> <p>0 = Kompatibilität 1 = Z1 (ink) 2 = Z1 (abs)</p> <hr/> <p>ZEHNER: reserviert</p> <hr/> <p>HUNDERTER: Eintauchtiefe bei Anfasen ZFS</p> <hr/> <p>0 = ZFS (abs) 1 = ZFS (ink)</p>

16.1.18 Kreiszapfen fräsen - CYCLE77

Programmierung

```
CYCLE77 (REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, REAL _DPR, REAL
_CDIAM, REAL _PA, REAL _PO, REAL _MID, REAL _FAL, REAL _FALD, REAL
_FFP1, REAL _FFD, INT _CDIR, INT _VARI, REAL _AP1, REAL _FS, REAL
_ZFS, INT _GMODE, INT _DMODE, INT _AMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	RP	_RTP	Rückzugsebene (abs)
2	Z0	_RFP	Bezugspunkt der Werkzeugachse (abs)
3	SC	_SDIS	Sicherheitsabstand (additiv auf Bezugspunkt, ohne Vorzeichen einzugeben)
4	Z1	_DP	Zapftiefe(abs)
5		_DPR	Zapftiefe (ink), bezogen auf Z0 (ohne Vorzeichen einzugeben)
6	Ø	_CDIAM	Durchmesser des Zapfens (ohne Vorzeichen einzugeben)
7	X0	_PA	Bezugspunkt Zapfen, 1. Achse der Ebene (abs)
8	Y0	_PO	Bezugspunkt Zapfen, 2. Achse der Ebene (abs)
9	DZ	_MID	maximale Tiefenzustellung (ink, ohne Vorzeichen einzugeben)
10	UXY	_FAL	Schlichtaufmaß Ebene (ink), Aufmaß an der Randkontur
11	UZ	_FALD	Schlichtaufmaß Tiefe (ink), Aufmaß am Grund (ohne Vorzeichen einzugeben)
12	FX	_FFP1	Vorschub an der Kontur
13	FZ	_FFD	Zustellvorschub Tiefe
14		_CDIR	Fräsrichtung (ohne Vorzeichen einzugeben)
			EINER:
			0 = Gleichlauf
			1 = Gegenlauf
15		_VARI	Bearbeitung
			EINER:
			1 = Schruppen bis auf Schlichtaufmaß
			2 = Schlichten (Aufmaß X/Y/Z = 0)
			5 = Anfasen
16	Ø1	_AP1	Durchmesser des Rohteilzapfens
17	FS	_FS	Fasenbreite (ink)
18	ZFS	_ZFS	Eintauchtiefe (Werkzeugspitze) bei Anfasen (abs/ink) siehe _AMODE)

Nr.	Param Maske	Param intern	Erklärung
19		<code>_GMODE</code>	<p>Mode für Bewertung der programmierten Geometriewerte</p> <hr/> <p>EINER: reserviert</p> <hr/> <p>ZEHNER: reserviert</p> <hr/> <p>HUNDERTER: Auswahl Bearbeitung/nur Startpunktberechnung</p> <hr/> <p>0 = Kompatibilitätsmode 1 = normale Bearbeitung</p> <hr/> <p>TAUSENDER: reserviert</p> <hr/> <p>ZEHNTAUSENDER: Komplettbearbeitung/Nachbearbeitung</p> <hr/> <p>0 = Kompatibilitätsmode (<code>_AP1</code> wie bisher behandeln) 1 = Komplettbearbeitung 2 = Nachbearbeitung</p>
20		<code>_DMODE</code>	<p>Displaymode</p> <hr/> <p>EINER: Bearbeitungsebene G17/G18/G19</p> <hr/> <p>0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebenen aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv)</p>
21		<code>_AMODE</code>	<p>Alternativmode</p> <hr/> <p>EINER: Endtiefe Z1 (abs/ink)</p> <hr/> <p>0 = Kompatibilität 1 = Z1 (ink) 2 = Z1 (abs)</p> <hr/> <p>ZEHNER: reserviert</p> <hr/> <p>HUNDERTER: Eintauchtiefe bei Anfasen ZFS</p> <hr/> <p>0 = ZFS (abs) 1 = ZFS (ink)</p>

16.1.19 Mehrkant - CYCLE79

Programmierung

```
CYCLE79 (REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, INT _NUM, REAL
_SWL, REAL _PA, REAL _PO, REAL _STA, REAL _RC, REAL _AP1, REAL
_MIDA, REAL _MID, REAL _FAL, REAL _FALD, REAL _FFP1, INT _CDIR, INT
_VARI, REAL _FS, REAL _ZFS, INT _GMODE, INT _DMODE, INT _AMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	RP	_RTP	Rückzugsebene (abs)
2	Z0	_RFP	Bezugspunkt der Werkzeugachse (abs)
3	SC	_SDIS	Sicherheitsabstand (additiv auf Bezugspunkt, ohne Vorzeichen einzugeben)
4	Z1	_DP	Mehrkanttiefe (abs/ink), siehe _AMODE
5	N	_NUM	Anzahl der Kanten (1...n)
6	SW/L	_SWL	Schlüsselweite oder Kantenlänge (je nach _VARI) ("SW" bei Schlüsselweite, "L" bei Kantenlänge) Schlüsselweite nur bei gerader Anzahl Kanten, und Einkant
7	X0	_PA	Zapfenbezugspunkt, 1. Achse (abs)
8	Y0	_PO	Zapfenbezugspunkt, 2. Achse (abs)
9	$\alpha 0$	_STA	Drehwinkel Kantenmitte gegen 1. Achse (X-Achse)
10	R1/FS1	_RC	Eckenverrundung bei _NUM > 2 (Radius/Fase, siehe _AMODE) (ink, ohne Vorzeichen einzugeben) ("R1" bei Radius, "FS1" bei Fase)
11	\emptyset	_AP1	Rohdurchmesser des Zapfens
12	DXY	_MIDA	maximale Zustellbreite (Einheit, siehe _AMODE)
13	DZ	_MID	maximale Tiefenzustellung
14	UXY	_FAL	Schlichtaufmaß Ebene
15	UZ	_FALD	Schlichtaufmaß Tiefe
16	F	_FFP1	Bearbeitungsvorschub
17		_CDIR	Fräsrichtung 0 = Gleichlauf 1 = Gegenlauf

Nr.	Param Maske	Param intern	Erklärung
18		<code>_VARI</code>	<p>Bearbeitungsart</p> <hr/> <p>EINER: Bearbeitung</p> <hr/> <p>1 = Schruppen 2 = Schlichten 3 = Schlichten Rand 5 = Anfasen</p> <hr/> <p>ZEHNER: Schlüsselweite oder Kantenlänge</p> <hr/> <p>0 = Schlüsselweite 1 = Kantenlänge</p>
19	<code>FS</code>	<code>_FS</code>	Fasenbreite (ink)
20	<code>ZFS</code>	<code>_ZFS</code>	Eintauchtiefe (Werkzeugspitze) bei Anfasen (abs/ink), siehe <code>_AMODE</code>
21		<code>_GMODE</code>	<p>Geometriemode</p> <hr/> <p>EINER: reserviert</p> <hr/> <p>ZEHNER: reserviert</p> <hr/> <p>HUNDERTER: Bearbeitung/Startpunktberechnung</p> <hr/> <p>1 = normale Bearbeitung</p>
22		<code>_DMODE</code>	<p>Displaymode</p> <hr/> <p>EINER: Bearbeitungsebene G17/G18/G19</p> <hr/> <p>0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv)</p>
23		<code>_AMODE</code>	<p>Alternativmode</p> <hr/> <p>EINER: Endtiefe (<code>_DP</code>)</p> <hr/> <p>0 = absolut 1 = inkrementell</p> <hr/> <p>ZEHNER: Einheit für Ebenenzustellung (<code>_MIDA</code>)</p> <hr/> <p>0 = mm 1 = % vom Werkzeugdurchmesser</p> <hr/> <p>HUNDERTER: Eintauchtiefe bei Anfasen (<code>_ZFS</code>)</p> <hr/> <p>0 = absolut 1 = inkrementell</p> <hr/> <p>TAUSENDER: Eckenverrundung (<code>_RC</code>)</p> <hr/> <p>0 = Radius 1 = Fase</p>

16.1.20 Längsnut - SLOT1

Programmierung

```
SLOT1 (REAL RTP, REAL RFP, REAL SDIS, REAL _DP, REAL _DPR, INT NUM,
REAL LENG, REAL WID, REAL _CPA, REAL _CPO, REAL RAD, REAL STA1, REAL
INDA, REAL FFD, REAL FFP1, REAL _MID, INT CDIR, REAL _FAL, INT VARI,
REAL _MIDF, REAL FFP2, REAL SSF, REAL _FALD, REAL _STA2, REAL _DP1,
INT _UMODE, REAL _FS, REAL _ZFS, INT _GMODE, INT _DMODE, INT _AMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	RP	RTP	Rückzugsebene (abs)
2	Z0	RFP	Bezugspunkt der Werkzeugachse (abs)
3	SC	SDIS	Sicherheitsabstand (additiv auf Bezugspunkt, ohne Vorzeichen einzugeben)
4	Z1	_DP	Nuttiefe (abs)
5		_DPR	Nuttiefe (ink), bezogen auf Z0 (ohne Vorzeichen einzugeben)
6		NUM	Anzahl der Nuten = 1
7	L	LENG	Länge der Nut
8	W	WID	Breite der Nut
9	X0	_CPA	Bezugspunkt, 1. Achse der Ebene
10	Y0	_CPO	Bezugspunkt, 2. Achse der Ebene
11		_RAD	reserviert
12	α	STA1	Drehwinkel
13		INDA	reserviert
14	FZ	FFD	Zustellvorschub Tiefe
15	F	FFP1	Vorschub
16	DZ	_MID	maximale Tiefenzustellung
17		CDIR	Fräsrichtung 0 = Gleichlauf 1 = Gegenlauf
18	UXY	_FAL	Schlichtaufmass Ebene oder Nutrand

Nr.	Param Maske	Param intern	Erklärung
19		VARI	<p>Bearbeitungsart</p> <hr/> <p>EINER:</p> <p>0 = reserviert 1 = Schruppen 2 = Schlichten 4 = Schlichten Rand (nur Rand bearbeiten) 5 = Anfasen</p> <hr/> <p>ZEHNER: Anfahren</p> <p>0 = vorgebohrt, Zustellung mit G0 (Nut ist vorbearbeitet) 1 = senkrecht, Zustellung mit G1 2 = helikal 3 = pendelnd</p> <hr/> <p>HUNDERTER: reserviert</p>
20	DZF	MIDF	reserviert
21	FF	FFP2	reserviert
22	SF	SSF	reserviert
23	UZ	_FALD	Schlichtaufmaß Tiefe
24	ER EW	_STA2	<p>Radius der Helixbahn beim Eintauchen helikal</p> <hr/> <p>maximaler Eintauchwinkel für pendelnd</p>
25	EP	_DP1	Eintauchtiefe pro Umdrehung für Helix
26		_UMODE	reserviert
27	FS	_FS	Fasenbreite (ink) bei Anfasen
28	ZFS	_ZFS	Eintauchtiefe (Werkzeugspitze) bei Anfasen (abs/ink), siehe _AMODE)
29		_GMODE	<p>Geometriemode</p> <hr/> <p>EINER: reserviert</p> <hr/> <p>ZEHNER: reserviert</p> <hr/> <p>HUNDERTER: Auswahl Bearbeitung oder nur Startpunktberechnung</p> <p>1 = normale Bearbeitung</p> <hr/> <p>TAUSENDER: Bemaßung Bezugspunkt, Nutlage</p> <p>0 = Mitte 1 = links innen +L 2 = rechts innen -L 3 = linker Rand +L 4 = rechter Rand -L</p>

Nr.	Param Maske	Param intern	Erklärung
30		_DMODE	<p>Displaymode</p> <hr/> <p>EINER: Bearbeitungsebene G17/18/19</p> <hr/> <p>0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebenen aktiv</p> <p>1 = G17 (nur im Zyklus aktiv)</p> <p>2 = G18 (nur im Zyklus aktiv)</p> <p>3 = G19 (nur im Zyklus aktiv)</p> <hr/> <p>ZEHNER: reserviert</p> <hr/> <p>HUNDERTER: reserviert</p> <hr/> <p>TAUSENDER: Kennung SW-Version</p> <hr/> <p>1 = Funktionserweiterung SLOT1</p>
31		_AMODE	<p>Alternativmode</p> <hr/> <p>EINER: Endtiefe Z1 (abs/ink)</p> <hr/> <p>0 = Kompatibilität</p> <p>1 = Z1 (ink)</p> <p>2 = Z1 (abs)</p> <hr/> <p>ZEHNER: reserviert</p> <hr/> <p>HUNDERTER: Eintauchtiefe bei Anfasen ZFS</p> <hr/> <p>0 = ZFS (abs)</p> <p>1 = ZFS (ink)</p>

Hinweis

Der Zyklus ist gegenüber früheren SW-Versionen mit neuen Funktionen ausgestattet. Das hat zur Folge, dass bestimmte Parameter in der Eingabemaske nicht mehr angezeigt werden (NUM, RAD, INDA). Mehrere Nuten auf einem Positionsmuster sind mittels "MCALL" und Aufruf des gewünschten Positionsmusters, z B HOLES2, programmierbar.

16.1.21 Kreisnut - SLOT2

Programmierung

```
SLOT2 (REAL RTP, REAL RFP, REAL SDIS, REAL _DP, REAL _DPR, INT NUM,
REAL AFSL, REAL WID, REAL _CPA, REAL _CPO, REAL RAD, REAL STA1, REAL
INDA, REAL FFD, REAL FFP1, REAL _MID, INT CDIR, REAL _FAL, INT VARI,
REAL _MIDF, REAL FFP2, REAL SSF, REAL _FFCP, INT _UMODE, REAL _FS,
REAL _ZFS, INT _GMODE, INT _DMODE, INT _AMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	RP	RTP	Rückzugsebene (abs)
2	Z0	RFP	Bezugspunkt der Werkzeugachse (abs)
3	SC	SDIS	Sicherheitsabstand (additiv auf Bezugspunkt, ohne Vorzeichen einzugeben)
4	Z1	_DP	Nuttiefe (abs)
5		_DPR	Nuttiefe (ink), bezogen auf Z0 (ohne Vorzeichen einzugeben)
6	N	NUM	Anzahl der Nuten
7	$\alpha 1$	AFSL	Öffnungswinkel der Nut
8	W	WID	Breite der Nut
9	X0	_CPA	Bezugspunkt = Mittelpunkt des Kreises, 1. Achse der Ebene
10	Y0	_CPO	Bezugspunkt = Mittelpunkt des Kreises, 2. Achse der Ebene
11	R	RAD	Radius des Kreises
12	$\alpha 0$	STA1	Anfangswinkel
13	$\alpha 2$	INDA	Fortschaltwinkel
14	FZ	FFD	Zustellvorschub Tiefe
15	F	FFP1	Vorschub
16	DZ	_MID	maximale Tiefenzustellung
17		CDIR	Fräsrichtung 0 = Gleichlauf 1 = Gegenlauf
18	UXY	_FAL	Schlichtaufmaß Ebene oder Nutrand
19		VARI	Bearbeitungsart EINER: 0 = Komplettbearbeitung 1 = Schruppen 2 = Schlichten 3 = Schlichten Rand 5 = Anfasen

Zyklen extern programmieren

16.1 Technologische Zyklen

Nr.	Param Maske	Param intern	Erklärung
			ZEHNER: 0 = Zwischenpositionieren mit G0-Gerade 1 = Zwischenpositionieren auf Kreisbahn
			HUNDERTER: reserviert
			TAUSENDER: 0 = Kompatibilitätsmode, wenn INDA = 0 dann Vollkreis, INDA <> 0 dann Teilkreis 1 = Vollkreis 2 = Teilkreis
20	DZF	_MIDF	reserviert
21		FFP2	reserviert
22		SSF	reserviert
23	FF	_FFCP	reserviert
24		_UMODE	reserviert
25	FS	_FS	Fasenbreite (Ink)
26	ZFS	_ZFS	Eintauchtiefe (Werkzeugspitze) bei Anfasen (abs/ink), siehe _AMODE)
27		_GMODE	Geometriemode EINER: reserviert ZEHNER: reserviert HUNDERTER: Auswahl Bearbeitung oder nur Startpunktberechnung 0 = Kompatibilitätsmode 1 = normale Bearbeitung
28		_DMODE	Displaymode EINER: Bearbeitungsebene G17/18/19 0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebenen aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv) ZEHNER: reserviert HUNDERTER: reserviert TAUSENDER: Kennung SW-Version 1 = Funktionen SLOT2 ab SW 2.5
29		_AMODE	Alternativmode EINER: Endtiefe Z1 (abs/ink) 0 = Kompatibilität 1 = Z1 (ink) 2 = Z1 (abs) ZEHNER: reserviert HUNDERTER: Eintauchtiefe bei Anfasen ZFS 0 = ZFS (abs) 1 = ZFS (ink)

16.1.22 Offene Nut fräsen - CYCLE899

Programmierung

```
CYCLE899 (REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, REAL _LENG,
REAL _WID, REAL _PA, REAL _PO, REAL _STA, REAL _MID, REAL _MIDA,
REAL _FAL, REAL _FALD, REAL _FFP1, INT _CDIR, INT _VARI, INT _GMODE,
INT _DMODE, INT _AMODE, INT _UMODE, REAL _FS, REAL _ZFS)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	RP	_RTP	Rückzugsebene (abs)
2	Z0	_RFP	Bezugspunkt der Werkzeugachse(abs)
3	SC	_SDIS	Sicherheitsabstand (additiv auf Bezugspunkt, ohne Vorzeichen einzugeben)
4	Z1	_DP	Nuttiefe(abs/ink), siehe _AMODE
5	L	_LENG	Länge der Nut (ink)
6	W	_WID	Breite der Nut (ink)
7	X0	_PA	Bezugspunkt/Startposition 1. Achse (abs)
8	Y0	_PO	Bezugspunkt/Startposition 2. Achse (abs)
9	α0	_STA	Drehwinkel zur 1.Achse
10	DZ	_MID	maximale Zustelltiefe (ink), nur für Wirbelfräsen
11	DXY	_MIDA	maximale Ebenenzustellung, siehe _AMODE
12	UXY	_FAL	Schlichtaufmaß Ebene
13	UZ	_FALD	Schlichtaufmaß Tiefe
14	F	_FFP1	Vorschub
15		_CDIR	Fräsrichtung EINER: 0 = Gleichlauf 1 = Gegenlauf 4 = Wechselweise
16		_VARI	Bearbeitung EINER: 1 = Schruppen 2 = Schlichten 3 = Schlichten Boden 4 = Schlichten Rand 5 = Vorschlichten 6 = Anfassen ZEHNER: reserviert HUNDERTER: reserviert TAUSENDER: 1 = Wirbelfräsen 2 = Tauchfräsen

Zyklen extern programmieren

16.1 Technologische Zyklen

Nr.	Param Maske	Param intern	Erklärung
17		<code>_GMODE</code>	Bewertung der Geometriewerte EINER: reserviert ZEHNER: reserviert HUNDERTER: Auswahl Bearbeitung/nur Startpunktberechnung 1 = normale Bearbeitung TAUSENDER: Bemaßung über Mitte / Kante 0 = Bemaßung über Mitte 1 = Bemaßung über Kante "links" ("- " Richtung der 1.Achse) 2 = Bemaßung über Kante "rechts" ("+" Richtung der 1.Achse)
18		<code>_DMODE</code>	Displaymode EINER: Bearbeitungsebene G17/G18/G19 0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebenen aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv)
19		<code>_AMODE</code>	Alternativmode EINER: Nutttiefe Z1 0 = absolut 1 = inkrementell ZEHNER: Einheit für Ebenenzustellung (<code>_MIDA</code>) DXY 0 = mm 1 = % vom Werkzeugdurchmesser HUNDERTER: Eintauchtiefe bei Anfasen ZFS 0 = absolut 1 = inkrementell
20		<code>_UMODE</code>	reserviert
21	FS	<code>_FS</code>	Fasenbreite (ink)
22	ZFS	<code>_ZFS</code>	Eintauchtiefe (Werkzeugspitze) bei Anfasen (abs/ink), siehe <code>_AMODE</code>)

16.1.23 Langloch - LONGHOLE

Programmierung

```
LONGHOLE (REAL RTP, REAL RFP, REAL SDIS, REAL _DP, REAL _DPR,
          INT NUM, REAL LENG, REAL _CPA, REAL _CPO, REAL RAD, REAL STA1,
          REAL INDA, REAL FFD, REAL FFP1, REAL MID, INT _VARI, INT _UMODE,
          INT _GMODE, INT _DMODE, INT _AMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	RP	RTP	Rückzugsebene (abs)
2	Z0	_RFP	Bezugspunkt der Werkzeugachse (abs)
3	SC	SDIS	Sicherheitsabstand (additiv auf Bezugspunkt, ohne Vorzeichen einzugeben)
4	Z1	_DP	Langlochtiefe (abs)
5		_DPR	Langlochtiefe (ink), bezogen auf Z0 (ohne Vorzeichen einzugeben)
6		NUM	Anzahl der Langlöcher = 1
7	L	LENG	Länge Langloch
8	X0	_CPA	Bezugspunkt, 1. Achse der Ebene
9	Y0	_CPO	Bezugspunkt, 2. Achse der Ebene
10		RAD	reserviert
11	$\alpha 0$	STA1	Drehwinkel
12		INDA	reserviert
13	FZ	FFD	Zustellvorschub Tiefe
14	F	FFP1	Vorschub
15	DZ	MID	maximale Tiefenzustellung
16		_VARI	Bearbeitungsart EINER: Zustellart 1 = senkrecht mit G1 3 = pendelnd HUNDERT: reserviert
17		_UMODE	reserviert

Nr.	Param Maske	Param intern	Erklärung
18		<code>_GMODE</code>	<p>Geometriemode</p> <p>EINER: reserviert</p> <p>ZEHNER: reserviert</p> <p>HUNDERT Auswahl Bearbeitung oder nur Startpunktberechnung</p> <p>0 = Kompatibilitätsmode</p> <p>1 = normale Bearbeitung</p> <p>TAUSENDER: Bemaßung Bezugspunkt, Nutlage</p> <p>0 = Mitte</p> <p>1 = links innen +L</p> <p>2 = rechts innen -L</p> <p>3 = linker Rand +L</p> <p>4 = rechter Rand -L</p>
19		<code>_DMODE</code>	<p>Displaymode</p> <p>EINER: Bearbeitungsebene G17/18/19</p> <p>0 = Kompatibilität, es bleibt die vor Zyklusauf Ruf wirksame Ebene aktiv</p> <p>1 = G17 (nur im Zyklus aktiv)</p> <p>2 = G18 (nur im Zyklus aktiv)</p> <p>3 = G19 (nur im Zyklus aktiv)</p> <p>ZEHNER: Vorschubart: G-Gruppe (G94/G95) für Flächen- und Tiefenvorschub</p> <p>0 = Kompatibilitätsmode</p> <p>1 = G-Code wie vor Zyklusauf Ruf. G94/G95 für Flächen- und Tiefenvorschub gleich</p> <p>HUNDERTER: reserviert</p> <p>TAUSENDER: Kennung SW-Version</p> <p>1 = Funktionserweiterung LONGHOLE (Bemaßung Bezugspunkt)</p>
20		<code>_AMODE</code>	<p>Alternativmode</p> <p>EINER: Endtiefe Z1 (abs/ink)</p> <p>0 = Kompatibilität</p> <p>1 = Z1 (ink)</p> <p>2 = Z1 (abs)</p>

Hinweis

Der Zyklus ist gegenüber früheren SW-Versionen mit neuen Funktionen ausgestattet. Das hat zur Folge, dass bestimmte Parameter in der Eingabemaske nicht mehr angezeigt werden (`NUM`, `RAD`, `INDA`). Mehrere Nuten auf einem Positionsmuster sind mittels "MCALL" und Aufruf des gewünschten Positionsmusters, z. B. HOLES2, programmierbar.

16.1.24 Gewindefräsen - CYCLE70

Programmierung

```
CYCLE70 (REAL _RTP, REAL _RFP, REAL _SDIS, REAL _DP, REAL _DIATH,
REAL _H1, REAL _FAL, REAL _PIT, INT _NT, REAL _MID, REAL _FFR, INT
_TYPH, REAL _PA, REAL _PO, REAL _NSP, INT _VARI, INT _PITA,
STRING[15] _PITM, STRING[20] _PTAB, STRING[20] _PTABA, INT _GMODE,
INT _DMODE, INT _AMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	RP	_RTP	Rückzugsebene (abs)
2	Z0	_RFP	Bezugspunkt der Werkzeugachse (abs)
3	SC	_SDIS	Sicherheitsabstand (additiv auf Bezugspunkt, ohne Vorzeichen einzugeben)
4	Z1	_DP	Gewindelänge (abs, ink), siehe _AMODE Auslauf am Bohrungsgrund berücksichtigen (minimum halbe Steigung)
5	Ø	_DIATH	Nenndurchmesser des Gewindes
6	H1	_H1	Gewindetiefe
7	U	_FAL	Schlichtaufmaß
8	P	_PIT	Gewindesteigung (Auswahl _PITA: mm, inch, MODUL, Gänge/Zoll)
9	NT	_NT	Anzahl Zähne auf der Schneidplatte Werkzeuglänge immer auf unteren Zahn bezogen!
10	DXY	_MID	Maximale Zustellung pro Schnitt _MID > _H1: alles mit einem Schnitt
11	F	_FFR	Fräsvorschub
12		_TYPH	Gewindetyp 0 = Innengewinde 1 = Außengewinde
13	X0	_PA	Kreismittelpunkt, 1. Achse (abs)
14	Y0	_PO	Kreismittelpunkt, 2. Achse (abs)
15	αS	_NSP	Startwinkel (mehrgängige Gewinde)
16		_VARI	Bearbeitungsart EINER: 1 = Schruppen 2 = Schlichten ZEHNER: 1 = von oben nach unten 2 = von unten nach oben HUNDERTER: 0 = Rechtsgewinde 1 = Linksgewinde

Zyklen extern programmieren

16.1 Technologische Zyklen

Nr.	Param Maske	Param intern	Erklärung
17		_PITA	Bewertung der Gewindesteigung 0 = Kompatibilitätsmode 1 = Steigung in mm 2 = Steigung in Gänge pro Zoll (TPI) 3 = Steigung in inch 4 = Steigung als MODUL
18		_PITM	String als Merker für Gewindesteigungseingabe (nur für die Oberfläche)
19		_RTAB	String für Gewindetabelle ("", "ISO", "BSW", "BSP", "UNC") (nur für die Oberfläche)
20		_PTABA	String für Auswahl in der Gewindetabelle (z.B. "M 10", "M 12", ...) (nur für die Oberfläche)
21		_GMODE	Geometriemode EINER: reserviert ZEHNER: reserviert HUNDERTER: Bearbeitung/Startpunktberechnung 0 = Kompatibilitätsmode 1 = normale Bearbeitung
22		_DMODE	Displaymode EINER: Bearbeitungsebene G17/18/19 0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv)
23		_AMODE	Alternativmode EINER: Gewindelänge (_DP) 0 = absolut 1 = inkrementell

16.1.25 Gravurzyklus - CYCLE60

Programmierung

```
CYCLE60 (STRING[200] _TEXT, REAL _RTP, REAL _RFP, REAL _SDIS, REAL
_DP, REAL _DPR, REAL _PA, REAL _PO, REAL _STA, REAL _CP1, REAL _CP2,
REAL _WID, REAL _DF, REAL _FFD, REAL _FFP1, INT _VARI, INT _CODEP,
INT _UMODE, INT _GMODE, INT _DMODE, INT _AMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1		_TEXT	zu gravierender Text (maximal 100 Zeichen)
2	RP	_RTP	Rückzugsebene (abs)
3	Z0	_RFP	Bezugspunkt der Werkzeugachse (abs)
4	SC	_SDIS	Sicherheitsabstand (additiv auf Referenzebene, ohne Vorzeichen einzugeben)
5	Z1	_DP	Tiefe (abs), siehe _AMODE
6	Z1	_DPR	Tiefe (ink), siehe _AMODE
7	X0	_PA	Bezugspunkt, 1. Achse der Ebene (abs) - rechtwinklig, siehe _VARI
	R		Bezugspunkt, Länge (Radius) - polar, siehe _VARI
8	Y0	_PO	Bezugspunkt, 2. Achse der Ebene (abs) - rechtwinklig, siehe _VARI
	$\alpha 0$		Bezugspunkt, Winkel bezogen auf die 1. Achse - polar, siehe _VARI
9	$\alpha 1$	_STA	Textrichtung, Winkel der Textlinie bezogen auf die 1. Achse, siehe _VARI
10	XM	_CP1	Mittelpunkt des Textkreises, 1. Achse der Ebene (abs) - rechtwinklig, siehe _VARI
	LM		Mittelpunkt des Textkreises, Länge (Radius) bezogen auf WNP - polar, siehe _VARI
11	YM	_CP2	: Mittelpunkt des Textkreises, 2. Achse der Ebene (abs) - rechtwinklig, siehe _VARI
	αM		Mittelpunkt des Textkreises, Winkel bezogen auf die 1. Achse - polar, siehe _VARI
12	W	_WID	Zeichenhöhe (ohne Vorzeichen einzugeben)
13	DX1 DX2	_DF	Zeichenabstand / Gesamtbreite, siehe _VARI
	$\alpha 2$		Öffnungswinkel, siehe _VARI
14	FZ	_FFD	Zustellvorschub Tiefe, siehe _DMODE
15	F	_FFP1	Vorschub für Flächenbearbeitung
16		_VARI	Bearbeitung (Ausrichtung und Bezugspunkt des Gravurtextes))
			EINER: Bezugspunkt
			0: rechtwinklig
			1: polar
		ZEHNER: Textausrichtung	
		0: Text auf einer Linie	
		1: Text auf einem Kreisbogen oben	
		2: Text auf einem Kreisbogen unten	
		HUNDERTER: reserviert	

Nr.	Param Maske	Param intern	Erklärung
			TAUSENDER: : Bezugspunkt des Textes horizontal 0: links 1: mittig 2: rechts
			ZEHNTAUSENDER: Bezugspunkt des Textes vertikal 0: unten 1: mittig 2: oben
			HUNDERTTAUSENDER: Textlänge 0: Zeichenabstand 1: Gesamtbreite des Textes (nur bei Text linear) 2: Öffnungswinkel (nur bei Text auf Kreisbogen)
			MILLION: Kreismittelpunkt 0: rechtwinklig (kartesisch) 1: polar
			ZEHNMILLION: Spiegelschrift 0 = Kompatibilität 1 = Spiegelschrift EIN 2 = Spiegelschrift AUS
17		_CODEP	Nummer der Codepage für die Schrift (z. Z. nur 1252)
18		_UMODE	reserviert
19		_GMODE	Mode für Bewertung der programmierten Geometriewerte EINER: reserviert ZEHNER : reserviert HUNDERTER: Auswahl Bearbeitung / nur Startpunktberechnung 0 = Kompatibilitätsmode 1 = normale Bearbeitung
20		_DMODE	Displaymode EINER: Bearbeitungsebene G17/18/19 0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebene aktiv 1 = G17 2 = G18 3 = G19 ZEHNER: Vorschubart: G-Gruppe (G94/G95) für Flächen- und Tiefenvorschub 0 = Kompatibilitätsmode 1 = G-Code wie vor Zyklusaufwurf. G94/G95 für Flächen- und Tiefenvorschub gleich
21		_AMODE	Alternativmode EINER: Endtiefe (_DP, _DPR) 0 = Kompatibilität 1 = inkrementell (_DPR) 2 = absolut (_DP)

16.1.26 Konturaufruf - CYCLE62

Programmierung

```
CYCLE62 (STRING [140] _KNAME, INT _TYPE, STRING [32] _LAB1, STRING [32]  
_LAB2)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	PRG/ CON	_KNAME	Konturname oder Unterprogrammname, muss nicht programmiert werden bei _TYPE = 2
2		_TYPE	Bestimmung der Kontureingabe 0 = Unterprogramm 1 = Konturname 2 = Labels 3 = Labels in Unterprogramm
3	LAB1	_LAB1	Label 1, Konturanfang
4	LAB2	_LAB2	Label 2, Konturende

16.1.27 Bahnfräsen - CYCLE72

Programmierung

```
CYCLE72 (STRING [141] _KNAME, REAL _RTP, REAL _RFP, REAL _SDIS, REAL
_DP, REAL _MID, REAL _FAL, REAL _FALD, REAL _FFP1, REAL _FFD, INT
_VARI, INT _RL, INT _AS1, REAL _LP1, REAL _FF3, INT _AS2, REAL
_LP2, INT _UMODE, REAL _FS, REAL _ZFS, INT _GMODE, INT _DMODE, INT
_AMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1		_KNAME	Name des Kontur-Unterprogrammes
2	RP	_RTP	Rückzugsebene (abs)
3	Z0	_RFP	Bezugspunkt der Werkzeugachse (abs)
4	SC	_SDIS	Sicherheitsabstand (additiv auf Bezugspunkt, ohne Vorzeichen einzugeben)
5	Z1	_DP	Endpunkt, Endtiefe(abs/ink), siehe _AMODE
6	DZ	_MID	maximale Tiefenzustellung (ink, ohne Vorzeichen einzugeben)
7	UXY	_FAL	Schlichtaufmaß Ebene (ink), Aufmaß an der Randkontur
8	UZ	_FALD	Schlichtaufmaß Tiefe (ink), Aufmaß am Grund ohne Vorzeichen einzugeben)
9	FX	_FFP1	Vorschub an der Kontur
10	FZ	_FFD	Vorschub für Tiefenzustellung (oder räumliche Zustellung)
11		_VARI	Bearbeitungsart
			EINER: Bearbeitung
			1 = Schruppen
			2 = Schlichten
			5 = Anfassen
			ZEHNER:
			0 = Zwischenwege mit G0
			1 = Zwischenwege mit G1
			HUNDETER:
			0 = Rückzug am Konturende auf Bezugspunkt
			1 = Rückzug am Konturende auf Bezugspunkt + _SDIS
			2 = Rückzug am Konturende um _SDIS
			3 = kein Rückzug am Konturende, nächster Startpunkt wird mit Konturvorschub angefahren
			TAUSENDER: reserviert
			ZEHNTAUSENDER:
			0 = Kontur vorwärts bearbeiten
			1 = Kontur rückwärts bearbeiten
			Einschränkungen bei rückwärts:
			• max. 170 Konturelemente (inclusive Fasen oder Verrundungen)
			• es werden nur Werte in der Ebene (X/Y) und F ausgewertet

Nr.	Param Maske	Param intern	Erklärung
12		_RL	Bearbeitungsrichtung 40 = mittig der Kontur (G40, An- und Abfahren: Gerade oder senkrecht) 41 = links der Kontur (G41, An- und Abfahren: Gerade oder Kreis) 42 = rechts der Kontur (G42, An- und Abfahren: Gerade oder Kreis)
13		_AS1	Kontur-Anfahrbewegung EINER: 1 = Gerade 2 = Viertelkreis 3 = Halbkreis 4 = senkrecht An- und Abfahren ZEHNER: 0 = letzte Bewegung, in der Ebene 1 = letzte Bewegung, räumlich
14	L1	_LP1	Anfahrweg, oder Anfahradius (ink, ohne Vorzeichen einzugeben)
15	FZ	_FF3	Vorschub für Zwischenwege (G94/G95 wie an Kontur)
16		_AS2	Kontur-Abfahrbewegung (nicht bei senkrecht An-/Abfahren) EINER: 1 = Gerade 2 = Viertelkreis 3 = Halbkreis ZEHNER: 0 = letzte Bewegung, in der Ebene 1 = letzte Bewegung, räumlich
17	L2	_LP2	Abfahrweg, oder Abfahradius (ink, ohne Vorzeichen einzugeben)
18		_UMODE	reserviert
19	FS	_FS	Fasenbreite (Ink)
20	ZFS	_ZFS	Eintauchtiefe (Werkzeugspitze) bei Anfasen (abs/ink), siehe _AMODE
21		_GMODE	Mode für Bewertung der programmierten Geometriewerte EINER: reserviert ZEHNER: reserviert HUNDERTER: Auswahl Bearbeitung/nur Startpunktberechnung 0 = Kompatibilitätsmode 1 = normale Bearbeitung

Zyklen extern programmieren

16.1 Technologische Zyklen

Nr.	Param Maske	Param intern	Erklärung
22		<code>_DMODE</code>	<p>Displaymode</p> <hr/> <p>EINER: Bearbeitungsebene G17/G18/G19</p> <p>0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv)</p> <hr/> <p>ZEHNER: Vorschubart: G-Gruppe (G94/G95) für Flächen- und Tiefenvorschub</p> <p>0 = Kompatibilitätsmode 1 = G-Code wie vor Zyklusaufwurf. G94/G95 für Flächen- und Tiefenvorschub gleich</p> <hr/> <p>TAUSENDER:</p> <p>0 = Kompatibilitätsmode: Konturname steht in <code>_KNAME</code> 1 = Konturname wird im <code>CYCLE62</code> programmiert und in <code>_SC_CONT_NAME</code> übergeben</p>
23		<code>_AMODE</code>	<p>Alternativmode</p> <hr/> <p>EINER: Endpunkt Z1 (<code>_DP</code>)</p> <p>0 = absolut (Kompatibilitätsmode) 1 = inkrementell</p> <hr/> <p>ZEHNER: Einheit für Ebenenzustellung</p> <p>0 = mm, inch 1 = reserviert</p> <hr/> <p>HUNDERTER: Eintauchtiefe bei Anfasen (<code>_ZFS</code>)</p> <p>0 = absolut 1 = inkrementell</p>

Hinweis

Sind folgende Übergabeparameter indirekt (als Parameter) programmiert, wird die Eingabemaske nicht rückübersetzt:

`_VARI, _RL, _AS1, _AS2, _UMODE, _GMODE, _DMODE, _AMODE`

16.1.28 Konturtasche vorbohren - CYCLE64

Programmierung

```
CYCLE64 (STRING[100] _PRG, INT _VARI, REAL _RP, REAL _Z0, REAL _SC,
REAL _Z1, REAL _F, REAL _DXY, REAL _UXY, REAL _UZ, INT _CDIR,
STRING[20] _TR, INT _DR, INT _UMODE, INT _GMODE, INT _DMODE, INT
_AMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	PRG	_PRG	Name Bohr-/Zentrierprogramm
2		_VARI	Bearbeitungsart EINER: reserviert ZEHNER: reserviert HUNDERTER: reserviert TAUSENDER: Abhebemodus 0 = Abheben auf Rückzugsebene 1 = Abheben auf Bezugspunkt + Sicherheitsabstand
3	RP	_RP	Rückzugsebene (abs)
4	Z0	_Z0	Bezugspunkt (abs)
5	SC	_SC	Sicherheitsabstand (additiv auf Bezugspunkt, ohne Vorzeichen einzugeben)
6	Z1	_Z1	Bohr-/Zentriertiefe, (siehe _AMODE EINER)
7	F	_F	Vorschub Bohren/Zentrieren
8	DXY	_DXY	Zustellung Ebene - Einheit, (siehe AMODE ZEHNER)
9	UXY	_UXY	Schlichtaufmaß Ebene
10	UZ	_UZ	Schlichtaufmaß Tiefe
11		_CDIR	Fräsrichtung 0 = Gleichlauf 1 = Gegenlauf
12	TR	_TR	Referenzwerkzeugname
13	DR	_DR	Referenzwerkzeug D-Nummer
14		_UMODE	reserviert
15		_GMODE	Mode für Bewertung der programmierten Geometriewerte EINER: reserviert ZEHNER: reserviert HUNDERTER: Auswahl Bearbeitung/nur Startpunktberechnung 0 = normale Bearbeitung (kein Kompatibilitätsmode erforderlich) 1 = normale Bearbeitung 2 = reserviert

Nr.	Param Maske	Param intern	Erklärung
25		_DMODE	<p>Displaymode</p> <hr/> <p>EINER: Bearbeitungsebene G17/18/19</p> <hr/> <p>0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebene aktiv</p> <p>1 = G17 (nur im Zyklus aktiv)</p> <p>2 = G18 (nur im Zyklus aktiv)</p> <p>3 = G19 (nur im Zyklus aktiv)</p> <hr/> <p>ZEHNER: Technologiemojus)</p> <hr/> <p>1 = Vorbohren</p> <p>2 = Zentrieren</p>
26		_AMODE	<p>Alternativmode</p> <hr/> <p>EINER: Bohr-/Zentriertiefe Z1</p> <hr/> <p>0 = absolut (Kompatibilitätsmode)</p> <p>1 = inkrementell</p> <hr/> <p>ZEHNER: : Einheit für Ebenenzustellung (_DXY)</p> <hr/> <p>0 = mm</p> <p>1 = % vom Werkzeugdurchmesser</p>

16.1.29 Konturtasche fräsen - CYCLE63

Programmierung

```

CYCLE63 (STRING[100] _PRG, INT _VARI, REAL _RP, REAL _Z0, REAL _SC,
REAL _Z1, REAL _F, REAL _FZ, REAL _DXY, REAL _DZ, REAL _UXY, REAL
_UZ, INT _CDIR, REAL _XS, REAL _YS, REAL _ER, REAL _EP, REAL _EW,
REAL _FS, REAL _ZFS, STRING[20] _TR, INT _DR, INT _UMODE, INT
_GMODE, INT _DMODE, INT _AMODE)
    
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	PRG	_PRG	Name Ausräumprogramm
2		_VARI	<p>Bearbeitungsart</p> <hr/> <p>EINER: Technologische Bearbeitung</p> <hr/> <p>1 = Schruppen</p> <p>3 = Schlichten Boden</p> <p>4 = Schlichten Rand</p> <p>5 = Anfasen</p>

Nr.	Param Maske	Param intern	Erklärung
			ZEHNER: Zustellungsart 0 = Eintauchen mittig 1 = Eintauchen helikal 2 = Eintauchen pendelnd
			HUNDERTER: reserviert
			TAUSENDER: Abhebemodus 0 = Abheben auf Rückzugsebene 1 = Abheben auf Bezugspunkt + Sicherheitsabstand
			ZEHNTAUSENDER: Startpunkt bei Schruppen und Schlichten Boden 0 = auto 1 = manuell
3	RP	_RP	Rückzugsebene (abs)
4	Z0	_Z0	Bezugspunkt der Werkzeugachse (abs)
5	SC	_SC	Sicherheitsabstand (additiv auf Bezugspunkt, ohne Vorzeichen einzugeben)
6	Z1	_Z1	Endtiefe, (siehe _AMODE EINER)
7	F	_F	Vorschub in der Ebene Schruppen/Schlichten
8	FZ	_FZ	Zustellvorschub Tiefe
9	DXY	_DXY	Zustellung Ebene - Einheit, (siehe AMODE ZEHNER)
10	DZ	_DZ	Zustellung Tiefe
11	UXY	_UXY	Schlichtaufmaß Ebene
12	UZ	_UZ	Schlichtaufmaß Tiefe
13		_CDIR	Fräsrichtung 0 = Gleichlauf 1 = Gegenlauf
14	XS	_XS	Startpunkt X, absolut
15	YS	_YS	Startpunkt Y, absolut
16	ER	_ER	Eintauchen helikal: Radius
17	EP	_EP	Eintauchen helikal: Steigung
18	EW	_EW	Eintauchen pendelnd: maximaler Eintauchwinkel
19	FS	_FS	Fasbreite (ink) bei Anfasen
20	ZFS	_ZFS	Eintauchtiefe Werkzeugspitze bei Anfasen, (siehe AMODE HUNDERTER)
21	TR	_TR	Referenzwerkzeugname bei Restmaterialbearbeitung
22	DR	_DR	Referenzwerkzeug D-Nummer bei Restmaterialbearbeitung
23		_UMODE	reserviert

Zyklen extern programmieren

16.1 Technologische Zyklen

Nr.	Param Maske	Param intern	Erklärung
24		<code>_GMODE</code>	Mode für Bewertung der programmierten Geometriewerte EINER: reserviert ZEHNER: reserviert HUNDERTER: Auswahl Bearbeitung/nur Startpunktberechnung 0 = normale Bearbeitung (kein Kompatibilitätsmode erforderlich) 1 = normale Bearbeitung 2 = reserviert
25		<code>_DMODE</code>	Displaymode EINER: Bearbeitungsebene G17/18/19 0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv) ZEHNER: reserviert HUNDERTER: Technologiemodus 1 = Tasche 2 = Zapfen TAUSENDER: Restmaterial bearbeiten 0 = nein 1 = ja
26		<code>_AMODE</code>	Alternativmode EINER: Endtiefe Z1 0 = absolut (Kompatibilitätsmode) 1 = inkrementell ZEHNER: Einheit für Ebenenzustellung (<code>_DXY</code>) 0 = mm 1 = % vom Werkzeugdurchmesser HUNDERTER: Eintauchtiefe bei Anfasen (<code>_ZFS</code>) 0 = absolut 1 = inkrementell

16.1.30 Abspannen - CYCLE951

Programmierung

```
CYCLE951 (REAL _SPD, REAL _SPL, REAL _EPD, REAL _EPL, REAL _ZPD, REAL
_ZPL, INT _LAGE, REAL _MID, REAL _FALX, REAL _FALZ, INT _VARI, REAL
_RF1, REAL _RF2, REAL _RF3, REAL _SDIS, REAL _FF1, INT _NR, INT
_DMODE, INT _AMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	X0	_SPD	Bezugspunkt (abs, immer Durchmesser)
2	Z0	_SPL	Bezugspunkt (abs)
3	X1	_EPD	Endpunkt
4	Z1	_EPL	Endpunkt
5	XM α1 α2	_ZPD	Zwischenpunkt, siehe _DMODE (ZEHNER)
6	ZM α1 α2	_ZPL	Zwischenpunkt, siehe _DMODE (ZEHNER)
7	Lage	_LAGE	Lage der Abspannecke 0 = außen/hinten 1 = außen/vorne 2 = innen/hinten 3 = innen/vorne
8	D	_MID	maximale Tiefenzustellung beim Eintauchen
9	UX	_FALX	Schlichtaufmaß in X
10	UZ	_FALZ	Schlichtaufmaß in Z
11		_VARI	Bearbeitungsart EINER: Abspanrichtung (längs oder plan) im Koordinatensystem 1 = längs 2 = plan ZEHNER: 1 = Schruppen bis Schlichtaufmaß 2 = Schlichten HUNDERTR: 0 = mit Nachziehen an der Kontur, ohne Restecken 1 = ohne Nachziehen an der Kontur TAUSENDER: 0 = mit Radius/Fase an Ecke 2 1 = mit Freistich an Ecke 2 ZEHNTAUSENDER: 0 = Nach der Bearbeitung stehenbleiben 1 = Auf Startposition zurückfahren
12	R1/FS1	_RF1	Verrundungsradius oder Fasenbreite 1, siehe _AMODE (ZEHNTAUSENDER)

Zyklen extern programmieren

16.1 Technologische Zyklen

Nr.	Param Maske	Param intern	Erklärung
13	R2/FS2	_RF2	Verrundungsradius oder Fasenbreite 2, siehe _AMODE (HUNDERTTAUSENDER)
14	R3/FS3	_RF3	Verrundungsradius oder Fasenbreite 3, siehe _AMODE (EINEMILLION)
15	SC	_SDIS	Sicherheitsabstand
16	F	_FF1	Vorschub für Schruppen/Schlichten
17		_NR	Kennung der Abspanart (entspricht den Vertikal-Softkeys zur Auswahl der Form): 0 = Abspannen 1, 90 Grad Ecke ohne Fasen/Verrundungen 1 = Abspannen 2, 90 Grad Ecke mit Fasen/Verrundungen 2 = Abspannen 3, beliebige Ecke mit Fasen/Verrundungen
18		_DMODE	Displaymode EINER: Bearbeitungsebene G17/G18/G19 0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv) ZEHNER: Eingabeform _ZPD/_ZPL 0 = Xm/Zm 1 = Xm/α1 2 = Xm/α2 3 = α1/Zm 4 = α2/Zm 5 = α1/α2
21		_AMODE	Alternativmode EINER: Zwischenpunkt in X 0 = absolut, Wert der Planachse im Durchmesser 1 = inkrementell, Wert der Planachse im Radius ZEHNER: Zwischenpunkt in Z 0 = absolut 1 = inkrementell HUNDERTER: Endpunkt in X 0 = absolut, Wert der Planachse im Durchmesser 1 = inkrementell, Wert der Planachse im Radius TAUSENDER: Endpunkt in Z 0 = absolut 1 = inkrementell ZEHNTAUSENDER: Radius/Fase 1 0 = Radius 1 = Fase HUNDERTTAUSENDER: Radius/Fase 2 0 = Radius 1 = Fase EINEMILLION: Radius/Fase 3 0 = Radius 1 = Fase

16.1.31 Einstich - CYCLE930

Programmierung

```
CYCLE930 (REAL _SPD, REAL _SPL, REAL _WIDG, REAL _WIDG2, REAL _DIAG,
REAL _DIAG2, REAL _STA, REAL _ANG1, REAL _ANG2, REAL _RCO1, REAL
_RCI1, REAL _RCI2, REAL _RCO2, REAL _FAL, REAL _IDEP1, REAL _SDIS,
INT _VARI, INT _DN, INT _NUM, REAL _DBH, REAL _FF1, INT _NR, REAL
_FALX, REAL _FALZ, INT _DMODE, INT _AMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	X0	_SPD	Bezugspunkt in der Planachse (immer Durchmesser)
2	Z0	_SPL	Bezugspunkt in der Längsachse
3	B1	_WIDG	Einstichbreite unten
4	B2	_WIDG2	Einstichbreite oben (nur für die Oberfläche)
5	T1	_DIAG	Einstichtiefe am Bezugspunkt, bei abs und Längsbearbeitung = Durchmesser, sonst ink
6	T2	_DIAG2	Einstichtiefe gegenüber dem Bezugspunkt (nur für die Oberfläche), bei abs und Längsbearbeitung = Durchmesser, sonst ink
7	$\alpha 0$	_STA	Winkel der Schräge ($-180 \leq _STA \leq 180$)
8	$\alpha 1$	_ANG1	Flankenwinkel 1 ($0 \leq _ANG1 < 90$) an der durch den Bezugspunkt bestimmten Seite des Einstichs
9	$\alpha 2$	_ANG2	Flankenwinkel 2 ($0 \leq _ANG2 < 90$) gegenüber dem Bezugspunkt
10	R1/FS1	_RCO1	Verrundungsradius oder Fasenbreite 1, außen am Bezugspunkt
11	R2/FS2	_RCI1	Verrundungsradius oder Fasenbreite 2, innen am Bezugspunkt
12	R3/FS3	_RCI2	Verrundungsradius oder Fasenbreite 3, innen gegenüber dem Bezugspunkt
13	R4/FS4	_RCO2	Verrundungsradius oder Fasenbreite 4, außen gegenüber dem Bezugspunkt
14	U	_FAL	Schlichtaufmaß in X und Z, siehe _VARI (ZEHNTAUSENDER) (ohne Vorzeichen eingeben)
15	D	_IDEP1	maximale Tiefenzustellung beim Eintauchen (ohne Vorzeichen eingeben) 0 = 1. Schnitt direkt auf ganze Tiefe > 0 = 1. Schnitt _IDEP1, 2. Schnitt 2 · _IDEP1 usw.
16	SC	_SDIS	Sicherheitsabstand (ohne Vorzeichen eingeben)

Zyklen extern programmieren

16.1 Technologische Zyklen

Nr.	Param Maske	Param intern	Erklärung
17		_VARI	<p>Bearbeitungsart</p> <hr/> <p>EINER: reserviert</p> <hr/> <p>ZEHNER: Technologische Bearbeitung</p> <hr/> <p>1 = Schruppen 2 = Schlichten 3 = Schruppen und Schlichten</p> <hr/> <p>HUNDERTER: Lage Längs/Plan Außen/Innen +Z/+Z bzw. +X/-X</p> <hr/> <p>1 = Längs/Außen +Z 2 = Plan/Innen -X 3 = Längs/Innen +Z 4 = Plan/Innen +X 5 = Längs/Außen -Z 6 = Plan/Außen -X 7 = Längs/Innen -Z 8 = Plan/Außen +X</p> <hr/> <p>TAUSENDER: Lage Bezugspunkt</p> <hr/> <p>0 = Bezugspunkt oben 1 = Bezugspunkt unten</p> <hr/> <p>ZEHTAUSENDER: Definition wie Schlichtaufmaße wirken</p> <hr/> <p>0 = Schlichtaufmaß U konturparallel 1 = Schlichtaufmaße UX und UZ getrennt</p>
18		_DN	<p>D-Nummer für 2. Schneide des Werkzeugs</p> <p>> 0 = D-Nummer für Werkzeugkorrektur der 2. Schneide des Einstechers</p> <p>0 = keine 2. Schneide programmiert</p>
19	N	_NUM	Anzahl der Einstiche (0 = 1 Einstich)
20	DP	_DBH	Abstand der Einstiche (nur erforderlich, wenn _NUM > 1)
21	F	_FF1	Vorschub
22		_NR	<p>Kennung für die Einstichform entspricht den Vertikal-Softkeys zur Auswahl der Form</p> <p>0 = 90°-Flanken ohne Fasen/Verrundungen 1 = schräge Flanken mit Fasen/Verrundungen (ohne α_0) 2 = wie 1, aber auf Kegel (mit α_0)</p>
23	UX	_FALX	Schlichtaufmaß in X Achse, siehe _VARI (ZEHTAUSENDER) (ohne Vorzeichen einzugeben)
24	UZ	_FALZ	Schlichtaufmaß in Z Achse, siehe _VARI (ZEHTAUSENDER) (ohne Vorzeichen einzugeben)

Nr.	Param Maske	Param intern	Erklärung
25		<code>_DMODE</code>	<p>Displaymode</p> <hr/> <p>EINER: Bearbeitungsebene G17/G18/G19</p> <hr/> <p>0 = Kompatibilität, es bleibt die vor Zyklusauf Ruf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv)</p>
26		<code>_AMODE</code>	<p>Alternativmode</p> <hr/> <p>EINER: Bemaßung Tiefe (nur für die Oberfläche)</p> <hr/> <p>0 = am Bezugspunkt 1 = gegenüber dem Bezugspunkt</p> <hr/> <p>ZEHNER: Tiefe</p> <hr/> <p>0 = absolut 1 = inkrementell</p> <hr/> <p>HUNDERTER: Bemaßung Breite (nur für die Oberfläche)</p> <hr/> <p>0 = am Außendurchmesser (oben) 1 = am Innendurchmesser (unten)</p> <hr/> <p>TAUSENDER: Radius/Fase 1 (<code>_RCO1</code>)</p> <hr/> <p>0 = Radius 1 = Fase</p> <hr/> <p>ZEHNTAUSENDER: Radius/Fase 2 (<code>_RCI1</code>)</p> <hr/> <p>0 = Radius 1 = Fase</p> <hr/> <p>HUNDERTTAUSENDER: Radius/Fase 3 (<code>_RCI2</code>)</p> <hr/> <p>0 = Radius 1 = Fase</p> <hr/> <p>MILLIONSTE STELLE: Radius/Fase 4 (<code>_RCO2</code>)</p> <hr/> <p>0 = Radius 1 = Fase</p>

16.1.32 Freistich Formen - CYCLE940

Mit dem Zyklus CYCLE940 können verschiedene Freistiche programmiert werden. Diese unterscheiden sich in der Parametrierung zum Teil erheblich.

Die zusätzlichen Spalten in der Tabelle zeigen, welcher Parameter bei welcher Freistichform benötigt wird. Sie entsprechen den vertikalen Auswahlsoftkeys in der Zyklenmaske:

- E: Freistich Form E
- F: Freistich Form F
- A-D: Freistich Gewinde DIN (Formen A-D)
- T: Freistich Gewinde (freie Definition der Form)

Programmierung

```
CYCLE940 (REAL _SPD, REAL _SPL, CHAR _FORM, INT _LAGE, REAL _SDIS,
REAL _FFP, INT _VARI, REAL _EPD, REAL _EPL, REAL _R1, REAL _R2, REAL
_STA, REAL _VRT, REAL _MID, REAL _FAL, REAL _FALX, REAL _FALZ, INT
_PITI, STRING[5] _PTAB, STRING[20] _PTABA, INT _DMODE, INT _AMODE)
```

Parameter

Nr.	Param Maske	Param intern	Progr. bei Form				Erklärung
			E	F	A-D	T	
1	X0	_SPD	x	x	x	x	Bezugspunkt in der Planachse (immer Durchmesser)
2	Z0	_SPL	x	x	x	x	Bezugspunkt Längsachse (abs)
3	FORM	_FORM	x	x	x	x	Form des Freistichs (Großbuchstaben, z.B. "T") Auswahl, aus welcher Tabelle die Freistichwerte genommen werden sollen A = außen, Bezug DIN76, A = normal B = außen, Bezug DIN76, B = kurz C = innen, Bezug DIN76, C = normal D = innen, Bezug DIN76, D = kurz E = Bezug DIN509 F = Bezug DIN509 T = Freiform
4	LAGE	_LAGE	x	x	x	x	Lage des Freistichs (parallel Z) 0 = Außen +Z: ___ 1 = Außen -Z: ___/ 2 = Innen +Z: /---- 3 = Innen -Z: ----\
5	SC	_SDIS	x	x	x	x	Sicherheitsabstand (ink)
6	F	_FFP	x	x	x	x	Bearbeitungsvorschub (mm/U)

Progr. bei Form								
7		_VARI	-	-	x	x	Bearbeitungsart	
					EINER: Bearbeitung			
					1 = Schruppen			
					2 = Schlichten			
					3 = Schruppen + Schlichten			
					ZEHNER: Bearbeitungsstrategie			
					0 = Konturparallel			
					1 = Längs			
					Freistiche Form E und F werden immer in einem Zug wie Schlichten bearbeitet.			
8	X1	_EPD	x	x	-	-	Aufmaß X (abs/ink), siehe _AMODE	
					-	-	Freistichtiefe (abs/ink), siehe _AMODE	
9	Z1	_EPL	-	x	-	-	Aufmaß Z	
					-	-	Freistichbreite (abs/ink), siehe _AMODE	
10	R1	_R1	-	-	-	x	Verrundungsradius an der Schrägen	
11	R2	_R2	-	-	-	x	Verrundungsradius in der Ecke	
12	α	_STA	-	-	x	x	Eintauchwinkel	
13	VX	_VRT	x	x	-	-	Planzug X (abs/ink), siehe _AMODE	
					-	-	Planzug X bei Schlichten, (abs/ink), siehe _AMODE	
14	D	_MID	-	-	x	x	Tiefenzustellung	
15	U	_FAL	-	-	x	x	Schlichtaufmaß konturparallel, siehe _AMODE	
16	UX	_FALX	-	-	x	x	Schlichtaufmaß X	
17	UZ	_FALZ	-	-	x	x	Schlichtaufmaß Z	
18	P	_PITI	-	-	x	-	Auswahl der Steigung, Form A-D, entspricht M1 ... M68	
					0 = 0.20	6 = 0.50	12 = 1.25	18 = 3.50
					1 = 0.25	7 = 0.60	13 = 1.50	19 = 4.00
					2 = 0.30	8 = 0.70	14 = 1.75	20 = 4.50
					3 = 0.35	9 = 0.75	15 = 2.00	21 = 5.00
					4 = 0.40	10 = 0.80	16 = 2.50	22 = 5.50
					5 = 0.45	11 = 1.00	17 = 3.00	23 = 6.00
					x	x	-	-
					Auswahl Radius/Tiefe, Form E, F			
					0 = 0.6 · 0.3	4 = 2.5 · 0.4	8 = 0.1 · 0.1	
					1 = 1.0 · 0.4	5 = 4.0 · 0.5	9 = 0.2 · 0.1	
					2 = 1.0 · 0.2	6 = 0.4 · 0.2		
					3 = 1.6 · 0.3	7 = 0.6 · 0.2		
19		_PTAB					String für Gewindetabelle ("", "ISO", "BSW", "BSP", "UNC") (nur für Oberfläche)	
20		_PTABA					String für Auswahl in der Gewindetabelle (z.B. "M 10", "M 12", ...) (nur für die Oberfläche)	

		Progr. bei Form				
21	_DMODE				Displaymode	
		x	x	x	x	EINER: Bearbeitungsebene G17/18/19
						0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv)
22	_AMODE					Alternativmode
		x	x	-	x	EINER: Parameter _EPD Aufmaß X oder Freistichtiefe
						0 = absolut (immer Durchmesser) 1 = inkrementell
		x	x	-	x	ZEHNER: Parameter _EPL Aufmaß Z oder Freistichbreite
						0 = absolut 1 = inkrementell
		x	x	x	x	HUNDERTER: Parameter _VRT Planzug X
						0 = absolut (immer Durchmesser) 1 = inkrementell
		-	-	x	x	TAUSENDER: Schlichtaufmaß
						0 = Schlichtaufmaß konturparallel (_FAL) = Schlichtaufmaß getrennt (_FALX/_FALZ)

16.1.33 Gewindedrehen - CYCLE99

Programmierung

```
CYCLE99 (REAL _SPL, REAL _SPD, REAL _FPL, REAL _FPD, REAL _APP, REAL
  _ROP, REAL _TDEP, REAL _FAL, REAL _IANG, REAL _NSP, INT _NRC, INT
  _NID, REAL _PIT, INT _VARI, INT _NUMTH, REAL _SDIS, REAL _MID, REAL
  _GDEP, REAL _PIT1, REAL _FDEP, INT _GST, INT _GUD, REAL _IFLANK, INT
  _PITA, STRING[15] _PITM, STRING[20] _PTAB, STRING[20] _PTABA, INT
  _DMODE, INT _AMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	Z0	_SPL	Bezugspunkt (abs)
2	X0	_SPD	Bezugspunkt (abs, immer Durchmesser)
3	Z1	_FPL	Endpunkt, siehe _AMODE (EINER)
4	X1	_FPD	Endpunkt, siehe _AMODE (ZEHNER)
5	LW/LW2	_APP	Gewindevorlauf, siehe _AMODE (HUNDERTER) oder Gewindeeinlauf = Gewindeauslauf, siehe _AMODE (HUNDERTER)
6	LR	_ROP	Gewindeauslauf
7	H1	_TDEP	Gewindetiefe
8	U	_FAL	Schlichtaufmaß in X und Z
9	DP αP	_IANG	Zustellschräge als Abstand oder Winkel, siehe _AMODE (TAUSENDER) > 0 = Zustellung an der positiven Flanke < 0 = Zustellung an der negativen Flanke 0 = mittige Zustellung
10	α0	_NSP	Startwinkelversatz (wirkt nur bei "eingängig")
11	ND	_NRC	Anzahl Schruppschnitte, in Verbindung mit _VARI (ZEHNTAUSENDER)
12	NN	_NID	Anzahl Leerschnitte
13	P	_PIT	Gewindesteigung als Wert, siehe _PITA
14		_VARI	Bearbeitungsart EINER: Technologie 1 = Außengewinde mit linearer Zustellung 2 = Innengewinde mit linearer Zustellung 3 = Außengewinde mit degressiver Zustellung, Spanquerschnitt bleibt konstant 4 = Innengewinde mit degressiver Zustellung, Spanquerschnitt bleibt konstant ZEHNER: reserviert HUNDERTER: Zustellungsart 1 = einseitige Zustellung 2 = wechselnde Zustellung TAUSENDER: reserviert

Zyklen extern programmieren

16.1 Technologische Zyklen

Nr.	Param Maske	Param intern	Erklärung
			ZEHNTAUSENDER: Alternative Tiefenzustellung
			0 = Vorgabe Anzahl der Schruppschnitte (_NRC)
			1 = Vorgabe Wert für 1. Zustellung (_MID)
			HUNDERTTAUSENDER: Bearbeitungsart
			1 = Schruppen
			2 = Schlichten
			3 = Schruppen und Schlichten
			EINEMILLION: Bearbeitungsreihenfolge bei mehrgängigen Gewinde
			0 = Reihenfolge der Gänge aufsteigend
			1 = Reihenfolge der Gänge gegenüber
15	N	_NUMTH	Anzahl Gewindegänge
16	VR	_SDIS	Rücklaufabstand, ink
17	D1	_MID	Erste Zustelltiefe, siehe _VARI (ZEHNTAUSENDER)
18	DA	_GDEP	Gangwechselftiefe 0 = keine Gangwechselftiefe berücksichtigen > 0 = Gangwechselftiefe berücksichtigen
19	G	_PIT1	Änderung der Steigung pro Umdrehung 0 = Gewindesteigung ist konstant (G33) > 0 = Gewindesteigung wird größer (G34) < 0 = Gewindesteigung wird kleiner (G35)
20		_FDEP	Eintauchtiefe (ohne Vorzeichen einzugeben)
21	N1	_GST	Startgang N1 = 1...N, siehe _AMODE (HUNDERTTAUSENDER)
22		_GUD	reserviert
23		_IFLANK	Zustellschräge als Breite (nur für die Oberfläche)
24		_PITA	Maßeinheit der Gewindesteigung (Auswertung PIT und/oder MPIT) 0 = Steigung in mm - Auswertung MPIT/PIT 1 = Steigung in mm - Auswertung PIT 2 = Steigung in TPI - Auswertung PIT (Gewindegänge pro inch) 3 = Steigung in inch - Auswertung PIT 4 = MODUL - Auswertung PIT
25		_PITM	String als Merker für Gewindesteigungseingabe (nur für die Oberfläche) ¹⁾
26		_PTAB	String für Gewindetabelle (nur für die Oberfläche) ¹⁾
27		_PTABA	String für Auswahl in der Gewindetabelle (nur für die Oberfläche) ¹⁾

Nr.	Param Maske	Param intern	Erklärung
28		<code>_DMODE</code>	<p>Displaymode</p> <hr/> <p>EINER: Bearbeitungsebene G17/G18/G19</p> <p>0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 nur im Zyklus aktiv)</p> <hr/> <p>ZEHNER: Gewindeart</p> <p>0 = Längsgewinde 1 = Plangewinde 2 = Kegengewinde</p>
29		<code>_AMODE</code>	<p>Alternativmode</p> <hr/> <p>EINER: Gewindelänge in Z</p> <p>0 = absolut 1 = inkrementell</p> <hr/> <p>ZEHNER: Gewindelänge in X</p> <p>0 = absolut, Wert der Planachse im Durchmesser 1 = inkrementell, Wert der Planachse im Radius 2 = α</p> <hr/> <p>HUNDERTER: Vorlauf-/Einlaufwegbewertung <code>_APP</code></p> <p>0 = Gewindevorlauf <code>_APP</code> 1 = Gewindeeinlauf = Gewindeauslauf <code>_APP = -_ROP</code> 2 = Gewindeeinlaufweg vorgeben <code>_APP = -_APP</code></p> <hr/> <p>TAUSENDER: Auswahl Zustellschräge als Winkel oder Breite</p> <p>0 = Zustellwinkel <code>_IANG</code> 1 = Zustellschräge <code>_IFLANK</code></p> <hr/> <p>ZEHNTAUSENDER: eingängig/mehrgängig</p> <p>0 = eingängig (mit Startwinkelversatz <code>_NSP</code>) 1 = mehrgängig</p> <hr/> <p>HUNDERTTAUSENDER Startgang <code>_GST</code></p> <p>0 = komplette Bearbeitung 1 = ab diesem Gang die Bearbeitung starten 2 = nur diesen Gang bearbeiten</p>

Hinweis

1) Die Parameter `_PITM`, `_PTAB` und `_PTABA` werden nur bei der Gewindeauswahl in den Gewindetabellen der Eingabemaske verwendet.
 Ein Zugriff auf die Gewindetabellen über Zyklusdefinition bei Zykluslaufzeit ist nicht möglich.

16.1.34 Gewindekette - CYCLE98

Programmierung

```
CYCLE98 (REAL _PO1, REAL _DM1, REAL _PO2, REAL _DM2, REAL _PO3, REAL
_DM3, REAL _PO4, REAL _DM4, REAL APP, REAL ROP, REAL TDEP, REAL FAL,
REAL _IANG, REAL NSP, INT NRC, INT NID, REAL _PP1, REAL _PP2, REAL
_PP3, INT _VARI, INT _NUMTH, REAL _VRT, REAL _MID, REAL _GDEP, REAL
_IFLANK, INT _PITA, STRING[15] _PITM1, STRING[15] _PITM2, STRING[15]
_PITM3, INT _DMODE, INT _AMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	Z0	_PO1	Bezugspunkt in Z (abs)
2	X1	_DM1	Bezugspunkt in X (abs), im Durchmesser
3	Z1	_PO2	Zwischenpunkt 1 in Z, (abs/ink), siehe _AMODE (EINER)
4	X1 X1α	_DM2	Zwischenpunkt 1 in X (abs/ink), siehe _AMODE (ZEHNER) oder Gewindeschräge 1 (-90° bis 90°) abs immer Durchmesser, ink immer Radius
5	Z2	_PO3	Zwischenpunkt 2 in Z, (abs/ink), siehe _AMODE (HUNDERTER)
6	X2 X2α	_DM3	Zwischenpunkt 2 in X (abs/ink), siehe _AMODE (TAUSENDER) oder Gewindeschräge 2 (-90° bis 90°) abs immer Durchmesser, ink immer Radius
7	Z3	_PO4	Endpunkt in Z (abs/ink), siehe _AMODE (ZEHNTAUSENDER)
8	X3 X3α	_DM4	Endpunkt in X, (abs/ink), siehe _AMODE (HUNDERTTAUSENDER) oder Gewindeschräge 3 (-90° bis 90°) abs immer Durchmesser, ink immer Radius
9	LW	APP	Gewindevorlauf (ink, ohne Vorzeichen einzugeben)
10	LR	ROP	Gewindeauslauf (ink, ohne Vorzeichen einzugeben)
11	H1	TDEP	Gewindetiefe (ink, ohne Vorzeichen einzugeben)
12	U	FAL	Schlichtaufmaß in X und Z
13	DP αP	_IANG	Zustellschräge als Abstand oder Winkel, siehe _AMODE (MILLION) Die Zustellschräge wirkt entsprechend der Einstellung Parameter _VARI (HUNDERTER). Definition für _VARI_HUNDERTER = 0 - Kompatibilitätsmode: > 0 = Flankenzustellung an einer Flanke 0 = Zustellung senkrecht im Gewinde < 0 = Flankenzustellung mit alternierenden Flanken Definition für _VARI_HUNDERTER<>0: > 0 = Zustellung an der positiven Flanke 0 = mittige Zustellung < 0 = Zustellung an der negativen Flanke
14	α0	NSP	Startwinkelversatz für den 1. Gewindegang

Nr.	Param Maske	Param intern	Erklärung
15		NRC	Anzahl der Schruppschnitte, siehe _VARI (ZEHNTAUSENDER)
16	NN	NID	Anzahl Leerschnitte
17	P0	_PP1	Gewindesteigung 1. Gewindeabschnitt, siehe _PITA
18	P1	_PP2	Gewindesteigung 2. Gewindeabschnitt, siehe _PITA
19	P2	_PP3	Gewindesteigung 3. Gewindeabschnitt, siehe _PITA
20		_VARI	<p>Bearbeitung</p> <hr/> <p>EINER: Technologie</p> <p>1 = Außengewinde mit linearer Zustellung 2 = Innengewinde mit linearer Zustellung 3 = Außengewinde mit degressiver Zustellung, Spanquerschnitt bleibt konstant 4 = Innengewinde mit degressiver Zustellung, Spanquerschnitt bleibt konstant</p> <hr/> <p>ZEHNER: reserviert</p> <hr/> <p>HUNDERTER: Zustellungsart</p> <p>0 = Kompatibilitätsmode für _IANG 1 = einseitige Zustellung 2 = wechselnde Zustellung</p> <hr/> <p>TAUSENDER: reserviert</p> <hr/> <p>ZEHNTAUSENDER: Alternative Tiefenzustellung</p> <p>0 = Kompatibilität, Vorgabe Anzahl der Schruppschnitte (_NRC) 1 = Vorgabe Wert für 1. Zustellung (_MID)</p> <hr/> <p>HUNDERTTAUSENDER: Bearbeitungsart</p> <p>0 = Kompatibilität (Schruppen und Schlichten) 1 = Schruppen 2 = Schlichten 3 = Schruppen und Schlichten</p> <hr/> <p>EINEMILLION: Bearbeitungsreihenfolge bei mehrgängigen Gewinde</p> <p>0 = Reihenfolge der Gänge aufsteigend 1 = Reihenfolge der Gänge gegenüber</p>
21	N	_NUMTH	Anzahl Gewindegänge
22		_VRT	<p>Rücklaufabstand (ink)</p> <p>0 = intern wird ein Abhebeweg von 1 mm unabhängig vom Maßsystem inch/metrisch verwendet > 0 = Abhebeweg</p>
23	D1	_MID	erste Zustellung, siehe _VARI (ZEHNTAUSENDE)
24	DA	_GDEP	<p>Gangwechselliefe (wirkt nur bei "mehrgängig")</p> <p>0 = keine Gangwechselliefe berücksichtigen >0 = Gangwechselliefe berücksichtigen</p>
25		_IFLANK	Zustellschräge als Breite (nur für die Oberfläche)

Zyklen extern programmieren

16.1 Technologische Zyklen

Nr.	Param Maske	Param intern	Erklärung
26		_PITA	Bewertung der Gewindesteigung 0 = Kompatibilitätsmode für Gewindesteigung, Auswertung _PP1 bis _PP3 wie bisher entsprechend aktivem System metrisch/inch 1 = Steigung in mm 2 = Steigung in TPI (Gewindegänge pro inch) 3 = Steigung in inch 4 = MODUL
27		_PITM1	String als Merker für Gewindesteigungseingabe (nur für die Oberfläche)
28		_PITM2	String als Merker für Gewindesteigungseingabe (nur für die Oberfläche)
29		_PITM3	String als Merker für Gewindesteigungseingabe (nur für die Oberfläche)
30		_DMODE	Displaymode EINER: Bearbeitungsebene G17/18/19 0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv)
31		_AMODE	Alternativmode EINER: 1. Zwischenpunkt in Z (Z1) 0 = absolut 1 = inkrementell ZEHNER: 1. Zwischenpunkt in X (X1) 0 = absolut 1 = inkrementell 2 = α HUNDERTER: 2. Zwischenpunkt in Z (Z2) 0 = absolut 1 = inkrementell TAUSENDER: 2. Zwischenpunkt in X (X2) 0 = absolut 1 = inkrementell 2 = α ZEHNTAUSENDER: Endpunkt in Z (Z3) 0 = absolut 1 = inkrementell HUNDERTTAUSENDER: Endpunkt in X (X3) 0 = absolut 1 = inkrementell 2 = α EINEMILLION: Auswahl Zustellschräge als Winkel oder Breite 0 = Zustellwinkel _IANG 1 = Zustellschräge _IFLANK ZEHNMILLIONEN: eingängig/mehrgängig 0 = Kompatibilitätsmode (Startwinkel _NSP wird ausgewertet) 1 = eingängig (mit Startwinkelpersatz _NSP) 2 = mehrgängig

16.1.35 Abstich - CYCLE92

Programmierung

CYCLE92 (REAL _SPD, REAL _SPL, REAL _DIAG1, REAL _DIAG2, REAL _RC,
REAL _SDIS, REAL _SV1, REAL _SV2, INT _SDAC, REAL _FF1, REAL _FF2,
REAL _SS2, REAL _DIAGM, INT _VARI, INT _DN, INT _DMODE, INT _AMODE)

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	X0	_SPD	Bezugspunkt (abs, immer Durchmesser)
2	Y0	_SPL	Bezugspunkt (abs)
3	X1	_DIAG1	Tiefe für Drehzahlreduzierung, siehe _AMODE (EINER)
4	X2	_DIAG2	Endtiefe, siehe _AMODE (ZEHNER)
5	R/FS	_RC	Verrundusradius oder Fasebreite, siehe _AMODE (TAUSENDER)
6	SC	_SDIS	Sicherheitsabstand (additiv auf Bezugspunkt, ohne Vorzeichen einzugeben)
7	S	_SV1	konstante Spindeldrehzahl, siehe _AMODE (ZEHNTAUSENDER)
	V		konstante Schnittgeschwindigkeit
8	SV	_SV2	Maximaldrehzahl bei konstanter Schnittgeschwindigkeit
9	DIR	_SDAC	Spindeldrehrichtung 3 = für M3 4 = für M4
10	F	_FF1	Vorschub bis Tiefe für Drehzahlreduzierung
11	FR	_FF2	reduzierter Vorschub bis Endtiefe
12	SR	_SS2	reduzierte Drehzahl bis Endtiefe
13	XM	_DIAGM	Tiefe Teilefänger ausfahren (abs, immer Durchmesser)
14		_VARI	Bearbeitungsart EINER: Rückzug 0 = Rückzug auf _SPD+ _SDIS 1 = kein Rückzug am Ende ZEHNER: Teilefänger 0 = Nein, kein M-Befehl ausführen 1 = Ja, Aufruf von CUST_TECHCYC(101)- Schublade ausfahren, CUST_TECHCYC(102)- Schublade schließen
15		_DN	D-Nummer für 2. Schneide, wenn nicht programmiert => D+1

Zyklen extern programmieren

16.1 Technologische Zyklen

Nr.	Param Maske	Param intern	Erklärung
20		<code>_DMODE</code>	Displaymode EINER: Bearbeitungsebene G17/G18/G19 0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv)
21		<code>_AMODE</code>	Alternativmode EINER: Tiefe für Drehzahlreduzierung (<code>_DIAG1</code>) 0 = absolut, Wert der Planachse im Durchmesser 1 = inkrementell, Wert der Planachse im Radius ZEHNER: Endtiefe (<code>_DIAG2</code>) 0 = absolut, Wert der Planachse im Durchmesser 1 = inkrementell, Wert der Planachse im Radius HUNDERTER: reserviert TAUSENDER: Radius/Fase (<code>_RC</code>) 0 = Radius 1 = Fase ZEHNTAUSENDER: Spindeldrehzahl/Schnittgeschwindigkeit (<code>_SV1</code>) 0 = konstante Spindeldrehzahl 1 = konstante Schnittgeschwindigkeit

16.1.36 Konturstechen - CYCLE952

Programmierung

```
CYCLE952 (STRING[100] _PRG, STRING[100] _CON, STRING[100] _CONR, INT
_VARI, REAL _F, REAL _FR, REAL _RP, REAL _D, REAL _DX, REAL _DZ,
REAL _UX, REAL _UZ, REAL _U, REAL _U1, INT _BL, REAL _XD, REAL _ZD,
REAL _XA, REAL _ZA, REAL _XB, REAL _ZB, REAL _XDA, REAL _XDB, INT
_N, REAL _DP, REAL _DI, REAL _SC, INT _DN, INT _GMODE, INT _DMODE,
INT _AMODE, INT _PK, REAL _DCH)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	PRG	_PRG	Name des Abspanprogramms
2	CON	_CON	Name des Programms, aus dem die aktualisierte Rohteilkontur gelesen wird (bei Restbearbeitung)
3	CONR	_CONR	Name des Programms, in das die aktualisierte Rohteilkontur (siehe _AMODE ZEHNTAUSENDER) geschrieben wird
4		_VARI	Bearbeitungsart <hr/> EINER: Abspanart <hr/> 1 = längs 2 = plan 3 = konturparallel <hr/> ZEHNER: Technologische Bearbeitung (siehe _GMODE HUNDERTER) <hr/> 1 = Schruppen 2 = Schlichten 3 = reserviert 4 = Schruppen zweikanalig 5 = Schlichten zweikanalig <hr/> HUNDERTER: Bearbeitungsrichtung <hr/> 1 = Bearbeitungsrichtung X - 2 = Bearbeitungsrichtung X + 3 = Bearbeitungsrichtung Z - 4 = Bearbeitungsrichtung Z + <hr/> TAUSENDER: Zustellrichtung <hr/> 1 = Außen X- 2 = Innen X + 3 = Stirnseite Z - 4 = Rückseite Z + <hr/> ZEHNTAUSENDER: Definition, wie Schlichtaufmaße wirken <hr/> 0 = Schlichtaufmaße UX und UZ getrennt 1 = Schlichtaufmaß U konturparallel

Zyklen extern programmieren

16.1 Technologische Zyklen

Nr.	Param Maske	Param intern	Erklärung
			HUNDERTTAUSENDER: Nachziehen
			0 = Kompatibilität, Nachziehen automatisch
			1 = mit Nachziehen an der Kontur
			2 = ohne Nachziehen
			3 = Nachziehen automatisch
			EINEMILLION: Hinterschnitte
			0 = Stelle wird nicht ausgewertet bei Stechen, - Rest und Stechdrehen, - Rest
			1 = Hinterschnitte bearbeiten
			2 = Hinterschnitte nicht bearbeiten
			ZEHNMILLION: hinter/vor Drehmitte
			0 = Bearbeitung vor der Drehmitte
			1 = reserviert
5	F FZ	_F	Vorschub für Schruppen/Schlichten Vorschub Abszisse Stechdrehen
6	FR FX	_FR	Vorschub zum Eintauchen in Hinterschnitte Schruppen Vorschub Ordinate Stechdrehen
7	RP	_RP	Rückzugsebene bei Innenbearbeitung (abs., immer Durchmesser)
8	D	_D	Zustellung Schruppen (siehe _AMODE EINER)
9	DX	_DX	Zustellung X (siehe _AMODE EINER)
10	DZ	_DZ	Zustellung Z (siehe _AMODE EINER)
11	UX	_UX	Schlichtaufmaß X (siehe _VARI ZEHNTAUSENDER)
12	UZ	_UZ	Schlichtaufmaß Z (siehe _VARI ZEHNTAUSENDER)
13	U	_U	Schlichtaufmaß konturparallel (siehe _VARI ZEHNTAUSENDER)
14	U1	_U1	zusätzliches Schlichtaufmaß beim Schlichten (siehe _AMODE TAUSENDER)
15	BL	_BL	Rohteildefinition 1 = Zylinder mit Aufmaß 2 = Aufmaß auf Fertigteilkontur 3 = Rohteilkontur ist angegeben
16	XD	_XD	Rohteildefinition X (siehe _AMODE HUNDERTTAUSENDER)
17	ZD	_ZD	Rohteildefinition Z (siehe _AMODE EINEMILLION)
18	XA	_XA	Grenze 1 X (abs., immer Durchmesser)
19	ZA	_ZA	Grenze 1 Z (abs.)
20	XB	_XB	Grenze 2 X (siehe _AMODE ZEHNMILLION)
21	ZB	_ZB	Grenze 2 Z (siehe _AMODE HUNDERTMILLION)
22	XDA	_XDA	Einstechgrenze 1 für die 1. Einstichposition auf der Stirnseite (abs., immer Durchmesser)
23	XDB	_XDB	Einstechgrenze 2 für die 1. Einstichposition auf der Stirnseite (abs., immer Durchmesser)
24	N	_N	Anzahl der Einstiche
25	DP	_DP	Abstand der Einstiche Längs-Einstich: parallel zur Z-Achse Plan-Einstich: parallel zur X-Achse

Nr.	Param Maske	Param intern	Erklärung
26	DI	_DI	Abstand für Vorschubunterbrechung 0 = keine Unterbrechung > 0 = mit Unterbrechung
27	SC	_SC	Sicherheitsabstand zum Umfahren von Hindernissen, inkremental
28	D2	_DN	D-Nummer für 2. Schneide, wenn nicht programmiert ⇒ D+1
29		_GMODE	Geometriemodus (Bewertung der programmierten Geometriewerte) EINER: reserviert ZEHNER: reserviert HUNDERTER: Auswahl Bearbeitung / nur Startpunktberechnung 0 = normale Bearbeitung (kein Kompatibilitätsmodus erforderlich) 1 = normale Bearbeitung 2 = Startposition berechnen - keine Bearbeitung (nur für Aufruf aus ShopMill/ShopTurn) TAUSENDER: Eingrenzung 0 = nein 1 = ja ZEHNTAUSENDER: Grenze 1 X eingeben 0 = nein 1 = ja HUNDERTTAUSENDER: Grenze 2 X eingeben 0 = nein 1 = ja EINEMILLION: Grenze 1 Z eingeben 0 = nein 1 = ja ZEHNMILLION: Grenze 2 Z eingeben 0 = nein 1 = ja
30		_DMODE	Display-Modus EINER: Bearbeitungsebene G17/18/19 0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv) ZEHNER: Technologiemodus 1 = Konturabspannen 2 = Konturstechen 3 = Stechdrehen HUNDERTER: Restmaterial bearbeiten 0 = nein 1 = ja

Nr.	Param Maske	Param intern	Erklärung
31		<code>_AMODE</code>	<p>Alternativmodus</p> <p>EINER: Auswahl Zustellung</p> <p>0 = Zustellung DX und DZ bei Abspanart konturparallel 1 = Zustellung D</p> <p>ZEHNER: Zustellstrategie</p> <p>0 = variable Schnitttiefe (90 ... 100 %) 1 = konstante Schnitttiefe</p> <p>HUNDETER: Schnittaufteilung</p> <p>0 = gleichmäßig 1 = an Kanten ausrichten</p> <p>TAUSENDER: Auswahl Konturaufmaß U1, Doppelschichten</p> <p>0 = nein 1 = ja</p> <p>ZEHNTAUSENDER: Auswahl Rohteil aktualisieren</p> <p>0 = nein 1 = ja</p> <p>HUNDERTTAUSENDER: Auswahl Aufmaß auf Rohteil XD</p> <p>0 = absolut, Wert der Planachse im Durchmesser 1 = inkremental, Wert der Planachse im Radius</p> <p>EINEMILLION : Auswahl Aufmaß auf Rohteil ZD</p> <p>0 = absolut 1 = inkremental</p> <p>ZEHNMILLION: Auswahl Grenze 2 XB</p> <p>0 = absolut, Wert der Planachse im Durchmesser 1 = inkremental, Wert der Planachse im Radius</p> <p>HUNDERTMILLION: Auswahl Grenze 2 ZB</p> <p>0 = absolut 1 = inkremental</p> <p>EINEMILLIARDE</p> <p>0 = Führungskanal 1 = Folgekanal</p>
32		<code>_PK</code>	Nummer des Partnerkanals, wenn mehr als 2 Kanäle an der Maschine vorhanden sind
33	DCH	<code>_DCH</code>	Kanalversatz

16.1.37 Schwenken - CYCLE800

Programmierung

```
CYCLE800 (INT _FR, STRING[32] _TC, INT _ST, INT _MODE, REAL _X0, REAL
_Y0, REAL _Z0, REAL _A, REAL _B, REAL _C, REAL _X1, REAL _Y1, REAL
_Z1, INT _DIR, REAL _FR_I , INT _DMODE)
```

Parameter

Nr.	Param Maske	Param intern	Erklärung
1		_FR	<p>Freifahrmodus:</p> <p>0 = kein Freifahren 1 = Freifahren Maschinenachse Z 2 = Freifahren Maschinenachse Z und danach XY 3 = reserviert 4 = Freifahren in Werkzeugrichtung maximal 5 = Freifahren in Werkzeugrichtung inkrementell</p>
2		_TC	<p>Name Schwenkdatensatz:</p> <p>"" (kein Name) wenn nur 1 Schwenkdatensatz vorhanden "0" Abwahl Schwenkdatensatz (Löschen der Schwenkframes)</p>
3		_ST	<p>Status Transformationen</p> <p>EINER:</p> <p>0 = Neu , Schwenkebene wird gelöscht und mit den aktuellen Parametern neu berechnet 1 = Additiv , Schwenkebene setzt additiv auf aktive Schwenkebene auf</p> <p>ZEHNER: Werkzeugspitze nachführen ja/nein (nur aktiv, wenn in IBN SCHWENKEN Funktion eingerichtet ist)</p> <p>0 = Werkzeugspitze nicht nachführen 1 = Werkzeugspitze nachführen (TRAORI)</p> <p>HUNDERTER: Werkzeug anstellen / ausrichten (Funktion wird in Eingabemaske SCHWENKEN Werkzeug angezeigt)</p> <p>0 = Werkzeug nicht anstellen 1 = Werkzeug (vorzugweise Radiumfräser) anstellen 2 = Drehwerkzeug ausrichten (wenn B-Achskinematik für Drehtechnologie in IBN Schwenken eingerichtet ist) 3 = Fräswerkzeug ausrichten (wenn B-Achskinematik für Drehtechnologie in IBN Schwenken eingerichtet ist) 9 = reserviert</p> <p>TAUSENDER: interner Parameter Schwenken in JOG</p> <p>ZEHNTAUSENDER: siehe Parameter Richtung _DIR</p> <p>0 = Schwenken "ja" 1 = Schwenken "nein" Richtung "Minus" ³⁾ 2 = Schwenken "nein" Richtung "Plus" ³⁾</p>

Zyklen extern programmieren

16.1 Technologische Zyklen

Nr.	Param Maske	Param intern	Erklärung
			HUNDERTTAUSENDER: siehe Parameter Richtung _DIR
			0 = Kompatibilität 1 = Richtungsauswahl "Minus" optimiert ⁴⁾ 2 = Richtungsauswahl "Plus" optimiert ⁴⁾
4		_MODE ⁵⁾	Schwenkmodus: Auswertung der Schwenkwinkel und der Schwenkreihenfolge (bitcodiert!)
			Bit: 7 6
			0 0: Schwenkwinkel achsweise -> siehe Parameter _A, _B, _C 0 1: Raumwinkel -> siehe Parameter _A, _B ¹⁾ 1 0: Projektionswinkel -> siehe Parameter _A, _B, _C ¹⁾ 1 1: Schwenkmodus Rundachsen direkt -> siehe Parameter _A, _B ¹⁾
			Bit: 5 4 3 2 1 0 (bei Raumwinkeln ohne Bedeutung!)
			x x x x 0 1 1.Drehung _A um X x x x x 1 0 1. Drehung _A um Y x x x x 1 1 1.Drehung _A um Z x x 0 1 x x 2.Drehung _B um X x x 1 0 x x 2.Drehung _B um Y x x 1 1 x x 2.Drehung _B um Z 0 1 x x x x 3.Drehung _C um X 1 0 x x x x 3.Drehung _C um Y 1 1 x x x x 3.Drehung _C um Z
5	X0	_X0	Bezugspunkt X vor der Drehung
6	Y0	_Y0	Bezugspunkt Y vor der Drehung
7	Z0	_Z0	Bezugspunkt Z vor der Drehung
8	X(A)	_A	1. Drehung laut Einstellung in Parameter _MODE
9	Y(B)	_B	2. Drehung laut Einstellung in Parameter _MODE
10	Z(C)	_C	3. Drehung laut Einstellung in Parameter _MODE
11	X1	_X1	Bezugspunkt X nach der Drehung
12	Y1	_Y1	Bezugspunkt Y nach der Drehung
13	Z1	_Z1	Bezugspunkt Z nach der Drehung
14	- oder +	_DIR	Verfahrbewegung der Rundachsen auslösen (default = -1!): -1 = auf kleineren Wert der Rundachse 1 oder 2 positionieren ²⁾ +1 = auf größeren Wert der Rundachse 1 oder 2 positionieren ²⁾ 0 = Schwenken nein (nur Schwenkframe berechnen) ^{1) 3)}
15	FR	_FR_I	Wert (ink) Freifahren in Werkzeugrichtung inkrementell
16		_DMODE	Displaymode
			EINER: Bearbeitungsebene G17/G18/G19
			0 = Kompatibilität, es bleibt die vor Zyklusaufwurf wirksame Ebene aktiv 1 = G17 (nur im Zyklus aktiv) 2 = G18 (nur im Zyklus aktiv) 3 = G19 (nur im Zyklus aktiv)

Hinweis

Sind folgende Übergabeparameter indirekt (als Parameter) programmiert, wird die Eingabemaske nicht rückübersetzt: _FR, _ST, _TC, _MODE, _DIR

- 1) Auswahl möglich, wenn in IBN SCHWENKEN Funktion eingerichtet ist.
- 2) Auswahl möglich, wenn in IBN SCHWENKEN Richtungsbezug auf Rundachse 1 oder 2 eingestellt ist.

kein Auswahlfeld bei Richtungsbezug nein

- 3) Auswahl Schwenken "nein" kann ausgeblendet sein SD 55221 Bit 0

Schwenken "nein" Richtung "Minus" entspricht _DIR = 0 und _ST ZEHNTAUSENDER = 1

Schwenken "nein" Richtung "Plus" entspricht _DIR = 0 und _ST ZEHNTAUSENDER = 2

- 4) Die Richtungsauswahl der Rundachse 1 oder 2 erfolgt auch, wenn sich die Rundachse mit dem Richtungsbezug in Polstellung (Positionswert gleich Null) befindet.

- 5) Beispiel Codierung: Drehung achsweise, Drehreihenfolge ZYX

Binär: 00011011; Dezimal: 27

Die Achsbezeichner XYZ entsprechen den Geometrieachsen des NC-Kanals. Drehungen um die Achsen XYZ dürfen einzeln ausgeführt werden. Z.B. Drehreihenfolge um ZXZ ist in einem Aufruf des CYCLE800 nicht erlaubt

16.1.38 High Speed Settings - CYCLE832

Programmierung

```
CYCLE832 (REAL S_TOL, INT S_TOLM, REAL S_OTOL)
```

Hinweis

Der CYCLE832 entlastet den Maschinenhersteller nicht von notwendigen Optimierungsaufgaben bei der Inbetriebnahme der Maschine. Dies betrifft die Optimierung der an der Bearbeitung beteiligten Achsen und die Einstellungen der NCU (Vorsteuerung, Ruckbegrenzung, usw.).

Parameter

Nr.	Param Maske	Param intern	Erklärung
1	Toleranz	S_TOL	Konturtoleranz Die Konturtoleranz entspricht der Achstoleranz der Geometrieachsen.
2		S_TOLM	Bearbeitungsart (Technologie) EINER: 0 = Abwahl 1 = Schlichten (Finish) 2 = Vorschlichten (Semifinish) 3 = Schruppen (Rough) ZEHNER: 0 = Kompatibilität ¹⁾ bzw. keine Orientierungstoleranz 1 = Orientierungstoleranz im 3.Parameter Um die Lesbarkeit des Zyklenaufrufs zu verbessern, kann der Parameter "Bearbeitungsart" auch im Klartext eingegeben werden. Klartexte sind nicht sprachabhängig. Folgende Eingaben sind zulässig: _FINISH = Schlichten _SEMIFIN = Vorschlichten _ROUGH = Schruppen _ORI_FINISH = Schlichten mit Eingabe einer Orientierungstoleranz _ORI_SEMIFIN = Vorschlichten mit Eingabe einer Orientierungstoleranz _ORI_ROUGH = Schruppen mit Eingabe einer Orientierungstoleranz OFF = Abwahl Hinweis: Die Begriffe sind an die G-Funktionsgruppe 59 (Dynamikmodus für Bahninterpolation) angelehnt. Mit diesen Klartexten werden 3-Achs-Maschinen und Maschinen mit Mehrachsorientierungstransformation (TRAORI) in der Anwendung klar getrennt.

Nr.	Param Maske	Param intern	Erklärung
3	ORI-Toleranz	S_OTOL	Orientierungstoleranz bzw. Versionskennzeichen CYCLE832 Toleranzparameter für die Orientierung des Werkzeugs. Wird benötigt bei der Abarbeitung eines Hochgeschwindigkeitsbearbeitungsprogramms auf Maschinen mit dynamischer Orientierungstransformation (z. B. der 5-Achs-Bearbeitung). Der Parameter S_OTOL muss programmiert werden. Dies gilt auch für Anwendungen auf 3-Achs-Maschinen bei Programmen ohne Orientierung des Werkzeugs (S_OTOL = 1).

1) Orientierungstoleranz abgeleitet von der Konturtoleranz multipliziert mit dem Faktor aus den Zyklen-Settingdaten SD55441 bis SD55443.

Literatur:

Inbetriebnahmehandbuch Basesoftware und Bedien-Software; SINUMERIK Operate (IM9), Kapitel "Funktion High Speed Setting (CYCLE832) konfigurieren"

Hinweis

Bei Abwahl CYCLE832 muss der Parameter S_TOL mit Null übergeben werden.

Beispiel: `CYCLE832(0,0,1)`

Die Syntax `CYCLE832()` ist für Abwahl CYCLE832 ebenfalls erlaubt.

Beispiele

Beispiel 1: CYCLE832 auf 3-Achs-Maschine ohne Orientierungstransformation

a) Zyklenaufruf mit Klartexteingabe

Programmcode	Kommentar
G710	; Maßsystem ist metrisch.
CYCLE832(0.004,_FINISH,1)	; Aufruf CYCLE832 mit: Konturtoleranz = 0,004 mm, Bearbeitungsart: Schlichten
...	; Abarbeitung eines Hochgeschwindigkeitsbearbeitungsprogramms

b) Zyklenaufruf ohne Klartexteingabe

Programmcode	Kommentar
G710	; s. o.
CYCLE832(0.004,1,1)	; s. o.
...	; s. o.

Beispiel 2: CYCLE832 auf 5-Achs-Maschine mit Orientierungstransformation

a) Zyklenaufruf und Abwahl mit Klartexteingabe

Programmcode	Kommentar
G710	; Maßsystem ist metrisch.
TRAORI	; Orientierungstransformation aktivieren.
CYCLE832 (0.3, _ORI_ROUGH, 0.8)	; Aufruf CYCLE832 mit: Konturtoleranz = 0,3 mm, Bearbeitungsart: Schruppen mit Eingabe einer Orientierungstoleranz, Orientierungstoleranz = 0,8 Grad
...	; Abarbeitung eines Hochgeschwindigkeits- bearbeitungsprogramms
CYCLE832 (0, _OFF, 1)	; Konturtoleranz = 0, Bearbeitungsart: Abwahl CYCLE832, Orientierungstoleranz = 0 Grad

b) Zyklenaufruf und Abwahl ohne Klartexteingabe

Programmcode	Kommentar
G710	; s. o.
TRAORI	; s. o.
CYCLE832 (0.3, 13, 0.8)	; s. o.
...	; s. o.
CYCLE832 (0, 0, 1)	; s. o.

Tabellen

17.1 Anweisungen

Legende:

¹⁾ Art der Anweisung:

- A** Adresse
Bezeichner, dem ein Wert zugewiesen wird (z. B. OVR=10). Es gibt auch einige Adressen, die ohne Wertzuweisung eine Funktion ein- oder ausschalten (z. B. CPLON und CPLOF).
- C** Technologischer Zyklus
Vorgefertigtes Teileprogramm (Unterprogramm), in dem ein bestimmter Bearbeitungsvorgang (Zyklus), wie z. B. das Bohren eines Gewindes oder das Fräsen einer Tasche, allgemeingültig realisiert wurde. Die Anpassung an eine konkrete Problemstellung erfolgt über Parameter, die dem Zyklus beim Aufruf übergeben werden.
- F** vordefinierte Funktion
Vordefinierte NCK-Funktion, die einen Rückgabewert liefert.
Der Aufruf der vordefinierten Funktion kann als Operand im Ausdruck stehen.
- G** G-Funktion
G-Funktionen sind in Funktionsgruppen eingeteilt. Es kann nur eine G-Funktion einer Gruppe in einem Satz geschrieben werden. Eine G-Funktion kann modal wirksam sein (bis auf Widerruf durch eine andere Funktion derselben Gruppe), oder sie ist nur für den Satz wirksam, in dem sie steht (satzweise wirksam).
- K** Schlüsselwort
Bezeichner, der die Syntax eines Satzes bestimmt. Einem Schlüsselwort wird kein Wert zugewiesen und mit einem Schlüsselwort kann auch keine NC-Funktion ein-/ausgeschaltet werden.
Beispiele: Kontrollstrukturen (IF, ELSE, ENDIF, WHEN, ...), Programmablauf (GOTOB, GOTO, RET ...)
- O** Operator
Operator für eine mathematische, Vergleichs- oder logische Operation
- P** vordefinierte Prozedur
Vordefinierte NCK-Funktion, die im Unterschied zur vordefinierten Funktion **keinen** Rückgabewert liefert.
- PA** Programmattribut
Programmattribute stehen am Ende der Definitionszeile eines Unterprogramms:
PROC <Programmname>(…) <Programmattribut>
Sie bestimmen das Verhalten beim Ablauf des Unterprogramms.

17.1 Anweisungen

- 2) Wirksamkeit der Anweisung:
 - m modal
 - s satzweise
- 3) Programmierbarkeit im Teileprogramm:
 - + programmierbar
 - nicht programmierbar
- 4) Programmierbarkeit in Synchronaktionen:
 - + programmierbar
 - nicht programmierbar
 - T programmierbar nur in Technologiezyklen
- 5) Verweis auf das Dokument, das die ausführliche Beschreibung der Anweisung enthält:
 - PGs/* Programmierhandbuch Grundlagen
 - PGAs/* Programmierhandbuch Arbeitsvorbereitung
 - BNMs/* Programmierhandbuch Messzyklen
 - BHDs/* Bedienhandbuch Drehen
 - BHFsl* Bedienhandbuch Fräsen
 - FB1sl ()* Funktionshandbuch Grundfunktionen (mit dem alphanumerischen Kürzel der betreffenden Funktionsbeschreibung in Klammern)
 - FB2sl ()* Funktionshandbuch Erweiterungsfunktionen (mit dem alphanumerischen Kürzel der betreffenden Funktionsbeschreibung in Klammern)
 - FB3sl ()* Funktionshandbuch Sonderfunktionen (mit dem alphanumerischen Kürzel der betreffenden Funktionsbeschreibung in Klammern)
 - FBSisl* Funktionshandbuch Safety Integrated
 - FBSY* Funktionshandbuch Synchronaktionen
 - FBWsl* Funktionshandbuch Werkzeugverwaltung
- 6) Standardeinstellung bei Programmanfang (im Auslieferungsstand der Steuerung, wenn nichts anderes programmiert ist).

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
:	O	NC-Hauptsatznummer, Sprungmarkenabschluss, Kettungsoperator		+		<i>PGAs/</i> Rechenfunktionen (Seite 73)
*	O	Operator für Multiplikation		+		<i>PGAs/</i> Rechenfunktionen (Seite 73)
+	O	Operator für Addition		+		<i>PGAs/</i> Rechenfunktionen (Seite 73)
-	O	Operator für Subtraktion		+		<i>PGAs/</i> Rechenfunktionen (Seite 73)
<	O	Vergleichsoperator, kleiner		+		<i>PGAs/</i> Rechenfunktionen (Seite 73)
<<	O	Verkettungsoperator für Strings		+		<i>PGAs/</i> Rechenfunktionen (Seite 73)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
<=	O	Vergleichsoperator, kleiner gleich		+		<i>PGAs/</i> Rechenfunktionen (Seite 73)
=	O	Zuweisungsoperator		+		<i>PGAs/</i> Rechenfunktionen (Seite 73)
>=	O	Vergleichsoperator, größer gleich		+		<i>PGAs/</i> Rechenfunktionen (Seite 73)
/	O	Operator für Division		+		<i>PGAs/</i> Rechenfunktionen (Seite 73)
/0 /7		Satz wird ausgeblendet (1. Ausblende Ebene) Satz wird ausgeblendet (8. Ausblende Ebene)		+		<i>PGs/</i>
A	A	Achsname	m/s	+		<i>PGAs/</i> Programmierung der Werkzeugorientierung (A..., B..., C..., LEAD, TILT) (Seite 338)
A2	A	Werkzeugorientierung: RPY- oder Eulerwinkel	s	+		<i>PGAs/</i> Programmierung der Werkzeugorientierung (A..., B..., C..., LEAD, TILT) (Seite 338)
A3	A	Werkzeugorientierung: Vektorkomponente Richtung- /Flächennormal	s	+		<i>PGAs/</i> Programmierung der Werkzeugorientierung (A..., B..., C..., LEAD, TILT) (Seite 338)
A4	A	Werkzeugorientierung: Flächennormalvektor für den Satzanfang	s	+		<i>PGAs/</i> Stirnfräsen (A4, B4, C4, A5, B5, C5) (Seite 345)
A5	A	Werkzeugorientierung: Flächennormalenvektor für das Satzende	s	+		<i>PGAs/</i> Stirnfräsen (A4, B4, C4, A5, B5, C5) (Seite 345)
ABS	F	Absolutwert (Betrag)		+	+	<i>PGAs/</i> Rechenfunktionen (Seite 73)
AC	K	absolute Maßangabe von Koordinaten/Positionen	s	+		<i>PGs/</i>
ACC	K	Beeinflussung der aktuellen axialen Beschleunigung	m	+	+	<i>PGs/</i>
ACCLIMA	K	Beeinflussung der aktuellen maximalen axialen Beschleunigung	m	+	+	<i>PGAs/</i> Beeinflussung der Beschleunigung bei Folgeachsen (VELOLIMA, ACCLIMA, JERKLIMA) (Seite 478)
ACN	K	absolute Maßangabe für Rundachsen, Position in negativer Richtung anfahren	s	+		<i>PGs/</i>
ACOS	F	Arcus-Cosinus (Trigon. Funktion)		+	+	<i>PGAs/</i> Rechenfunktionen (Seite 73)

17.1 Anweisungen

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
ACP	K	absolute Maßangabe für Rundachsen, Position in positiver Richtung anfahren	s	+		<i>PGs/</i>
ACTBLOCNO	P	Ausgabe der aktuellen Satznummer eines Alarmsatzes, auch wenn "aktuelle Satzanzeige unterdrückt" (DISPLOF) aktiv ist!		+		<i>PGAs/</i> Aktuelle Satzanzeige unterdrücken (DISPLOF, DISPLON, ACTBLOCNO) (Seite 187)
ADDFRAME	F	Einrechnung und evtl. Aktivierung eines gemessenen Frames		+	-	<i>PGAs/</i> , <i>FB1s/</i> (K2) Frame-Berechnung aus 3 Messpunkten im Raum (MEAFRAME) (Seite 311)
ADIS	A	Überschleifabstand für Bahnfunktionen G1, G2, G3, ...	m	+		<i>PGs/</i>
ADISPOS	A	Überschleifabstand für Eilgang G0	m	+		<i>PGs/</i>
ADISPOSA	P	Größe des Toleranzfenster für IPOBRKA	m	+	+	<i>PGAs/</i> Programmierbares Bewegungsendekriterium (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA) (Seite 289)
ALF	A	Schnellabhebewinkel	m	+		<i>PGAs/</i> Schnellabheben von der Kontur (SETINT LIFTFAST, ALF) (Seite 136)
AMIRROR	G	Programmierbare Spiegelung	s	+		<i>PGs/</i>
AND	K	Logisches UND		+		<i>PGAs/</i> Vergleichs- und logische Operationen (Seite 75)
ANG	A	Konturzug-Winkel	s	+		<i>PGs/</i>
AP	A	Polarwinkel	m/s	+		<i>PGs/</i>
APR	K	Zugriffsschutz lesen / anzeigen		+		<i>PGAs/</i> Attribut: Zugriffsrechte (APR, APW, APRP, APWP, APRB, APWB) (Seite 41)
APRB	K	Zugriffsrecht lesen, BTSS		+		<i>PGAs/</i> Attribut: Zugriffsrechte (APR, APW, APRP, APWP, APRB, APWB) (Seite 41)
APRP	K	Zugriffsrecht lesen, Teileprogramm		+		<i>PGAs/</i> Attribut: Zugriffsrechte (APR, APW, APRP, APWP, APRB, APWB) (Seite 41)
APW	K	Zugriffsschutz schreiben		+		<i>PGAs/</i> Attribut: Zugriffsrechte (APR, APW, APRP, APWP, APRB, APWB) (Seite 41)
APWB	K	Zugriffsrecht schreiben, BTSS		+		<i>PGAs/</i> Attribut: Zugriffsrechte (APR, APW, APRP, APWP, APRB, APWB) (Seite 41)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
APWP	K	Zugriffsrecht schreiben, Teileprogramm		+		<i>PGAsI</i> Attribut: Zugriffsrechte (APR, APW, APRP, APWP, APRB, APWB) (Seite 41)
APX	K	Definition des Zugriffsschutzes für die Ausführung des angegebenen Sprachelements		+		<i>PGAsI</i> Redefinition von Systemvariablen, Anwendervariablen und NC-Sprachbefehlen (REDEF) (Seite 31)
AR	A	Öffnungswinkel	m/s	+		<i>PGsI</i>
AROT	G	Programmierbare Drehung	s	+		<i>PGsI</i>
AROTS	G	Programmierbare Framedrehungen mit Raumwinkeln	s	+		<i>PGsI</i>
AS	K	Makro-Definition		+		<i>PGAsI</i> Makrotechnik (DEFINE ... AS) (Seite 221)
ASCALE	G	Programmierbare Skalierung	s	+		<i>PGsI</i>
ASIN	F	Rechenfunktion, Arcussinus		+	+	<i>PGAsI</i> Rechenfunktionen (Seite 73)
ASPLINE	G	Akima-Spline	m	+		<i>PGAsI</i> Spline-Interpolation (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) (Seite 250)
ATAN2	F	Arcus-Tangens2		+	+	<i>PGAsI</i> Rechenfunktionen (Seite 73)
ATOL	K	achsenspezifische Toleranz für Kompressor-Funktionen, Orientierungsglättung und Überschleifarten		+		<i>PGAsI</i> Programmierbare Kontur-/Orientierungstoleranz (CTOL, OTOL, ATOL) (Seite 509)
ATRANS	G	additive programmierbare Verschiebung	s	+		<i>PGsI</i>
AUXFUDEL	P	Hilfsfunktion kanalspezifisch aus der globalen Liste löschen		+	-	<i>FB1sI (H2)</i>
AUXFUDELG	P	Alle Hilfsfunktionen einer Hilfsfunktionsgruppe kanalspezifisch aus der globalen Liste löschen		+	-	<i>FB1sI (H2)</i>
AUXFUMSEQ	P	Ausgabe-Reihenfolge für M-Hilfsfunktionen ermitteln		+	-	<i>FB1sI (H2)</i>
AUXFUSYNC	P	Aus der globalen Liste der Hilfsfunktionen einen vollständigen Teileprogrammsatz für das kanalspezifische SERUPRO-Ende-ASUP als String generieren		+	-	<i>FB1sI (H2)</i>

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
AX	K	Variabler Achsbezeichner	m/s	+		<i>PGAs/</i> Achsfunktionen (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) (Seite 599)
AXCTSWE	P	Achscontainer drehen		+	-	<i>PGAs/</i> Achscontainer (AXCTSWE, AXCTSWED, AXCTSWEC) (Seite 606)
AXCTSWEC	P	Freigabe zur Achscontainer-Drehung zurücknehmen		+	+	<i>PGAs/</i> Achscontainer (AXCTSWE, AXCTSWED, AXCTSWEC) (Seite 606)
AXCTSWED	P	Achscontainer drehen (Befehlsvariante für die Inbetriebnahme!)		+	-	<i>PGAs/</i> Achscontainer (AXCTSWE, AXCTSWED, AXCTSWEC) (Seite 606)
AXIS	K	Achsbezeichner, Achsadresse		+		<i>PGAs/</i> Definition von Anwendervariablen (DEF) (Seite 25)
AXNAME	F	Konvertiert Eingangsstring in Achsbezeichner		+	-	<i>PGAs/</i> Achsfunktionen (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) (Seite 599)
AXSTRING	F	Konvertiert den String Spindelnummer		+	-	<i>PGAs/</i> Achsfunktionen (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) (Seite 599)
AXTOCHAN	P	Achse für einen bestimmten Kanal anfordern. Ist vom NC-Programm und aus Synchronaktion möglich.		+	+	<i>PGAs/</i> Achse einem anderen Kanal übergeben (AXTOCHAN) (Seite 148)
AXTOINT	F	Datentyp einer Achsvariablen von AXIS zu INT wandeln		+	-	<i>PGAs/</i> Explizite Datentypwandlungen (AXTOINT, INTTOAX) (Seite 56)
AXTOSPI	F	konvertiert Achsbezeichner in einen Spindelindex um		+	-	<i>PGAs/</i> Achsfunktionen (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) (Seite 599)
B	A	Achsname	m/s	+		<i>PGAs/</i> Programmierung der Werkzeugorientierung (A..., B..., C..., LEAD, TILT) (Seite 338)
B2	A	Werkzeugorientierung: RPY- oder Eulerwinkel	s	+		<i>PGAs/</i> Programmierung der Werkzeugorientierung (A..., B..., C..., LEAD, TILT) (Seite 338)
B3	A	Werkzeugorientierung: Vektorkomponente Richtung-/Flächen-normal	s	+		<i>PGAs/</i> Programmierung der Werkzeugorientierung (A..., B..., C..., LEAD, TILT) (Seite 338)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
B4	A	Werkzeugorientierung: Flächennormalvektor für den Satzanfang	s	+		<i>PGAs/</i> Stirnfräsen (A4, B4, C4, A5, B5, C5) (Seite 345)
B5	A	Werkzeugorientierung: Flächennormalenvektor für das Satzende	s	+		<i>PGAs/</i> Stirnfräsen (A4, B4, C4, A5, B5, C5) (Seite 345)
B_AND	O	Bitweises UND		+		<i>PGAs/</i> Vergleichs- und logische Operationen (Seite 75)
B_OR	O	Bitweises ODER		+		<i>PGAs/</i> Vergleichs- und logische Operationen (Seite 75)
B_NOT	O	Bitweise Negierung		+		<i>PGAs/</i> Vergleichs- und logische Operationen (Seite 75)
B_XOR	O	Bitweises Exklusiv-ODER		+		<i>PGAs/</i> Vergleichs- und logische Operationen (Seite 75)
BAUTO	G	Definieren des ersten Spline- Abschnitts durch die nachfolgenden 3 Punkte	m	+		<i>PGAs/</i> Spline-Interpolation (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) (Seite 250)
BLOCK	K	Definiert zusammen mit dem Schlüsselwort TO den abzuarbeitenden Programmteil in einem indirekten Unterprogrammlauf		+		<i>PGAs/</i> Indirekter Unterprogrammaufruf mit Angabe des auszuführenden Programmteils (CALL BLOCK ... TO ...) (Seite 207)
BLSYNC	K	Bearbeitung der Interruptroutine soll erst mit dem nächsten Satzwechsel beginnen		+		<i>PGAs/</i> Interruptroutine zuordnen und starten (SETINT, PRIO, BLSYNC) (Seite 132)
BNAT ⁶⁾	G	Natürlicher Übergang zum ersten Spline-Satz	m	+		<i>PGAs/</i> Spline-Interpolation (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) (Seite 250)
BOOL	K	Datentyp: Wahrheitswerte TRUE/FALSE bzw. 1/0		+		<i>PGAs/</i> Definition von Anwendervariablen (DEF) (Seite 25)
BOUND	F	Prüft, ob Wert innerhalb des definierten Wertebereichs liegt. Gleichheit gibt Prüfwert zurück.		+	+	<i>PGAs/</i> Minimum, Maximum und Bereich von Variablen (MINVAL, MAXVAL, BOUND) (Seite 79)
BRISK ⁶⁾	G	Sprungförmige Bahnbeschleunigung	m	+		<i>PGAs/</i> Beschleunigungsmodus (BRISK, BRISKA, SOFT, SOFTA, DRIVE, DRIVEA) (Seite 475)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
BRISKA	P	Sprungförmige Bahnbeschleunigung für die programmierten Achsen einschalten		+	-	<i>PGAs/</i> Beschleunigungsmodus (BRISK, BRISKA, SOFT, SOFTA, DRIVE, DRIVEA) (Seite 475)
BSPLINE	G	B-Spline	m	+		<i>PGAs/</i> Spline-Interpolation (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) (Seite 250)
BTAN	G	Tangentialem Übergang zum ersten Spline-Satz	m	+		<i>PGAs/</i> Spline-Interpolation (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) (Seite 250)
C	A	Achsname	m/s	+		<i>PGAs/</i> Programmierung der Werkzeugorientierung (A..., B..., C..., LEAD, TILT) (Seite 338)
C2	A	Werkzeugorientierung: RPY- oder Eulerwinkel	s	+		<i>PGAs/</i> Programmierung der Werkzeugorientierung (A..., B..., C..., LEAD, TILT) (Seite 338)
C3	A	Werkzeugorientierung: Vektorkomponente Richtung-/Flächen-normal	s	+		<i>PGAs/</i> Programmierung der Werkzeugorientierung (A..., B..., C..., LEAD, TILT) (Seite 338)
C4	A	Werkzeugorientierung: Flächennormalvektor für den Satzanfang	s	+		<i>PGAs/</i> Stirnfräsen (A4, B4, C4, A5, B5, C5) (Seite 345)
C5	A	Werkzeugorientierung: Flächennormalvektor für das Satzende	s	+		<i>PGAs/</i> Stirnfräsen (A4, B4, C4, A5, B5, C5) (Seite 345)
CAC	K	Absolutes Anfahren einer Position		+		<i>PGAs/</i> Codierte Positionen anfahren (CAC, CIC, CDC, CACP, CACN) (Seite 249)
CACN	K	In Tabelle abgelegter Wert wird absolut in negativer Richtung angefahren		+		<i>PGAs/</i> Codierte Positionen anfahren (CAC, CIC, CDC, CACP, CACN) (Seite 249)
CACP	K	In Tabelle abgelegter Wert wird absolut in positiver Richtung angefahren		+		<i>PGAs/</i> Codierte Positionen anfahren (CAC, CIC, CDC, CACP, CACN) (Seite 249)
CALCDAT	F	Berechnet Radius und Mittelpunkt eines Kreises aus 3 oder 4 Punkten		+	-	<i>PGAs/</i> Kreisdaten berechnen (CALCDAT) (Seite 659)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
CALCPOSI	F	Überprüfung auf Schutzbereichsverletzung, Arbeitsfeldbegrenzung und Softwarelimits		+	-	<i>PGAs/</i> Überprüfung auf Schutzbereichsverletzung, Arbeitsfeldbegrenzung und Softwarelimits (CALCPOSI) (Seite 241)
CALL	K	Indirekter Unterprogrammaufruf		+		<i>PGAs/</i> Indirekter Unterprogrammaufruf (CALL) (Seite 206)
CALLPATH	P	Programmierbarer Suchpfad bei Unterprogrammaufrufen		+	-	<i>PGAs/</i> Suchpfad bei Unterprogrammaufrufen erweitern (CALLPATH) (Seite 211)
CANCEL	P	Modale Synchronaktion abbrechen		+	-	<i>FBSY</i>
CASE	K	Bedingte Programmverzweigung		+		<i>PGAs/</i> Programmverzweigung (CASE ... OF ... DEFAULT ...) (Seite 106)
CDC	K	Direktes Anfahren einer Position		+		<i>PGAs/</i> Codierte Positionen anfahren (CAC, CIC, CDC, CACP, CACN) (Seite 249)
CDOF ⁶⁾	G	Kollisionsüberwachung AUS	m	+		<i>PGs/</i>
CDOF2	G	Kollisionsüberwachung AUS, bei 3D-Umfangsfräsen	m	+		<i>PGs/</i>
CDON	G	Kollisionsüberwachung EIN	m	+		<i>PGs/</i>
CFC ⁶⁾	G	Konstanter Vorschub an der Kontur	m	+		<i>PGs/</i>
CFIN	G	Konstanter Vorschub nur bei Innenkrümmung, nicht bei Außenkrümmung	m	+		<i>PGs/</i>
CFINE	F	Zuweisung der Feinverschiebung an eine FRAME-Variable		+	-	<i>PGAs/</i> Grob- und Feinverschiebung (CFINE, CTRANS) (Seite 306)
CFTCP	G	Konstanter Vorschub im Werkzeugschneiden-Bezugspunkt, Mittelpunkts-bahn	m	+		<i>PGs/</i>
CHAN	K	Spezifizierung des Gültigkeitsbereichs von Daten		+		<i>PGAs/</i> Definition von Anwendervariablen (DEF) (Seite 25)
CHANDATA	P	Kanalnummer für Kanaldatenzugriffe einstellen		+	-	<i>PGAs/</i> Arbeitsspeicher (CHANDATA, COMPLETE, INITIAL) (Seite 230)
CHAR	K	Datentyp: ASCII-Zeichen		+		<i>PGAs/</i> Definition von Anwendervariablen (DEF) (Seite 25)
CHECKSUM	F	Bildet die Checksumme über ein Feld als STRING mit einer festgesetzten Länge		+	-	<i>PGAs/</i> Checksummenberechnung über ein Feld (CHECKSUM) (Seite 166)

17.1 Anweisungen

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
CHF	A	Fase; Wert = Länge der Fase	s	+		<i>PGs/</i>
CHKDM	F	Prüfung der Eindeutigkeit innerhalb eines Magazins		+	-	<i>FBWs/</i>
CHKDNO	F	Eindeutigkeitsprüfung der D- Nummern		+	-	<i>PGAs/</i> Freie D-Nummernvergabe: D-Nummern prüfen (CHKDNO) (Seite 444)
CHR	A	Fase; Wert = Länge der Fase in Bewegungsrichtung		+		<i>PGs/</i>
CIC	K	Inkrementelles Anfahren einer Position		+		<i>PGAs/</i> Codierte Positionen anfahren (CAC, CIC, CDC, CACP, CACN) (Seite 249)
CIP	G	Kreisinterpolation über Zwischenpunkt	m	+		<i>PGs/</i>
CLEARM	P	Rücksetzen einer/mehrerer Marken für Kanalkoordinierung		+	+	<i>PGAs/</i> Programmkoordinierung (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) (Seite 124)
CLRINT	P	Interrupt abwählen		+	-	<i>PGAs/</i> Zuordnung einer Interruptroutine löschen (CLRINT) (Seite 135)
CMIRROR	F	Spiegeln an einer Koordinatenachse		+	-	<i>PGAs/</i> Rechenfunktionen (Seite 73)
COARSEA	K	Bewegungsende beim Erreichen von "Genauhalt Grob"	m	+		<i>PGAs/</i> Programmierbares Bewegungsendekriterium (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA) (Seite 289)
COMPCAD	G	Kompressor EIN: Optimierte Oberflächen-güte bei CAD- Programmen	m	+		<i>PGAs/</i> NC-Satz-Kompression (COMPON, COMPCURV, COMPCAD, COMPOF) (Seite 262)
COMPCURV	G	Kompressor EIN: krümmungsstetige Polynome	m	+		<i>PGAs/</i> NC-Satz-Kompression (COMPON, COMPCURV, COMPCAD, COMPOF) (Seite 262)
COMPLETE		Steueranweisung für das Aus- und Einlesen von Daten		+		<i>PGAs/</i> Arbeitsspeicher (CHANDATA, COMPLETE, INITIAL) (Seite 230)
COMPOF ⁶⁾	G	Kompressor AUS	m	+		<i>PGAs/</i> NC-Satz-Kompression (COMPON, COMPCURV, COMPCAD, COMPOF) (Seite 262)
COMPON	G	Kompressor EIN		+		<i>PGAs/</i> NC-Satz-Kompression (COMPON, COMPCURV, COMPCAD, COMPOF) (Seite 262)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
CONTDCON	P	Konturdecodierung in Tabellenform EIN		+	-	<i>PGAs/</i> Codierte Konturtable erstellen (CONTDCON) (Seite 652)
CONTPRON	P	Referenzaufbereitung einschalten		+	-	<i>PGAs/</i> Konturtable erstellen (CONTPRON) (Seite 646)
CORROF	P	Alle aktiven überlagerten Bewegungen werden abgewählt.		+	-	<i>PGs/</i>
COS	F	Cosinus (Trigon. Funktion)		+	+	<i>PGAs/</i> Rechenfunktionen (Seite 73)
COUPDEF	P	Definition ELG-Verband / Synchronspindel-Verband		+	-	<i>PGAs/</i> Synchronspindel: Programmierung (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) (Seite 554)
COUPDEL	P	ELG-Verband löschen		+	-	<i>PGAs/</i> Synchronspindel: Programmierung (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) (Seite 554)
COUPOF	P	ELG-Verband / Synchronspindel-paar EIN		+	-	<i>PGAs/</i> Synchronspindel: Programmierung (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) (Seite 554)
COUPOFS	P	Ausschalten ELG-Verband / Synchronspindel-paar mit Stopp der Folgespindel		+	-	<i>PGAs/</i> Synchronspindel: Programmierung (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) (Seite 554)
COUPON	P	ELG-Verband / Synchronspindel-paar EIN		+	-	<i>PGAs/</i> Synchronspindel: Programmierung (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) (Seite 554)
COUPONC	P	Einschalten ELG-Verband / Synchronspindel-paar mit vorhergehender Programmierung übernehmen		+	-	<i>PGAs/</i> Synchronspindel: Programmierung (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) (Seite 554)
COUPRES	P	ELG-Verband rücksetzen		+	-	<i>PGAs/</i> Synchronspindel: Programmierung (COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC) (Seite 554)
CP ⁶⁾	G	Bahnbewegung	m	+		<i>PGAs/</i> Kartesisches PTP-Fahren (Seite 390)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
CPBC	K	Generische Kopplung: Satzwechselkriterium		+	+	FB3sl (M3)
CPDEF	K	Generische Kopplung: Anlegen eines Koppelmoduls		+	+	FB3sl (M3)
CPDEL	K	Generische Kopplung: Löschen eines Koppelmoduls		+	+	FB3sl (M3)
CPFMOF	K	Generische Kopplung: Verhalten der Folgeachse beim vollständigen Ausschalten		+	+	FB3sl (M3)
CPFMON	K	Generische Kopplung: Verhalten der Folgeachse beim Einschalten		+	+	FB3sl (M3)
CPFMSON	K	Generische Kopplung: Synchronisationsmodus		+	+	FB3sl (M3)
CPFPOS	K	Generische Kopplung: Synchronposition der Folgeachse		+	+	FB3sl (M3)
CPFRS	K	Generische Kopplung: Koordinatenbezugssystem		+	+	FB3sl (M3)
CPLA	K	Generische Kopplung: Definition einer Leitachse		+	-	FB3sl (M3)
CPLCTID	K	Generische Kopplung: Nummer der Kurventabelle		+	+	FB3sl (M3)
CPLDEF	K	Generische Kopplung: Definition einer Leitachse und Anlegen eines Koppelmoduls		+	+	FB3sl (M3)
CPLDEL	K	Generische Kopplung: Löschen einer Leitachse eines Koppelmoduls		+	+	FB3sl (M3)
CPLDEN	K	Generische Kopplung: Nenner des Koppelfaktors		+	+	FB3sl (M3)
CPLINSC	K	Generische Kopplung: Skalierfaktor für den Eingangswert einer Leitachse		+	+	FB3sl (M3)
CPLINTR	K	Generische Kopplung: Verschiebewert für den Eingangswert einer Leitachse		+	+	FB3sl (M3)
CPLNUM	K	Generische Kopplung: Zähler des Koppelfaktors		+	+	FB3sl (M3)
CPLOF	K	Generische Kopplung: Ausschalten einer Leitachse eines Koppelmoduls		+	+	FB3sl (M3)
CPLON	K	Generische Kopplung: Einschalten einer Leitachse eines Koppelmoduls		+	+	FB3sl (M3)
CPLOUTSC	K	Generische Kopplung: Skalierfaktor für den Ausgangswert einer Kopplung		+	+	FB3sl (M3)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
CPLOUTTR	K	Generische Kopplung: Verschiebewert für den Ausgangswert einer Kopplung		+	+	<i>FB3sl (M3)</i>
CPLPOS	K	Generische Kopplung: Synchronposition der Leitachse		+	+	<i>FB3sl (M3)</i>
CPLSETVAL	K	Generische Kopplung: Kopplungsbezug		+	+	<i>FB3sl (M3)</i>
CPMALARM	K	Generische Kopplung: Unterdrückung spezieller kopplungsbezogener Alarmausgaben		+	+	<i>FB3sl (M3)</i>
CPMBRAKE	K	Generische Kopplung: Verhalten der Folgeachse bei bestimmten Stoppsignalen und -kommandos		+	-	<i>FB3sl (M3)</i>
CPMPRT	K	Generische Kopplung: Kopplungsverhalten beim Teileprogrammstart unter Suchlauf via Programmtest		+	+	<i>FB3sl (M3)</i>
CPMRESET	K	Generische Kopplung: Kopplungsverhalten bei RESET		+	+	<i>FB3sl (M3)</i>
CPMSTART	K	Generische Kopplung: Kopplungsverhalten beim Teileprogrammstart		+	+	<i>FB3sl (M3)</i>
CPMVDI	K	Generische Kopplung: Verhalten der Folgeachse auf bestimmte NC/PLC- Nahtstellensignale		+	+	<i>FB3sl (M3)</i>
CPOF	K	Generische Kopplung: Ausschalten eines Koppelmoduls		+	+	<i>FB3sl (M3)</i>
CPON	K	Generische Kopplung: Einschalten eines Koppelmoduls		+	+	<i>FB3sl (M3)</i>
CPRECOF ⁶⁾	G	Programmierbare Konturgenauigkeit AUS	m	+		<i>PGAsl</i> Programmierbare Konturgenauigkeit (CPRECON, CPRECOF) (Seite 483)
CPRECON	G	Programmierbare Konturgenauigkeit EIN	m	+		<i>PGAsl</i> Programmierbare Konturgenauigkeit (CPRECON, CPRECOF) (Seite 483)
CPRES	K	Generische Kopplung: Aktiviert die projektierten Daten der Synchronspindelkopplung		+	-	<i>FB3sl (M3)</i>
CPROT	P	Kanalspezifischer Schutzbereich EIN/AUS		+	-	<i>PGAsl</i> Schutzbereiche aktivieren/deaktivieren (CPROT, NPROT) (Seite 237)
CPROTDEF	P	Definition eines kanalspezifischen Schutzbereichs		+	-	<i>PGAsl</i> Festlegung der Schutzbereiche (CPROTDEF, NPROTDEF) (Seite 233)

17.1 Anweisungen

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
CPSETTYPE	K	Generische Kopplung: Kopplungstyp		+	+	<i>FB3sl (M3)</i>
CPSYNCOF	K	Generische Kopplung: Schwellwert für den Positionssynchronlauf "Grob"		+	+	<i>FB3sl (M3)</i>
CPSYNCOF2	K	Generische Kopplung: Schwellwert für den Positionssynchronlauf "Grob" 2		+	+	<i>FB3sl (M3)</i>
CPSYNCOV	K	Generische Kopplung: Schwellwert für den Geschwindigkeitssynchronlauf "Grob"		+	+	<i>FB3sl (M3)</i>
CPSYNFIP	K	Generische Kopplung: Schwellwert für den Positionssynchronlauf "Fein"		+	+	<i>FB3sl (M3)</i>
CPSYNFIP2	K	Generische Kopplung: Schwellwert für den Positionssynchronlauf "Fein" 2		+	+	<i>FB3sl (M3)</i>
CPSYNFIV	K	Generische Kopplung: Schwellwert für den Geschwindigkeitssynchronlauf "Fein"		+	+	<i>FB3sl (M3)</i>
CR	A	Kreisradius	s	+		<i>PGsl</i>
CROT	F	Drehung des aktuellen Koordinatensystems		+	-	<i>PGAsl</i> Rechenfunktionen (Seite 73)
CROTS	F	Programmierbare Framedrehungen mit Raumwinkeln (Drehung in den angegebenen Achsen)	s	+	-	<i>PGsl</i>
CRPL	F	Frame-Drehung in einer beliebigen Ebene		+	-	<i>FB1sl (K2)</i>
CSCALE	F	Maßstabsfaktor für mehrere Achsen		+	-	<i>PGAsl</i> Rechenfunktionen (Seite 73)
CSPLINE	F	Kubischer Spline	m	+		<i>PGAsl</i> Spline-Interpolation (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) (Seite 250)
CT	G	Kreis mit tangentialem Übergang	m	+		<i>PGsl</i>
CTAB	F	Ermittle Folgeachsposition anhand der Leitachsposition aus Kurventabelle		+	+	<i>PGAsl</i> Kurventabellenwerte lesen (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) (Seite 532)
CTABDEF	P	Tabellendefinition EIN		+	-	<i>PGAsl</i> Kurventabellen definieren (CTABDEF, CATBEND) (Seite 521)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
CTABDEL	P	Kurventabelle löschen		+	-	<i>PGAs/</i> Kurventabellen löschen (CTABDEL) (Seite 528)
CTABEND	P	Tabellendefinition AUS		+	-	<i>PGAs/</i> Kurventabellen definieren (CTABDEF, CATBEND) (Seite 521)
CTABEXISTS	F	Prüft die Kurventabelle mit der Nummer n		+	+	<i>PGAs/</i> Vorhandensein einer Kurventabelle prüfen (CTABEXISTS) (Seite 527)
CTABFNO	F	Anzahl der noch möglichen Kurventabellen im Speicher		+	+	<i>PGAs/</i> Kurventabellen: Ressourcennutzung prüfen (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) (Seite 537)
CTABFPOL	F	Anzahl der noch möglichen Polynome im Speicher		+	+	<i>PGAs/</i> Kurventabellen: Ressourcennutzung prüfen (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) (Seite 537)
CTABFSEG	F	Anzahl der noch möglichen Kurvensegmente im Speicher		+	+	<i>PGAs/</i> Kurventabellen: Ressourcennutzung prüfen (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) (Seite 537)
CTABID	F	Liefert Tabellen-Nummer der n- ten Kurventabelle		+	+	<i>PGAs/</i> Kurventabellen: Tabelleneigenschaften ermitteln (CTABID, CTABISLOCK, CTABMENTYP, CTABPERIOD) (Seite 530)
CTABINV	F	Ermittle Leitachsposition anhand der Folgeachsposition aus Kurventabelle		+	+	<i>PGAs/</i> Kurventabellenwerte lesen (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) (Seite 532)
CTABISLOCK	F	Gibt den Sperrzustand der Kurventabelle mit der Nummer n zurück		+	+	<i>PGAs/</i> Kurventabellen: Tabelleneigenschaften ermitteln (CTABID, CTABISLOCK, CTABMENTYP, CTABPERIOD) (Seite 530)
CTABLOCK	P	Löschen und Über-schreiben, sperrern		+	+	<i>PGAs/</i> Kurventabellen gegen Löschen und Überschreiben sperren (CTABLOCK, CTABUNLOCK) (Seite 529)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
CTABMEMTYP	F	Gibt den Speicher zurück, in dem die Kurventabelle mit der Nummer n angelegt ist.		+	+	<i>PGAs/</i> Kurventabellen: Tabelleneigenschaften ermitteln (CTABID, CTABISLOCK, CTABMEMTYP, CTABPERIOD) (Seite 530)
CTABMPOL	F	Anzahl der maximal möglichen Polynome im Speicher		+	+	<i>PGAs/</i> Kurventabellen: Ressourcennutzung prüfen (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) (Seite 537)
CTABMSEG	F	Anzahl der maximal möglichen Kurvensegmente im Speicher		+	+	<i>PGAs/</i> Kurventabellen: Ressourcennutzung prüfen (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) (Seite 537)
CTABNO	F	Anzahl der definierten Kurventabellen im SRAM oder DRAM		+	+	<i>FB3s/ (M3)</i>
CTABNOMEM	F	Anzahl der definierten Kurventabellen im SRAM oder DRAM		+	+	<i>PGAs/</i> Kurventabellen: Ressourcennutzung prüfen (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) (Seite 537)
CTABPERIOD	F	Gibt die Tabellenperiodizität der Kurventabelle mit der Nummer n zurück		+	+	<i>PGAs/</i> Kurventabellen: Tabelleneigenschaften ermitteln (CTABID, CTABISLOCK, CTABMEMTYP, CTABPERIOD) (Seite 530)
CTABPOL	F	Anzahl der bereits verwendeten Polynome im Speicher		+	+	<i>PGAs/</i> Kurventabellen: Ressourcennutzung prüfen (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) (Seite 537)
CTABPOLID	F	Anzahl der von der Kurventabelle mit der Nummer n verwendeten Kurvenpolynome		+	+	<i>PGAs/</i> Kurventabellen: Ressourcennutzung prüfen (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) (Seite 537)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
CTABSEG	F	Anzahl der bereits verwendeten Kurvensegmente im Speicher		+	+	<i>PGAs/</i> Kurventabellen: Ressourcennutzung prüfen (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) (Seite 537)
CTABSEGID	F	Anzahl der von der Kurventabelle mit der Nummer n verwendeten Kurvensegmente		+	+	<i>PGAs/</i> Kurventabellen: Ressourcennutzung prüfen (CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL) (Seite 537)
CTABSEV	F	Liefert den Endwert der Folgeachse eines Segments der Kurventabelle		+	+	<i>PGAs/</i> Kurventabellenwerte lesen (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) (Seite 532)
CTABSSV	F	Liefert den Startwert der Folgeachse eines Segments der Kurventabelle		+	+	<i>PGAs/</i> Kurventabellenwerte lesen (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) (Seite 532)
CTABTEP	F	Liefert den Wert der Leitachse am Kurventabellen-Ende		+	+	<i>PGAs/</i> Kurventabellenwerte lesen (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) (Seite 532)
CTABTEV	F	Liefert den Wert der Folgeachse am Kurventabellen-Ende		+	+	<i>PGAs/</i> Kurventabellenwerte lesen (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) (Seite 532)
CTABTMAX	F	Liefert Maximalwert der Folgeachse der Kurventabelle		+	+	<i>PGAs/</i> Kurventabellenwerte lesen (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) (Seite 532)
CTABTMIN	F	Liefert Minimalwert der Folgeachse der Kurventabelle		+	+	<i>PGAs/</i> Kurventabellenwerte lesen (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) (Seite 532)
CTABTSP	F	Liefert den Wert der Leitachse am Kurventabellen-Anfang		+	+	<i>PGAs/</i> Kurventabellenwerte lesen (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) (Seite 532)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
CTABTSV	F	Liefert den Wert der Folgeachse am Kurventabellen-Anfang		+	+	<i>PGAs!</i> Kurventabellenwerte lesen (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX) (Seite 532)
CTABUNLOCK	P	Aufheben der Lösch- und Überschreibsperre		+	+	<i>PGAs!</i> Kurventabellen gegen Löschen und Überschreiben sperren (CTABLOCK, CTABUNLOCK) (Seite 529)
CTOL	K	Konturtoleranz für Kompressor-Funktionen, Orientierungsglättung und Überschleifarten		+		<i>PGAs!</i> Programmierbare Kontur-/Orientierungstoleranz (CTOL, OTOL, ATOL) (Seite 509)
CTRANS	F	Nullpunktverschiebung für mehrere Achsen		+	-	<i>PGAs!</i> Grob- und Feinverschiebung (CFINE, CTRANS) (Seite 306)
CUT2D ⁶⁾	G	2D-Werkzeugkorrektur	m	+		<i>PGs!</i>
CUT2DF	G	2D-Werkzeugkorrektur Die Werkzeugkorrektur wirkt relativ zum aktuellen Frame (schräge Ebene).	m	+		<i>PGs!</i>
CUT3DC	G	3D-Werkzeugkorrektur Umfangsfräsen	m	+		<i>PGAs!</i> Aktivierung 3D-Werkzeugkorrekturen (CUT3DC..., CUT3DF...) (Seite 424)
CUT3DCC	G	3D-Werkzeugkorrektur Umfangsfräsen mit Begrenzungsflächen	m	+		<i>PGAs!</i> 3D-Werkzeugkorrektur: Berücksichtigung einer Begrenzungsfläche (CUT3DCC, CUT3DCCD) (Seite 433)
CUT3DCCD	G	3D-Werkzeugkorrektur Umfangsfräsen mit Begrenzungsflächen mit Differenzwerkzeug	m	+		<i>PGAs!</i> 3D-Werkzeugkorrektur: Berücksichtigung einer Begrenzungsfläche (CUT3DCC, CUT3DCCD) (Seite 433)
CUT3DF	G	3D-Werkzeugkorrektur Stirnfräsen	m	+		<i>PGAs!</i> Aktivierung 3D-Werkzeugkorrekturen (CUT3DC..., CUT3DF...) (Seite 424)
CUT3DFF	G	3D-Werkzeugkorrektur Stirnfräsen mit konstanter Werkzeugorientierung abhängig vom aktiven Frame	m	+		<i>PGAs!</i> Aktivierung 3D-Werkzeugkorrekturen (CUT3DC..., CUT3DF...) (Seite 424)
CUT3DFS	G	3D-Werkzeugkorrektur Stirnfräsen mit konstanter Werkzeugorientierung unabhängig vom aktiven Frame	m	+		<i>PGAs!</i> Aktivierung 3D-Werkzeugkorrekturen (CUT3DC..., CUT3DF...) (Seite 424)
CUTCONOF ⁶⁾	G	Konstante Radiuskorrektur AUS	m	+		<i>PGs!</i>
CUTCONON	G	Konstante Radiuskorrektur EIN	m	+		<i>PGs!</i>

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
CUTMOD	K	Funktion "Modifikation der Korrekturdaten bei drehbaren Werkzeugen" einschalten		+		<i>PGAs/</i> Schneidendaten-Modifikation bei drehbaren Werkzeugen (CUTMOD) (Seite 458)
CYCLE60	C	Gravurzyklus		+		<i>PGAs/</i> Gravurzyklus - CYCLE60 (Seite 707)
CYCLE61	C	Planfräsen		+		<i>PGAs/</i> Planfräsen - CYCLE61 (Seite 683)
CYCLE62	C	Konturaufruf		+		<i>PGAs/</i> Konturaufruf - CYCLE62 (Seite 709)
CYCLE63	C	Konturtasche fräsen		+		<i>PGAs/</i> Konturtasche fräsen - CYCLE63 (Seite 714)
CYCLE64	C	Konturtasche vorbohren		+		<i>PGAs/</i> Konturtasche vorbohren - CYCLE64 (Seite 713)
CYCLE70	C	Gewindefräsen		+		<i>PGAs/</i> Gewindefräsen - CYCLE70 (Seite 705)
CYCLE72	C	Bahnfräsen		+		<i>PGAs/</i> Bahnfräsen - CYCLE72 (Seite 710)
CYCLE76	C	Rechteckzapfen fräsen		+		<i>PGAs/</i> Rechteckzapfen fräsen - CYCLE76 (Seite 690)
CYCLE77	C	Kreiszapfen fräsen		+		<i>PGAs/</i> Kreiszapfen fräsen - CYCLE77 (Seite 692)
CYCLE78	C	Bohrgewindefräsen		+		<i>PGAs/</i> Bohrgewinde fräsen - CYCLE78 (Seite 676)
CYCLE79	C	Mehrkant		+		<i>PGAs/</i> Mehrkant - CYCLE79 (Seite 694)
CYCLE81	C	Bohren, Zentrieren		+		<i>PGAs/</i> Bohren, Zentrieren - CYCLE81 (Seite 665)
CYCLE82	C	Bohren, Plansenken		+		<i>PGAs/</i> Bohren, Plansenken - CYCLE82 (Seite 666)
CYCLE83	C	Tieflochbohren		+		<i>PGAs/</i> Tieflochbohren - CYCLE83 (Seite 668)
CYCLE84	C	Gewindebohren ohne Ausgleichsfutter		+		<i>PGAs/</i> Gewindebohren ohne Ausgleichsfutter - CYCLE84 (Seite 671)
CYCLE85	C	Reiben		+		<i>PGAs/</i> Reiben - CYCLE85 (Seite 667)

Tabellen

17.1 Anweisungen

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
CYCLE86	C	Ausdrehen		+		<i>PGAs/</i> Ausdrehen - CYCLE86 (Seite 670)
CYCLE92	C	Abstich		+		<i>PGAs/</i> Abstich - CYCLE92 (Seite 731)
CYCLE98	C	Gewindekette		+		<i>PGAs/</i> Gewindekette - CYCLE98 (Seite 728)
CYCLE99	C	Gewindedrehen		+		<i>PGAs/</i> Gewindedrehen - CYCLE99 (Seite 725)
CYCLE800	C	Schwenken		+		<i>PGAs/</i> Schwenken - CYCLE800 (Seite 737)
CYCLE801	C	Gitter oder Rahmen		+		<i>PGAs/</i> Gitter oder Rahmen - CYCLE801 (Seite 681)
CYCLE802	C	Beliebige Positionen		+		<i>PGAs/</i> Beliebige Positionen- CYCLE802 (Seite 678)
CYCLE832	C	High Speed Settings		+		<i>PGAs/</i> High Speed Settings - CYCLE832 (Seite 740)
CYCLE840	C	Gewindebohren mit Ausgleichsfutter		+		<i>PGAs/</i> Gewindebohren mit Ausgleichsfutter - CYCLE840 (Seite 674)
CYCLE899	C	Offene Nut fräsen		+		<i>PGAs/</i> Offene Nut fräsen - CYCLE899 (Seite 701)
CYCLE930	C	Einstich		+		<i>PGAs/</i> Einstich - CYCLE930 (Seite 719)
CYCLE940	C	Freistich Formen		+		<i>PGAs/</i> Freistich Formen - CYCLE940 (Seite 722)
CYCLE951	C	Abspannen		+		<i>PGAs/</i> Abspannen - CYCLE951 (Seite 717)
CYCLE952	C	Konturstechen		+		<i>PGAs/</i> Konturstechen - CYCLE952 (Seite 733)
D	A	Werkzeugkorrekturnummer		+		<i>PGs/</i>
D0	A	Bei D0 sind die Korrekturen für das Werkzeug unwirksam		+		<i>PGs/</i>
DAC	K	Absolut satzweise achsspezifische Durchmesser- programmierung	s	+		<i>PGs/</i>
DC	K	Absolute Maßangabe für Rundachsen, Position direkt anfahren	s	+		<i>PGs/</i>

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
DEF	K	Variablen definition		+		<i>PGAsl</i> Definition von Anwendervariablen (DEF) (Seite 25)
DEFAULT	K	Zweig in der CASE-Verzweigung		+		<i>PGAsl</i> Programmverzweigung (CASE ... OF ... DEFAULT ...) (Seite 106)
DEFINE	K	Schlüsselwort für Makrodefinitionen		+		<i>PGAsl</i> Makrotechnik (DEFINE ... AS) (Seite 221)
DELAYFSTOF	P	Ende eines Stopp-Delay-Bereichs definieren	m	+	-	<i>PGAsl</i> Bedingt unterbrechbare Programmabschnitte (DELAYFSTON, DELAYFSTOF) (Seite 488)
DELAYFSTON	P	Beginn eines Stopp-Delay-Bereichs definieren	m	+	-	<i>PGAsl</i> Programmverzweigung (CASE ... OF ... DEFAULT ...) (Seite 106)
DELDL	F	Additive Korrekturen löschen		+	-	<i>PGAsl</i> Additive Korrekturen löschen (DELDL) (Seite 410)
DELDTG	P	Restweglöschen		-	+	<i>FBSY</i>
DELETE	P	Die angegebene Datei löschen. Der Dateiname kann mit Pfad und Datei-Kennung angegeben werden.		+	-	<i>PGAsl</i> Datei löschen (DELETE) (Seite 156)
DELMLOWNER	F	Eigentümergelegenplatz des Werkzeugs löschen		+	-	<i>FBWsl</i>
DEMLRES	F	Magazinplatz-Reservierung löschen		+	-	<i>FBWsl</i>
DELMT	P	Multitool löschen		+	-	<i>FBWsl</i>
DELT	P	Werkzeug löschen		+	-	<i>FBWsl</i>
DELTC	P	Werkzeugträgerdatensatz löschen		+	-	<i>FBWsl</i>
DELTOOLENV	F	Datensätze zur Beschreibung von Werkzeugumgebungen löschen		+	-	<i>FB1sl (W1)</i>
DIACYCOFA	K	Achsspezifische modale Durchmesserprogrammierung: AUS in Zyklen	m	+		<i>FB1sl (P1)</i>
DIAM90	G	Durchmesserprogrammierung für G90, Radiusprogrammierung für G91	m	+		<i>PGAsl</i>
DIAM90A	K	Achsspezifische modale Durchmesserprogrammierung für G90 und AC, Radiusprogrammierung für G91 und IC	m	+		<i>PGsl</i>

17.1 Anweisungen

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
DIAMCHAN	K	Übernahme aller Achsen aus MD Achsfunktionen in den Kanalzustand der Durchmesserprogrammierung		+		PGs/
DIAMCHANA	K	Übernahme Kanalzustand der Durchmesserprogrammierung		+		PGs/
DIAMCYCOF	G	Kanalspezifische Durchmesserprogrammierung: AUS in Zyklen	m	+		FB1s/ (P1)
DIAMOF ⁶⁾	G	Durchmesserprogrammierung: AUS Grundstellung siehe Maschinenhersteller	m	+		PGs/
DIAMOFA	K	Achsspezifische modale Durchmesserprogrammierung: AUS Grundstellung siehe Maschinenhersteller	m	+		PGs/
DIAMON	G	Durchmesserprogrammierung: EIN	m	+		PGs/
DIAMONA	K	Achsspezifische modale Durchmesserprogrammierung: EIN Freischaltung siehe Maschinenhersteller	m	+		PGs/
DIC	K	Relativ satzweise achsspezifische Durchmesserprogrammierung	s	+		PGs/
DILF	A	Rückzugsweg (Länge)	m	+		PGs/
DISABLE	P	Interrupt AUS		+	-	PGAs/ Zuordnung einer Interruptroutine deaktivieren/reaktivieren (DISABLE, ENABLE) (Seite 134)
DISC	A	Überhöhung Übergangskreis Werkzeug-Radiuskorrektur	m	+		PGs/
DISCL	A	Abstand des Endpunkts der schnellen Zustellbewegung, von der Bearbeitungsebene		+		PGs/
DISPLOF	PA	Aktuelle Satzanzeige unterdrücken		+		PGAs/ Aktuelle Satzanzeige unterdrücken (DISPLOF, DISPLON, ACTBLOCNO) (Seite 187)
DISPLON	PA	Unterdrückung der aktuellen Satzanzeige aufheben		+		PGAs/ Aktuelle Satzanzeige unterdrücken (DISPLOF, DISPLON, ACTBLOCNO) (Seite 187)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
DISPR	A	Repos-Bahndifferenz	s	+		<i>PGAs/</i> Wiederanfahren an Kontur (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) (Seite 495)
DISR	A	Repos-Abstand	s	+		<i>PGAs/</i> Wiederanfahren an Kontur (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) (Seite 495)
DISRP	A	Abstand der Rückzugsebene von der Bearbeitungsebene beim weichen An- und Abfahren		+		<i>PGs/</i>
DITE	A	Gewindeauslaufweg	m	+		<i>PGs/</i>
DITS	A	Gewindeeinlaufweg	m	+		<i>PGs/</i>
DIV	K	Integer-Division		+		<i>PGAs/</i> Rechenfunktionen (Seite 73)
DL	A	Ortsabhängige additive Werkzeugkorrektur an-wählen (DL, Summen-Einrichtekorrektur)	m	+		<i>PGAs/</i> Additive Korrekturen anwählen (DL) (Seite 407)
DO	A	Schlüsselwort für Synchronaktion, löst bei erfüllter Bedingung Aktion aus		-	+	<i>FBSY</i>
DRFOF	P	Ausschalten der Handradverschiebungen (DRF)	m	+	-	<i>PGs/</i>
DRIVE	G	Geschwindigkeitsabhängige Bahnbeschleunigung	m	+		<i>PGAs/</i> Beschleunigungsmodus (BRISK, BRISKA, SOFT, SOFTA, DRIVE, DRIVEA) (Seite 475)
DRIVEA	P	Geknickte Beschleunigungskennlinie für die programmierten Achsen einschalten		+	-	<i>PGAs/</i> Beschleunigungsmodus (BRISK, BRISKA, SOFT, SOFTA, DRIVE, DRIVEA) (Seite 475)
DYNFINISH	G	Dynamik für Feinschichten	m	+		<i>PGAs/</i> Aktivierung von Technologie-spezifischen Dynamikwerten (DYNNORM, DYNPOS, DYNROUGH, DYNSEMIFIN, DYNFINISH) (Seite 480)
DYNNORM ⁶⁾	G	Normale Dynamik	m	+		<i>PGAs/</i> Aktivierung von Technologie-spezifischen Dynamikwerten (DYNNORM, DYNPOS, DYNROUGH, DYNSEMIFIN, DYNFINISH) (Seite 480)

17.1 Anweisungen

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
DYNPOS	G	Dynamik für Positionierbetrieb, Gewindebohren	m	+		<i>PGAs/</i> Aktivierung von Technologie-spezifischen Dynamikwerten (DYNNORM, DYNPOS, DYNROUGH, DYNSEMIFIN, DYNFINISH) (Seite 480)
DYNROUGH	G	Dynamik für Schruppen	m	+		<i>PGAs/</i> Aktivierung von Technologie-spezifischen Dynamikwerten (DYNNORM, DYNPOS, DYNROUGH, DYNSEMIFIN, DYNFINISH) (Seite 480)
DYNSEMIFIN	G	Dynamik für Schlichten	m	+		<i>PGAs/</i> Aktivierung von Technologie-spezifischen Dynamikwerten (DYNNORM, DYNPOS, DYNROUGH, DYNSEMIFIN, DYNFINISH) (Seite 480)
DZERO	P	Kennzeichnet alle D-Nummern der TO-Einheit als ungültig		+	-	<i>PGAs/</i> Freie D-Nummernvergabe: D-Nummern ungültig setzen (DZERO) (Seite 446)
EAUTO	G	Festlegung des letzten Spline-Abschnitts durch die letzten 3 Punkte	m	+		<i>PGAs/</i> Spline-Interpolation (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) (Seite 250)
EGDEF	P	Definition eines elektronischen Getriebes		+	-	<i>PGAs/</i> Elektronisches Getriebe definieren (EGDEF) (Seite 545)
EGDEL	P	Kopplungsdefinition für die Folgeachse löschen		+	-	<i>PGAs/</i> Definition eines Elektronischen Getriebes löschen (EGDEL) (Seite 552)
EGOFC	P	Elektronisches Getriebe kontinuierlich ausschalten		+	-	<i>PGAs/</i> Elektronisches Getriebe ausschalten (EGOFS, EGOFC) (Seite 551)
EGOFS	P	Elektronisches Getriebe selektiv ausschalten		+	-	<i>PGAs/</i> Elektronisches Getriebe ausschalten (EGOFS, EGOFC) (Seite 551)
EGON	P	Elektronisches Getriebe einschalten		+	-	<i>PGAs/</i> Elektronisches Getriebe ausschalten (EGOFS, EGOFC) (Seite 551)
EGONSYN	P	Elektronisches Getriebe einschalten		+	-	<i>PGAs/</i> Elektronisches Getriebe ausschalten (EGOFS, EGOFC) (Seite 551)
EGONSYNE	P	Elektronisches Getriebe einschalten, mit Vorgabe von Anfahrmodus		+	-	<i>PGAs/</i> Elektronisches Getriebe ausschalten (EGOFS, EGOFC) (Seite 551)
ELSE	K	Programmverzweigung, wenn IF-Bedingung nicht erfüllt		+		<i>PGAs/</i> Programmschleife mit Alternative (IF, ELSE, ENDIF) (Seite 116)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
ENABLE	P	Interrupt EIN		+	-	<i>PGAs/</i> Zuordnung einer Interruptroutine deaktivieren/reactivieren (DISABLE, ENABLE) (Seite 134)
ENAT ⁶⁾	G	Natürlicher Kurvenübergang zum nächsten Verfahrssatz	m	+		<i>PGAs/</i> Spline-Interpolation (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) (Seite 250)
ENDFOR	K	Endezeile der FOR-Zählschleife		+		<i>PGAs/</i> Zählschleife (FOR ... TO ..., ENDFOR) (Seite 119)
ENDIF	K	Endezeile der IF-Verzweigung		+		<i>PGAs/</i> Programmschleife mit Alternative (IF, ELSE, ENDIF) (Seite 116)
ENDLABEL	K	Endmarke für Teilprogramm- wiederholungen über REPEAT		+		<i>PGAs/</i> , <i>FB1s/ (K1)</i> Programmteilwiederholung (REPEAT, REPEATB, ENDLABEL, P) (Seite 108)
ENDLOOP	K	Endezeile der Endlos- Programmschleife LOOP		+		<i>PGAs/</i> Endlos-Programmschleife (LOOP, ENDLOOP) (Seite 118)
ENDPROC	K	Endezeile eines Programms mit der Anfangszeile PROC		+		
ENDWHILE	K	Endezeile der WHILE-Schleife		+		<i>PGAs/</i> Programmschleife mit Bedingung am Schleifenanfang (WHILE, ENDWHILE) (Seite 121)
ESRR	P	Antriebsautarkes ESR- Rückziehen im Antrieb parametrieren		+		<i>PGAs/</i> Antriebsautarkes Rückziehen projektieren (ESRR) (Seite 643)
ESRS	P	Antriebsautarkes ESR- Stillsetzen im Antrieb parametrieren		+		<i>PGAs/</i> Antriebsautarkes Stillsetzen projektieren (ESRS) (Seite 642)
ETAN	G	Tangentialer Kurvenübergang zum nächsten Verfahrssatz bei Spline-Beginn	m	+		<i>PGAs/</i> Spline-Interpolation (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) (Seite 250)
EVERY	K	Synchronaktion ausführen bei Übergang der Bedingung von FALSE zu TRUE		-	+	<i>FBSY</i>
EX	K	Schlüsselwert für die Wertzuzuweisung in exponentieller Schreibweise		+		<i>PGAs/</i> Vordefinierte Anwendervariablen: Rechenparameter (R) (Seite 21)
EXECSTRING	P	Übergabe einer String- Variablen mit der auszuführenden Teileprogrammzeile		+	-	<i>PGAs/</i> Indirekte Programmierung von Teileprogrammzeilen (EXECSTRING) (Seite 72)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
EXECTAB	P	Ein Element aus einer Bewegungstabelle abarbeiten		+	-	<i>PGAs/</i> Indirekte Programmierung von Teileprogrammzeilen (EXECSTRING) (Seite 72)
EXECUTE	P	Programmausführung EIN		+	-	<i>PGAs/</i> Konturaufbereitung ausschalten (EXECUTE) (Seite 661)
EXP	F	Exponentialfunktion ex		+	+	<i>PGAs/</i> Rechenfunktionen (Seite 73)
EXTCALL	A	Externes Unterprogramm abarbeiten		+	+	<i>PGAs/</i> Externes Unterprogramm abarbeiten (840D sl) (EXTCALL) (Seite 213)
EXTCLOSE	P	Zum Schreiben geöffnetes externes Gerät/Datei schließen		+	-	<i>PGAs/</i> Process DataShare - Ausgabe auf ein externes Gerät/Datei (EXTOPEN, WRITE, EXTCLOSE) (Seite 623)
EXTERN	K	Bekanntmachung eines Unterprogramms mit Parameterübergabe		+		<i>PGAs/</i> Unterprogrammaufruf ohne Parameterübergabe (Seite 199)
EXTOPEN	P	Externes Gerät/Datei für den Kanal zum Schreiben öffnen		+	-	<i>PGAs/</i> Process DataShare - Ausgabe auf ein externes Gerät/Datei (EXTOPEN, WRITE, EXTCLOSE) (Seite 623)
F	A	Vorschubwert (in Verbindung mit G4 wird mit F auch die Verweilzeit programmiert)		+	+	<i>PGs/</i>
FA	K	Axialer Vorschub	m	+	+	<i>PGs/</i>
FAD	A	Zustell-Vorschub für Weiches An- und Abfahren		+		<i>PGs/</i>
FALSE	K	Logische Konstante: falsch		+		<i>PGAs/</i> Definition von Anwendervariablen (DEF) (Seite 25)
FB	A	Satzweiser Vorschub		+		<i>PGs/</i>
FCTDEF	P	Polynomfunktion definieren		+	-	<i>PGAs/</i> Online-Werkzeugkorrektur (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) (Seite 419)
FCUB	G	Vorschub nach kubischem Spline veränderlich	m	+		<i>PGAs/</i> Vorschubverlauf (FNORM, FLIN, FCUB, FPO) (Seite 470)
FD	A	Bahnvorschub für Handradüberlagerung	s	+		<i>PGs/</i>
FDA	K	Axialer Vorschub für Handradüberlagerung	s	+		<i>PGs/</i>

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
FENDNORM ⁶⁾	G	Eckenverzögerung AUS	m	+		<i>PGAs/</i> Vorschubreduzierung mit Eckenverzögerung (FENDNORM, G62, G621) (Seite 288)
FFWOF ⁶⁾	G	Vorsteuerung AUS	m	+		<i>PGAs/</i> Fahren mit Vorsteuerung (FFWON, FFWOF) (Seite 482)
FFWON	G	Vorsteuerung Ein	m	+		<i>PGAs/</i> Fahren mit Vorsteuerung (FFWON, FFWOF) (Seite 482)
FGREF	K	Bezugsradius bei Rundachsen oder Bahnbezugsfaktoren bei Orientierungsachsen (Vektorinterpolation)	m	+		<i>PGs/</i>
FGROUP	P	Festlegung der Achse(n) mit Bahnvorschub		+	-	<i>PGs/</i>
FI	K	Parameter für Zugriff auf Framedaten: Feinverschiebung		+		<i>PGAs/</i> Framekomponenten lesen und verändern (TR, FI, RT, SC, MI) (Seite 302)
FIFOCTRL	G	Steuerung des Vorlaufpuffers	m	+		<i>PGAs/</i> Programmablauf mit Vorlaufspeicher (STOPFIFO, STARTFIFO, FIFOCTRL, STOPRE) (Seite 485)
FILEDATE	P	Liefert Datum des zuletzt schreibenden Zugriffs auf die Datei		+	-	<i>PGAs/</i> Datei-Informationen auslesen (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO) (Seite 163)
FILEINFO	P	Liefert Summe von FILEDATE, FILESIZE, FILESTAT und FILETIME zusammen		+	-	<i>PGAs/</i> Datei-Informationen auslesen (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO) (Seite 163)
FILESIZE	P	Liefert aktuelle Größe der Datei		+	-	<i>PGAs/</i> Datei-Informationen auslesen (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO) (Seite 163)
FILESTAT	P	Liefert Filestatus der Rechte Lesen, Schreiben, Execute, Anzeigen, Löschen (rwxsd)		+	-	<i>PGAs/</i> Datei-Informationen auslesen (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO) (Seite 163)
FILETIME	P	Liefert Uhrzeit des zuletzt schreibenden Zugriffs auf die Datei		+	-	<i>PGAs/</i> Datei-Informationen auslesen (FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO) (Seite 163)
FINEA	K	Bewegungsende beim Erreichen von "Genauhalt Fein"	m	+		<i>PGAs/</i> Programmierbares Bewegungsendekriterium (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA) (Seite 289)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
FL	K	Grenzgeschwindigkeit für Synchronachsen	m	+		<i>PGs/</i>
FLIN	G	Vorschub linear veränderlich	m	+		<i>PGAs/</i> Vorschubverlauf (FNORM, FLIN, FCUB, FPO) (Seite 470)
FMA	K	Mehrere Vorschübe axial	m	+		<i>PGs/</i>
FNORM ⁶⁾	G	Vorschub normal nach DIN66025	m	+		<i>PGAs/</i> Vorschubverlauf (FNORM, FLIN, FCUB, FPO) (Seite 470)
FOC	K	Satzweise wirksame Moment/Kraft-Begrenzung	s	-	+	<i>FBSY</i>
FOCOF	K	Modale Moment/Kraft-Begrenzung ausschalten	m	-	+	<i>FBSY</i>
FOCON	K	Modale Moment/Kraft-Begrenzung einschalten	m	-	+	<i>FBSY</i>
FOR	K	Zählschleife mit fester Anzahl von Durchläufen		+		<i>PGAs/</i> Zählschleife (FOR ... TO ..., ENDFOR) (Seite 119)
FP	A	Festpunkt: Nummer des anzufahrenden Festpunkts	s	+		<i>PGs/</i>
FPO	K	Über ein Polynom programmierter Vorschubverlauf		+		<i>PGAs/</i> Vorschubverlauf (FNORM, FLIN, FCUB, FPO) (Seite 470)
FPR	P	Kennzeichnung Rundachse		+	-	<i>PGs/</i>
FPRAOF	P	Umdrehungsvorschub ausschalten		+	-	<i>PGs/</i>
FPRAON	P	Umdrehungsvorschub einschalten		+	-	<i>PGs/</i>
FRAME	K	Datentyp zur Festlegung von Koordinatensystemen		+		<i>PGAs/</i> Definition neuer Frames (DEF FRAME) (Seite 305)
FRC	A	Vorschub für Radius und Fase	s	+		<i>PGs/</i>
FRCM	A	Vorschub für Radius und Fase modal	m	+		<i>PGs/</i>
FROM	K	Die Aktion wird ausgeführt, wenn die Bedingung einmal erfüllt ist und solange die Synchronaktion aktiv ist		-	+	<i>FBSY</i>
FTOC	P	Werkzeugfeinkorrektur ändern		-	+	<i>FBSY</i>
FTOCOF ⁶⁾	G	Online wirksame Werkzeugfeinkorrektur AUS	m	+		<i>PGAs/</i> Online-Werkzeugkorrektur (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) (Seite 419)
FTOCON	G	Online wirksame Werkzeugfeinkorrektur EIN	m	+		<i>PGAs/</i> Online-Werkzeugkorrektur (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) (Seite 419)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
FXS	K	Fahren auf Festanschlag ein	m	+	+	PGsl
FXST	K	Momentgrenze für Fahren auf Festanschlag	m	+	+	PGsl
FXSW	K	Überwachungsfenster für Fahren auf Festanschlag		+	+	PGsl
FZ	K	Zahnvorschub	m	+		PGsl
G0	G	Linearinterpolation mit Eilgang (Eilgangsbewegung)	m	+		PGsl
G1 ⁶⁾	G	Linearinterpolation mit Vorschub (Geradeninterpolation)	m	+		PGsl
G2	G	Kreisinterpolation im Uhrzeigersinn	m	+		PGsl
G3	G	Kreisinterpolation gegen Uhrzeigersinn	m	+		PGsl
G4	G	Verweilzeit, zeitlich vorbestimmt	s	+		PGsl
G5	G	Schrägeinstechschleifen	s	+		PGAsl Schräge Achse (TRAANG) (Seite 385)
G7	G	Ausgleichsbewegung beim Schrägeinstechschleifen	s	+		PGAsl Schräge Achse (TRAANG) (Seite 385)
G9	G	Genauhalt - Geschwindigkeitsabnahme	s	+		PGsl
G17 ⁶⁾	G	Wahl der Arbeitsebene X/Y	m	+		PGsl
G18	G	Wahl der Arbeitsebene Z/X	m	+		PGsl
G19	G	Wahl der Arbeitsebene Y/Z	m	+		PGsl
G25	G	Untere Arbeitsfeldbegrenzung	s	+		PGsl
G26	G	Obere Arbeitsfeldbegrenzung	s	+		PGsl
G33	G	Gewindeschneiden mit konstanter Steigung	m	+		PGsl
G34	G	Gewindeschneiden mit linear zunehmender Steigung	m	+		PGsl
G35	G	Gewindeschneiden mit linear abnehmender Steigung	m	+		PGsl
G40 ⁶⁾	G	Werkzeugradiuskorrektur AUS	m	+		PGsl
G41	G	Werkzeugradiuskorrektur links von der Kontur	m	+		PGsl
G42	G	Werkzeugradiuskorrektur rechts von der Kontur	m	+		PGsl
G53	G	Unterdrückung der aktuellen Nullpunktverschiebung (satzweise)	s	+		PGsl
G54	G	1. Einstellbare Nullpunktverschiebung	m	+		PGsl
G55	G	2. Einstellbare Nullpunktverschiebung	m	+		PGsl

17.1 Anweisungen

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
G56	G	3. Einstellbare Nullpunktverschiebung	m	+		PGsl
G57	G	4. Einstellbare Nullpunktverschiebung	m	+		PGsl
G58 (840D sl)	G	Axiale programmierbare Nullpunktverschiebung absolut, Grobverschiebung	s	+		PGsl
G58 (828D)	G	5. Einstellbare Nullpunktverschiebung	m	+		PGsl
G59 (840D sl)	G	Axiale programmierbare Nullpunktverschiebung additiv, Feinverschiebung	s	+		PGsl
G59 (828D)	G	6. Einstellbare Nullpunktverschiebung	m	+		PGsl
G60 ⁶⁾	G	Genauhalt - Geschwindigkeitsabnahme	m	+		PGsl
G62	G	Eckenverzögerung an Innenecken bei aktiver Werkzeugradiuskorrektur (G41, G42)	m	+		PGAs/ Vorschubreduzierung mit Eckenverzögerung (FENDNORM, G62, G621) (Seite 288)
G63	G	Gewindebohren mit Ausgleichsfutter	s	+		PGsl
G64	G	Bahnsteuerbetrieb	m	+		PGsl
G70	G	Inch-Maßangabe für geometrische Angaben (Längen)	m	+	+	PGsl
G71 ⁶⁾	G	Metrische Maßangabe für geometrische Angaben (Längen)	m	+	+	PGsl
G74	G	Referenzpunktanfahren	s	+		PGsl
G75	G	Festpunktanfahren	s	+		PGsl
G90 ⁶⁾	G	Maßangabe absolut	m/s	+		PGsl
G91	G	Kettenmaßangabe	m/s	+		PGsl
G93	G	Zeitreziproker Vorschub 1/min	m	+		PGsl
G94 ⁶⁾	G	Linearvorschub F in mm/min oder inch/min und Grad/min	m	+		PGsl
G95	G	Umdrehungsvorschub F in mm/U oder inch/U	m	+		PGsl
G96	G	konstante Schnittgeschwindigkeit (wie bei G95) EIN	m	+		PGsl
G97	G	konstante Schnittgeschwindigkeit (wie bei G95) AUS	m	+		PGsl
G110	G	Polprogrammierung relativ zur letzten programmierten Sollposition	s	+		PGsl

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
G111	G	Polprogrammierung relativ zum Nullpunkt des aktuellen Werkstück-Koordinatensystems	s	+		<i>PGsl</i>
G112	G	Polprogrammierung relativ zum letzten gültigen Pol	s	+		<i>PGsl</i>
G140 ⁶⁾	G	Anfahrriichtung WAB festgelegt durch G41/G42	m	+		<i>PGsl</i>
G141	G	Anfahrriichtung WAB links der Kontur	m	+		<i>PGsl</i>
G142	G	Anfahrriichtung WAB rechts der Kontur	m	+		<i>PGsl</i>
G143	G	Anfahrriichtung WAB tangentialabhängig	m	+		<i>PGsl</i>
G147	G	Weiches Anfahren mit Gerade	s	+		<i>PGsl</i>
G148	G	Weiches Abfahren mit Gerade	s	+		<i>PGsl</i>
G153	G	Unterdrückung aktueller Frames inklusive Basisframe	s	+		<i>PGsl</i>
G247	G	Weiches Anfahren mit Viertelkreis	s	+		<i>PGsl</i>
G248	G	Weiches Abfahren mit Viertelkreis	s	+		<i>PGsl</i>
G290 ⁶⁾	G	Umschalten auf SINUMERIK-Mode EIN	m	+		<i>FBWsl</i>
G291	G	Umschalten auf ISO2/3-Mode EIN	m	+		<i>FBWsl</i>
G331	G	Gewindebohren ohne Ausgleichsfutter, positive Steigung, Rechtslauf	m	+		<i>PGsl</i>
G332	G	Gewindebohren ohne Ausgleichsfutter, negative Steigung, Linkslauf	m	+		<i>PGsl</i>
G340 ⁶⁾	G	Anfahrersatz räumlich (Tiefe und in der Ebene zugleich (Helix))	m	+		<i>PGsl</i>
G341	G	Zuerst in der senkrechten Achse zustellen (z), dann Anfahren in der Ebene	m	+		<i>PGsl</i>
G347	G	Weiches Anfahren mit Halbkreis	s	+		<i>PGsl</i>
G348	G	Weiches Abfahren mit Halbkreis	s	+		<i>PGsl</i>
G450 ⁶⁾	G	Übergangskreis	m	+		<i>PGsl</i>
G451	G	Schnittpunkt der Äquidistanten	m	+		<i>PGsl</i>
G460 ⁶⁾	G	Einschalten der Kollisionsüberwachung für An- und Abfahrersatz	m	+		<i>PGsl</i>
G461	G	Einfügen eines Kreises im WRK-Satz	m	+		<i>PGsl</i>

17.1 Anweisungen

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
G462	G	Einfügen einer Geraden im WRK-Satz	m	+		PGsl
G500 ⁶⁾	G	Ausschalten aller einstellbaren Frames, Basisframes sind aktiv	m	+		PGsl
G505 ... G599	G	5 ... 99. Einstellbare Nullpunktverschiebung	m	+		PGsl
G601 ⁶⁾	G	Satzwechsel bei Genauhalt fein	m	+		PGsl
G602	G	Satzwechsel bei Genauhalt grob	m	+		PGsl
G603	G	Satzwechsel bei IPO-Satzende	m	+		PGsl
G621	G	Eckenverzögerung an allen Ecken	m	+		PGAs/ Vorschubreduzierung mit Eckenverzögerung (FENDNORM, G62, G621) (Seite 288)
G641	G	Bahnsteuerbetrieb mit Überschleifen nach Wegkriterium (= programmierbarer Überschleifabstand)	m	+		PGsl
G642	G	Bahnsteuerbetrieb mit Überschleifen unter Einhaltung definierter Toleranzen	m	+		PGsl
G643	G	Bahnsteuerbetrieb mit Überschleifen unter Einhaltung definierter Toleranzen (satzintern)	m	+		PGsl
G644	G	Bahnsteuerbetrieb mit Überschleifen mit maximal möglicher Dynamik	m	+		PGsl
G645	G	Bahnsteuerbetrieb mit Überschleifen von Ecken und tangentialer Satzübergänge unter Einhaltung definierter Toleranzen	m	+		PGsl
G700	G	Inch-Maßangabe für geometrische und technologische Angaben (Längen, Vorschub)	m	+	+	PGsl
G710 ⁶⁾	G	Metrische Maßangabe für geometrische und technologische Angaben (Längen, Vorschub)	m	+	+	PGsl
G751	G	Festpunkt über Zwischenpunkt anfahren	s	+		PGsl
G810 ⁶⁾ , ..., G819	G	Für den OEM-Anwender reservierte G-Gruppe		+		PGAs/ Spezielle Funktionen für den OEM-Anwender (OMA1 ... OMA5, OEMIPO1, OEMIPO2, G810 ... G829) (Seite 287)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
G820 ⁶⁾ , ..., G829	G	Für den OEM-Anwender reservierte G-Gruppe		+		<i>PGAs!</i> Spezielle Funktionen für den OEM-Anwender (OMA1 ... OMA5, OEMIPO1, OEMIPO2, G810 ... G829) (Seite 287)
G931	G	Vorschubvorgabe durch Verfahrszeit	m	+		
G942	G	Linear-Vorschub und konstante Schnittgeschwindigkeit oder Spindeldrehzahl einfrieren	m	+		
G952	G	Umdrehungsvorschub und konstante Schnittgeschwindigkeit oder Spindeldrehzahl einfrieren	m	+		
G961	G	konstante Schnittgeschwindigkeit und Linear-Vorschub	m	+		<i>PGs!</i>
G962	G	Linear-Vorschub oder Umdrehungsvorschub und konstante Schnittgeschwindigkeit	m	+		<i>PGs!</i>
G971	G	Spindeldrehzahl einfrieren und Linear-Vorschub	m	+		<i>PGs!</i>
G972	G	Linear-Vorschub oder Umdrehungsvorschub und konstante Spindeldrehzahl einfrieren	m	+		<i>PGs!</i>
G973	G	Umdrehungsvorschub ohne Spindeldrehzahlbegrenzung	m	+		<i>PGs!</i>
GEOAX	P	Den Geometrieachsen 1 - 3 neue Kanalachsen zuordnen		+	-	<i>PGAs!</i> Umschaltbare Geometrieachsen (GEOAX) (Seite 602)
GET	P	Freigegebene Achse zwischen Kanälen tauschen		+	+	<i>PGAs!</i> Achstausch, Spindeltausch (RELEASE, GET, GETD) (Seite 143)
GETACTT	F	Bestimmt das aktive Werkzeug aus einer Gruppe von gleichnamigen Werkzeugen		+	-	<i>FBWs!</i>
GETACTTD	F	Bestimmt zu einer absoluten D-Nummer die zugehörige T-Nummer		+	-	<i>PGAs!</i> Freie D-Nummernvergabe: T-Nummer zur vorgegebenen D-Nummer ermitteln (GETACTTD) (Seite 446)
GETD	P	Achse direkt zwischen Kanälen tauschen		+	-	<i>PGAs!</i> Achstausch, Spindeltausch (RELEASE, GET, GETD) (Seite 143)
GETDNO	F	Liefert D-Nummer einer Schneide (CE) eines Werkzeugs (T)		+	-	<i>PGAs!</i> Freie D-Nummernvergabe: D-Nummern umbenennen (GETDNO, SETDNO) (Seite 445)

17.1 Anweisungen

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
GETEXET	P	Lesen der eingewechselten T-Nummer		+	-	<i>FBWs/</i>
GETFREELOC	P	Für ein gegebenes Werkzeug einen Leerplatz in den Magazinen suchen		+	-	<i>FBWs/</i>
GETSELT	P	Vorgewählte T-Nummer liefern		+	-	<i>FBWs/</i>
GETT	F	T-Nummer zu Werkzeugnamen bestimmen		+	-	<i>FBWs/</i>
GETTCOR	F	Werkzeuglängen bzw. Werkzeuglängenkomponten auslesen		+	-	<i>FB1s/ (W1)</i>
GETTENV	F	T-, D-, und DL-Nummern lesen		+	-	<i>FB1s/ (W1)</i>
GETVARAP	F	Zugriffsrecht auf eine System-/Anwendervariable lesen		+	-	<i>PGAs/</i> Attributwerte/Datentyp lesen (GETVARPHU, GETVARAP, GETVARLIM, GETVARDFT, GETVARTYP) (Seite 59)
GETVARDFT	F	Standardwert einer System-/Anwendervariablen lesen		+	-	<i>PGAs/</i> Attributwerte/Datentyp lesen (GETVARPHU, GETVARAP, GETVARLIM, GETVARDFT, GETVARTYP) (Seite 59)
GETVARLIM	F	Grenzwerte einer System-/Anwendervariablen lesen		+	-	<i>PGAs/</i> Attributwerte/Datentyp lesen (GETVARPHU, GETVARAP, GETVARLIM, GETVARDFT, GETVARTYP) (Seite 59)
GETVARPHU	F	Physikalische Einheit einer System-/Anwendervariablen lesen		+	-	<i>PGAs/</i> Attributwerte/Datentyp lesen (GETVARPHU, GETVARAP, GETVARLIM, GETVARDFT, GETVARTYP) (Seite 59)
GETVARTYP	F	Datentyp einer System-/Anwendervariablen lesen		+	-	<i>PGAs/</i> Attributwerte/Datentyp lesen (GETVARPHU, GETVARAP, GETVARLIM, GETVARDFT, GETVARTYP) (Seite 59)
GOTO	K	Sprunganweisung erst vorwärts dann rückwärts (Richtung erst zum Programm-Ende und dann zum Programm-Anfang)		+		<i>PGAs/</i> Programmsprünge auf Sprungmarken (GOTOB, GOTOF, GOTO, GOTOC) (Seite 103)
GOTOB	K	Sprunganweisung rückwärts (Richtung Programm-Anfang)		+		<i>PGAs/</i> Programmsprünge auf Sprungmarken (GOTOB, GOTOF, GOTO, GOTOC) (Seite 103)
GOTOC	K	Wie GOTO, aber Alarm 14080 "Sprungziel nicht gefunden" unterdrücken		+		<i>PGAs/</i> Programmsprünge auf Sprungmarken (GOTOB, GOTOF, GOTO, GOTOC) (Seite 103)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
GOTOF	K	Sprunganweisung vorwärts (Richtung Programm-Ende)		+		<i>PGAs/</i> Programmsprünge auf Sprungmarken (GOTOB, GOTOF, GOTO, GOTOC) (Seite 103)
GOTOS	K	Rücksprung auf Programmanfang		+		<i>PGAs/</i> Rücksprung auf Programmanfang (GOTOS) (Seite 102)
GP	K	Schlüsselwort zur indirekten Programmierung von Positionsattributen		+		<i>PGAs/</i> Indirekte Programmierung von Positionsattributen (GP) (Seite 69)
GWPSOF	P	Konstante Scheibenumfangs- geschwindigkeit (SUG) abwählen	s	+	-	<i>PGs/</i>
GWPSON	P	Konstante Scheibenumfangs- geschwindigkeit (SUG) anwählen	s	+	-	<i>PGs/</i>
H...	A	Hilfsfunktionsausgabe an die PLC		+	+	<i>PGs//FB1s/ (H2)</i>
HOLES1	C	Lochreihe		+		<i>PGAs/</i> Lochreihe - HOLES1 (Seite 680)
HOLES2	C	Lochkreis		+		<i>PGAs/</i> Lochkreis - HOLES2 (Seite 682)
I	A	Interpolationsparameter	s	+		<i>PGs/</i>
I1	A	Zwischenpunktcoordinate	s	+		<i>PGs/</i>
IC	K	Kettenmaßeingabe	s	+		<i>PGs/</i>
ICYCOF	P	Alle Sätze eines Technologiezyklus nach ICYCOF in einem IPO-Takt abarbeiten		+	+	<i>FBSY</i>
ICYCON	P	Jeden Satz eines Technologiezyklus nach ICYCON in einem separaten IPO-Takt abarbeiten		+	+	<i>FBSY</i>
ID	K	Kennzeichnung für modale Synchronaktionen	m	-	+	<i>FBSY</i>
IDS	K	Kennzeichnung für modale statische Synchronaktionen		-	+	<i>FBSY</i>
IF	K	Einleitung eines bedingten Sprungs im Teileprogramm / Technologiezyklus		+	+	<i>PGAs/</i> Programmschleife mit Alternative (IF, ELSE, ENDIF) (Seite 116)
INDEX	F	Index eines Zeichens im Eingangsstring bestimmen		+	-	<i>PGAs/</i> Zeichen/String in String suchen (INDEX, RINDEX, MINDEX, MATCH) (Seite 89)
INICF	K	Initialisierung der Variablen bei NewConfig		+		<i>PGAs/</i> Definition von Anwendervariablen (DEF) (Seite 25)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
INIPO	K	Initialisierung der Variablen bei PowerOn		+		<i>PGAs/</i> Definition von Anwendervariablen (DEF) (Seite 25)
INIRE	K	Initialisierung der Variablen bei Reset		+		<i>PGAs/</i> Definition von Anwendervariablen (DEF) (Seite 25)
INIT	P	Anwahl eines bestimmten NC-Programms zur Abarbeitung in einem bestimmten Kanal		+	-	<i>PGAs/</i> Programmkoordinierung (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) (Seite 124)
INITIAL		Erzeugen eines INI-Files über alle Bereiche		+		<i>PGAs/</i> Arbeitsspeicher (CHANDATA, COMPLETE, INITIAL) (Seite 230)
INT	K	Datentyp: Ganzzahliger Wert mit Vorzeichen		+		<i>PGAs/</i> Definition von Anwendervariablen (DEF) (Seite 25)
INTERSEC	F	Schnittpunkt zwischen zwei Konturelementen berechnen		+	-	<i>PGAs/</i> Schnittpunkt zwischen zwei Konturelementen ermitteln (INTERSEC) (Seite 656)
INTTOAX	F	Datentyp einer Achsvariablen von INT zu AXIS wandeln		+	-	<i>PGAs/</i> Explizite Datentypwandlungen (AXTOINT, INTTOAX) (Seite 56)
INVCCW	G	Evolvente fahren, gegen den Uhrzeigersinn	m	+		<i>PGs/</i>
INVCW	G	Evolvente fahren, im Uhrzeigersinn	m	+		<i>PGs/</i>
INVFRAME	F	Aus einem Frame den inversen Frame berechnen		+	-	<i>FB1s/ (K2)</i>
IP	K	Variabler Interpolationsparameter		+		<i>PGAs/</i> Indirekte Programmierung (Seite 65)
IPOBRKA	P	Bewegungskriterium ab Einsatzpunkt der Bremsrampe	m	+	+	
IPOENDA	K	Bewegungsende beim Erreichen von "IPO-Stopp"	m	+		<i>PGAs/</i> Programmierbares Bewegungsendekriterium (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA) (Seite 289)
IPTRLOCK	P	Beginn des suchunfähigen Programmabschnitts auf nächsten Maschinenfunktionssatz einfrieren.	m	+	-	<i>PGAs/</i> Programmstelle für SERUPRO verhindern (IPTRLOCK, IPTRUNLOCK) (Seite 493)
IPTRUNLOCK	P	Ende des suchunfähigen Programmabschnitts auf aktuellen Satz zum Unterbrechungszeitpunkt setzen.	m	+	-	<i>PGAs/</i> Programmstelle für SERUPRO verhindern (IPTRLOCK, IPTRUNLOCK) (Seite 493)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
ISAXIS	F	Prüfen, ob die als Parameter angegebene Geometrieachse 1 ist		+	-	<i>PGAs/</i> Achsfunktionen (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) (Seite 599)
ISD	A	Eintauchtiefe	m	+		<i>PGAs/</i> Aktivierung von 3D-Werkzeugkorrekturen (CUT3DC, CUT3DF, CUT3DFS, CUT3DFF, ISD) (Seite 424)
ISFILE	F	Prüfen, ob eine Datei im NCK-Anwendungs-speicher vorhanden ist		+	-	<i>PGAs/</i> Vorhandensein einer Datei prüfen (ISFILE) (Seite 161)
ISNUMBER	F	Prüfen, ob Eingangsstring in Zahl umgewandelt werden kann		+	-	<i>PGAs/</i> Typenkonvertierung von STRING (NUMBER, ISNUMBER, AXNAME) (Seite 85)
ISOCALL	K	Indirekter Aufruf eines in ISO-Sprache programmierten Programms		+		<i>PGAs/</i> Indirekter Aufruf eines in ISO-Sprache programmierten Programms (ISOCALL) (Seite 209)
ISVAR	F	Prüfen, ob der Übergabeparameter eine in der NC bekannte Variable enthält		+	-	<i>PGAs/</i> Vorhandensein einer Variablen prüfen (ISVAR) (Seite 57)
J	A	Interpolationsparameter	s	+		<i>PGs/</i>
J1	A	Zwischenpunktcoordinate	s	+		<i>PGs/</i>
JERKA	P	Über MD eingestelltes Beschleunigungsverhalten für die programmierten Achsen aktivieren		+	-	
JERKLIM	K	Reduktion oder Überhöhung des maximalen axialen Rucks	m	+		<i>PGAs/</i> Prozentuale Ruckkorrektur (JERKLIM) (Seite 505)
JERKLIMA	K	Reduktion oder Überhöhung des maximalen axialen Rucks	m	+	+	<i>PGAs/</i> Beeinflussung der Beschleunigung bei Folgeachsen (VELOLIMA, ACCLIMA, JERKLIMA) (Seite 478)
K	A	Interpolationsparameter	s	+		<i>PGs/</i>
K1	A	Zwischenpunktcoordinate	s	+		<i>PGs/</i>
KONT	G	Kontur umfahren bei der Werkzeugkorrektur	m	+		<i>PGs/</i>
KONTC	G	Mit krümmungsstetigem Polynom an-/abfahren	m	+		<i>PGs/</i>
KONTT	G	Mit tangentenstetigem Polynom an-/abfahren	m	+		<i>PGs/</i>
L	A	Unterprogramm-Nummer	s	+	+	<i>PGAs/</i> Unterprogrammaufruf ohne Parameterübergabe (Seite 199)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
LEAD	A	Voreilwinkel 1. Werkzeugorientierung 2. Orientierungspolynome	m	+		<i>PGAs/</i> Programmierung der Werkzeugorientierung (A..., B..., C..., LEAD, TILT) (Seite 338)
LEADOF	P	Axiale Leitwertkopplung AUS		+	+	<i>PGAs/</i> Axiale Leitwertkopplung (LEADON, LEADOF) (Seite 539)
LEADON	P	Axiale Leitwertkopplung EIN		+	+	<i>PGAs/</i> Axiale Leitwertkopplung (LEADON, LEADOF) (Seite 539)
LENTOAX	F	Liefert Informationen über die Zuordnung der Werkzeuglängen L1, L2 und L3 des aktiven Werkzeugs zur Abszisse, Ordinate und Applikate		+	-	<i>FB1s/ (W1)</i>
LFOF ⁶⁾	G	Schnellrückzug für Gewindeschneiden AUS	m	+		<i>PGs/</i>
LFON	G	Schnellrückzug für Gewindeschneiden EIN	m	+		<i>PGs/</i>
LFPOS	G	Rückzug der mit POLFMASK oder POLFMLIN bekannt gemachten Achse auf die mit POLF programmierte absolute Achspolition	m	+		<i>PGs/</i>
LFTXT ⁶⁾	G	Ebene der Rückzugsbewegung beim Schnellabheben wird bestimmt aus der Bahntangente und der aktuellen Werkzeugrichtung	m	+		<i>PGs/</i>
LFWP	G	Ebene der Rückzugsbewegung beim Schnellabheben wird bestimmt durch die aktuelle Arbeitsebene (G17/G18/G19)	m	+		<i>PGs/</i>
LIFTFAST	K	Schnellabheben		+		<i>PGs/</i> Schnellabheben von der Kontur (SETINT LIFTFAST, ALF) (Seite 136)
LIMS	K	Drehzahlbegrenzung bei G96/G961 und G97	m	+		<i>PGs/</i>
LLI	K	Unterer Grenzwert von Variablen		+		<i>PGAs/</i> Attribut: Grenzwerte (LLI, ULI) (Seite 37)
LN	F	Natürlicher Logarithmus		+	+	<i>PGAs/</i> Rechenfunktionen (Seite 73)
LOCK	P	Synchronaktion mit ID sperren (Technologiezyklus stoppen)		-	+	<i>FBSY</i>

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
LONGHOLE	C	Langloch		+		<i>PGAsl</i> Langloch - LONGHOLE (Seite 703)
LOOP	K	Einleitung einer Endlosschleife		+		<i>PGAsl</i> Endlos-Programmschleife (LOOP, ENDLOOP) (Seite 118)
M0		Programmierer Halt		+	+	<i>PGsl</i>
M1		Wahlweiser Halt		+	+	<i>PGsl</i>
M2		Programmende Hauptprogramm mit Rücksetzen auf Programmanfang		+	+	<i>PGsl</i>
M3		Spindeldrehrichtung rechts		+	+	<i>PGsl</i>
M4		Spindeldrehrichtung links		+	+	<i>PGsl</i>
M5		Spindel Halt		+	+	<i>PGsl</i>
M6		Werkzeugwechsel		+	+	<i>PGsl</i>
M17		Unterprogrammende		+	+	<i>PGsl</i>
M19		Spindelpositionierung auf die im SD43240 eingetragene Position		+	+	<i>PGsl</i>
M30		Programmende, wie M2		+	+	<i>PGsl</i>
M40		Automatische Getriebschaltung		+	+	<i>PGsl</i>
M41 ... M45		Getriebestufe 1 ... 5		+	+	<i>PGsl</i>
M70		Übergang in Achsbetrieb		+	+	<i>PGsl</i>
MASLDEF	P	Master/Slave-Achsverband definieren		+	+	<i>PGAsl</i> Master/Slave-Kopplung (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS) (Seite 565)
MASLDEL	P	Master/Slave-Achsverband trennen und Definition des Verbandes löschen		+	+	<i>PGAsl</i> Master/Slave-Kopplung (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS) (Seite 565)
MASLOF	P	Ausschalten einer temporären Kopplung		+	+	<i>PGAsl</i> Master/Slave-Kopplung (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS) (Seite 565)
MASLOFS	P	Ausschalten einer temporären Kopplung mit automatischem Stillsetzen der Slave-Achse		+	+	<i>PGAsl</i> Master/Slave-Kopplung (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS) (Seite 565)
MASLON	P	Einschalten einer temporären Kopplung		+	+	<i>PGAsl</i> Master/Slave-Kopplung (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS) (Seite 565)
MATCH	F	Suchen eines String im String		+	-	<i>PGAsl</i> Zeichen/String in String suchen (INDEX, RINDEX, MINDEX, MATCH) (Seite 89)

17.1 Anweisungen

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
MAXVAL	F	Größerer Wert zweier Variablen (arithm. Funktion)		+	+	<i>PGAs/</i> Minimum, Maximum und Bereich von Variablen (MINVAL, MAXVAL, BOUND) (Seite 79)
MCALL	K	Modaler Unterprogrammaufruf		+		<i>PGAs/</i> Modaler Unterprogrammaufruf (MCALL) (Seite 204)
MEAC	K	Axiales kontinuierliches Messen ohne Restweglösschen	s	+	+	<i>PGAs/</i> Axiales Messen (MEASA, MEAWA, MEAC) (Option) (Seite 277)
MEAFRAME	F	Frame-Berechnung aus Messpunkten		+	-	<i>PGAs/</i> Frame-Berechnung aus 3 Messpunkten im Raum (MEAFRAME) (Seite 311)
MEAS	A	Messen mit Restweglösschen	s	+		<i>PGAs/</i> Messen mit schaltendem Taster (MEAS, MEAW) (Seite 274)
MEASA	K	Axiales Messen mit Restweglösschen	s	+	+	<i>PGAs/</i> Axiales Messen (MEASA, MEAWA, MEAC) (Option) (Seite 277)
MEASURE	F	Berechnungsmethode für die Werkstück- und Werkzeugvermessung		+	-	<i>FB1s/ (M5)</i> Messen mit schaltendem Taster (MEAS, MEAW) (Seite 274)
MEAW	A	Messen ohne Restweglösschen	s	+		<i>PGAs/</i> Messen mit schaltendem Taster (MEAS, MEAW) (Seite 274)
MEAWA	K	Axiales Messen ohne Restweglösschen	s	+	+	<i>PGAs/</i> Axiales Messen (MEASA, MEAWA, MEAC) (Option) (Seite 277)
MI	K	Zugriff auf Frame-Daten: Spiegelung		+		<i>PGAs/</i> Framekomponenten lesen und verändern (TR, FI, RT, SC, MI) (Seite 302)
MINDEX	F	Index eines Zeichens im Eingangsstring bestimmen		+	-	<i>PGAs/</i> Zeichen/String in String suchen (INDEX, RINDEX, MINDEX, MATCH) (Seite 89)
MINVAL	F	Kleinerer Wert zweier Variablen (arithm. Funktion)		+	+	<i>PGAs/</i> Minimum, Maximum und Bereich von Variablen (MINVAL, MAXVAL, BOUND) (Seite 79)
MIRROR	G	Programmierbare Spiegelung	s	+		<i>PGAs/</i>
MMC	P	Aus dem Teileprogramm interaktiv Dialogfenster am HMI aufrufen		+	-	<i>PGAs/</i> Fenster aus dem Teileprogramm interaktiv aufrufen (MMC) (Seite 616)
MOD	K	Modulo-Division		+		<i>PGAs/</i> Rechenfunktionen (Seite 73)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
MODAXVAL	F	Modulo-Position einer Modulo-Rundachse ermitteln		+	-	<i>PGAs/</i> Achsfunktionen (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) (Seite 599)
MOV	K	Positionierachse starten		-	+	<i>FBSY</i>
MOVT	A	Endpunkt einer Verfahrbewegung in Werkzeugrichtung angeben				<i>FB1(K2)</i>
MSG	P	Programmierbare Meldungen	m	+	-	<i>PGs/</i>
MVTOOL	P	Sprachbefehl zum Bewegen eines Werkzeugs		+	-	<i>FBWs/</i>
N	A	NC-Nebensatznummer		+		<i>PGs/</i>
NCK	K	Spezifizierung des Gültigkeitsbereichs von Daten		+		<i>PGAs/</i> Definition von Anwendervariablen (DEF) (Seite 25)
NEWCONF	P	Geänderte Maschinendaten übernehmen (entspricht "Maschinendatum wirksam setzen")		+	-	<i>PGAs/</i> Maschinendaten wirksam setzen (NEWCONF) (Seite 150)
NEWMT	F	Neues Multitool anlegen		+	-	<i>FBWs/</i>
NEWT	F	Neues Werkzeug anlegen		+	-	<i>FBWs/</i>
NORM ⁶⁾	G	Normaleinstellung im Anfangs-, Endpunkt bei der Werkzeugkorrektur	m	+		<i>PGs/</i>
NOT	K	Logisches NICHT (Negation)		+		<i>PGAs/</i> Vergleichs- und logische Operationen (Seite 75)
NPROT	P	Maschinenspezifischer Schutzbereich EIN/AUS		+	-	<i>PGAs/</i> Schutzbereiche aktivieren/deaktivieren (CPROT, NPROT) (Seite 237)
NPROTDEF	P	Definition eines maschinenspezifischen Schutzbereichs		+	-	<i>PGAs/</i> Festlegung der Schutzbereiche (CPROTDEF, NPROTDEF) (Seite 233)
NUMBER	F	Eingangsstring in Zahl umwandeln		+	-	<i>PGAs/</i> Typenkonvertierung von STRING (NUMBER, ISNUMBER, AXNAME) (Seite 85)
OEMIPO1	G	OEM-Interpolation 1	m	+		<i>PGAs/</i> Spezielle Funktionen für den OEM-Anwender (OMA1 ... OMA5, OEMIPO1, OEMIPO2, G810 ... G829) (Seite 287)
OEMIPO2	G	OEM-Interpolation 2	m	+		<i>PGAs/</i> Spezielle Funktionen für den OEM-Anwender (OMA1 ... OMA5, OEMIPO1, OEMIPO2, G810 ... G829) (Seite 287)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
OF	K	Schlüsselwort in der CASE-Verzweigung		+		<i>PGAs/</i> Programmverzweigung (CASE ... OF ... DEFAULT ...) (Seite 106)
OFFN	A	Aufmaß zur programmierten Kontur	m	+		<i>PGs/</i>
OMA1	A	OEM-Adresse 1	m	+		<i>PGAs/</i> Spezielle Funktionen für den OEM-Anwender (OMA1 ... OMA5, OEMIPO1, OEMIPO2, G810 ... G829) (Seite 287)
OMA2	A	OEM-Adresse 2	m	+		<i>PGAs/</i> Spezielle Funktionen für den OEM-Anwender (OMA1 ... OMA5, OEMIPO1, OEMIPO2, G810 ... G829) (Seite 287)
OMA3	A	OEM-Adresse 3	m	+		<i>PGAs/</i> Spezielle Funktionen für den OEM-Anwender (OMA1 ... OMA5, OEMIPO1, OEMIPO2, G810 ... G829) (Seite 287)
OMA4	A	OEM-Adresse 4	m	+		<i>PGAs/</i> Spezielle Funktionen für den OEM-Anwender (OMA1 ... OMA5, OEMIPO1, OEMIPO2, G810 ... G829) (Seite 287)
OMA5	A	OEM-Adresse 5	m	+		<i>PGAs/</i> Spezielle Funktionen für den OEM-Anwender (OMA1 ... OMA5, OEMIPO1, OEMIPO2, G810 ... G829) (Seite 287)
OR	K	Logischer Operator, ODER-Verknüpfung		+		<i>PGAs/</i> Vergleichs- und logische Operationen (Seite 75)
ORIXES	G	Lineare Interpolation der Maschinenachsen oder Orientierungsachsen	m	+		<i>PGAs/</i> Programmierung der Orientierungsachsen (ORIXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) (Seite 348)
ORIXPOS	G	Orientierungswinkel über virtuelle Orientierungsachsen mit Rundachspositionen	m	+		<i>PGAs/</i> Programmierung der Werkzeugorientierung (A..., B..., C..., LEAD, TILT) (Seite 338)
ORIC ⁶⁾	G	Orientierungsänderungen an Außenecken werden dem einzufügenden Kreissatz überlagert	m	+		<i>PGAs/</i> Werkzeugorientierung (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) (Seite 437)
ORICONCCW	G	Interpolation auf einer Kreismantelfläche im Gegenuhrzeigersinn	m	+		<i>PGAs//FB3s/ (F3)</i> Orientierungsprogrammierung entlang einer Kegelmantelfläche (ORIPANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) (Seite 351)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
ORICONCW	G	Interpolation auf einer Kreismantelfläche im Uhrzeigersinn	m	+		<i>PGAs//FB3sI (F4)</i> Orientierungsprogrammierung entlang einer Kegelmantelfläche (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) (Seite 351)
ORICONIO	G	Interpolation auf einer Kreismantelfläche mit Angabe einer Zwischenorientierung	m	+		<i>PGAs//FB3sI (F4)</i> Orientierungsprogrammierung entlang einer Kegelmantelfläche (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) (Seite 351)
ORICONTO	G	Interpolation auf einer Kreismantelfläche im tangentialen Übergang (Angabe der Endorientierung)	m	+		<i>PGAs//FB3sI (F5)</i> Orientierungsprogrammierung entlang einer Kegelmantelfläche (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) (Seite 351)
ORICURVE	G	Interpolation der Orientierung mit Vorgabe der Bewegung zweier Kontaktpunkte des Werkzeugs	m	+		<i>PGAs//FB3sI (F6)</i> Orientierungsvorgabe zweier Kontaktpunkte (ORICURVE, PO[XH]=, PO[YH]=, PO[ZH]=) (Seite 355)
ORID	G	Orientierungsänderungen werden vor dem Kreissatz ausgeführt	m	+		<i>PGAs/</i> Werkzeugorientierung (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) (Seite 437)
ORIEULER ⁶⁾	G	Orientierungswinkel über Euler-Winkel	m	+		<i>PGAs/</i> Programmierung der Orientierungsachsen (ORIAxes, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) (Seite 348)
ORIMKS	G	Werkzeugorientierung im Maschinen-Koordinatensystem	m	+		<i>PGAs/</i> Bezug der Orientierungsachsen (ORIWKS, ORIMKS) (Seite 346)
ORIPATH	G	Werkzeugorientierung bezogen auf die Bahn	m	+		<i>PGAs/</i> Bahnrelative Drehung der Werkzeugorientierung (ORIPATH, ORIPATHS, Drehwinkel) (Seite 363)
ORIPATHS	G	Werkzeugorientierung bezogen auf die Bahn, ein Knick im Orientierungsverlauf wird geglättet	m	+		<i>PGAs/</i> Bahnrelative Drehung der Werkzeugorientierung (ORIPATH, ORIPATHS, Drehwinkel) (Seite 363)
ORIPLANE	G	Interpolation in einer Ebene (entspricht ORIVECT) Großkreisinterpolation	m	+		<i>PGAs/</i> Orientierungsprogrammierung entlang einer Kegelmantelfläche (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) (Seite 351)
ORIRESET	P	Grundstellung der Werkzeugorientierung mit bis zu 3 Orientierungsachsen		+	-	<i>PGAs/</i> Varianten der Orientierungsprogrammierung und Grundstellung (ORIRESET) (Seite 336)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
ORIROTA ⁶⁾	G	Drehwinkel zu einer absolut vorgegebenen Drehrichtung	m	+		<i>PGAs/</i> Drehungen der Werkzeugorientierung (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) (Seite 359)
ORIROTC	G	Tangentiale Drehvektor zur Bahntangente	m	+		<i>PGAs/</i> Drehungen der Werkzeugorientierung (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) (Seite 359)
ORIROTR	G	Drehwinkel relativ zur Ebene zwischen Start- und Endorientierung	m	+		<i>PGAs/</i> Drehungen der Werkzeugorientierung (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) (Seite 359)
ORIROTT	G	Drehwinkel relativ zur Änderung des Orientierungsvektors	m	+		<i>PGAs/</i> Drehungen der Werkzeugorientierung (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) (Seite 359)
ORIRPY	G	Orientierungswinkel über RPY-Winkel (XYZ)	m	+		<i>PGAs/</i> Programmierung der Orientierungsachsen (ORIXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) (Seite 348)
ORIRPY2	G	Orientierungswinkel über RPY-Winkel (ZYX)	m	+		<i>PGAs/</i> Programmierung der Orientierungsachsen (ORIXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) (Seite 348)
ORIS	A	Orientierungsänderung	m	+		<i>PGAs/</i> Werkzeugorientierung (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) (Seite 437)
ORISOF ⁶⁾	G	Glättung des Orientierungsverlaufs AUS	m	+		<i>PGAs/</i> Glättung des Orientierungsverlaufs (ORISON, ORISOF) (Seite 371)
ORISON	G	Glättung des Orientierungsverlaufs EIN	m	+		<i>PGAs/</i> Glättung des Orientierungsverlaufs (ORISON, ORISOF) (Seite 371)
ORIVECT ⁶⁾	G	Großkreisinterpolation (identisch mit ORIPLANE)	m	+		<i>PGAs/</i> Programmierung der Orientierungsachsen (ORIXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) (Seite 348)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
ORIVIRT1	G	Orientierungswinkel über virtuelle Orientierungsachsen (Definition 1)	m	+		<i>PGAs/</i> Programmierung der Orientierungsachsen (ORIAxes, ORIVect, ORIEuler, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) (Seite 348)
ORIVIRT2	G	Orientierungswinkel über virtuelle Orientierungsachsen (Definition 1)	m	+		<i>PGAs/</i> Programmierung der Orientierungsachsen (ORIAxes, ORIVect, ORIEuler, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2) (Seite 348)
ORIWKS ⁶⁾	G	Werkzeugorientierung im Werkstück-Koordinatensystem	m	+		<i>PGAs/</i> Bezug der Orientierungsachsen (ORIWKS, ORIMKS) (Seite 346)
OS	K	Pendeln ein/aus		+		<i>PGAs/</i> Asynchrones Pendeln (OS, OSP1, OSP2, OST1, OST2, OSCtrl, OSnSC, OSE, OSB) (Seite 571)
OSB	K	Pendeln: Startpunkt	m	+		<i>FB1s/ (P5)</i>
OSC	G	Konstante Glättung Werkzeugorientierung	m	+		<i>PGAs/</i> Werkzeugorientierung (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) (Seite 437)
OSCILL	K	Axis: 1 - 3 Zustellachsen	m	+		<i>PGAs/</i> Über Synchronaktionen gesteuertes Pendeln (OSCILL) (Seite 577)
OSCTRL	K	Optionen pendeln	m	+		<i>PGAs/</i> Asynchrones Pendeln (OS, OSP1, OSP2, OST1, OST2, OSCtrl, OSnSC, OSE, OSB) (Seite 571)
OSD	G	Überschleifen de Werkzeugorientierung durch Vorgabe der Überschleiflänge mit SD	m	+		<i>PGAs/</i> Werkzeugorientierung (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) (Seite 437)
OSE	K	Pendeln Endpunkt	m	+		<i>PGAs/</i> Asynchrones Pendeln (OS, OSP1, OSP2, OST1, OST2, OSCtrl, OSnSC, OSE, OSB) (Seite 571)
OSnSC	K	Pendeln: Ausfunktanzahl	m	+		<i>PGAs/</i> Asynchrones Pendeln (OS, OSP1, OSP2, OST1, OST2, OSCtrl, OSnSC, OSE, OSB) (Seite 571)
OSOF ⁶⁾	G	Glättung der Werkzeugorientierung AUS	m	+		<i>PGAs/</i> Asynchrones Pendeln (OS, OSP1, OSP2, OST1, OST2, OSCtrl, OSnSC, OSE, OSB) (Seite 571)

17.1 Anweisungen

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
OSP1	K	Pendeln: linker Umkehrpunkt	m	+		<i>PGAs/</i> Asynchrones Pendeln (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) (Seite 571)
OSP2	K	Pendeln rechter Umkehrpunkt	m	+		<i>PGAs/</i> Asynchrones Pendeln (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) (Seite 571)
OSS	G	Glättung der Werkzeugorientierung am Satzende	m	+		<i>PGAs/</i> Werkzeugorientierung (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) (Seite 437)
OSSE	G	Glättung der Werkzeugorientierung am Satzanfang und Satzende	m	+		<i>PGAs/</i> Werkzeugorientierung (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) (Seite 437)
OST	G	Überschleifen der Werkzeugorientierung durch Vorgabe der Winkeltoleranz in Grad mit dem SD (maximale Abweichung vom programmiert. Orientierungsverlauf)	m	+		<i>PGAs/</i> Werkzeugorientierung (ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST) (Seite 437)
OST1	K	Pendeln: Haltepunkt im linken Umkehrpunkt	m	+		<i>PGAs/</i> Asynchrones Pendeln (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) (Seite 571)
OST2	K	Pendeln: Haltepunkt im rechten Umkehrpunkt	m	+		<i>PGAs/</i> Asynchrones Pendeln (OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) (Seite 571)
OTOL	K	Orientierungstoleranz für Kompressor-Funktionen, Orientierungsglättung und Überschleifarten		+		<i>PGAs/</i> Programmierbare Kontur-/Orientierungstoleranz (CTOL, OTOL, ATOL) (Seite 509)
OVR	K	Drehzahlkorrektur	m	+		<i>PGAs/</i>
OVRA	K	Axiale Drehzahlkorrektur	m	+	+	<i>PGAs/</i>
OVRRAP	K	Eilgang-Korrektur	m	+		<i>PGAs/</i>
P	A	Anzahl Unterprogramm-durchläufe		+		<i>PGAs/</i> Anzahl der Programmwiederholungen (P) (Seite 203)
PAROT	G	Werkstückkoordinatensystem am Werkstück ausrichten	m	+		<i>PGs/</i>
PAROTOF ⁶⁾	G	Werkstückbezogene Frame-Drehung ausschalten	m	+		<i>PGs/</i>
PCALL	K	Unterprogramme mit absoluter Pfadangabe und Parameterübergabe aufrufen		+		<i>PGAs/</i> Unterprogramm mit Pfadangabe und Parametern aufrufen (PCALL) (Seite 210)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
PDELAYOF	G	Verzögerung beim Stanzen AUS	m	+		<i>PGAs/</i> Stanzen und Nibbeln ein oder aus (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) (Seite 585)
PDELAYON ⁶⁾	G	Verzögerung beim Stanzen EIN	m	+		<i>PGAs/</i> Stanzen und Nibbeln ein oder aus (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) (Seite 585)
PHI	K	Drehwinkel der Orientierung um die Richtungsachse des Kegels		+		<i>PGAs/</i> Orientierungsprogrammierung entlang einer Kegelmantelfläche (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) (Seite 351)
PHU	K	Physikalische Einheit einer Variablen		+		<i>PGAs/</i> Definition von Anwendervariablen (DEF) (Seite 25)
PL	A	1. B-Spline: Knotenabstand 2. Polynom-Interpolation: Länge des Parameterintervalls bei Polynom-Interpolation	s	+		<i>PGAs/</i> 1. Spline-Interpolation (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) (Seite 250) 2. Polynom-Interpolation (POLY, POLYPATH, PO, PL) (Seite 265)
PM	K	pro Minute		+		<i>PGs/</i>
PO	K	Polynomkoeffizient bei Polynom-Interpolation	s	+		<i>PGAs/</i> Polynom-Interpolation (POLY, POLYPATH, PO, PL) (Seite 265)
POCKET3	C	Technologischer Zyklus: Rechtecktasche fräsen		+		<i>PGAs/</i> Rechtecktasche fräsen - POCKET3 (Seite 685)
POCKET4	C	Technologischer Zyklus: Kreistasche fräsen		+		<i>PGAs/</i> Kreistasche fräsen - POCKET4 (Seite 687)
POLF	K	Rückzugsposition LIFTFAST	m	+		<i>PGs//PGAs/</i>
POLFA	P	Rückzugsposition von Einzelachsen mit \$AA_ESR_TRIGGER starten	m	+	+	<i>PGs/</i>
POLFMASK	P	Achsen für den Rückzug ohne Zusammenhang zwischen den Achsen freigeben	m	+	-	<i>PGs/</i>
POLFMLIN	P	Achsen für den Rückzug mit linearen Zusammenhang zwischen den Achsen freigeben	m	+	-	<i>PGs/</i>
POLY	G	Polynom-Interpolation	m	+		<i>PGAs/</i> Polynom-Interpolation (POLY, POLYPATH, PO, PL) (Seite 265)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
POLYPATH	P	Polynom-Interpolation selektierbar für die Achsgruppen AXIS oder VECT	m	+	-	<i>PGAs/</i> Polynom-Interpolation (POLY, POLYPATH, PO, PL) (Seite 265)
PON	G	Stanzen EIN	m	+		<i>PGAs/</i> Stanzen und Nibbeln ein oder aus (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) (Seite 585)
PONS	G	Stanzen EIN im IPO-Takt	m	+		<i>PGAs/</i> Stanzen und Nibbeln ein oder aus (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) (Seite 585)
POS	K	Achse positionieren		+	+	<i>PGs/</i>
POSA	K	Achse positionieren über Satzgrenze		+	+	<i>PGs/</i>
POSM	P	Magazin positionieren		+	-	<i>FBWs/</i>
POSMT	P	Multitool auf WZ-Halter auf Platznummer positionieren		+	-	<i>FBWs/</i>
POSP	K	Positionieren in Teilstücken (Pendeln)		+		<i>PGs/</i>
POSRANGE	F	Ermitteln, ob sich die aktuell interpolierte Sollposition einer Achse in einem Fenster um eine vorgegebene Referenzposition befindet		+	+	<i>FBSY</i>
POT	F	Quadrat (Arithmetische Funktion)		+	+	<i>PGAs/</i> Rechenfunktionen (Seite 73)
PR	K	Pro Umdrehung		+		<i>PGs/</i>
PREPRO	PA	Unterprogramme mit Vorbereitung kennzeichnen		+		<i>PGAs/</i> Unterprogramme mit Vorbereitung kennzeichnen (PREPRO) (Seite 190)
PRESETON	P	Istwertsetzen für programmierte Achsen		+	+	<i>PGAs/</i> Orientierungsprogrammierung entlang einer Kegelmantelfläche (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) (Seite 351)
PRIO	K	Schlüsselwort zum Setzen der Priorität bei der Behandlung von Interrupts		+		<i>PGAs/</i> Interruptroutine zuordnen und starten (SETINT, PRIO, BLSYNC) (Seite 132)
PRLOC	K	Initialisierung der Variablen bei Reset nur nach lokaler Änderung		+		<i>PGAs/</i> Attribut: Initialisierungswert (Seite 34)
PROC	K	Erste Anweisung eines Programms		+		<i>PGAs/</i> Unterprogramm mit Pfadangabe und Parametern aufrufen (PCALL) (Seite 210)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
PSI	K	Öffnungswinkel des Kegels		+		<i>PGAs/</i> Orientierungsprogrammierung entlang einer Kegelmantelfläche (ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) (Seite 351)
PTP	G	Punkt-zu-Punkt-Bewegung	m	+		<i>PGAs/</i> Kartesisches PTP-Fahren (Seite 390)
PTPG0	G	Punkt-zu-Punkt-Bewegung nur bei G0, sonst CP	m	+		<i>PGAs/</i> PTP bei TRANSMIT (Seite 395)
PUNCHACC	P	Wegabhängige Beschleunigung beim Nibbeln		+	-	<i>PGAs/</i> Stanzen und Nibbeln ein oder aus (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) (Seite 585)
PUTFTOC	P	Werkzeugfeinkorrektur für paralleles Abrichten		+	-	<i>PGAs/</i> Online-Werkzeugkorrektur (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) (Seite 419)
PUTFTOCF	P	Werkzeugfeinkorrektur in Abhängigkeit einer mit FCTDEF festgelegten Funktion für paralleles Abrichten		+	-	<i>PGAs/</i> Online-Werkzeugkorrektur (PUTFTOCF, FCTDEF, PUTFTOC, FTOCON, FTOCOF) (Seite 419)
PW	A	B-Spline, Punkt-Gewicht	s	+		<i>PGAs/</i> Spline-Interpolation (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) (Seite 250)
QU	K	Schnelle Zusatz-(Hilfs-)funktionsausgabe		+		<i>PGs/</i>
R...	A	Rechenparameter auch als einstellbarer Adressbezeichner und mit numerischer Erweiterung		+		<i>PGAs/</i> Vordefinierte Anwendervariablen: Rechenparameter (R) (Seite 21)
RAC	K	Absolut satzweise achsspezifische Radiusprogrammierung	s	+		<i>PGs/</i>
RDISABLE	P	Einlesesperre		-	+	<i>FBSY</i>
READ	P	Liest in der angegebenen Datei eine oder mehrere Zeilen ein und legt gelesene Informationen im Feld ab		+	-	<i>PGAs/</i> Zeilen in Datei lesen (READ) (Seite 158)
REAL	K	Datentyp: Gleitpunktvariable mit Vorzeichen (reale Zahlen)		+		<i>PGAs/</i> Definition von Anwendervariablen (DEF) (Seite 25)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
REDEF	K	Einstellung für Maschinendaten, NC-Sprachelemente und Systemvariablen, bei welchen Benutzergruppen sie angezeigt werden		+		<i>PGAs/</i> Redefinition von Systemvariablen, Anwendervariablen und NC-Sprachbefehlen (REDEF) (Seite 31)
RELEASE	P	Maschinenachsen zum Achstausch freigeben		+	+	<i>PGAs/</i> Achstausch, Spindeltausch (RELEASE, GET, GETD) (Seite 143)
REP	K	Schlüsselwort zur Initialisierung aller Elemente eines Feldes mit demselben Wert		+		<i>PGAs/</i> Definition und Initialisierung von Feldvariablen (DEF, SET, REP) (Seite 47)
REPEAT	K	Wiederholung einer Programmschleife		+		<i>PGAs/</i> Programmteilwiederholung (REPEAT, REPEATB, ENDLABEL, P) (Seite 108)
REPEATB	K	Wiederholung einer Programmzeile		+		<i>PGAs/</i> Programmteilwiederholung (REPEAT, REPEATB, ENDLABEL, P) (Seite 108)
REPOSA	G	Wiederanfahren an die Kontur linear mit allen Achsen	s	+		<i>PGAs/</i> Wiederanfahren an Kontur (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) (Seite 495)
REPOSH	G	Wiederanfahren an die Kontur mit Halbkreis	s	+		<i>PGAs/</i> Wiederanfahren an Kontur (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) (Seite 495)
REPOSHA	G	Wiederanfahren an die Kontur mit allen Achsen; Geometrieachsen im Halbkreis	s	+		<i>PGAs/</i> Wiederanfahren an Kontur (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) (Seite 495)
REPOSL	G	Wiederanfahren an die Kontur linear	s	+		<i>PGAs/</i> Wiederanfahren an Kontur (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) (Seite 495)
REPOSQ	G	Wiederanfahren an die Kontur im Viertelkreis	s	+		<i>PGAs/</i> Wiederanfahren an Kontur (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) (Seite 495)
REPOSQA	G	Wiederanfahren an die Kontur linear mit allen Achsen; Geometrieachsen im Viertelkreis	s	+		<i>PGAs/</i> Wiederanfahren an Kontur (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) (Seite 495)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
RESET	P	Technologiezyklus rücksetzen		-	+	<i>FBSY</i>
RESETMON	P	Sprachbefehl zur Sollwertaktivierung		+	-	<i>FBWsl</i>
RET	P	Unterprogrammende		+	+	<i>PGAsl</i> Parametrierbarer Unterprogrammrücksprung (RET ...) (Seite 193)
RIC	K	Relativ satzweise achsspezifische Radiusprogrammierung	s	+		<i>PGsl</i> Achstausch, Spindeltausch (RELEASE, GET, GETD) (Seite 143)
RINDEX	F	Index eines Zeichens im Eingangsstring bestimmen		+	-	<i>PGAsl</i> Zeichen/String in String suchen (INDEX, RINDEX, MINDEX, MATCH) (Seite 89)
RMB	G	Wiederanfahren an Satzanfangspunkt	m	+		<i>PGAsl</i> Wiederanfahren an Kontur (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) (Seite 495)
RME	G	Wiederanfahren an Satzendpunkt	m	+		<i>PGAsl</i> Wiederanfahren an Kontur (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) (Seite 495)
RMI ⁶⁾	G	Wiederanfahren an Unterbrechungspunkt	m	+		<i>PGAsl</i> Wiederanfahren an Kontur (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) (Seite 495)
RMN	G	Wiederanfahren an nächstliegenden Bahnpunkt	m	+		<i>PGAsl</i> Wiederanfahren an Kontur (REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMI, RMB, RME, RMN) (Seite 495)
RND	A	Konturecke verrunden	s	+		<i>PGsl</i>
RNDM	A	Modales Verrunden	m	+		<i>PGsl</i>
ROT	G	Programmierbare Drehung	s	+		<i>PGsl</i>
ROTS	G	Programmierbare Frame-Drehungen mit Raumwinkeln	s	+		<i>PGsl</i>
ROUND	F	Runden der Nachkommastellen		+	+	<i>PGAsl</i> Rechenfunktionen (Seite 73)
ROUNDUP	F	Aufrunden eines Eingabewerts		+	+	<i>PGAsl</i> Aufrunden (ROUNDUP) (Seite 168)
RP	A	Polarradius	m/s	+		<i>PGsl</i>
RPL	A	Drehung in der Ebene	s	+		<i>PGsl</i>
RT	K	Parameter für Zugriff auf Framedaten: Drehung		+		<i>PGAsl</i> Framekomponenten lesen und verändern (TR, FI, RT, SC, MI) (Seite 302)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
RTLIOF	G	G0 ohne Linearinterpolation (Einzelachsinterpolation)	m	+		<i>PGs/</i>
RTLION ⁶⁾	G	G0 mit Linearinterpolation	m	+		<i>PGs/</i>
S	A	Spindeldrehzahl (bei G4, G96/G961 andere Bedeutung)	m/s	+	+	<i>PGs/</i>
SAVE	PA	Attribut zur Rettung von Informationen bei Unterprogrammaufrufen		+		<i>PGAs/</i> Modale G-Funktionen sichern (SAVE) (Seite 180)
SBLOF	P	Einzelsatz unterdrücken		+	-	<i>PGAs/</i> Einzelsatzbearbeitung unterdrücken (SBLOF, SBLON) (Seite 181)
SBLON	P	Einzelsatzunterdrückung aufheben		+	-	<i>PGAs/</i> Einzelsatzbearbeitung unterdrücken (SBLOF, SBLON) (Seite 181)
SC	K	Parameter für Zugriff auf Framedaten: Skalierung		+		<i>PGAs/</i> Framekomponenten lesen und verändern (TR, FI, RT, SC, MI) (Seite 302)
SCALE	G	Programmierbare Skalierung	s	+		<i>PGs/</i>
SCC	K	Selektive Zuordnung einer Planachse zu G96/G961/G962. Achsbezeichner können Geo-, Kanal oder Maschinenachse sein.		+		<i>PGs/</i>
SCPARA	K	Servo-Parameterersatz programmieren		+	+	<i>PGAs/</i> Programmierbare Parametersatzumschaltung (SCPARA) (Seite 610)
SD	A	Spline-Grad	s	+		<i>PGAs/</i> Spline-Interpolation (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) (Seite 250)
SET	K	Schlüsselwort zur Initialisierung aller Elemente eines Feldes mit aufgelisteten Werten		+		<i>PGAs/</i> Definition und Initialisierung von Feldvariablen (DEF, SET, REP) (Seite 47)
SETAL	P	Alarm setzen		+	+	<i>PGAs/</i> Alarmer (SETAL) (Seite 633)
SETDNO	F	D-Nummer der Schneide (CE) eines Werkzeugs (T) zuordnen		+	-	<i>PGAs/</i> Freie D-Nummernvergabe: D-Nummern umbenennen (GETDNO, SETDNO) (Seite 445)
SETINT	K	Festlegung, welche Interruptroutine aktiviert werden soll, wenn ein NCK-Eingang ansteht		+		<i>PGAs/</i> Interruptroutine zuordnen und starten (SETINT, PRIO, BLSYNC) (Seite 132)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
SETM	P	Setzen von Markern im eigenen Kanal		+	+	PGAs/ Programmkoordinierung (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) (Seite 124)
SETMS	P	Zurückschalten auf die im Maschinendatum festgelegte Masterspindel		+	-	PGs/
SETMS(n)	P	Spindel n soll als Masterspindel gelten		+		PGs/
SETMTH	P	Masterwerkzeughalternummer setzen		+	-	FBWs/
SETPIECE	P	Stückzahl für alle Werkzeuge berücksichtigen, die der Spindel zugeordnet sind		+	-	FBWs/
SETTA	P	Werkzeug aus Verschleißverbund aktiv setzen		+	-	FBWs/
SETTCOR	F	Veränderung von Werkzeugkomponenten unter Berücksichtigung aller Randbedingungen		+	-	FB1s/ (W1)
SETTIA	P	Werkzeug aus Verschleißverbund inaktiv setzen		+	-	FBWs/
SF	A	Startpunktversatz für Gewindeschneiden	m	+		PGs/
SIN	F	Sinus (Trigon. Funktion)		+	+	PGAs/ Rechenfunktionen (Seite 73)
SIRELAY	F	Die mit SIRELIN, SIRELOUT und SIRELTIME parametrisierten Sicherheitfunktionen aktivieren		-	+	FBSIs/
SIRELIN	P	Eingangsgrößen des Funktionsbausteins initialisieren		+	-	FBSIs/
SIRELOUT	P	Ausgangsgrößen des Funktionsbausteins initialisieren		+	-	FBSIs/
SIRELTIME	P	Timer des Funktionsbausteins initialisieren		+	-	FBSIs/
SLOT1	C	Technologischer Zyklus: Längsnut		+		PGAs/ Längsnut - SLOT1 (Seite 696)
SLOT2	C	Technologischer Zyklus: Kreisnut		+		PGAs/ Kreisnut - SLOT2 (Seite 699)
SOFT	G	Ruckbegrenzte Bahnbeschleunigung	m	+		PGs/ Beschleunigungsmodus (BRISK, BRISKA, SOFT, SOFTA, DRIVE, DRIVEA) (Seite 475)
SOFTA	P	Ruckbegrenzte Achsbeschleunigung für die programmierten Achsen einschalten		+	-	PGs/ Beschleunigungsmodus (BRISK, BRISKA, SOFT, SOFTA, DRIVE, DRIVEA) (Seite 475)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
SOFTENDPR	P	Programmierbarer Softwareendschalter		+	-	
SOFTENDSA	P	Programmierbarer Softwareendschalter für Synchronaktionen		+	+	
SON	G	Nibbeln EIN	m	+		<i>PGAs/</i> Stanzen und Nibbeln ein oder aus (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) (Seite 585)
SONS	G	Nibbeln EIN im IPO-Takt	m	+		<i>PGAs/</i> Stanzen und Nibbeln ein oder aus (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) (Seite 585)
SPATH ⁶⁾	G	Bahnbezug für FGROU- Achsen ist Bogenlänge	m	+		<i>PGAs/</i> Einstellbarer Bahnbezug (SPATH, UPATH) (Seite 271)
SPCOF	P	Masterspindel oder Spindel (n) von Lageregelung in Drehzahlregelung umschalten	m	+	-	<i>PGs/</i>
SPCON	P	Masterspindel oder Spindel (n) von Drehzahlregelung in Lageregelung umschalten	m	+	-	<i>PGAs/</i>
SPI	F	Konvertiert Spindelnummer in Achsbezeichner		+	-	<i>PGAs/</i> Achsfunktionen (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) (Seite 599)
SPIF1 ⁶⁾	G	Schnelle NCK-Ein-/Ausgänge für Stanzen/Nibbeln Byte 1	m	+		<i>FB2sl (N4)</i>
SPIF2	G	Schnelle NCK-Ein-/Ausgänge für Stanzen/Nibbeln Byte 2	m	+		<i>FB2sl (N4)</i>
SPLINEPATH	P	Spline-Verband festlegen		+	-	<i>PGAs/</i> Spline-Verbund (SPLINEPATH) (Seite 261)
SPN	A	Anzahl der Teilstrecken pro Satz	s	+		<i>PGAs/</i> Automatische Wegaufteilung (Seite 590)
SPOF ⁶⁾	G	Hub AUS, Stanzen, Nibbeln AUS	m	+		<i>PGAs/</i> Stanzen und Nibbeln ein oder aus (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) (Seite 585)
SPOS	K	Spindelposition	m	+	+	<i>PGs/</i>
SPOSA	K	Spindelposition über Satzgrenzen hinweg	m	+		<i>PGs/</i>

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
SPP	A	Länge einer Teilstrecke	m	+		<i>PGAs/</i> Automatische Wegaufteilung (Seite 590)
SPRINT	F	Liefert einen Eingangsstring formatiert zurück		+		<i>PGAs/</i> String formatieren (SPRINT) (Seite 93)
SQRT	F	Quadratwurzel (arithmetische Funktion) (square root)		+	+	<i>PGAs/</i> Rechenfunktionen (Seite 73)
SR	A	Pendelrückzugsweg für Synchronaktion	s	+		<i>PGs/</i>
SRA	K	Pendelrückzugsweg bei externem Eingang axial für Synchronaktion	m	+		<i>PGs/</i>
ST	A	Pendelausfeuerzeit für Synchronaktion	s	+		<i>PGs/</i>
STA	K	Pendelausfeuerzeit axial für Synchronaktion	m	+		<i>PGs/</i>
START	P	Starten der ausgewählten Programme in mehreren Kanälen gleichzeitig aus dem laufenden Programm		+	-	<i>PGAs/</i> Programmkoordinierung (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) (Seite 124)
STARTFIFO ⁶⁾	G	Abarbeiten; parallel dazu Auffüllen des Vorlaufpuffers	m	+		<i>PGAs/</i> Programmablauf mit Vorlaufspeicher (STOPFIFO, STARTFIFO, FIFOCTRL, STOPRE) (Seite 485)
STAT		Stellung der Gelenke	s	+		<i>PGAs/</i> Kartesisches PTP-Fahren (Seite 390)
STOLF	K	G0-Toleranzfaktor	m	+		<i>PGAs/</i> Toleranz bei G0-Bewegungen (STOLF) (Seite 512)
STOPFIFO	G	Anhalten der Bearbeitung; Auffüllen des Vorlaufpuffers, bis STARTFIFO erkannt wird, Vorlaufpuffer voll oder Programmende	m	+		<i>PGAs/</i> Programmablauf mit Vorlaufspeicher (STOPFIFO, STARTFIFO, FIFOCTRL, STOPRE) (Seite 485)
STOPRE	P	Vorlaufstopp, bis alle vorbereiteten Sätze vom Hauptlauf abgearbeitet sind		+	-	<i>PGAs/</i> Programmablauf mit Vorlaufspeicher (STOPFIFO, STARTFIFO, FIFOCTRL, STOPRE) (Seite 485)
STOPREOF	P	Vorlaufstopp aufheben		-	+	<i>FBSY</i>
STRING	K	Datentyp: Zeichenkette		+		<i>PGAs/</i> Definition von Anwendervariablen (DEF) (Seite 25)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
STRINGIS	F	Prüft vorhandenen NC-Sprachumfang und speziell für diesen Befehl gehörende NC-Zyklennamen, Anwendervariablen, Makros und Labelnamen, ob diese existieren, gültig, definiert oder aktiv sind.		+	-	<i>PGAs/</i> Vorhandenen NC-Sprachumfang prüfen (STRINGIS) (Seite 612)
STRLEN	F	Länge eines Strings bestimmen		+	-	<i>PGAs/</i> Länge eines Strings bestimmen (STRLEN) (Seite 89)
SUBSTR	F	Index eines Zeichens im Eingangsstring bestimmen		+	-	<i>PGAs/</i> Auswahl eines Teilstrings (SUBSTR) (Seite 91)
SUPA	G	Unterdrückung der aktuellen Nullpunktverschiebung, einschließlich programmierter Verschiebungen, Systemframes, Handradverschiebungen (DRF), externer Nullpunktverschiebung und überlagerte Bewegung	s	+		<i>PGs/</i>
SVC	K	Werkzeug-Schnittgeschwindigkeit	m	+		<i>PGs/</i>
SYNFCT	P	Auswertung eines Polynoms abhängig von einer Bedingung in der Bewegungssynchronaktion		-	+	<i>FBSY</i>
SYNR	K	Lesen der Variable erfolgt synchron, d. h. zum Abarbeitungszeitpunkt		+		<i>PGAs/</i> Definition von Anwendervariablen (DEF) (Seite 25)
SYNRW	K	Lesen und Schreiben der Variable erfolgt synchron, d. h. zum Abarbeitungszeitpunkt		+		<i>PGAs/</i> Definition von Anwendervariablen (DEF) (Seite 25)
SYNW	K	Schreiben der Variable erfolgt synchron, d. h. zum Abarbeitungszeitpunkt		+		<i>PGAs/</i> Definition von Anwendervariablen (DEF) (Seite 25)
T	A	Werkzeug aufrufen (wechseln nur, wenn im Maschinendatum festgelegt; ansonsten M6-Befehl nötig)		+		<i>PGs/</i>
TAN	F	Tangens (Trigon. Funktion)		+	+	<i>PGAs/</i> Rechenfunktionen (Seite 73)
TANG	P	Definition des Achsverbandes Tangentiales Nachführen		+	-	<i>PGAs/</i> Tangentialsteuerung (TANG, TANGON, TANGOF, TLIFT, TANGDEL) (Seite 463)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
TANGDEL	P	Löschen der Definition des Achsverbandes Tangentiales Nachführen		+	-	<i>PGAs/</i> Tangentialsteuerung (TANG, TANGON, TANGOF, TLIFT, TANGDEL) (Seite 463)
TANGOF	P	Tangentielles Nachführen AUS		+	-	<i>PGAs/</i> Tangentialsteuerung (TANG, TANGON, TANGOF, TLIFT, TANGDEL) (Seite 463)
TANGON	P	Tangentielles Nachführen EIN		+	-	<i>PGAs/</i> Tangentialsteuerung (TANG, TANGON, TANGOF, TLIFT, TANGDEL) (Seite 463)
TCA (828D: _TCA)	P	Werkzeuganwahl / Werkzeugwechsel unabhängig vom Status des Werkzeugs		+	-	<i>FBWs/</i>
TCARR	A	Werkzeugträger (Nummer "m") anfordern		+		<i>PGAs/</i> Werkzeuflängenkorrektur für orientierbare Werkzeugträger (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) (Seite 452)
TCI	P	Wechsle Werkzeug aus Zwischenspeicher in das Magazin		+	-	<i>FBWs/</i>
TCOABS ⁶⁾	G	Werkzeuflängenkomponenten aus der aktuellen Werkzeugorientierung bestimmen	m	+		<i>PGAs/</i> Werkzeuflängenkorrektur für orientierbare Werkzeugträger (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) (Seite 452)
TCOFR	G	Werkzeuflängenkomponenten aus der Orientierung des aktiven Frames bestimmen	m	+		<i>PGAs/</i> Werkzeuflängenkorrektur für orientierbare Werkzeugträger (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) (Seite 452)
TCOFRX	G	Werkzeugorientierung eines aktiven Frames bei der Werkzeugwahl bestimmen, Werkzeug zeigt in X-Richtung	m	+		<i>PGAs/</i> Werkzeuflängenkorrektur für orientierbare Werkzeugträger (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) (Seite 452)
TCOFRY	G	Werkzeugorientierung eines aktiven Frames bei der Werkzeugwahl bestimmen, Werkzeug zeigt in Y-Richtung	m	+		<i>PGAs/</i> Werkzeuflängenkorrektur für orientierbare Werkzeugträger (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) (Seite 452)
TCOFRZ	G	Werkzeugorientierung eines aktiven Frames bei der Werkzeugwahl bestimmen, Werkzeug zeigt in Z-Richtung	m	+		<i>PGAs/</i> Werkzeuflängenkorrektur für orientierbare Werkzeugträger (TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ) (Seite 452)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
THETA	A	Drehwinkel	s	+		<i>PGAs/</i> Drehungen der Werkzeugorientierung (ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) (Seite 359)
TILT	A	Seitwärtswinkel	m	+		<i>PGAs/</i> Programmierung der Werkzeugorientierung (A..., B..., C..., LEAD, TILT) (Seite 338)
TLIFT	P	Bei Tangentialsteuerung Zwischensatz an Konturrecken einfügen		+	-	<i>PGAs/</i> Tangentialsteuerung (TANG, TANGON, TANGOF, TLIFT, TANGDEL) (Seite 463)
TML	P	Werkzeuganwahl mit Magazin-Platznummer		+	-	<i>FBWs/</i>
TMOF	P	Werkzeugüberwachung abwählen		+	-	<i>PGAs/</i> Schleifenspezifische Werkzeugüberwachung im Teileprogramm (TMON, TMOF) (Seite 597)
TMON	P	Werkzeugüberwachung anwählen		+	-	<i>PGAs/</i> Schleifenspezifische Werkzeugüberwachung im Teileprogramm (TMON, TMOF) (Seite 597)
TO	K	Bezeichnet den Endwert in einer FOR-Zählschleife		+		<i>PGAs/</i> Zählschleife (FOR ... TO ..., ENDFOR) (Seite 119)
TOFF	K	Werkzeuglängen-Offset in Richtung der Werkzeuglängenkomponente, die parallel zu der im Index angegebenen Geometrieachse wirkt.	m	+		<i>PGs/</i>
TOFFL	K	Werkzeuglängen-Offset in Richtung der Werkzeuglängenkomponente L1, L2 bzw. L3	m	+		<i>PGs/</i>
TOFFOF	P	Online-Werkzeuglängenkorrektur rücksetzen		+	-	<i>PGAs/</i> Online-Werkzeuglängenkorrektur (TOFFON, TOFFOF) (Seite 455)
TOFFON	P	Online-Werkzeuglängenkorrektur aktivieren		+	-	<i>PGAs/</i> Online-Werkzeuglängenkorrektur (TOFFON, TOFFOF) (Seite 455)
TOFFR	A	Werkzeugradius-Offset	m	+		<i>PGs/</i>
TOFRAME	G	Z-Achse des WKS durch Frame-Drehung parallel zur Werkzeugorientierung ausrichten	m	+		<i>PGs/</i>

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
TOFRAMEX	G	X-Achse des WKS durch Frame-Drehung parallel zur Werkzeugorientierung ausrichten	m	+		<i>PGsl</i>
TOFRAMEY	G	Y-Achse des WKS durch Frame-Drehung parallel zur Werkzeugorientierung ausrichten	m	+		<i>PGsl</i>
TOFRAMEZ	G	wie TOFRAME	m	+		<i>PGsl</i>
TOLOWER	F	Buchstaben eines Strings in Kleinbuchstaben wandeln		+	-	<i>PGAsl</i> Wandlung in Klein-/Großbuchstaben (TOLOWER, TOUPPER) (Seite 88)
TOOLENV	F	Alle aktuellen Zustände speichern, die für die Bewertung der im Speicher abgelegten Werkzeugdaten von Bedeutung sind		+	-	<i>FB1sl (W1)</i>
TOOLGNT	F	Anzahl der Werkzeuge einer Werkzeuggruppe ermitteln		+	-	<i>FBWsl</i>
TOOLGT	F	T-Nummer eines Werkzeugs aus einer Werkzeuggruppe ermitteln		+	-	<i>FBWsl</i>
TOROT	G	Z-Achse des WKS durch Frame-Drehung parallel zur Werkzeugorientierung ausrichten	m	+		<i>PGsl</i>
TOROTOF ⁶⁾	G	Framedrehungen in Werkzeugrichtung AUS	m	+		<i>PGsl</i>
TOROTX	G	X-Achse des WKS durch Frame-Drehung parallel zur Werkzeugorientierung ausrichten	m	+		<i>PGsl</i>
TOROTY	G	Y-Achse des WKS durch Frame-Drehung parallel zur Werkzeugorientierung ausrichten	m	+		<i>PGsl</i>
TOROTZ	G	wie TOROT	m	+		<i>PGsl</i>
TOUPPER	F	Buchstaben eines Strings in Großbuchstaben wandeln		+	-	<i>PGAsl</i> Wandlung in Klein-/Großbuchstaben (TOLOWER, TOUPPER) (Seite 88)
TOWBCS	G	Verschleißwerte im Basiskoordinatensystem (BKS)	m	+		<i>PGAsl</i> Koordinatensystem der aktiven Bearbeitung (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) (Seite 415)

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
TOWKCS	G	Verschleißwerte im Koordinatensystem des Werkzeugkopfes bei kinetischer Transformation (unterscheidet sich vom MKS durch Werkzeugdrehung)	m	+		<i>PGAs/</i> Koordinatensystem der aktiven Bearbeitung (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) (Seite 415)
TOWMCS	G	Verschleißwerte im Maschinen-Koordinatensystem (MKS)	m	+		<i>PGAs/</i> Koordinatensystem der aktiven Bearbeitung (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) (Seite 415)
TOWSTD ⁶⁾	G	Grundstellungswert für Korrekturen in der Werkzeuglänge	m	+		<i>PGAs/</i> Koordinatensystem der aktiven Bearbeitung (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) (Seite 415)
TOWTCS	G	Verschleißwerte im Werkzeug-Koordinatensystem (Werkzeugträgerbezugspunkt T an der Werkzeughalteraufnahme)	m	+		<i>PGAs/</i> Koordinatensystem der aktiven Bearbeitung (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) (Seite 415)
TOWWCS	G	Verschleißwerte im Werkstück-Koordinatensystem (WKS)	m	+		<i>PGAs/</i> Koordinatensystem der aktiven Bearbeitung (TOWSTD, TOWMCS, TOWWCS, TOWBCS, TOWTCS, TOWKCS) (Seite 415)
TR	K	Verschiebungskomponente einer Frame-Variablen		+		<i>PGAs/</i> Framekomponenten lesen und verändern (TR, FI, RT, SC, MI) (Seite 302)
TRAANG	P	Transformation schräge Achse		+	-	<i>PGAs/</i> Schräge Achse (TRAANG) (Seite 385)
TRACON	P	Kaskadierte Transformation		+	-	<i>PGAs/</i> Verkettete Transformationen (TRACON, TRAFOOF) (Seite 401)
TRACYL	P	Zylinder: Mantelflächen-Transformation		+	-	<i>PGAs/</i> Zylindermanteltransformation (TRACYL) (Seite 377)
TRAFOOF	P	Im Kanal aktive Transformationen ausschalten		+	-	<i>PGAs/</i> Verkettete Transformationen (TRACON, TRAFOOF) (Seite 401)
TRAILOF	P	Achssynchrones Mitschleppen AUS		+	+	<i>PGAs/</i> Mitschleppen (TRAILON, TRAILOF) (Seite 515)
TRAILON	P	Achssynchrones Mitschleppen EIN		+	+	<i>PGAs/</i> Mitschleppen (TRAILON, TRAILOF) (Seite 515)
TRANS	G	Programmierbare Verschiebung	s	+		<i>PGs/</i>

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
TRANSMIT	P	Polar-Transformation (Stirnflächenbearbeitung)		+	-	<i>PGAs/</i> Fräsbearbeitung an Drehteilen (TRANSMIT) (Seite 373)
TRAORI	P	4-, 5-Achstransformation, Generische Transformation		+	-	<i>PGAs/</i> Drei, Vier, und Fünf- Achs- Transformation (TRAORI) (Seite 334)
TRUE	K	Logische Konstante: wahr		+		<i>PGAs/</i> Definition von Anwendervariablen (DEF) (Seite 25)
TRUNC	F	Abschneiden der Nachkommastellen		+	+	<i>PGAs/</i> Genauigkeitskorrektur bei Vergleichsfehlern (TRUNC) (Seite 77)
TU		Achswinkel	s	+		<i>PGAs/</i> Kartesisches PTP-Fahren (Seite 390)
TURN	A	Windungszahl für Schraubenlinie	s	+		<i>PGs/</i>
ULI	K	Oberer Grenzwert von Variablen		+		<i>PGAs/</i> Attribut: Grenzwerte (LLI, ULI) (Seite 37)
UNLOCK	P	Synchronaktion mit ID freigeben (Technologiezyklus fortsetzen)		-	+	<i>FBSY</i>
UNTIL	K	Bedingung zur Beendigung einer REPEAT-Schleife		+		<i>PGAs/</i> Programmschleife mit Bedingung am Schleifenanfang (WHILE, ENDWHILE) (Seite 121)
UPATH	G	Bahnbezug für FGROU- Achsen ist Kurvenparameter	m	+		<i>PGAs/</i> Einstellbarer Bahnbezug (SPATH, UPATH) (Seite 271)
VAR	K	Schlüsselwort: Art der Parameterübergabe		+		<i>PGAs/</i> Unterprogrammaufruf mit Parameterübergabe (EXTERN) (Seite 201)
VELOLIM	K	Reduktion der maximalen axialen Geschwindigkeit	m	+		<i>PGAs/</i> Prozentuale Geschwindigkeitskorrektur (VELOLIM) (Seite 506)
VELOLIMA	K	Reduktion oder Überhöhung der maximalen axialen Geschwindigkeit der Folgeachse	m	+	+	<i>PGAs/</i> Beeinflussung der Beschleunigung bei Folgeachsen (VELOLIMA, ACCLIMA, JERKLIMA) (Seite 478)
WAITC	P	Warten, bis Kopplungssatzwechselkriterium für die Achsen/Spindeln erfüllt ist		+	-	<i>PGAs/</i> Programmkoordinierung (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) (Seite 124)
WAITE	P	Warten auf das Programmende in einem anderen Kanal.		+	-	<i>PGAs/</i> Programmkoordinierung (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) (Seite 124)

17.1 Anweisungen

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
WAITENC	P	Warten auf synchronisierte bzw. restaurierte Achspositionen		+	-	<i>PGs!</i> Warten auf gültige Achsposition (WAITENC) (Seite 608)
WAITM	P	Warten auf Marker im angegebenen Kanal; vorhergehenden Satz mit Genauhalt beenden.		+	-	<i>PGs!</i> Programmkoordinierung (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) (Seite 124)
WAITMC	P	Warten auf Marker im angegebenen Kanal; Genauhalt nur, wenn die anderen Kanäle den Marker noch nicht erreicht haben.		+	-	<i>PGs!</i> Programmkoordinierung (INIT, START, WAITM, WAITMC, WAITE, SETM, CLEARM) (Seite 124)
WAITP	P	Warten auf Verfahrnde der Positionierachse		+	-	<i>PGs!</i>
WAITS	P	Warten auf Erreichen der Spindelposition		+	-	<i>PGs!</i>
WALCS0 ⁶⁾	G	WKS-Arbeitsfeldbegrenzung abgewählt	m	+		<i>PGs!</i>
WALCS1	G	WKS-Arbeitsfeldbegrenzungsgruppe 1 aktiv	m	+		<i>PGs!</i>
WALCS2	G	WKS-Arbeitsfeldbegrenzungsgruppe 2 aktiv	m	+		<i>PGs!</i>
WALCS3	G	WKS-Arbeitsfeldbegrenzungsgruppe 3 aktiv	m	+		<i>PGs!</i>
WALCS4	G	WKS-Arbeitsfeldbegrenzungsgruppe 4 aktiv	m	+		<i>PGs!</i>
WALCS5	G	WKS-Arbeitsfeldbegrenzungsgruppe 5 aktiv	m	+		<i>PGs!</i>
WALCS6	G	WKS-Arbeitsfeldbegrenzungsgruppe 6 aktiv	m	+		<i>PGs!</i>
WALCS7	G	WKS-Arbeitsfeldbegrenzungsgruppe 7 aktiv	m	+		<i>PGs!</i>
WALCS8	G	WKS-Arbeitsfeldbegrenzungsgruppe 8 aktiv	m	+		<i>PGs!</i>
WALCS9	G	WKS-Arbeitsfeldbegrenzungsgruppe 9 aktiv	m	+		<i>PGs!</i>
WALCS10	G	WKS-Arbeitsfeldbegrenzungsgruppe 10 aktiv	m	+		<i>PGs!</i>
WALIMOF	G	BKS-Arbeitsfeldbegrenzung AUS	m	+		<i>PGs!</i>
WALIMON ⁶⁾	G	BKS-Arbeitsfeldbegrenzung EIN	m	+		<i>PGs!</i>
WHEN	K	Die Aktion wird zyklisch ausgeführt, wenn die Bedingung erfüllt ist.		-	+	<i>FBSY</i>
WHENEVER	K	Die Aktion wird einmal ausgeführt, wenn die Bedingung einmal erfüllt ist.		-	+	<i>FBSY</i>

Anweisung	Art ¹⁾	Bedeutung	W ²⁾	TP ³⁾	SA ⁴⁾	Beschreibung siehe ⁵⁾
WHILE	K	Beginn der WHILE- Programmschleife		+		<i>PGAs/</i> Programmschleife mit Bedingung am Schleifenanfang (WHILE, ENDWHILE) (Seite 121)
WRITE	P	Text ins Dateisystem schreiben. Fügt einen Satz am Ende der angegebenen Datei an.		+	-	<i>PGAs/</i> Datei schreiben (WRITE) (Seite 151)
WRTPR	P	Verzögert den Bearbeitungsauftrag ohne dabei den Bahnsteuerbetrieb zu unterbrechen		+	-	<i>PGAs/</i>
X	A	Achsname	m/s	+		<i>PGs/</i>
XOR	O	Logisches Exklusiv- ODER		+		<i>PGAs/</i> Vergleichs- und logische Operationen (Seite 75)
Y	A	Achsname	m/s	+		<i>PGs/</i>
Z	A	Achsname	m/s	+		<i>PGs/</i>

17.2 Anweisungen: Verfügbarkeit bei SINUMERIK 828D

Anweisung	828D-Steuerungsvariante					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Drehen	Fräsen	Drehen	Fräsen
:	•	•	•	•	•	•
*	•	•	•	•	•	•
+	•	•	•	•	•	•
-	•	•	•	•	•	•
<	•	•	•	•	•	•
<<	•	•	•	•	•	•
<=	•	•	•	•	•	•
=	•	•	•	•	•	•
>=	•	•	•	•	•	•
/	•	•	•	•	•	•
/0	•	•	•	•	•	•
...						
...						
/7	○	○	○	○	○	○
A	•	•	•	•	•	•
A2	-	-	-	-	-	-
A3	-	-	-	-	-	-
A4	-	-	-	-	-	-
A5	-	-	-	-	-	-
ABS	•	•	•	•	•	•
AC	•	•	•	•	•	•
ACC	•	•	•	•	•	•
ACCLIMA	•	•	•	•	•	•
ACN	•	•	•	•	•	•
ACOS	•	•	•	•	•	•
ACP	•	•	•	•	•	•
ACTBLOCNO	•	•	•	•	•	•
ADDFRAME	•	•	•	•	•	•
ADIS	•	•	•	•	•	•
ADISPOS	•	•	•	•	•	•
ADISPOSA	•	•	•	•	•	•
ALF	•	•	•	•	•	•
AMIRROR	•	•	•	•	•	•
AND	•	•	•	•	•	•
ANG	•	•	•	•	•	•
AP	•	•	•	•	•	•
APR	•	•	•	•	•	•
APRB	•	•	•	•	•	•

Anweisung	828D-Steuerungsvariante					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Drehen	Fräsen	Drehen	Fräsen
APRP	•	•	•	•	•	•
APW	•	•	•	•	•	•
APWB	•	•	•	•	•	•
APWP	•	•	•	•	•	•
APX	•	•	•	•	•	•
AR	•	•	•	•	•	•
AROT	•	•	•	•	•	•
AROTS	•	•	•	•	•	•
AS	•	•	•	•	•	•
ASCALE	•	•	•	•	•	•
ASIN	•	•	•	•	•	•
ASPLINE	-	○	-	○	-	○
ATAN2	•	•	•	•	•	•
ATOL	-	•	-	•	-	•
ATRANS	•	•	•	•	•	•
AX	•	•	•	•	•	•
AXCTSWE	-	-	-	-	-	-
AXCTSWEC	-	-	-	-	-	-
AXCTSWED	-	-	-	-	-	-
AXIS	•	•	•	•	•	•
AXNAME	•	•	•	•	•	•
AXSTRING	•	•	•	•	•	•
AXTOCHAN	•	•	•	•	•	•
AXTOSPI	•	•	•	•	•	•
B	•	•	•	•	•	•
B2	-	-	-	-	-	-
B3	-	-	-	-	-	-
B4	-	-	-	-	-	-
B5	-	-	-	-	-	-
B_AND	•	•	•	•	•	•
B_OR	•	•	•	•	•	•
B_NOT	•	•	•	•	•	•
B_XOR	•	•	•	•	•	•
BAUTO	-	○	-	○	-	○
BLOCK	•	•	•	•	•	•
BLSYNC	•	•	•	•	•	•
BNAT	-	○	-	○	-	○
BOOL	•	•	•	•	•	•
BOUND	•	•	•	•	•	•
BRISK	•	•	•	•	•	•

Anweisung	828D-Steuerungsvariante					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Drehen	Fräsen	Drehen	Fräsen
BRISKA	•	•	•	•	•	•
BSPLINE	-	○	-	○	-	○
BTAN	-	○	-	○	-	○
C	•	•	•	•	•	•
C2	-	-	-	-	-	-
C3	-	-	-	-	-	-
C4	-	-	-	-	-	-
C5	-	-	-	-	-	-
CAC	•	•	•	•	•	•
CACN	•	•	•	•	•	•
CACP	•	•	•	•	•	•
CALCDAT	•	•	•	•	•	•
CALCPOSI	•	•	•	•	•	•
CALL	•	•	•	•	•	•
CALLPATH	•	•	•	•	•	•
CANCEL	•	•	•	•	•	•
CASE	•	•	•	•	•	•
CDC	•	•	•	•	•	•
CDOF	•	•	•	•	•	•
CDOF2	•	•	•	•	•	•
CDON	•	•	•	•	•	•
CFC	•	•	•	•	•	•
CFIN	•	•	•	•	•	•
CFINE	•	•	•	•	•	•
CFTCP	•	•	•	•	•	•
CHAN	•	•	•	•	•	•
CHANDATA	•	•	•	•	•	•
CHAR	•	•	•	•	•	•
CHECKSUM	•	•	•	•	•	•
CHF	•	•	•	•	•	•
CHKDM	•	•	•	•	•	•
CHKDNO	•	•	•	•	•	•
CHR	•	•	•	•	•	•
CIC	•	•	•	•	•	•
CIP	•	•	•	•	•	•
CLEARM	-	-	-	-	-	-
CLRINT	•	•	•	•	•	•
CMIRROR	•	•	•	•	•	•
COARSEA	•	•	•	•	•	•
COMPCAD	-	○	-	○	-	○

Anweisung	828D-Steuerungsvariante					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Drehen	Fräsen	Drehen	Fräsen
COMPCURV	-	○	-	○	-	○
COMPLETE	●	●	●	●	●	●
COMPOF	-	○	-	○	-	○
COMPON	-	○	-	○	-	○
CONTDCON	●	●	●	●	●	●
CONTPRON	●	●	●	●	●	●
CORROF	●	●	●	●	●	●
COS	●	●	●	●	●	●
COUPDEF	○	-	○	-	○	-
COUPDEL	○	-	○	-	○	-
COUPOF	○	-	○	-	○	-
COUPOFS	○	-	○	-	○	-
COUPON	○	-	○	-	○	-
COUPONC	○	-	○	-	○	-
COUPRES	○	-	○	-	○	-
CP	●	●	●	●	●	●
CPRECOF	●	●	●	●	●	●
CPRECON	●	●	●	●	●	●
CPROT	●	●	●	●	●	●
CPROTDEF	●	●	●	●	●	●
CR	●	●	●	●	●	●
CROT	●	●	●	●	●	●
CROTS	●	●	●	●	●	●
CRPL	●	●	●	●	●	●
CSCALE	●	●	●	●	●	●
CSPLINE	-	○	-	○	-	○
CT	●	●	●	●	●	●
CTAB	-	-	-	-	-	-
CTABDEF	-	-	-	-	-	-
CTABDEL	-	-	-	-	-	-
CTABEND	-	-	-	-	-	-
CTABEXISTS	-	-	-	-	-	-
CTABFNO	-	-	-	-	-	-
CTABFPOL	-	-	-	-	-	-
CTABFSEG	-	-	-	-	-	-
CTABID	-	-	-	-	-	-
CTABINV	-	-	-	-	-	-
CTABISLOCK	-	-	-	-	-	-
CTABLOCK	-	-	-	-	-	-
CTABMEMTYP	-	-	-	-	-	-

Anweisung	828D-Steuerungsvariante					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Drehen	Fräsen	Drehen	Fräsen
CTABMPOL	-	-	-	-	-	-
CTABMSEG	-	-	-	-	-	-
CTABNO	-	-	-	-	-	-
CTABNOMEM	-	-	-	-	-	-
CTABPERIOD	-	-	-	-	-	-
CTABPOL	-	-	-	-	-	-
CTABPOLID	-	-	-	-	-	-
CTABSEG	-	-	-	-	-	-
CTABSEGID	-	-	-	-	-	-
CTABSEV	-	-	-	-	-	-
CTABSSV	-	-	-	-	-	-
CTABTEP	-	-	-	-	-	-
CTABTEV	-	-	-	-	-	-
CTABTMAX	-	-	-	-	-	-
CTABTMIN	-	-	-	-	-	-
CTABTSP	-	-	-	-	-	-
CTABTSV	-	-	-	-	-	-
CTABUNLOCK	-	-	-	-	-	-
CTOL	-	o	-	o	-	o
CTRANS	•	•	•	•	•	•
CUT2D	•	•	•	•	•	•
CUT2DF	•	•	•	•	•	•
CUT3DC	-	-	-	-	-	-
CUT3DCC	-	-	-	-	-	-
CUT3DCCD	-	-	-	-	-	-
CUT3DF	-	-	-	-	-	-
CUT3DFF	-	-	-	-	-	-
CUT3DFS	-	-	-	-	-	-
CUTCONOF	•	•	•	•	•	•
CUTCONON	•	•	•	•	•	•
CUTMOD	•	•	•	•	•	•
CYCLE...	•	•	•	•	•	•
D	•	•	•	•	•	•
D0	•	•	•	•	•	•
DAC	•	•	•	•	•	•
DC	•	•	•	•	•	•
DEF	•	•	•	•	•	•
DEFINE	•	•	•	•	•	•
DEFAULT	•	•	•	•	•	•
DELAYFSTON	•	•	•	•	•	•

Anweisung	828D-Steuerungsvariante					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Drehen	Fräsen	Drehen	Fräsen
DELAYSTOF	•	•	•	•	•	•
DELDL	•	•	•	•	•	•
DELDTG	•	•	•	•	•	•
DELETE	•	•	•	•	•	•
DELTOOLENV	•	•	•	•	•	•
DIACYCOFA	•	•	•	•	•	•
DIAM90	•	•	•	•	•	•
DIAM90A	•	•	•	•	•	•
DIAMCHAN	•	•	•	•	•	•
DIAMCHANA	•	•	•	•	•	•
DIAMCYCOF	•	•	•	•	•	•
DIAMOF	•	•	•	•	•	•
DIAMOFA	•	•	•	•	•	•
DIAMON	•	•	•	•	•	•
DIAMONA	•	•	•	•	•	•
DIC	•	•	•	•	•	•
DILF	•	•	•	•	•	•
DISABLE	•	•	•	•	•	•
DISC	•	•	•	•	•	•
DISCL	•	•	•	•	•	•
DISPLOF	•	•	•	•	•	•
DISPLON	•	•	•	•	•	•
DISPR	•	•	•	•	•	•
DISR	•	•	•	•	•	•
DISRP	•	•	•	•	•	•
DITE	•	•	•	•	•	•
DITS	•	•	•	•	•	•
DIV	•	•	•	•	•	•
DL	-	-	-	-	-	-
DO	•	•	•	•	•	•
DRFOF	•	•	•	•	•	•
DRIVE	•	•	•	•	•	•
DRIVEA	•	•	•	•	•	•
DYNFINISH	•	•	•	•	•	•
DYNNORM	•	•	•	•	•	•
DYNPOS	•	•	•	•	•	•
DYNROUGH	•	•	•	•	•	•
DYNSEMIFIN	•	•	•	•	•	•
DZERO	•	•	•	•	•	•
EAUTO	-	○	-	○	-	○

Anweisung	828D-Steuerungsvariante					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Drehen	Fräsen	Drehen	Fräsen
EGDEF	-	-	-	-	-	-
EGDEL	-	-	-	-	-	-
EGOFC	-	-	-	-	-	-
EGOFS	-	-	-	-	-	-
EGON	-	-	-	-	-	-
EGONSYN	-	-	-	-	-	-
EGONSYNE	-	-	-	-	-	-
ELSE	•	•	•	•	•	•
ENABLE	•	•	•	•	•	•
ENAT	-	○	-	○	-	○
ENDFOR	•	•	•	•	•	•
ENDIF	•	•	•	•	•	•
ENDLABEL	•	•	•	•	•	•
ENDLOOP	•	•	•	•	•	•
ENDPROC	•	•	•	•	•	•
ENDWHILE	•	•	•	•	•	•
ESRR	•	•	•	•	•	•
ESRS	•	•	•	•	•	•
ETAN	-	○	-	○	-	○
EVERY	•	•	•	•	•	•
EX	•	•	•	•	•	•
EXECSTRING	•	•	•	•	•	•
EXECTAB	•	•	•	•	•	•
EXECUTE	•	•	•	•	•	•
EXP	•	•	•	•	•	•
EXTCALL	•	•	•	•	•	•
EXTCLOSE	•	•	•	•	•	•
EXTERN	•	•	•	•	•	•
EXTOPEN	•	•	•	•	•	•
F	•	•	•	•	•	•
FA	•	•	•	•	•	•
FAD	•	•	•	•	•	•
FALSE	•	•	•	•	•	•
FB	•	•	•	•	•	•
FCTDEF	-	-	-	-	-	-
FCUB	•	•	•	•	•	•
FD	•	•	•	•	•	•
FDA	•	•	•	•	•	•
FENDNORM	•	•	•	•	•	•
FFWOF	•	•	•	•	•	•

Anweisung	828D-Steuerungsvariante					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Drehen	Fräsen	Drehen	Fräsen
FFWON	•	•	•	•	•	•
FGREF	•	•	•	•	•	•
FGROUP	•	•	•	•	•	•
FI	•	•	•	•	•	•
FIFOCTRL	•	•	•	•	•	•
FILEDATE	•	•	•	•	•	•
FILEINFO	•	•	•	•	•	•
FILESIZE	•	•	•	•	•	•
FILESTAT	•	•	•	•	•	•
FILETIME	•	•	•	•	•	•
FINEA	•	•	•	•	•	•
FL	•	•	•	•	•	•
FLIN	•	•	•	•	•	•
FMA	-	-	-	-	-	-
FNORM	•	•	•	•	•	•
FOCOF	○	-	○	-	○	-
FOCON	○	-	○	-	○	-
FOR	•	•	•	•	•	•
FP	•	•	•	•	•	•
FPO	-	-	-	-	-	-
FPR	•	•	•	•	•	•
FPRAOF	•	•	•	•	•	•
FPRAON	•	•	•	•	•	•
FRAME	•	•	•	•	•	•
FRC	•	•	•	•	•	•
FRCM	•	•	•	•	•	•
FROM	•	•	•	•	•	•
FTOC	•	•	•	•	•	•
FTOCOF	•	•	•	•	•	•
FTOCON	•	•	•	•	•	•
FXS	•	•	•	•	•	•
FXST	•	•	•	•	•	•
FXSW	•	•	•	•	•	•
FZ	•	•	•	•	•	•
G0	•	•	•	•	•	•
G1	•	•	•	•	•	•
G2	•	•	•	•	•	•
G3	•	•	•	•	•	•
G4	•	•	•	•	•	•
G5	•	•	•	•	•	•

Anweisung	828D-Steuerungsvariante					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Drehen	Fräsen	Drehen	Fräsen
G7	•	•	•	•	•	•
G9	•	•	•	•	•	•
G17	•	•	•	•	•	•
G18	•	•	•	•	•	•
G19	•	•	•	•	•	•
G25	•	•	•	•	•	•
G26	•	•	•	•	•	•
G33	•	•	•	•	•	•
G34	•	•	•	•	•	•
G35	•	•	•	•	•	•
G40	•	•	•	•	•	•
G41	•	•	•	•	•	•
G42	•	•	•	•	•	•
G53	•	•	•	•	•	•
G54	•	•	•	•	•	•
G55	•	•	•	•	•	•
G56	•	•	•	•	•	•
G57	•	•	•	•	•	•
G58	•	•	•	•	•	•
G59	•	•	•	•	•	•
G60	•	•	•	•	•	•
G62	•	•	•	•	•	•
G63	•	•	•	•	•	•
G64	•	•	•	•	•	•
G70	•	•	•	•	•	•
G71	•	•	•	•	•	•
G74	•	•	•	•	•	•
G75	•	•	•	•	•	•
G90	•	•	•	•	•	•
G91	•	•	•	•	•	•
G93	•	•	•	•	•	•
G94	•	•	•	•	•	•
G95	•	•	•	•	•	•
G96	•	•	•	•	•	•
G97	•	•	•	•	•	•
G110	•	•	•	•	•	•
G111	•	•	•	•	•	•
G112	•	•	•	•	•	•
G140	•	•	•	•	•	•
G141	•	•	•	•	•	•

Anweisung	828D-Steuerungsvariante					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Drehen	Fräsen	Drehen	Fräsen
G142	•	•	•	•	•	•
G143	•	•	•	•	•	•
G147	•	•	•	•	•	•
G148	•	•	•	•	•	•
G153	•	•	•	•	•	•
G247	•	•	•	•	•	•
G248	•	•	•	•	•	•
G290	•	•	•	•	•	•
G291	•	•	•	•	•	•
G331	•	•	•	•	•	•
G332	•	•	•	•	•	•
G340	•	•	•	•	•	•
G341	•	•	•	•	•	•
G347	•	•	•	•	•	•
G348	•	•	•	•	•	•
G450	•	•	•	•	•	•
G451	•	•	•	•	•	•
G460	•	•	•	•	•	•
G461	•	•	•	•	•	•
G462	•	•	•	•	•	•
G500	•	•	•	•	•	•
G505 ... G599	•	•	•	•	•	•
G601	•	•	•	•	•	•
G602	•	•	•	•	•	•
G603	•	•	•	•	•	•
G621	•	•	•	•	•	•
G641	•	•	•	•	•	•
G642	•	•	•	•	•	•
G643	•	•	•	•	•	•
G644	•	•	•	•	•	•
G645	•	•	•	•	•	•
G700	•	•	•	•	•	•
G710	•	•	•	•	•	•
G751	•	•	•	•	•	•
G810 ... G819	-	-	-	-	-	-
G820 ... G829	-	-	-	-	-	-
G931	•	•	•	•	•	•
G942	•	•	•	•	•	•
G952	•	•	•	•	•	•
G961	•	•	•	•	•	•

Anweisung	828D-Steuerungsvariante					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Drehen	Fräsen	Drehen	Fräsen
G962	•	•	•	•	•	•
G971	•	•	•	•	•	•
G972	•	•	•	•	•	•
G973	•	•	•	•	•	•
GEOAX	•	•	•	•	•	•
GET	•	•	•	•	•	•
GETACTT	•	•	•	•	•	•
GETACTTD	•	•	•	•	•	•
GETD	•	•	•	•	•	•
GETDNO	•	•	•	•	•	•
GETEXET	•	•	•	•	•	•
GETFREELOC	•	•	•	•	•	•
GETSELT	•	•	•	•	•	•
GETT	•	•	•	•	•	•
GETTCOR	•	•	•	•	•	•
GETTENV	•	•	•	•	•	•
GOTO	•	•	•	•	•	•
GOTOB	•	•	•	•	•	•
GOTOC	•	•	•	•	•	•
GOTOF	•	•	•	•	•	•
GOTOS	•	•	•	•	•	•
GP	•	•	•	•	•	•
GWPSOF	•	•	•	•	•	•
GWPSON	•	•	•	•	•	•
H...	•	•	•	•	•	•
HOLES1	•	•	•	•	•	•
HOLES2	•	•	•	•	•	•
I	•	•	•	•	•	•
I1	•	•	•	•	•	•
IC	•	•	•	•	•	•
ICYCOF	•	•	•	•	•	•
ICYCON	•	•	•	•	•	•
ID	•	•	•	•	•	•
IDS	•	•	•	•	•	•
IF	•	•	•	•	•	•
INDEX	•	•	•	•	•	•
INIPO	•	•	•	•	•	•
INIRE	•	•	•	•	•	•
INICF	•	•	•	•	•	•
INIT	-	-	-	-	-	-

Anweisung	828D-Steuerungsvariante					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Drehen	Fräsen	Drehen	Fräsen
INITIAL	•	•	•	•	•	•
INT	•	•	•	•	•	•
INTERSEC	•	•	•	•	•	•
INVCCW	-	-	-	-	-	-
INVCW	-	-	-	-	-	-
INVFRAME	•	•	•	•	•	•
IP	•	•	•	•	•	•
IPOBRKA	•	•	•	•	•	•
IPOENDA	•	•	•	•	•	•
IPTRLOCK	•	•	•	•	•	•
IPTRUNLOCK	•	•	•	•	•	•
ISAXIS	•	•	•	•	•	•
ISD	-	-	-	-	-	-
ISFILE	•	•	•	•	•	•
ISNUMBER	•	•	•	•	•	•
ISOCALL	•	•	•	•	•	•
ISVAR	•	•	•	•	•	•
J	•	•	•	•	•	•
J1	•	•	•	•	•	•
JERKA	•	•	•	•	•	•
JERKLIM	•	•	•	•	•	•
JERKLIMA	•	•	•	•	•	•
K	•	•	•	•	•	•
K1	•	•	•	•	•	•
KONT	•	•	•	•	•	•
KONTC	•	•	•	•	•	•
KONTT	•	•	•	•	•	•
L	•	•	•	•	•	•
LEAD						
Werkzeugorientierung	-	-	-	-	-	-
Orientierungspolyn.	-	-	-	-	-	-
LEADOF	-	-	-	-	-	-
LEADON	-	-	-	-	-	-
LENTOAX	•	•	•	•	•	•
LFOF	•	•	•	•	•	•
LFON	•	•	•	•	•	•
LFPOS	•	•	•	•	•	•
LFTXT	•	•	•	•	•	•
LFWP	•	•	•	•	•	•
LIFTFAST	•	•	•	•	•	•

Anweisung	828D-Steuerungsvariante					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Drehen	Fräsen	Drehen	Fräsen
LIMS	•	•	•	•	•	•
LLI	•	•	•	•	•	•
LN	•	•	•	•	•	•
LOCK	•	•	•	•	•	•
LONGHOLE	-	-	-	-	-	-
LOOP	•	•	•	•	•	•
M0	•	•	•	•	•	•
M1	•	•	•	•	•	•
M2	•	•	•	•	•	•
M3	•	•	•	•	•	•
M4	•	•	•	•	•	•
M5	•	•	•	•	•	•
M6	•	•	•	•	•	•
M17	•	•	•	•	•	•
M19	•	•	•	•	•	•
M30	•	•	•	•	•	•
M40	•	•	•	•	•	•
M41 ... M45	•	•	•	•	•	•
M70	•	•	•	•	•	•
MASLDEF	•	•	•	•	•	•
MASLDEL	•	•	•	•	•	•
MASLOF	•	•	•	•	•	•
MASLOFS	•	•	•	•	•	•
MASLON	•	•	•	•	•	•
MATCH	•	•	•	•	•	•
MAXVAL	•	•	•	•	•	•
MCALL	•	•	•	•	•	•
MEAC	-	-	-	-	-	-
MEAFRAME	•	•	•	•	•	•
MEAS	•	•	•	•	•	•
MEASA	-	-	-	-	-	-
MEASURE	•	•	•	•	•	•
MEAW	•	•	•	•	•	•
MEAWA	-	-	-	-	-	-
MI	•	•	•	•	•	•
MINDEX	•	•	•	•	•	•
MINVAL	•	•	•	•	•	•
MIRROR	•	•	•	•	•	•
MMC	•	•	•	•	•	•
MOD	•	•	•	•	•	•

Anweisung	828D-Steuerungsvariante					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Drehen	Fräsen	Drehen	Fräsen
MODAXVAL	•	•	•	•	•	•
MOV	•	•	•	•	•	•
MSG	•	•	•	•	•	•
MVTOOL	•	•	•	•	•	•
N	•	•	•	•	•	•
NCK	•	•	•	•	•	•
NEWCONF	•	•	•	•	•	•
NEWT	•	•	•	•	•	•
NORM	•	•	•	•	•	•
NOT	•	•	•	•	•	•
NPROT	•	•	•	•	•	•
NPROTDEF	•	•	•	•	•	•
NUMBER	•	•	•	•	•	•
OEMIPO1	-	-	-	-	-	-
OEMIPO2	-	-	-	-	-	-
OF	•	•	•	•	•	•
OFFN	•	•	•	•	•	•
OMA1	-	-	-	-	-	-
OMA2	-	-	-	-	-	-
OMA3	-	-	-	-	-	-
OMA4	-	-	-	-	-	-
OMA5	-	-	-	-	-	-
OR	•	•	•	•	•	•
ORIXES	-	-	-	-	-	-
ORIXPOS	-	-	-	-	-	-
ORIC	-	-	-	-	-	-
ORICONCCW	-	-	-	-	-	-
ORICONCW	-	-	-	-	-	-
ORICONIO	-	-	-	-	-	-
ORICONTO	-	-	-	-	-	-
ORICURVE	-	-	-	-	-	-
ORID	-	-	-	-	-	-
ORIEULER	-	-	-	-	-	-
ORIMKS	-	-	-	-	-	-
ORIPATH	-	-	-	-	-	-
ORIPATHS	-	-	-	-	-	-
ORIPANE	-	-	-	-	-	-
ORIRESET	-	-	-	-	-	-
ORIROTA	-	-	-	-	-	-
ORIROTC	-	-	-	-	-	-

Anweisung	828D-Steuerungsvariante					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Drehen	Fräsen	Drehen	Fräsen
ORIROTR	-	-	-	-	-	-
ORIROTT	-	-	-	-	-	-
ORIRPY	-	-	-	-	-	-
ORIRPY2	-	-	-	-	-	-
ORIS	-	-	-	-	-	-
ORISOF	-	-	-	-	-	-
ORISON	-	-	-	-	-	-
ORIVECT	-	-	-	-	-	-
ORIVIRT1	-	-	-	-	-	-
ORIVIRT2	-	-	-	-	-	-
ORIWKS	-	-	-	-	-	-
OS	-	-	-	-	-	-
OSB	-	-	-	-	-	-
OSC	-	-	-	-	-	-
OSCILL	-	-	-	-	-	-
OSCTRL	-	-	-	-	-	-
OSD	-	-	-	-	-	-
OSE	-	-	-	-	-	-
OSNSC	-	-	-	-	-	-
OSOF	-	-	-	-	-	-
OSP1	-	-	-	-	-	-
OSP2	-	-	-	-	-	-
OSS	-	-	-	-	-	-
OSSE	-	-	-	-	-	-
OST	-	-	-	-	-	-
OST1	-	-	-	-	-	-
OST2	-	-	-	-	-	-
OTOL	-	•	-	•	-	•
OVR	•	•	•	•	•	•
OVRA	•	•	•	•	•	•
OVRRAP	•	•	•	•	•	•
P	•	•	•	•	•	•
PAROT	•	•	•	•	•	•
PAROTOF	•	•	•	•	•	•
PCALL	•	•	•	•	•	•
PDELAYOF	-	-	-	-	-	-
PDELAYON	-	-	-	-	-	-
PHI	-	-	-	-	-	-
PHU	•	•	•	•	•	•

Anweisung	828D-Steuerungsvariante					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Drehen	Fräsen	Drehen	Fräsen
PL	-	○	-	○	-	○
	-	-	-	-	-	-
PM	●	●	●	●	●	●
PO	-	-	-	-	-	-
POCKET3	●	●	●	●	●	●
POCKET4	●	●	●	●	●	●
POLF	●	●	●	●	●	●
POLFA	●	●	●	●	●	●
POLFMASK	●	●	●	●	●	●
POLFMLIN	●	●	●	●	●	●
POLY	-	-	-	-	-	-
POLYPATH	-	-	-	-	-	-
PON	-	-	-	-	-	-
PONS	-	-	-	-	-	-
POS	●	●	●	●	●	●
POSA	●	●	●	●	●	●
POSM	●	●	●	●	●	●
POSP	●	●	●	●	●	●
POSRANGE	●	●	●	●	●	●
POT	●	●	●	●	●	●
PR	●	●	●	●	●	●
PREPRO	●	●	●	●	●	●
PRESETON	●	●	●	●	●	●
PRIO	●	●	●	●	●	●
PROC	●	●	●	●	●	●
PSI	-	-	-	-	-	-
PTP	●	●	●	●	●	●
PTPG0	●	●	●	●	●	●
PUNCHACC	-	-	-	-	-	-
PUTFTOC	●	●	●	●	●	●
PUTFTOCF	●	●	●	●	●	●
PW	-	○	-	○	-	○
QU	●	●	●	●	●	●
R...	●	●	●	●	●	●
RAC	●	●	●	●	●	●
RDISABLE	●	●	●	●	●	●
READ	●	●	●	●	●	●
REAL	●	●	●	●	●	●
REDEF	●	●	●	●	●	●

Anweisung	828D-Steuerungsvariante					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Drehen	Fräsen	Drehen	Fräsen
RELEASE	•	•	•	•	•	•
REP	•	•	•	•	•	•
REPEAT	•	•	•	•	•	•
REPEATB	•	•	•	•	•	•
REPOSA	•	•	•	•	•	•
REPOSH	•	•	•	•	•	•
REPOSHA	•	•	•	•	•	•
REPOSL	•	•	•	•	•	•
REPOSQ	•	•	•	•	•	•
REPOSQA	•	•	•	•	•	•
RESET	•	•	•	•	•	•
RESETMON	•	•	•	•	•	•
RET	•	•	•	•	•	•
RIC	•	•	•	•	•	•
RINDEX	•	•	•	•	•	•
RMB	•	•	•	•	•	•
RME	•	•	•	•	•	•
RMI	•	•	•	•	•	•
RMN	•	•	•	•	•	•
RND	•	•	•	•	•	•
RNDM	•	•	•	•	•	•
ROT	•	•	•	•	•	•
ROTS	•	•	•	•	•	•
ROUND	•	•	•	•	•	•
ROUNDUP	•	•	•	•	•	•
RP	•	•	•	•	•	•
RPL	•	•	•	•	•	•
RT	•	•	•	•	•	•
RTLIOF	•	•	•	•	•	•
RTLION	•	•	•	•	•	•
S	•	•	•	•	•	•
SAVE	•	•	•	•	•	•
SBLOF	•	•	•	•	•	•
SBLON	•	•	•	•	•	•
SC	•	•	•	•	•	•
SCALE	•	•	•	•	•	•
SCC	•	•	•	•	•	•
SCPARA	•	•	•	•	•	•
SD	-	○	-	○	-	○
SET	•	•	•	•	•	•

Anweisung	828D-Steuerungsvariante					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Drehen	Fräsen	Drehen	Fräsen
SETAL	•	•	•	•	•	•
SETDNO	•	•	•	•	•	•
SETINT	•	•	•	•	•	•
SETM	-	-	-	-	-	-
SETMS	•	•	•	•	•	•
SETMS(n)	•	•	•	•	•	•
SETMTH	•	•	•	•	•	•
SETPIECE	•	•	•	•	•	•
SETTA	•	•	•	•	•	•
SETTCOR	•	•	•	•	•	•
SETTIA	•	•	•	•	•	•
SF	•	•	•	•	•	•
SIN	•	•	•	•	•	•
SIRELAY	-	-	-	-	-	-
SIRELIN	-	-	-	-	-	-
SIRELOUT	-	-	-	-	-	-
SIRELTIME	-	-	-	-	-	-
SLOT1	•	•	•	•	•	•
SLOT2	•	•	•	•	•	•
SOFT	•	•	•	•	•	•
SOFTA	•	•	•	•	•	•
SON	-	-	-	-	-	-
SONS	-	-	-	-	-	-
SPATH	•	•	•	•	•	•
SPCOF	•	•	•	•	•	•
SPCON	•	•	•	•	•	•
SPI	•	•	•	•	•	•
SPIF1	-	-	-	-	-	-
SPIF2	-	-	-	-	-	-
SPLINEPATH	-	○	-	○	-	○
SPN	-	-	-	-	-	-
SPOF	-	-	-	-	-	-
SPOS	•	•	•	•	•	•
SPOSA	•	•	•	•	•	•
SPP	-	-	-	-	-	-
SPRINT	•	•	•	•	•	•
SQRT	•	•	•	•	•	•
SR	-	-	-	-	-	-
SRA	-	-	-	-	-	-
ST	-	-	-	-	-	-

Anweisung	828D-Steuerungsvariante					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Drehen	Fräsen	Drehen	Fräsen
STA	-	-	-	-	-	-
START	-	-	-	-	-	-
STARTFIFO	•	•	•	•	•	•
STAT	•	•	•	•	•	•
STOLF	-	-	-	-	-	-
STOPFIFO	•	•	•	•	•	•
STOPRE	•	•	•	•	•	•
STOPREOF	•	•	•	•	•	•
STRING	•	•	•	•	•	•
STRINGFELD	•	•	•	•	•	•
STRINGIS	•	•	•	•	•	•
STRINGVAR	-	-	-	-	-	-
STRLEN	•	•	•	•	•	•
SUBSTR	•	•	•	•	•	•
SUPA	•	•	•	•	•	•
SVC	•	•	•	•	•	•
SYNFCT	•	•	•	•	•	•
SYNR	•	•	•	•	•	•
SYNRW	•	•	•	•	•	•
SYNW	•	•	•	•	•	•
T	•	•	•	•	•	•
TAN	•	•	•	•	•	•
TANG	-	-	-	-	-	-
TANGDEL	-	-	-	-	-	-
TANGOF	-	-	-	-	-	-
TANGON	-	-	-	-	-	-
TCA (828D: _TCA)	•	•	•	•	•	•
TCARR	-	•	-	•	-	•
TCI	•	•	•	•	•	•
TCOABS	-	•	-	•	-	•
TCOFR	-	•	-	•	-	•
TCOFRX	-	•	-	•	-	•
TCOFRY	-	•	-	•	-	•
TCOFRZ	-	•	-	•	-	•
THETA	-	-	-	-	-	-
TILT	-	-	-	-	-	-
TLIFT	-	-	-	-	-	-
TMOF	•	•	•	•	•	•
TMON	•	•	•	•	•	•

Anweisung	828D-Steuerungsvariante					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Drehen	Fräsen	Drehen	Fräsen
TO	•	•	•	•	•	•
TOFF	•	•	•	•	•	•
TOFFL	•	•	•	•	•	•
TOFFOF	•	•	•	•	•	•
TOFFON	•	•	•	•	•	•
TOFFR	•	•	•	•	•	•
TOFRAME	•	•	•	•	•	•
TOFRAMEX	•	•	•	•	•	•
TOFRAMEY	•	•	•	•	•	•
TOFRAMEZ	•	•	•	•	•	•
TOLOWER	•	•	•	•	•	•
TOOLENV	•	•	•	•	•	•
TOROT	•	•	•	•	•	•
TOROTOF	•	•	•	•	•	•
TOROTX	•	•	•	•	•	•
TOROTY	•	•	•	•	•	•
TOROTZ	•	•	•	•	•	•
TOUPPER	•	•	•	•	•	•
TOWBCS	-	•	-	•	-	•
TOWKCS	-	•	-	•	-	•
TOWMCS	-	•	-	•	-	•
TOWSTD	-	•	-	•	-	•
TOWTCS	-	•	-	•	-	•
TOWWCS	-	•	-	•	-	•
TR	•	•	•	•	•	•
TRAANG	-	-	-	-	○	-
TRACON	-	-	-	-	○	-
TRACYL	○	○	○	○	○	○
TRAFOOF	•	•	•	•	•	•
TRAILOF	•	•	•	•	•	•
TRAILON	•	•	•	•	•	•
TRANS	•	•	•	•	•	•
TRANSMIT	○	○	○	○	○	○
TRAORI	-	•	-	•	-	•
TRUE	•	•	•	•	•	•
TRUNC	•	•	•	•	•	•
TU	•	•	•	•	•	•
TURN	•	•	•	•	•	•
ULI	•	•	•	•	•	•
UNLOCK	•	•	•	•	•	•

Anweisung	828D-Steuerungsvariante					
	PPU240.2 / 241.2		PPU260.2 / 261.2		PPU280.2 / 281.2	
	basic T	basic M	Drehen	Fräsen	Drehen	Fräsen
UNTIL	•	•	•	•	•	•
UPATH	•	•	•	•	•	•
VAR	•	•	•	•	•	•
VELOLIM	•	•	•	•	•	•
VELOLIMA	•	•	•	•	•	•
WAITC	-	-	-	-	○	-
WAITE	-	-	-	-	-	-
WAITENC	-	-	-	-	-	-
WAITM	-	-	-	-	-	-
WAITMC	-	-	-	-	-	-
WAITP	•	•	•	•	•	•
WAITS	•	•	•	•	•	•
WALCS0	•	•	•	•	•	•
WALCS1	•	•	•	•	•	•
WALCS2	•	•	•	•	•	•
WALCS3	•	•	•	•	•	•
WALCS4	•	•	•	•	•	•
WALCS5	•	•	•	•	•	•
WALCS6	•	•	•	•	•	•
WALCS7	•	•	•	•	•	•
WALCS8	•	•	•	•	•	•
WALCS9	•	•	•	•	•	•
WALCS10	•	•	•	•	•	•
WALIMOF	•	•	•	•	•	•
WALIMON	•	•	•	•	•	•
WHEN	•	•	•	•	•	•
WHENEVER	•	•	•	•	•	•
WHILE	•	•	•	•	•	•
WRITE	•	•	•	•	•	•
WRTPR	•	•	•	•	•	•
X	•	•	•	•	•	•
XOR	•	•	•	•	•	•
Y	•	•	•	•	•	•
Z	•	•	•	•	•	•

- Standard
- Option
- Nicht verfügbar

17.3 Aktuelle Sprache im HMI

Die folgende Tabelle enthält alle auf der Bedienoberfläche verfügbaren Sprachen.

Die aktuell eingestellte Sprache ist im Teileprogramm und in Synchronaktionen über folgende Systemvariable abfragbar:

\$AN_LANGUAGE_ON_HMI = <Wert>

<Wert>	Sprache	Sprachkürzel
1	Deutsch (Deutschland)	DEU
2	Französisch	FRA
3	Englisch (Vereinigtes Königreich)	ENG
4	Spanisch	ESP
6	Italienisch	ITA
7	Niederländisch	NLD
8	Chinesisch (vereinfacht)	CHS
9	Schwedisch	SVE
18	Ungarisch	HUN
19	Finnisch	FIN
28	Tschechisch	CSY
50	Portugiesisch (Brasilien)	PTB
53	Polnisch	PLK
55	Dänisch	DAN
57	Russisch	RUS
68	Slowakisch	SKY
72	Rumänisch	ROM
80	Chinesisch (traditionell)	CHT
85	Koreanisch	KOR
87	Japanisch	JPN
89	Türkisch	TRK

Hinweis

Eine Aktualisierung von \$AN_LANGUAGE_ON_HMI erfolgt:

- nach Systemhochlauf.
- nach NCK- und/oder PLC-Reset.
- nach dem Umschalten auf einen anderen NCK im Rahmen von M2N.
- nach Sprachumschaltung auf HMI.

Anhang

A.1 Liste der Abkürzungen

A	
A	Ausgang
ADI4	Analog Drive Interface for 4 Axes
AC	Adaptive Control
ALM	Active Line Module
ARM	Asynchroner rotatorischer Motor
AS	Automatisierungssystem
ASCII	American Standard Code for Information Interchange: Amerikanische Code-Norm für den Informationsaustausch
ASIC	Application Specific Integrated Circuit: Anwender-Schaltkreis
ASUP	Asynchrones Unterprogramm
AUXFU	Auxiliary Function: Hilfsfunktion
AWL	Anweisungsliste
AWP	Anwenderprogramm

B	
BA	Betriebsart
BAG	Betriebsartengruppe
BCD	Binary Coded Decimals: Im Binärcode verschlüsselte Dezimalzahlen
BERO	Berührungsloser Endschalter mit rückgekoppeltem Oszillator
BI	Binector Input
BICO	Binector Connector
BIN	Binary Files: Binärdateien
BIOS	Basic Input Output System
BKS	Basiskoordinatensystem
BO	Binector Output
BTSS	Bedientafelschnittstelle

C	
CAD	Computer-Aided Design
CAM	Computer-Aided Manufacturing
CC	Compile Cycle: Compile-Zyklen
CI	Connector Input

CF-Card	Compact Flash-Card
CNC	Computerized Numerical Control: Computerunterstützte numerische Steuerung
CO	Connector Output
CoL	Certificate of License
COM	Communication
CPA	Compiler Projecting Data: Projektierdaten des Compilers
CRT	Cathode Ray Tube: Bildröhre
CSB	Central Service Board: PLC-Baugruppe
CU	Control Unit
CP	Communication Processor
CPU	Central Processing Unit: Zentrale Rechereinheit
CR	Carriage Return
CTS	Clear To Send: Meldung der Sendebereitschaft bei seriellen Daten-Schnittstellen
CUTCOM	Cutter Radius Compensation: Werkzeugradiuskorrektur

D	
DAU	Digital-Analog-Umwandler
DB	Datenbaustein (PLC)
DBB	Datenbaustein-Byte (PLC)
DBD	Datenbaustein-Doppelwort (PLC)
DBW	Datenbaustein-Wort (PLC)
DBX	Datenbaustein-Bit (PLC)
DDE	Dynamic Data Exchange
DIN	Deutsche Industrie Norm
DIO	Data Input/Output: Datenübertragungs-Anzeige
DIR	Directory: Verzeichnis
DLL	Dynamic Link Library
DO	Drive Object
DPM	Dual Port Memory
DPR	Dual Port RAM
DRAM	Dynamischer Speicher (ungepuffert)
DRF	Differential Resolver Function: Differential-Drehmelder-Funktion (Handrad)
DRIVE-CLiQ	Drive Component Link with IQ
DRY	Dry Run: Probelaufvorschub
DSB	Decoding Single Block: Dekodierungseinzelsatz
DSC	Dynamic Servo Control / Dynamic Stiffness Control
DW	Datenwort
DWORD	Doppelwort (aktuell 32 Bit)

E	
E	Eingang
E/A	Ein-/Ausgabe
ENC	Encoder: Istwertgeber
EFP	Einfach Peripheriemodul (PLC–E/A–Baugruppe)
EGB	Elektronisch gefährdete Baugruppen/Bauelemente
EMV	Elektromagnetische Verträglichkeit
EN	Europäische Norm
EnDat	Geberschnittstelle
EPROM	Erasable Programmable Read Only Memory: Löscharer, elektrisch programmierbarer nur Lesespeicher
ePS Network Services	Dienste zur internetgestützten Maschinen-Fernwartung
EQN	Typbezeichnung eines Absolutwertgebers mit 2048 Sinussignalen/Umdrehung
ES	Engineering System
ESR	Erweitertes Stillsetzen und Rückziehen
ETC	ETC–Taste ">"; Erweiterung der Softkeyleiste im gleichen Menü

F	
FB	Funktionsbaustein (PLC)
FC	Function Call: Funktionsbaustein (PLC)
FEPROM	Flash–EPROM: Les– und schreibbarer Speicher
FIFO	First In First Out: Speicher, der ohne Adressangabe arbeitet und dessen Daten in derselben Reihenfolge gelesen werden, in der sie gespeichert wurden
FIPO	Feininterpolator
FPU	Floating Point Unit: Gleitpunkteinheit
FRK	Fräsradiuskorrektur
FST	Feed Stop: Vorschub Halt
FUP	Funktionsplan (Programmiermethode für PLC)
FW	Firmware

G	
GC	Global Control (PROFIBUS: Broadcast-Telegramm)
GEO	Geometrie, z.B. Geometrieachse
GIA	Gear Interpolation Data: Getriebeinterpolationsdaten
GND	Signal Ground
GP	Grundprogramm (PLC)

Anhang

A.1 Liste der Abkürzungen

GS	Getriebestufe
GSD	Gerätstammdatei zur Beschreibung eines PROFIBUS Slaves
GSDML	Generic Station Description Markup Language: XML-basierte Beschreibungssprache zur Erstellung einer GSD-Datei
GUD	Global User Data: Globale Anwenderdaten

H	
HEX	Kurzbezeichnung für hexadezimale Zahl
HiFu	Hilfsfunktion
HLA	Hydraulischer Linearantrieb
HMI	Human Machine Interface: SINUMERIK-Bedienoberfläche
HSA	Hauptspindeltrieb
HW	Hardware

I	
IBN	Inbetriebnahme
IKA	Interpolatorische Kompensation
IM	Interface-Modul: Anschaltungsbaugruppe
IMR	Interface-Modul Receive: Anschaltungsbaugruppe für Empfangsbetrieb
IMS	Interface-Modul Send: Anschaltungsbaugruppe für Sendebetrieb
INC	Increment: Schrittmaß
INI	Initializing Data: Initialisierungsdaten
IPO	Interpolator
ISA	International Standard Architecture
ISO	International Standard Organization

J	
JOG	Jogging: Einrichtbetrieb

K	
K _v	Verstärkungsfaktor des Regelkreises
K _p	Proportionalverstärkung
K _Ü	Übersetzungsverhältnis
KOP	Kontaktplan (Programmiermethode für PLC)

L	
LAI	Logic Machine Axis Image: Logisches Maschinenachsen-Abbild
LAN	Local Area Network
LCD	Liquid-Crystal Display: Flüssigkristallanzeige
LED	Light Emitting Diode: Leuchtdiode
LF	Line Feed
LMS	Lagemesssystem
LR	Lageregler
LSB	Least Significant Bit: Niederwertigstes Bit
LUD	Local User Data: Anwenderdaten (lokal)

M	
MAC	Media Access Control
MAIN	Main program: Hauptprogramm (OB1, PLC)
MB	Megabyte
MCI	Motion Control Interface
MCIS	Motion–Control–Information–System
MCP	Machine Control Panel: Maschinensteuertafel
MD	Maschinendatum bzw. Maschinendaten
MDA	Manual Data Automatic: Handeingabe
MELDW	Meldungswort
MKS	Maschinenkoordinatensystem
MLFB	Maschinenlesbare Fabrikatbezeichnung
MM	Motor Module
MPF	Main Program File: Hauptprogramm (NC)
MSTT	Maschinensteuertafel

N	
NC	Numerical Control: Numerische Steuerung
NCK	Numerical Control Kernel: Numerik-Kern mit Satzaufbereitung, Verfahrbereich usw.
NCU	Numerical Control Unit: Hardware-Einheit des NCK
NRK	Bezeichnung des Betriebssystems des NCK
NST	Nahtstellensignal
NURBS	Non-Uniform Rational B-Spline
NV	Nullpunktverschiebung
NX	Numerical Extension: Achserweiterungsbaugruppe

O	
OB	Organisationsbaustein in der PLC
OEM	Original Equipment Manufacturer
OP	Operation Panel: Bedieneinrichtung
OPI	Operation Panel Interface: Bedientafel-Anschaltung
OPT	Options: Optionen
OLP	Optical Link Plug: Busstecker für Lichtleiter
OSI	Open Systems Interconnection: Normung für Rechnerkommunikation

P	
PAA	Prozessabbild der Ausgänge
PAE	Prozessabbild der Eingänge
PC	Personal Computer
PCIN	Name der SW für den Datenaustausch mit der Steuerung
PCMCIA	Personal Computer Memory Card International Association: Speichersteckkarten-Normierung
PCU	PC Unit: PC-Box (Rechereinheit)
PG	Programmiergerät
PKE	Parameterkennung: Teil eines PKW
PKW	Parameterkennung: Wert (Parametrierteil eines PPO)
PLC	Programmable Logic Control: Anpass-Steuerung
PN	PROFINET
PNO	PROFIBUS-Nutzerorganisation
PO	POWER ON
POE	Programmorganisationseinheit
POS	Position/Positionieren
POSMO A	Positioning Motor Actuator: Positioniermotor
POSMO CA	Positioning Motor Compact AC: Komplette Antriebseinheit mit integrierter Leistungs- und Reglungsbaugruppe sowie Positioniereinheit und Programmspeicher; Wechselstrom-Einspeisung
POSMO CD	Positioning Motor Compact DC: wie CA, jedoch Gleichstromspeisung
POSMO SI	Positioning Motor Servo Integrated: Positioniermotor; Gleichstromspeisung
PPO	Parameter Prozessdaten Objekt ; Zyklisches Datentelegramm bei der Übertragung mit PROFIBUS-DP und Profil "Drehzahlveränderbare Antriebe"
PPU	Panel Processing Unit (zentrale Hardware einer Panel-basierten CNC-Steuerung z.B. SINUMERIK 828D)
PROFIBUS	Process Field Bus: Serieller Datenbus
PRT	Programmtest
PSW	Programmsteuerwort
PTP	Point to Point: Punkt zu Punkt
PUD	Program Global User Data: Programmglobale Anwendervariable
PZD	Prozessdaten: Prozessdatenteil eines PPO

Q	
QFK	Quadrantenfehler Kompensation

R	
RAM	Random Access Memory: Schreib-/Lese-Speicher
REF	Funktion Referenzpunkt anfahren
REPOS	Funktion Repositionieren
RISC	Reduced Instruction Set Computer: Prozessortyp mit kleinem Befehlssatz und schnellem Befehlsdurchsatz
ROV	Rapid Override: Eingangskorrektur
RP	R-Parameter, Rechenparameter, vordefinierte Anwendervariable
RPA	R-Parameter Active: Speicherbereich in NCK für R-Parameternummern
RPY	Roll Pitch Yaw: Drehungsart eines Koordinatensystems
RTL	Rapid Traverse Linear Interpolation: Lineare Interpolation bei Eilgangbewegung
RTS	Request To Send: Sendeteil einschalten, Steuersignal von seriellen Daten-Schnittstellen
RTCP	Real Time Control Protocol

S	
SA	Synchronaktion
SBC	Safe Break Control: Sichere Bremsenansteuerung
SBL	Single Block: Einzelsatz
SBR	Subroutine: Unterprogramm (PLC)
SD	Settingdatum bzw. Settingdaten
SDB	System Datenbaustein
SEA	Setting Data Active: Kennzeichnung (Dateityp) für Settingdaten
SERUPRO	Search-Run by Program Test: Suchlauf via Programmtest
SFB	System Funktionsbaustein
SFC	System Function Call
SGE	Sicherheitsgerichteter Eingang
SGA	Sicherheitsgerichteter Ausgang
SH	Sicherer Halt
SIM	Single in Line Module
SK	Softkey
SKP	Skip: Funktion zum Ausblenden eines Teileprogrammsatzes
SLM	Synchroner Linearmotor

Anhang

A.1 Liste der Abkürzungen

SM	Schrittmotor
SMC	Sensor Module Cabinet Mounted
SME	Sensor Module Externally Mounted
SMI	Sensor Module Integrated
SPF	Sub Program File: Unterprogramm (NC)
SPS	Speicherprogrammierbare Steuerung = PLC
SRAM	Statischer Speicher (gepuffert)
SRK	Schneidenradiuskorrektur
SRM	Synchron rotatorischer Motor
SSFK	Spindelsteigungsfehlerkompensation
SSI	Serial Synchron Interface: Serielle synchrone Schnittstelle
SSL	Satzsuchlauf
STW	Steuerwort
SUG	Scheibenumfangsgeschwindigkeit
SW	Software
SYF	System Files: Systemdateien
SYNACT	Synchronized Action: Synchronaktion

T	
TB	Terminal Board (SINAMICS)
TCP	Tool Center Point: Werkzeugspitze
TCP/IP	Transport Control Protocol / Internet Protocol
TCU	Thin Client Unit
TEA	Testing Data Active: Kennung für Maschinendaten
TIA	Totally Integrated Automation
TM	Terminal Module (SINAMICS)
TO	Tool Offset: Werkzeugkorrektur
TOA	Tool Offset Active: Kennzeichnung (Dateityp) für Werkzeugkorrekturen
TRANSMIT	Transform Milling Into Turning: Koordinatentransformation für Fräsbearbeitungen an einer Drehmaschine
TTL	Transistor-Transistor-Logik (Schnittstellen-Typ)
TZ	Technologiezyklus

U	
UFR	User Frame: Nullpunktverschiebung
UP	Unterprogramm
USB	Universal Serial Bus
USV	Unterbrechungsfreie Stromversorgung

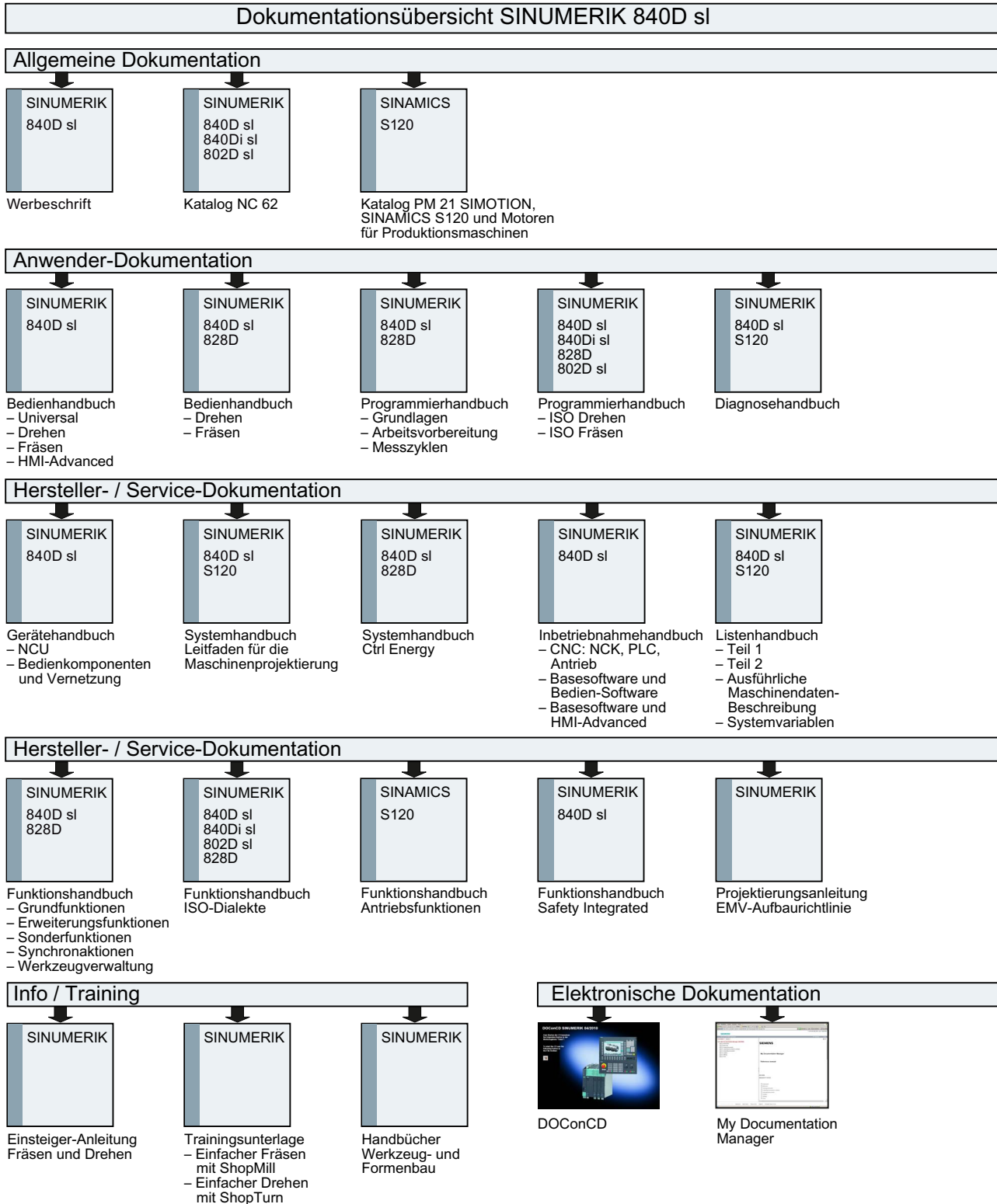
V	
VDI	Interne Kommunikationsschnittstelle zwischen NCK und PLC
VDI	Verein Deutscher Ingenieure
VDE	Verband Deutscher Elektrotechniker
VI	Voltage Input
VO	Voltage Output
VSA	Vorschubantrieb

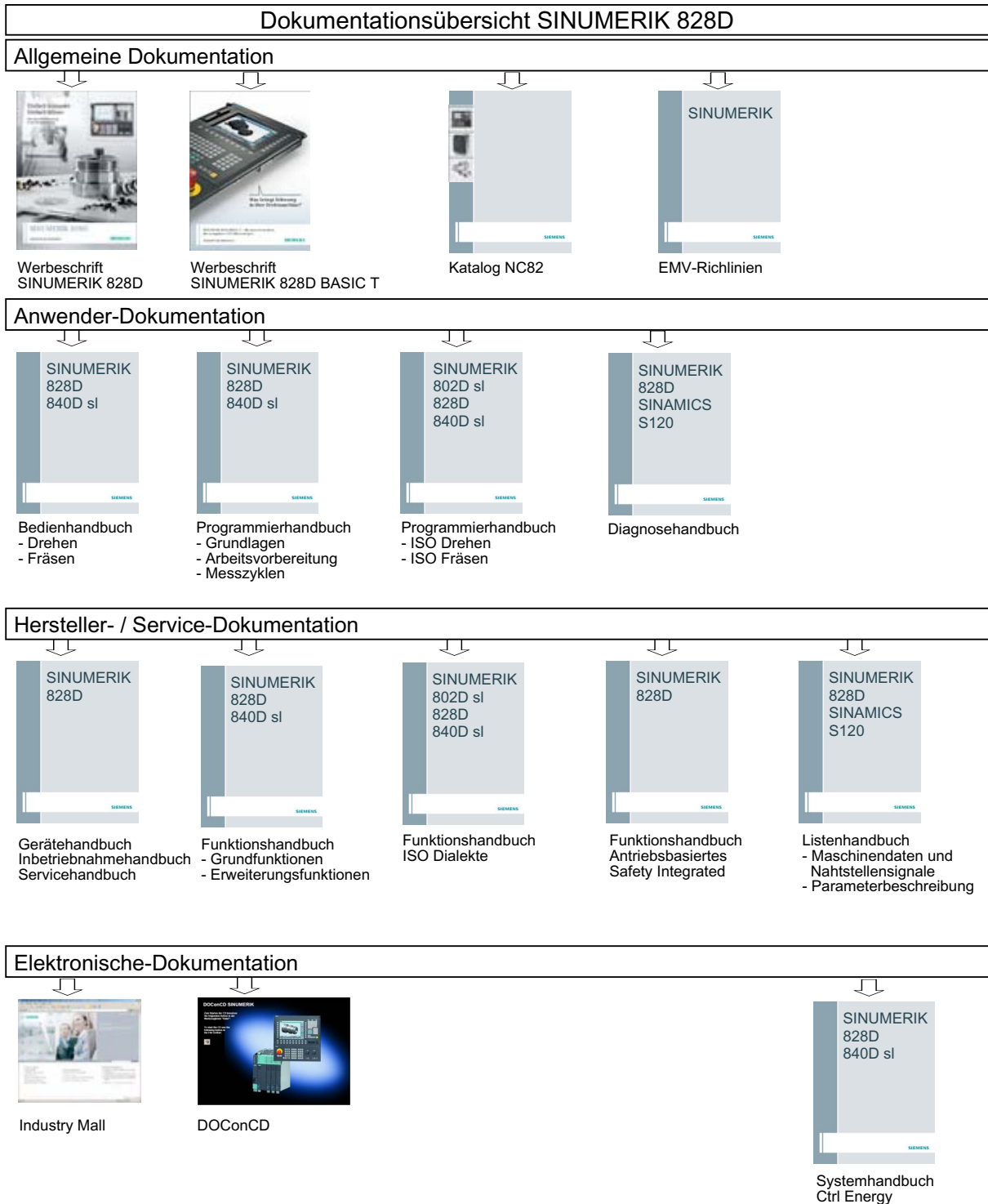
W	
WAB	Funktion Weiches An- und Abfahren
WKS	Werkstückkoordinatensystem
WKZ	Werkzeug
WLK	Werkzeuglängenkorrektur
WOP	Werkstatt-orientierte Programmierung
WPD	Work Piece Directory: Werkstückverzeichnis
WRK	Werkzeug-Radius-Korrektur
WZ	Werkzeug
WZK	Werkzeugkorrektur
WZV	Werkzeugverwaltung
WZW	Werkzeugwechsel

X	
XML	Extensible Markup Language

Z	
ZOA	Zero Offset Active: Kennung für Nullpunktverschiebungen
ZSW	Zustandswort (des Antriebs)

A.2 Dokumentationsübersicht





Glossar

Absolutmaß

Angabe des Bewegungsziels einer Achsbewegung durch ein Maß, das sich auf den Nullpunkt des momentan gültigen Koordinatensystems bezieht. Siehe → Kettenmaß.

Achsadresse

Siehe → Achsbezeichner

Achsbezeichner

Achsen werden nach DIN 66217 für ein rechtsdrehendes, rechtwinkliges → Koordinatensystem bezeichnet mit X, Y, Z.

Um X, Y, Z drehende → Rundachsen erhalten die Bezeichner A, B, C. Zusätzliche Achsen, parallel zu den angegebenen, können mit weiteren Adressbuchstaben gekennzeichnet werden.

Achsen

Die CNC-Achsen werden entsprechend ihres Funktionsumfangs abgestuft in:

- Achsen: interpolierende Bahnachsen
- Hilfsachsen: nicht interpolierende Zustell- und Positionierachsen mit achsspezifischem Vorschub. Hilfsachsen sind an der eigentlichen Bearbeitung nicht beteiligt, z. B. Werkzeugzubringer, Werkzeugmagazin.

Achsname

Siehe → Achsbezeichner

Adresse

Eine Adresse ist die Kennzeichnung für einen bestimmten Operanden oder Operandenbereich, z. B. Eingang, Ausgang usw.

Alarmer

Alle → Meldungen und Alarmer werden auf der Bedientafel im Klartext mit Datum und Uhrzeit und dem entsprechenden Symbol für das Löschkriterium angezeigt. Die Anzeige erfolgt getrennt nach Alarmen und Meldungen.

1. Alarmer und Meldungen im Teileprogramm

Alarmer und Meldungen können direkt aus dem Teileprogramm im Klartext zur Anzeige gebracht werden.

2. Alarmer und Meldungen von PLC

Alarmer- und Meldungen der Maschine können aus dem PLC-Programm im Klartext zur Anzeige gebracht werden. Dazu sind keine zusätzlichen Funktionsbaustein-Pakete notwendig.

Antrieb

Der Antrieb ist diejenige Einheit der CNC, welche die Drehzahl- und Momentenregelung aufgrund der Vorgaben der NC ausführt.

Anwenderdefinierte Variable

Anwender können für beliebige Nutzung im → Teileprogramm oder Datenbaustein (globale Anwenderdaten) anwenderdefinierte Variablen vereinbaren. Eine Definition enthält eine Datentypangabe und den Variablennamen. Siehe → Systemvariable.

Anwenderprogramm

Anwenderprogramme für Automatisierungssysteme S7-300 werden mit der Programmiersprache STEP 7 erstellt. Das Anwenderprogramm ist modular aufgebaut und besteht aus einzelnen Bausteinen.

Die grundlegenden Bausteintypen sind:

- Code-Bausteine

Diese Bausteine enthalten die STEP 7-Befehle.

- Datenbausteine

Diese Bausteine enthalten Konstanten und Variablen für das STEP 7-Programm.

Anwenderspeicher

Alle Programme und Daten wie Teileprogramme, Unterprogramme, Kommentare, Werkzeugkorrekturen, Nullpunktverschiebungen/Frames sowie Kanal- und Programmanwenderdaten können in den gemeinsamen CNC-Anwenderspeicher abgelegt werden.

Arbeitsfeldbegrenzung

Mit der Arbeitsfeldbegrenzung kann der Verfahrbereich der Achsen zusätzlich zu den Endschaltern eingeschränkt werden. Je Achse ist ein Wertepaar zur Beschreibung des geschützten Arbeitsraumes möglich.

Arbeitsraum

Dreidimensionaler Raum, in den die Werkzeugspitze aufgrund der Konstruktion der Werkzeugmaschine hineinfahren kann. Siehe → Schutzraum.

Arbeitsspeicher

Der Arbeitsspeicher ist ein RAM-Speicher in der → CPU, auf den der Prozessor während der Programmbearbeitung auf das Anwenderprogramm zugreift.

Archivieren

Auslesen von Dateien und/oder Verzeichnissen auf ein **externes** Speichergerät.

Asynchrones Unterprogramm

Teileprogramm, das asynchron (unabhängig) zum aktuellen Programmzustand durch ein Interruptsignal (z. B. Signal "schneller NC-Eingang") gestartet werden kann.

Automatik

Betriebsart der Steuerung (Satzfolgebetrieb nach DIN): Betriebsart bei NC-Systemen, in der ein → Teileprogramm angewählt und kontinuierlich abgearbeitet wird.

Bahnachse

Bahnachsen sind alle Bearbeitungsachsen des → Kanals, die vom → Interpolator so geführt werden, dass sie gleichzeitig starten, beschleunigen, stoppen und den Endpunkt erreichen.

Bahngeschwindigkeit

Die maximal programmierbare Bahngeschwindigkeit ist abhängig von der Eingabefeinheit. Bei einer Auflösung von beispielsweise 0,1 mm beträgt die maximal programmierbare Bahngeschwindigkeit 1000 m/min.

Bahnsteuerbetrieb

Ziel des Bahnsteuerbetriebes ist es, ein größeres Abbremsen der → Bahnachsen an den Teileprogramm-Satzgrenzen zu vermeiden und mit möglichst gleicher Bahngeschwindigkeit in den nächsten Satz zu wechseln.

Bahnvorschub

Bahnvorschub wirkt auf → Bahnachsen. Er stellt die geometrische Summe der Vorschübe der beteiligten → Geometrieachsen dar.

Basisachse

Achse, deren Soll- oder Istwert für die Berechnung eines Kompensationswertes herangezogen wird.

Basiskoordinatensystem

Kartesisches Koordinatensystem, wird durch Transformation auf das Maschinenkoordinatensystem abgebildet.

Im → Teileprogramm verwendet der Programmierer Achsnamen des Basiskoordinatensystems. Es besteht, wenn keine → Transformation aktiv ist, parallel zum → Maschinenkoordinatensystem. Der Unterschied zu diesem liegt in den → Achsbezeichnungen.

Baudrate

Geschwindigkeit bei der Datenübertragung (Bit/s).

Baustein

Als Bausteine werden alle Dateien bezeichnet, die für die Programmerstellung und Programmverarbeitung benötigt werden.

Bearbeitungskanal

Über eine Kanalstruktur können durch parallele Bewegungsabläufe Nebenzeiten verkürzt werden, z. B. Verfahren eines Ladeportals simultan zur Bearbeitung. Ein CNC-Kanal ist dabei als eigene CNC-Steuerung mit Dekodierung, Satzaufbereitung und Interpolation anzusehen.

Bedienoberfläche

Die Bedienoberfläche (BOF) ist das Anzeigemedium einer CNC-Steuerung in Gestalt eines Bildschirms. Sie ist mit horizontalen und vertikalen Softkeys gestaltet.

Beschleunigung mit Ruckbegrenzung

Zur Erzielung eines optimalen Beschleunigungsverhaltens an der Maschine bei gleichzeitiger Schonung der Mechanik kann im Bearbeitungsprogramm zwischen sprunghafter Beschleunigung und stetiger (ruckfreier) Beschleunigung umgeschaltet werden.

Betriebsart

Ablaufkonzept für den Betrieb einer SINUMERIK-Steuerung. Es sind die Betriebsarten → Jog, → MDA, → Automatik definiert.

Betriebsartengruppe

Technologisch zusammengehörige Achsen und Spindeln können zu einer Betriebsartengruppe (BAG) zusammengefasst werden. Achsen/Spindeln einer BAG können von einem oder mehreren → Kanälen gesteuert werden. Den Kanälen der BAG ist immer die gleiche → Betriebsart zugeordnet.

Bezeichner

Die Wörter nach DIN 66025 werden durch Bezeichner (Namen) für Variable (Rechenvariable, Systemvariable, Anwendervariable), für Unterprogramme, für Schlüsselwörter und Wörter mit mehreren Adressbuchstaben ergänzt. Diese Ergänzungen kommen in der Bedeutung den Wörtern beim Satzaufbau gleich. Bezeichner müssen eindeutig sein. Derselbe Bezeichner darf nicht für verschiedene Objekte verwendet werden.

Booten

Laden des Systemprogramms nach Power On.

C-Achse

Achse, um die eine gesteuerte Drehbewegung und Positionierung mit der Werkstückspindel erfolgt.

CNC

Siehe → NC

COM

Komponente der NC-Steuerung zur Durchführung und Koordination von Kommunikation.

CPU

Central Processing Unit, siehe → Speicherprogrammierbare Steuerung

C-Spline

Der C-Spline ist der bekannteste und am meisten verwendete Spline. Die Übergänge an den Stützpunkten sind tangential- und krümmungsstetig. Es werden Polynome 3. Grades verwendet.

Datenbaustein

1. Dateneinheit der → PLC, auf die → HIGHSTEP-Programme zugreifen können.
2. Dateneinheit der → NC: Datenbausteine enthalten Datendefinitionen für globale Anwenderdaten. Die Daten können bei der Definition direkt initialisiert werden.

Datenübertragungsprogramm PCIN

PCIN ist ein Hilfsprogramm zum Senden und Empfangen von CNC-Anwenderdaten über die serielle Schnittstelle, wie z. B. Teileprogramme, Werkzeugkorrekturen etc. Das PCIN-Programm ist unter MS-DOS auf Standard-Industrie-PCs lauffähig.

Datenwort

Zwei Byte große Dateneinheit innerhalb eines → Datenbausteins.

Diagnose

1. Bedienbereich der Steuerung
2. Die Steuerung besitzt sowohl ein Selbstdiagnose-Programm als auch Testhilfen für den Service: Status-, Alarm- und Serviceanzeigen

DRF

Differential Resolver Function: NC-Funktion, die in Verbindung mit einem elektronischen Handrad eine inkrementale Nullpunktverschiebung im Automatik-Betrieb erzeugt.

Editor

Der Editor ermöglicht das Erstellen, Ändern, Ergänzen, Zusammenschieben und Einfügen von Programmen/Texten/Programmsätzen.

EIA-Code

Spezieller Lochstreifencode. Die Lochanzahl pro Zeichen ist stets ungerade.

Eilgang

Schnellste Verfahrensgeschwindigkeit einer Achse. Sie wird z. B. verwendet, wenn das Werkzeug aus einer Ruhestellung an die → Werkstückkontur herangefahren oder von der Werkstückkontur zurückgezogen wird. Die Eilganggeschwindigkeit wird maschinenspezifisch über Maschinendatum eingestellt.

Externe Nullpunktverschiebung

Von der → PLC vorgegebene Nullpunktverschiebung.

Fertigteilkontur

Kontur des fertig bearbeiteten Werkstücks. Siehe → Rohteil.

Festpunkt-Anfahren

Werkzeugmaschinen können feste Punkte wie Werkzeugwechsellpunkt, Beladepunkt, Palettenwechsellpunkt etc. definiert anfahren. Die Koordinaten dieser Punkte sind in der Steuerung hinterlegt. Die Steuerung verfährt die betroffenen Achsen, wenn möglich, im → Eilgang.

Frame

Ein Frame stellt eine Rechenvorschrift dar, die ein kartesisches Koordinatensystem in ein anderes kartesisches Koordinatensystem überführt. Ein Frame enthält die Komponenten → Nullpunktverschiebung, → Rotation, → Skalierung, → Spiegelung.

Führungssachse

Die Führungssachse ist die → Gantry-Achse, die aus Sicht des Bedieners und des Programmierers vorhanden und damit entsprechend wie eine normale NC-Achse beeinflussbar ist.

Genauhalt

Bei programmierter Genauhalt-Anweisung wird die in einem Satz angegebene Position genau und ggf. sehr langsam angefahren. Zur Reduktion der Annäherungszeit werden für Eilgang und Vorschub → Genauhaltsgrenzen definiert.

Genauhaltgrenze

Erreichen alle Bahnachsen ihre Genauhaltgrenze, so verhält sich die Steuerung als habe sie einen Zielpunkt exakt erreicht. Es erfolgt Satzweitchaltung des → Teileprogramms.

Geometrie

Beschreibung eines → Werkstücks im → Werkstückkoordinatensystem.

Geometrieachse

Geometrieachsen dienen der Beschreibung eines 2- oder 3-dimensionalen Bereichs im Werkstückkoordinatensystem.

Geradeninterpolation

Das Werkzeug wird auf einer Geraden zum Zielpunkt verfahren und dabei das Werkstück bearbeitet.

Geschwindigkeitsführung

Um bei Verfahrbewegungen um sehr kleine Beträge je Satz eine akzeptable Verfahr­geschwindigkeit erreichen zu können, kann vorausschauende Auswertung über mehrere Sätze (→ Look Ahead) eingestellt werden.

Gewindebohren ohne Ausgleichsfutter

Mit dieser Funktion können Gewinde ohne Ausgleichsfutter gebohrt werden. Durch das interpolierende Verfahren der Spindel als Rundachse und der Bohrachse werden Gewinde exakt auf Endbohrtiefe geschnitten, z. B. Sacklochgewinde (Voraussetzung: Achsbetrieb der Spindel).

Gleichlaufachse

Die Gleichlaufachse ist die → Gantry-Achse, deren Sollposition stets von der Verfahr­bewegung der → Führungsachse abgeleitet und damit synchron verfahren wird. Aus Sicht des Bedieners und des Programmierers ist die Gleichlaufachse "nicht vorhanden".

Grenzdrehzahl

Maximale/minimale (Spindel-)Drehzahl: Durch Vorgaben von Maschinendaten, der → PLC oder → Settingdaten kann die maximale Drehzahl einer Spindel begrenzt sein.

Hauptprogramm

Die Bezeichnung Hauptprogramm stammt noch aus der Zeit, als Teileprogramm fest in Haupt- und → Unterprogramme unterteilt waren. Diese feste Einteilung besteht mit der heutigen SINUMERIK NC-Sprache nicht mehr. Prinzipiell kann jedes Teileprogramm im Kanal angewählt und gestartet werden. Es läuft dann in der → Programmebene 0 (Hauptprogramm-Ebene) ab. Im Hauptprogramm können weitere Teileprogramme oder → Zyklen als Unterprogramme aufgerufen werden

Hauptsatz

Durch ":" eingeleiteter Satz, der alle Angaben enthält, um den Arbeitsablauf in einem → Teileprogramm starten zu können.

HIGHSTEP

Zusammenfassung der Programmiermöglichkeiten für die → PLC des Systems AS300/AS400.

Hilfsfunktionen

Mit Hilfsfunktionen können in → Teileprogrammen → Parameter an die → PLC übergeben werden, die dort vom Maschinenhersteller definierte Reaktionen auslösen.

Hochsprache CNC

Die Hochsprache bietet: → Anwenderdefinierte Variable, → Systemvariable, → Makrotechnik.

HW-Konfig

SIMATIC S7-Tool zum Konfigurieren und Parametrieren von Hardware-Komponenten innerhalb eines S7-Projekts.

Interpolator

Logische Einheit des → NCK, die nach Angaben von Zielpositionen im Teileprogramm Zwischenwerte für die in den einzelnen Achsen zu fahrenden Bewegungen bestimmt.

Interpolatorische Kompensation

Mit Hilfe der interpolatorischen Kompensation können fertigungsbedingte Spindelsteigungsfehler und Messsystemfehler kompensiert werden (SSFK, MSFK).

Interruptroutine

Interruptroutinen sind spezielle → Unterprogramme, die durch Ereignisse (externe Signale) vom Bearbeitungsprozess gestartet werden können. Ein in Abarbeitung befindlicher Teileprogrammsatz wird abgebrochen, die Unterbrechungsposition der Achsen wird automatisch gespeichert.

ISO-Code

Spezieller Lochstreifencode. Die Lochanzahl pro Zeichen ist stets gerade.

JOG

Betriebsart der Steuerung (Einrichtebetrieb): In der Betriebsart JOG kann die Maschine eingerichtet werden. Einzelne Achsen und Spindeln können über die Richtungstasten im Tipbetrieb verfahren werden. Weitere Funktionen in der Betriebsart JOG sind das → Referenzpunktfahren, → Repos sowie → Preset (Istwert setzen).

Kanal

Ein Kanal ist dadurch gekennzeichnet, dass er unabhängig von anderen Kanälen ein → Teileprogramm abarbeiten kann. Ein Kanal steuert exklusiv die ihm zugeordneten Achsen und Spindeln. Teileprogrammabläufe verschiedener Kanäle können durch → Synchronisation koordiniert werden.

Kettenmaß

Auch Inkrementmaß: Angabe eines Bewegungsziels einer Achse durch eine zu verfahrenende Wegstrecke und Richtung bezogen auf einen bereits erreichten Punkt. Siehe → Absolutmaß.

Kompensationsachse

Achse, deren Soll- oder Istwert durch den Kompensationswert modifiziert wird.

Kompensationstabelle

Tabelle von Stützpunkten. Sie liefert für ausgewählte Positionen der Basisachse die Kompensationswerte der Kompensationsachse.

Kompensationswert

Differenz zwischen der durch den Messgeber gemessenen Achsposition und der gewünschten, programmierten Achsposition.

Kontur

Umriss des → Werkstücks

Konturüberwachung

Als Maß für die Konturtreue wird der Schleppfehler innerhalb eines definierbaren Toleranzbandes überwacht. Ein unzulässig hoher Schleppfehler kann sich z. B. durch Überlastung des Antriebs ergeben. In diesem Fall kommt es zu einem Alarm und die Achsen werden stillgesetzt.

Koordinatensystem

Siehe → Maschinenkoordinatensystem, → Werkstückkoordinatensystem

Korrekturspeicher

Datenbereich in der Steuerung, in dem Werkzeugkorrekturdaten hinterlegt sind.

Kreisinterpolation

Das → Werkzeug soll zwischen festgelegten Punkten der Kontur mit einem gegebenen Vorschub auf einem Kreis fahren und dabei das Werkstück bearbeiten.

Krümmung

Die Krümmung k einer Kontur ist das Inverse des Radius r des anschmiegenden Kreises in einem Konturpunkt ($k = 1/r$).

KÜ

Übersetzungsverhältnis

KV

Kreisverstärkungsfaktor, regelungstechnische Größe eines Regelkreises

Ladespeicher

Der Ladespeicher ist bei der CPU 314 der → SPS gleich dem → Arbeitsspeicher.

Linearachse

Die Linearachse ist eine Achse, welche im Gegensatz zur Rundachse eine Gerade beschreibt.

Look Ahead

Mit der Funktion **Look Ahead** wird durch das "Vorausschauen" über eine parametrierbare Anzahl von Verfahrssätzen ein Optimum an Bearbeitungsgeschwindigkeit erzielt.

Losekompensation

Ausgleich einer mechanischen Maschinenlose, z. B. Umkehrlose bei Kugelrollspindeln. Für jede Achse kann die Losekompensation getrennt eingegeben werden.

Makrotechnik

Zusammenfassung einer Menge von Anweisungen unter einem Bezeichner. Der Bezeichner repräsentiert im Programm die Menge der zusammengefassten Anweisungen.

Maschinenachsen

In der Werkzeugmaschine physikalisch existierende Achsen.

Maschinenfestpunkt

Durch die Werkzeugmaschine eindeutig definierter Punkt, z. B. Maschinen-Referenzpunkt.

Maschinenkoordinatensystem

Koordinatensystem, das auf die Achsen der Werkzeugmaschine bezogen ist.

Maschinennullpunkt

Fester Punkt der Werkzeugmaschine, auf den sich alle (abgeleiteten) Messsysteme zurückführen lassen.

Maschinensteuertafel

Bedientafel der Werkzeugmaschine mit den Bedienelementen Tasten, Drehschalter usw. und einfachen Anzeigeelementen wie LEDs. Sie dient der unmittelbaren Beeinflussung der Werkzeugmaschine über die PLC.

Maßangabe metrisch und inch

Im Bearbeitungsprogramm können Positions- und Steigungswerte in inch programmiert werden. Unabhängig von der programmierbaren Maßangabe (*G70/G71*) wird die Steuerung auf ein Grundsystem eingestellt.

Masse

Als Masse gilt die Gesamtheit aller untereinander verbundenen inaktiven Teile eines Betriebsmittels, die auch im Fehlerfall keine gefährliche Berührungsspannung annehmen können.

MDA

Betriebsart der Steuerung: Manual Data Automatic. In der Betriebsart MDA können einzelne Programmsätze oder Satzfolgen ohne Bezug auf ein Haupt- oder Unterprogramm eingegeben und anschließend über die Taste NC-Start sofort ausgeführt werden.

Meldungen

Alle im Teileprogramm programmierten Meldungen und vom System erkannte → Alarme werden auf der Bedientafel im Klartext mit Datum und Uhrzeit und dem entsprechenden Symbol für das Löschkriterium angezeigt. Die Anzeige erfolgt getrennt nach Alarmen und Meldungen.

Metrisches Messsystem

Genormtes System von Einheiten: für Längen z. B. mm (Millimeter), m (Meter).

NC

Numerical Control: NC-Steuerung umfasst alle Komponenten der Werkzeugmaschinensteuerung: → NCK, → PLC, HMI, → COM.

Hinweis

Für die SINUMERIK-Steuerungen wäre CNC-Steuerung korrekter: Computerized Numerical Control.

NCK

Numerical Control Kernel: Komponente der NC-Steuerung, die → Teileprogramme abarbeitet und im Wesentlichen die Bewegungsvorgänge für die Werkzeugmaschine koordiniert.

Nebensatz

Durch "N" eingeleiteter Satz mit Informationen für einen Arbeitsschritt, z. B. eine Positionsangabe.

Netz

Ein Netz ist die Verbindung von mehreren S7-300 und weiteren Endgeräten, z. B. einem PG, über → Verbindungskabel. Über das Netz erfolgt ein Datenaustausch zwischen den angeschlossenen Geräten.

NRK

Numeric Robotic Kernel (Betriebssystem des → NCK)

Nullpunktverschiebung

Vorgabe eines neuen Bezugspunkts für ein Koordinatensystem durch Bezug auf einen bestehenden Nullpunkt und ein → Frame.

1. Einstellbar

Es steht eine projektierbare Anzahl von einstellbaren Nullpunktverschiebungen für jede CNC-Achse zur Verfügung. Die über G-Funktionen anwählbaren Verschiebungen sind alternativ wirksam.

2. Extern

Zusätzlich zu allen Verschiebungen, die die Lage des Werkstücknullpunkts festlegen, kann eine externe Nullpunktverschiebung durch Handrad (DRF-Verschiebung) oder von der PLC überlagert werden.

3. Programmierbar

Mit der Anweisung `TRANS` sind für alle Bahn- und Positionierachsen Nullpunktverschiebungen programmierbar.

NURBS

Die steuerungsinterne Bewegungsführung und Bahninterpolation wird auf Basis von NURBS (**N**on **U**niform **R**ational **B**-**S**plines) durchgeführt. Damit steht steuerungsintern für alle Interpolationen ein einheitliches Verfahren zur Verfügung.

OEM

Für Maschinenhersteller, die ihre eigene Bedienoberfläche erstellen oder technologiespezifische Funktionen in die Steuerung einbringen wollen, sind Freiräume für individuelle Lösungen (OEM-Applikationen) vorgesehen.

Orientierter Spindelhalt

Halt der Werkstückspindel in vorgegebener Winkellage, z. B. um an bestimmter Stelle eine Zusatzbearbeitung vorzunehmen.

Orientierter Werkzeugrückzug

RETTTOOL: Bei Bearbeitungsunterbrechungen (z. B. bei Werkzeugbruch) kann das Werkzeug per Programmbefehl mit vorgegebener Orientierung um einen definierten Weg zurückgezogen werden.

Override

Manuelle bzw. programmierbare Eingriffsmöglichkeit, die es dem Bediener gestattet, programmierte Vorschübe oder Drehzahlen zu überlagern, um sie einem bestimmten Werkstück oder Werkstoff anzupassen.

Peripheriebaugruppe

Peripheriebaugruppen stellen die Verbindung zwischen CPU und Prozess her.

Peripheriebaugruppen sind:

- → Digital-Ein-/Ausgabebaugruppen
- → Analog-Ein-/Ausgabebaugruppen
- → Simulatorbaugruppen

PLC

Programmable Logic Control: → Speicherprogrammierbare Steuerung. Komponente der → NC: Anpass-Steuerung zur Bearbeitung der Kontroll-Logik der Werkzeugmaschine.

PLC-Programmierung

Die PLC wird mit der Software **STEP 7** programmiert. Die Programmiersoftware STEP 7 basiert auf dem Standardbetriebssystem **WINDOWS** und enthält die Funktionen der STEP 5 -Programmierung mit innovativen Weiterentwicklungen.

PLC-Programmspeicher

SINUMERIK 840D sl: Im PLC-Anwenderspeicher werden das PLC-Anwenderprogramm und die Anwenderdaten gemeinsam mit dem PLC-Grundprogramm abgelegt.

Polarkoordinaten

Koordinatensystem, das die Lage eines Punktes in einer Ebene durch seinen Abstand vom Nullpunkt und den Winkel festlegt, den der Radiusvektor mit einer festgelegten Achse bildet.

Polynom-Interpolation

Mit der Polynom-Interpolation können die unterschiedlichsten Kurvenverläufe erzeugt werden, wie **Gerade-, Parabel-, Potenzfunktionen** (SINUMERIK 840D sl).

Positionierachse

Achse, die eine Hilfsbewegung an einer Werkzeugmaschine ausführt. (z. B. Werkzeugmagazin, Palettentransport). Positionierachsen sind Achsen, die nicht mit den → Bahnachsen interpolieren.

Programmbaustein

Programmbausteine enthalten die Haupt- und Unterprogramme der → Teileprogramme.

Programmebene

Ein im Kanal gestartetes Teileprogramm läuft als → Hauptprogramm auf Programmebene 0 (Hauptprogramm-Ebene). Jedes im Hauptprogramm aufgerufene Teileprogramm läuft als → Unterprogramm auf einer eigenen Programmebene 1 ... n.

Programmierbare Arbeitsfeldbegrenzung

Begrenzung des Bewegungsraumes des Werkzeugs auf einen durch programmierte Begrenzungen definierten Raum.

Programmierbare Frames

Mit programmierbaren → Frames können dynamisch im Zuge der Teileprogramm-Abarbeitung neue Koordinatensystem-Ausgangspunkte definiert werden. Es wird unterschieden nach absoluter Festlegung anhand eines neuen Frames und additiver Festlegung unter Bezug auf einen bestehenden Ausgangspunkt.

Programmierschlüssel

Zeichen und Zeichenfolgen, die in der Programmiersprache für → Teileprogramme eine festgelegte Bedeutung haben.

Pufferbatterie

Die Pufferbatterie gewährleistet, dass das → Anwenderprogramm in der → CPU netzausfallsicher hinterlegt ist und festgelegte Datenbereiche und Merker, Zeiten und Zähler remanent gehalten werden.

Quadrantenfehlerkompensation

Konturfehler an Quadrantenübergängen, die durch wechselnde Reibverhältnisse an Führungsbahnen entstehen, sind mit der Quadrantenfehlerkompensation weitgehend eliminierbar. Die Parametrierung der Quadrantenfehlerkompensation erfolgt durch einen Kreisformtest.

Referenzpunkt

Punkt der Werkzeugmaschine, auf den sich das Messsystem der → Maschinenachsen bezieht.

Rohteil

Teil, mit dem die Bearbeitung eines Werkstücks begonnen wird.

Rotation

Komponente eines → Frames, die eine Drehung des Koordinatensystems um einen bestimmten Winkel definiert.

R-Parameter

Rechenparameter, kann vom Programmierer des → Teileprogramms für beliebige Zwecke im Programm gesetzt oder abgefragt werden.

Rundachse

Rundachsen bewirken eine Werkstück- oder Werkzeugdrehung in eine vorgegebene Winkellage.

Rundungsachse

Rundungsachsen bewirken eine Werkstück- oder Werkzeugdrehung in eine einem Teilungsraster entsprechende Winkellage. Beim Erreichen eines Rasters ist die Rundungsachse "in Position".

Satzsuchlauf

Zum Austesten von Teileprogrammen oder nach einem Abbruch der Bearbeitung kann über die Funktion "Satzsuchlauf" eine beliebige Stelle im Teileprogramm angewählt werden, an der die Bearbeitung gestartet oder fortgesetzt werden soll.

Schlüsselschalter

Der Schlüsselschalter auf der → Maschinensteuertafel besitzt 4 Stellungen, die vom Betriebssystem der Steuerung mit Funktionen belegt sind. Zum Schlüsselschalter gehören drei verschiedenfarbige Schlüssel, die in den angegebenen Stellungen abgezogen werden können.

Schlüsselwörter

Wörter mit festgelegter Schreibweise, die in der Programmiersprache für → Teileprogramme eine definierte Bedeutung haben.

Schneidenradiuskorrektur

Bei der Programmierung einer Kontur wird von einem spitzen Werkzeug ausgegangen. Da dies in der Praxis nicht realisierbar ist, wird der Krümmungsradius des eingesetzten Werkzeugs der Steuerung angegeben und von dieser berücksichtigt. Dabei wird der Krümmungsmittelpunkt um den Krümmungsradius verschoben äquidistant um die Kontur geführt.

Schnellabheben von der Kontur

Beim Eintreffen eines Interrupts kann über das CNC-Bearbeitungsprogramm eine Bewegung eingeleitet werden, die ein schnelles Abheben des Werkzeugs von der gerade bearbeiteten Werkstückkontur ermöglicht. Zusätzlich kann der Rückzugwinkel und der Betrag des Weges parametrisiert werden. Nach dem Schnellabheben kann zusätzlich eine Interruptroutine ausgeführt werden.

Schnelle digitale Ein-/Ausgänge

Über die digitalen Eingänge können z. B. schnelle CNC-Programmroutinen (Interruptroutinen) gestartet werden. Über die digitalen CNC-Ausgänge können schnelle, programmgesteuerte Schaltfunktionen ausgelöst werden.

Schrägenbearbeitung

Bohr- und Fräsbearbeitungen an Werkstückflächen, die nicht in den Koordinatenebenen der Maschine liegen, können mit Unterstützung der Funktion "Schrägenbearbeitung" komfortabel ausgeführt werden.

Schraubenlinien-Interpolation

Die Schraubenlinien-Interpolation eignet sich besonders zum einfachen Herstellen von Innen- oder Außengewinden mit Formfräsern und zum Fräsen von Schmiernuten.

Dabei setzt sich die Schraubenlinie aus zwei Bewegungen zusammen:

- Kreisbewegung in einer Ebene
- Linearbewegung senkrecht zu dieser Ebene

Schrittmaß

Verfahrweglängenangabe über Inkrementanzahl (Schrittmaß). Inkrementanzahl kann als → Settingdatum hinterlegt sein bzw. durch entsprechend beschriftete Tasten 10, 100, 1000, 10000 gewählt werden.

Schutzraum

Dreidimensionaler Raum innerhalb des → Arbeitsraumes, in den die Werkzeugspitze nicht hineinreichen darf.

Settingdaten

Daten, die Eigenschaften der Werkzeugmaschine auf durch die Systemsoftware definierte Weise der NC-Steuerung mitteilen.

Sicherheitsfunktionen

Die Steuerung enthält ständig aktive Überwachungen, die Störungen in der → CNC, der Anpass-Steuerung (→ PLC) und der Maschine so frühzeitig erkennen, dass Schäden an Werkstück, Werkzeug oder Maschine weitgehend ausgeschlossen werden. Im Störfall wird der Bearbeitungsablauf unterbrochen und die Antriebe werden stillgesetzt, die Störungsursache gespeichert und als Alarm angezeigt. Gleichzeitig wird der PLC mitgeteilt, dass ein CNC-Alarm ansteht.

Skalierung

Komponente eines → Frames, die achsspezifische Maßstabsveränderungen bewirkt.

Softkey

Taste, deren Beschriftung durch ein Feld im Bildschirm repräsentiert wird, das sich dynamisch der aktuellen Bediensituation anpasst. Die frei belegbaren Funktionstasten (Softkeys) werden softwaremäßig definierten Funktionen zugeordnet.

Software-Endschalter

Software-Endschalter begrenzen den Verfahrbereich einer Achse und verhindern ein Auffahren des Schlittens auf die Hardware-Endschalter. Je Achse sind 2 Wertepaare vorgebar, die getrennt über die → PLC aktiviert werden können.

Speicherprogrammierbare Steuerung

Speicherprogrammierbare Steuerungen (SPS) sind elektronische Steuerungen, deren Funktion als Programm im Steuerungsgerät gespeichert ist. Aufbau und Verdrahtung des Gerätes hängen also nicht von der Funktion der Steuerung ab. Die speicherprogrammierbare Steuerung hat die Struktur eines Rechners; sie besteht aus CPU (Zentralbaugruppe) mit Speicher, Ein-/Ausgabebaugruppen und internem Bus-System. Die Peripherie und die Programmiersprache sind auf die Belange der Steuerungstechnik ausgerichtet.

Spiegelung

Bei Spiegelung werden die Vorzeichen der Koordinatenwerte einer Kontur bezüglich einer Achse vertauscht. Es kann bezüglich mehrerer Achsen zugleich gespiegelt werden.

Spindelsteigungsfehler-Kompensation

Ausgleich mechanischer Ungenauigkeiten einer am Vorschub beteiligten Kugelrollspindel durch die Steuerung anhand von hinterlegten Messwerten der Abweichungen.

Spline-Interpolation

Mit der Spline-Interpolation kann die Steuerung aus nur wenigen vorgegebenen Stützpunkten einer Sollkontur einen glatten Kurvenverlauf erzeugen.

Standardzyklen

Für häufig wiederkehrende Bearbeitungsaufgaben stehen Standardzyklen zur Verfügung:

- für die Technologie Bohren/Fräsen
- für die Technologie Drehen

Im Bedienbereich "Programm" werden unter dem Menü "Zyklusunterstützung" die zur Verfügung stehenden Zyklen aufgelistet. Nach Anwahl des gewünschten Bearbeitungszyklus werden die notwendigen Parameter für die Wertzuweisung im Klartext angezeigt.

Synchronachsen

Synchronachsen benötigen für ihren Weg die gleiche Zeit wie die Geometrieachsen für ihren Bahnweg.

Synchronaktionen

1. Hilfsfunktionsausgabe

Während der Werkstückbearbeitung können aus dem CNC-Programm heraus technologische Funktionen (→ Hilfsfunktionen) an die PLC ausgegeben werden. Über diese Hilfsfunktionen werden beispielsweise Zusatzeinrichtungen der Werkzeugmaschine gesteuert, wie Pinole, Greifer, Spannfutter etc.

2. Schnelle Hilfsfunktionsausgabe

Für zeitkritische Schaltfunktionen können die Quittierungszeiten für die → Hilfsfunktionen minimiert und unnötige Haltepunkte im Bearbeitungsprozess vermieden werden.

Synchronisation

Anweisungen in → Teileprogrammen zur Koordination der Abläufe in verschiedenen → Kanälen an bestimmten Bearbeitungsstellen.

Systemspeicher

Der Systemspeicher ist ein Speicher in der CPU, in der folgende Daten abgelegt werden:

- Daten, die das Betriebssystem benötigt
- die Operanden Zeiten, Zähler, Merker

Systemvariable

Ohne Zutun des Programmierers eines → Teileprogramms existierende Variable. Sie ist definiert durch einen Datentyp und dem Variablennamen, der durch das Zeichen \$ eingeleitet wird. Siehe → Anwenderdefinierte Variable.

Teileprogramm

Folge von Anweisungen an die NC-Steuerung, die insgesamt die Erzeugung eines bestimmten → Werkstücks bewirken. Ebenso Vornahme einer bestimmten Bearbeitung an einem gegebenen → Rohteil.

Teileprogrammsatz

Teil eines → Teileprogramms, durch Line Feed abgegrenzt. Es werden → Hauptsätze und → Nebensätze unterschieden.

Teileprogrammverwaltung

Die Teileprogrammverwaltung kann nach → Werkstücken organisiert werden. Die Größe des Anwenderspeichers bestimmt die Anzahl der zu verwaltenden Programme und Daten. Jede Datei (Programme und Daten) kann mit einem Namen von maximal 24 alphanumerischen Zeichen versehen werden.

Text-Editor

Siehe → Editor

TOA-Bereich

Der TOA-Bereich umfasst alle Werkzeug- und Magazindaten. Standardmäßig fällt der Bereich bzgl. der Reichweite der Daten mit dem Bereich → Kanal zusammen. Über Maschinendaten kann jedoch festgelegt werden, dass sich mehrere Kanäle eine → TOA-Einheit teilen, so dass diesen Kanälen dann gemeinsame WZV-Daten zur Verfügung stehen.

TOA-Einheit

Jeder → TOA-Bereich kann mehrere TOA-Einheiten enthalten. Die Anzahl der möglichen TOA-Einheiten wird über die maximale Anzahl aktiver → Kanäle begrenzt. Eine TOA-Einheit umfasst genau einen WZ-Daten-Baustein und einen Magazindaten-Baustein. Zusätzlich kann noch ein WZ-Trägerdaten-Baustein enthalten sein (optional).

Transformation

Additive oder absolute Nullpunktverschiebung einer Achse.

Unterprogramm

Die Bezeichnung Unterprogramm stammt noch aus der Zeit, als Teileprogramm fest in → Haupt- und Unterprogramme unterteilt waren. Diese feste Einteilung besteht mit der heutigen SINUMERIK NC-Sprache nicht mehr. Prinzipiell kann jedes Teileprogramm oder jeder → Zyklus innerhalb eines anderen Teileprogramms als Unterprogramm aufgerufen werden. Es läuft dann in der nächsten → Programmebene (x+1) (Unterprogrammebene (x+1)) ab.

Urlöschen

Beim Urlöschen werden folgende Speicher der → CPU gelöscht:

- → Arbeitsspeicher
- Schreib-/Lesebereich des → Ladespeichers
- → Systemspeicher
- → Backup-Speicher

V.24

Serielle Schnittstelle für die Dateneingabe/-ausgabe. Über diese Schnittstelle können Bearbeitungsprogramme sowie Hersteller- und Anwenderdaten geladen und gesichert werden.

Variablendefinition

Eine Variablendefinition umfasst die Festlegung eines Datentyps und eines Variablennamens. Mit dem Variablennamen kann der Wert der Variablen angesprochen werden.

Verfahrbereich

Der maximal zulässige Verfahrbereich bei Linearachsen beträgt ± 9 Dekaden. Der absolute Wert ist abhängig von der gewählten Eingabe- und Lageregelfeinheit und dem Einheitensystem (inch oder metrisch).

Vorkoinzidenz

Satzwechsel bereits, wenn Bahnweg um ein vorgegebenes Delta der Endposition nahe gekommen ist.

Vorschub-Override

Der programmierten Geschwindigkeit wird die aktuelle Geschwindigkeitseinstellung über → Maschinensteuertafel oder von der → PLC überlagert (0-200%). Die Vorschubgeschwindigkeit kann zusätzlich im Bearbeitungsprogramm durch einen programmierbaren Prozentfaktor (1-200%) korrigiert werden.

Vorsteuerung, dynamisch

Ungenauigkeiten der → Kontur, bedingt durch Schleppfehler, lassen sich durch die dynamische, beschleunigungsabhängige Vorsteuerung nahezu eliminieren. Dadurch ergibt sich auch bei hohen → Bahngeschwindigkeiten eine hervorragende Bearbeitungsgenauigkeit. Die Vorsteuerung kann achsspezifisch über das → Teileprogramm an- und abgewählt werden.

Werkstück

Von der Werkzeugmaschine zu erstellendes/zu bearbeitendes Teil.

Werkstückkontur

Sollkontur des zu erstellenden/bearbeitenden → Werkstücks.

Werkstückkoordinatensystem

Das Werkstückkoordinatensystem hat seinen Ausgangspunkt im → Werkstücknullpunkt. Bei Programmierung im Werkstückkoordinatensystem beziehen sich Maße und Richtungen auf dieses System.

Werkstücknullpunkt

Der Werkstücknullpunkt bildet den Ausgangspunkt für das → Werkstückkoordinatensystem. Er ist durch Abstände zum → Maschinennullpunkt definiert.

Werkzeug

An der Werkzeugmaschine wirksames Teil, das die Bearbeitung bewirkt (z. B. Drehmeißel, Fräser, Bohrer, LASER-Strahl ...).

Werkzeugkorrektur

Berücksichtigung der Werkzeug-Abmessungen bei der Berechnung der Bahn.

Werkzeugradiuskorrektur

Um eine gewünschte → Werkstückkontur direkt programmieren zu können, muss die Steuerung unter Berücksichtigung des Radius des eingesetzten Werkzeugs eine äquidistante Bahn zur programmierten Kontur verfahren ($G41/G42$).

WinSCP

WinSCP ist ein frei verfügbares Open Source-Programm für Windows zum Transferieren von Dateien.

Zeitreziproker Vorschub

Anstelle der Vorschubgeschwindigkeit kann für die Achsbewegung auch die Zeit programmiert werden, die der Bahnweg eines Satzes benötigen soll (G93).

Zoll-Maßsystem

Maßsystem, das Entfernungen in "inch" und Bruchteilen davon definiert.

Zwischensätze

Verfahrbewegungen mit angewählter → Werkzeugkorrektur (G_{41}/G_{42}) dürfen durch eine begrenzte Anzahl Zwischensätze (Sätze ohne Achsbewegungen in der Korrektorebene) unterbrochen werden, wobei die Werkzeugkorrektur noch korrekt verrechnet werden kann. Die zulässige Anzahl Zwischensätze, die die Steuerung vorausliest, ist über Systemparameter einstellbar.

Zyklen

Geschützte Unterprogramme zur Ausführung von wiederholt auftretenden Bearbeitungsvorgängen am → Werkstück.

Index

\$

- \$AA_ATOL, 511
- \$AA_COUP_ACT
 - bei axialer Leitwertkopplung, 544
 - bei Tangentialsteuerung, 469
 - beim Mitschleppen, 519
- \$AA_ESR_ENABLE, 636
- \$AA_LEAD_SP, 544
- \$AA_LEAD_SV, 544
- \$SAC_ACT_PROG_NET_TIME, 619
- \$SAC_ACTUAL_PARTS, 622
- \$SAC_AXCTSWA, 607
- \$SAC_AXCTSWE, 607
- \$SAC_CTOL, 511
- \$SAC_CUT_INV, 462
- \$SAC_CUTMOD, 462
- \$SAC_CUTMOD_ANG, 461
- \$SAC_CUTTING_TIME, 619
- \$SAC_CYCLE_TIME, 618
- \$SAC_DELAYFST, 492
- \$SAC_ESR_TRIGGER, 636
- \$SAC_OLD_PROG_NET_TIME, 619
- \$SAC_OLD_PROG_NET_TIME_COUNT, 619
- \$SAC_OPERATING_TIME, 618
- \$SAC_OTOL, 511
- \$SAC_PROG_NET_TIME_TRIGGER, 619
- \$SAC_REPOS_PATH_MODE, 502
- \$SAC_REQUIRED_PARTS, 622
- \$SAC_SMAXVELO, 507
- \$SAC_SMAXVELO_INFO, 507
- \$SAC_SPECIAL_PARTS, 622
- \$SAC_STOLF, 513
- \$SAC_TOTAL_PARTS, 622
- \$AN_AXCTAS, 607
- \$AN_AXCTSWA, 607
- \$AN_ESR_TRIGGER, 636
- \$AN_LANGUAGE_ON_HMI, 827
- \$AN_POWERON_TIME, 618
- \$AN_SETUP_TIME, 618
- \$P_ACTBFRAME, 318
- \$P_AD, 462
- \$P_BFRAME, 318
- \$P_CHBFRAME, 318
- \$P_CHBFRMASK, 319
- \$P_CTOL, 511
- \$P_CUT_INV, 462
- \$P_CUTMOD, 462
- \$P_CUTMOD_ANG, 461
- \$P_DELAYFST, 492
- \$P_IFRAME, 319
- \$P_NCBFRAME, 317
- \$P_NCBFRMASK, 319
- \$P_OTOL, 511
- \$P_PFRAME, 319
- \$P_SIM, 287
- \$P_STOLF, 513
- \$P_SUBPAR, 174
- \$PA_ATOL, 511
- \$\$SA_LEAD_TYPE, 543
- \$\$SC_CONTPREC, 483
- \$\$SC_MINFEED, 483
- \$\$SC_PA_ACTIV_IMMED, 240
- \$\$SN_PA_ACTIV_IMMED, 240
- \$TC_CARR1...14, 447
- \$TC_CARR18...23, 447
- \$TC_CARR18[m], 451
- \$TC_DP1 ... 25, 403
- \$TC_ECPxy, 409
- \$TC_SCPxy, 409
- \$TC_TPG1 ... 9, 598

- *
 - * (Rechenfunktion), 73

- /
 - / (Rechenfunktion), 73

- +
 - + (Rechenfunktion), 73

- <
 - < (Vergleichsoperator), 75
 - << (Verkettungsoperator), 86
 - <= (Vergleichsoperator), 75
 - <> (Vergleichsoperator), 75

- =
- == (Vergleichsoperator), 75
- >
- > (Vergleichsoperator), 75
- >= (Vergleichsoperator), 75
- 0**
- 0-Zeichen, 83
- 3**
- 3D-Werkzeugkorrektur, 428
 - Bahnkrümmung, 430
 - Eintauchtiefe, 430
 - Korrektur auf der Bahn, 429
 - Schnittpunktverfahren, 432
- 3D-Werkzeugradiuskorrektur, 424
 - 3DSchnittpunkt der Äquidistanten, 432
 - Innenecken/Außenecken, 431
 - Stirnfräsen, 427
 - Übergangskreis, 432
 - Umfangsfräsen, 426
- A**
- ABS, 73
- Abspanen
 - unterstützende Funktionen, 645
- Abspanen - CYCLE951
 - extern programmieren, 717
- Abstich - CYCLE92
 - extern programmieren, 731
- ACCLIMA, 478
- Achs
 - tausch, 143
- Achsen
 - Mitschlepp-, 518
- ACOS, 73
- ACTBLOCNO, 187
- ACTFRAME, 295
- ADISPOSA, 289
- Alarme
 - im NC-Programm setzen, 633
- ALF
 - für Schnellabheben von der Kontur, 139
- AND, 75
- Anweisungen
 - Liste, 743
- APR, 41
- APRB, 41
- APRP, 41
- APW, 41
- APWB, 41
- APWP, 41
- Arbeitsspeicher, 230
- Array, 47
- AS, 221
- ASIN, 73
- A-Spline, 256
- ASPLINE, 250
- ASUP, 130
- Asynchrones Pendeln, 571
- ATAN2, 73
- ATOL, 509
- Aufrunden, 168
- Ausdrehen - CYCLE86
 - extern programmieren, 670
- Ausgabe
 - auf externes Gerät/Datei, 623
- Automatische Wegaufteilung, 590
- Automatischer Unterbrechungszeiger, 494
- AV, 554
- AX, 599
- AXCTSWE, 606
- AXCTSWEC, 606
- AXCTSWED, 606
- Axiale Leitwertkopplung, 539
- AXIS, 25
- AXNAME, 85
- AXSTRING, 599
- AXTOCHAN, 148
- AXTOINT, 56
- AXTOSPI, 599
- B**
- B_AND, 75
- B_NOT, 75
- B_OR, 75
- B_XOR, 75
- Bahnbezug
 - Einstellbarer, 271
- Bahnfräsen - CYCLE72
 - extern programmieren, 710
- Bahnrelative Werkzeugorientierung, 362
- BAUTO, 250
- Bearbeitungszeit, 619
- Beliebige Positionen - CYCLE802
 - extern programmieren, 678

- Beschleunigungsmodus, 475
 Betriebsmodus
 beim Messen, 282
 Bewegungsendekriterium
 programmierbar, 289
 BFRAME, 295
 BLOCK, 207
 BLSYNC, 132
 BNAT, 250
 Bohren - CYCLE82
 extern programmieren, 666
 Bohrgewinde fräsen - CYCLE78
 extern programmieren, 676
 BOOL, 25
 BOUND, 79
 BRISK, 475
 BRISKA, 475
 B-Spline, 257
 BSPLINE, 250
 BTAN, 250
- C**
- CAC, 249
 CACN, 249
 CACP, 249
 CALCDAT, 659
 CALL, 206
 CALLPATH, 211
 CASE, 106
 CDC, 249
 CFINE, 306
 CHAN, 25
 CHANDATA, 230
 CHAR, 25
 CHECKSUM, 166
 CHKDNO, 444
 CIC, 249
 CLEARM, 124
 CLRINT, 135
 COARSE, 554
 COARSEA, 289
 COMCAD, 262
 COMPCURV, 262
 COMPLETE, 230
 COMPOF, 262
 COMPON, 262
 CONTDCON, 652
 CONTPRON, 646
 COS, 73
 COUPDEF, 554
 COUPDEL, 554
 COUPOF, 554
 COUPOFS, 554
 COUPON, 554
 COUPONC, 554
 COUPRES, 554
 CP, 390
 CPRECOF, 483
 CPRECON, 483
 CPROT, 237
 CPROTDEF, 233
 C-Spline, 258
 CSPLINE, 250
 CTAB, 532
 CTABDEF, 521
 CTABDEL, 528
 CTABEND, 521
 CTABEXISTS, 527
 CTABFNO, 537
 CTABFPOL, 537
 CTABFSEG, 537
 CTABID, 530
 CTABINV, 532
 CTABISLOCK, 530
 CTABLOCK, 529
 CTABMENTYP, 530
 CTABMPOL, 537
 CTABMSEG, 537
 CTABNO, 537
 CTABNOMEM, 537
 CTABPERIOD, 530
 CTABPOL, 537
 CTABPOLID, 537
 CTABSEG, 537
 CTABSEGID, 537
 CTABSEV, 532
 CTABSSV, 532
 CTABTEP, 532
 CTABTEV, 532
 CTABTMAX, 532
 CTABTMIN, 532
 CTABTSP, 532
 CTABTSV, 532
 CTABUNLOCK, 529
 CTOL, 509
 CTRANS, 306
 CUT3DC, 424
 CUT3DCC, 433
 CUT3DCCD, 433
 CUT3DF, 424
 CUT3DFF, 424
 CUT3DFS, 424
 CUTMOD, 458

CYCLE60
 extern programmieren, 707
CYCLE61
 extern programmieren, 683
CYCLE62
 extern programmieren, 709
CYCLE63
 extern programmieren, 714
CYCLE64
 extern programmieren, 713
CYCLE70
 extern programmieren, 705
CYCLE72
 extern programmieren, 710
CYCLE76
 extern programmieren, 690
CYCLE77
 extern programmieren, 692
CYCLE78
 extern programmieren, 676
CYCLE79
 extern programmieren, 694
CYCLE800
 extern programmieren, 737
CYCLE801
 extern programmieren, 681
CYCLE802
 extern programmieren, 678
CYCLE81
 extern programmieren, 665
CYCLE82
 extern programmieren, 666
CYCLE83
 extern programmieren, 668
CYCLE832
 extern programmieren, 740
CYCLE84
 extern programmieren, 671
CYCLE840
 extern programmieren, 674
CYCLE85
 extern programmieren, 667
CYCLE86
 extern programmieren, 670
CYCLE899
 extern programmieren, 701
CYCLE92
 extern programmieren, 731
CYCLE930
 extern programmieren, 719
CYCLE940
 extern programmieren, 722

CYCLE951
 extern programmieren, 717
CYCLE952
 extern programmieren, 733
CYCLE98
 extern programmieren, 728
CYCLE99
 extern programmieren, 725

D

Datei
 -informationen, 163
DEF, 25
DEFAULT, 106
DEFINE ... AS, 221
DELAYFSTOF, 488
DELAYFSTON, 488
DELDL, 410
DELETE, 156
DISABLE, 134
DISPLOF, 187
DISPLON, 187
DISPR, 495
DIV, 73
DL, 407
D-Nummer
 frei vergeben, 443
D-Nummern
 prüfen, 444
 umbenennen, 445
DO, 569
Drehachsen
 Abstandsvektoren, 447
 Richtungsvektoren, 447
 Verdrehwinkel, 447
Drehung
 des Orientierungsvektors, 359
DRIVE, 475
DRIVEA, 475
DV, 554
DYNFINISH, 480
DYNNORM, 480
DYNPOS, 480
DYNROUGH, 480
DYNSEMIFIN, 480

E

EAUTO, 250
Eckenverzögerung an allen Ecken, 288

Eckenverzögerung an Innenecken, 288

EG

Elektronisches Getriebe, 545

EGDEF, 545

EGDEL, 552

EGOFC, 551

EGOFS, 551

EGON, 547

EGONSYN, 547

EGONSYNE, 547

Einrichtewert, 409

Einstich - CYCLE930

extern programmieren, 719

Eintauchtiefe, 430

Einzelachsbewegung, 595

Einzelatz

-unterdrückung, 181

Elektronisches Getriebe, 545

ELSE, 116

ENABLE, 134

ENAT, 250

ENDFOR, 119

ENDIF, 116

ENDLABEL, 108

ENDLOOP, 118

Endlosschleife, 118

ENDWHILE, 121

Erweiterte Messfunktion, 390

ESR, 635

ESRR, 643

ESRS, 642

ETAN, 250

Eulerwinkel, 341

EVERY, 569

EXECSTRING, 72

EXECTAB, 658

EXECUTE, 661

EXP, 73

EXTCALL, 213, 217

EXTCLOSE, 623

EXTERN, 201

Externe Nullpunktverschiebung, 308

EXTOPEN, 623

F

F10, 233

FALSE, 25

FCUB, 470

Feld

-definition, 47

-element, 47

Feldindex, 50

FENDNORM, 288

FFWOF, 482

FFWON, 482

FGROUP-Achsen, 271

FIFOCTRL, 485

FILEDATE, 163

FILEINFO, 163

FILESIZE, 163

FILESTAT, 163

FILETIME, 163

FINE, 554

FINEA, 289

FLIN, 470

FNORM, 470

Folgeachse

bei axialer Leitwertkopplung, 539

bei Tangentialsteuerung, 463

FOR, 119

FPO, 470

Frame

aufrufen, 303

-Kettung, 321

FRAME, 25

Framekomponente

FI, 302

MI, 302

RT, 302

SC, 302

TR, 302

Frames

Frameketten, 304

Kanalspezifische, 316

NCU-globale, 315

System-, 317

zuweisen, 304

Framevariable

Aufruf von Koordinatentransformationen, 293

Definition neuer Frames, 305

Nullpunktverschiebungen G54 bis G599, 298

Vordefinierte Framevariable, 295, 303

Werte zuweisen, 299

Zuordnungen zu den G-Befehlen G54 bis

G599, 298

Fräser

-hilfspunkt (FH), 430

-spitze (FS), 430

Fräserformen, 428

Freistich - CYCLE940

extern programmieren, 722

FROM, 569

FTOCOF, 419

FTOCON, 419

G

G0-Toleranzfaktor, 512
G5, 388
G62, 288
G621, 288
G7, 388
G810 ... G819, 287
G820 ... G829, 287
G-Code
 indirekt programmieren, 68
GEOAX, 602
Geometrieachse
 umschalten, 602
Geschwindigkeitskopplung, 556
GET, 143
GETACTTD, 446
GETD, 143
GETDNO, 445
GETVARAP, 60
GETVARDFT, 62
GETVARLIM, 61
GETVARPHU, 59
GETVARTYP, 64
Gewindebohren mit Ausgleichsfutter - CYCLE840
 extern programmieren, 674
Gewindebohren ohne Ausgleichsfutter - CYCLE84
 extern programmieren, 671
Gewindedrehen - CYCLE99
 extern programmieren, 725
Gewindefräsen - CYCLE70
 extern programmieren, 705
Gewindekette - CYCLE98
 extern programmieren, 728
G-Gruppe
 Technologie, 480
Glättung
 des Orientierungsverlaufs, 371
GOTO, 103
GOTOB, 103
GOTOC, 103
GOTOF, 103
GOTOS, 102
GP, 69
Gravurzyklus - CYCLE60
 extern programmieren, 707
Grundstellung der Werkzeugorientierung
ORIRESET, 336
GUD, 25

H

Haltesatz, 494
High Speed Settings – CYCLE832
 extern programmieren, 740
Hilfsfunktionen, 590
HOLES1
 extern programmieren, 680
HOLES2
 extern programmieren, 682
Hubauslösung, 588

I

ID, 569
IDS, 569
IF, 116
IFRAME, 295
INDEX, 89
Indirekte Programmierung
 von Adressen, 65
 von G-Codes, 68
 von Positionsattributen, 69
 von Teileprogrammzeilen, 72
INICF, 25
INIPO, 25
INIRE, 25
INIT, 124
INITIAL, 230
INITIAL_INI, 230
Initialisierung
 von Feldern, 47
Initialisierungsprogramm, 230
INT, 25
Interpolation des Drehvektors, 359
Interruptroutine, 130
 Aus-/Einschalten, 134
 Löschen, 135
 Neu Zuordnen, 133
 Programmierbare Verfahrrichtung, 139
 Rückzugsbewegung, 139
 Schnellabheben von der Kontur, 136
 Zuordnen und Starten, 132
INTERSEC, 656
INTTOAX, 56
IPOBRKA, 289
IPOENDA, 289
IPOSTOP, 554
IPTRLOCK, 493
IPTRUNLOCK, 493
ISAXIS, 599
ISD, 424

ISFILE, 161
 ISNUMBER, 85
 ISOCALL, 209
 Istwertkopplung, 556
 ISVAR, 57

J

JERKLIM, 505
 JERKLIMA, 478

K

Kartesisches PTP-Fahren, 328
 Kinematik
 Aufgelöste, 451
 Kinematiktyp, 451
 Kompressor, 262
 Kontroll
 -strukturen, 115
 Kontur
 -aufbereitung, 646
 -codierung, 652
 -tabelle, 646, 652
 wieder anfahren, 495
 Konturaufbereitung
 Fehlerrückmeldung, 661
 Konturaufruf - CYCLE62
 extern programmieren, 709
 Konturelement
 abfahren, 658
 Konturgenauigkeit
 programmierbare, 483
 Konturstechen - CYCLE952
 extern programmieren, 733
 Konturtasche vorbohren – CYCLE63
 extern programmieren, 714
 Konturtasche vorbohren – CYCLE64
 extern programmieren, 713
 Koppel, 463
 Koppelfaktor, 515
 Kopplungsart, 556
 Kopplungsstatus
 bei axialer Leitwertkopplung, 544
 beim Mitschleppen, 519
 Korrekturspeicher, 403
 Kreisdaten
 berechnen, 659
 Kreisnut - SLOT2
 extern programmieren, 699
 Kreistasche - POCKET4

 extern programmieren, 687
 Kreiszapfen - CYCLE77
 extern programmieren, 692
 KS, 463

L

L..., 199
 Label, 108
 Lagesynchronität, 554
 Langloch - LONGHOLE
 extern programmieren, 703
 Längsnut - SLOT1
 extern programmieren, 696
 Laufzeit
 -verhalten von Kontrollstrukturen, 116
 LEAD, 338
 LEADOF, 539
 LEADON, 539
 Leitachse
 bei axialer Leitwertkopplung, 539
 bei Tangentialsteuerung, 463
 Leitwertkopplung
 Ist- und Sollwertkopplung, 543
 Synchronisation Leit- und Folgeachse, 542
 Leitwertsimulation, 543
 LIFTFAST, 136
 Link
 -Variablen, 23
 LLI, 37
 LN, 73
 Logische Operatoren, 75
 LONGHOLE
 extern programmieren, 703
 LOOP, 118
 LUD, 25

M

M17, 191
 M30, 191
 Makro, 221
 MASLDEF, 565
 MASLDEL, 565
 MASLOF, 565
 MASLOFS, 565
 MASLON, 565
 MATCH, 89
 MAXVAL, 79
 MCALL, 204
 MEAC, 277

MEAFRAME, 311
MEAS, 274
MEASA, 277
MEAW, 274
MEAWA, 277
Mehrkant - CYCLE79
 extern programmieren, 694
Messauftragsstatus, 285
MINDEX, 89
MINVAL, 79
Mitschleppachsen, 518
Mitschleppen, 515
Mitschleppverband, 515
MMC, 616
MOD, 73
MODAXVAL, 599
MPF, 226

N

NCK, 25
NC-Satz-Kompressor, 262
Nennerpolynom, 269
NEWCONF, 150
Nibbeln
 aktivieren/deaktivieren, 585
 automatische Wegaufteilung, 590
NOC, 554
NOT, 75
NPROT, 237
NPROTDEF, 233
Nullpunktverschiebung
 Externe Nullpunktverschiebung, 308
 PRESETON, 309
NUMBER, 85
NUT, 351

O

OEM-Adressen, 287
OEM-Funktionen, 287
OEMIPO1/2, 287
Offene Nut fräsen - CYCLE899
 extern programmieren, 701
Offset Kontur-normal OFFN, 383
OMA1 ... OMA5, 287
Online-Werkzeuglängenkorrektur, 455
OR, 75
ORIAxes, 348
ORIC, 437
ORICONCCW, 351

ORICONCW, 351
ORICONIO, 351
ORICONTO, 351
ORICURVE, 355
ORID, 437
Orientierbare Werkzeugträger, 447
Orientierungsachsen, 348
Orientierungsprogrammierung, 348
Orientierungstransformation TRAORI
 Generische 5/6-Achs Transformation, 327
 Maschinenkinematik, 326
 Orientierungsprogrammierung, 336
 Varianten der Orientierungsprogrammierung, 336
Verfahrbewegungen und
 Orientierungsbewegungen, 326
Orientierungsvektor THETA, 359
ORIEULER, 348
ORIMKS, 346
ORIPATH, 363
ORIPATHS, 363
ORIPLANE, 351
ORIRESET(A, B, C), 336
ORIROTA, 359
ORIROTC
 bei Drehung der Werkzeugorientierung, 359
 bei Interpolation der Werkzeugdrehung, 365
ORIROTR, 359
ORIROTT, 359
ORIRPY, 348
ORIRPY2, 348
ORIS, 437
ORISOF, 371
ORISON, 371
ORIVECT, 348
ORIVIRT1, 348
ORIVIRT2, 348
ORIWKS, 346
OS, 571
OSB, 571
OSC, 437
OSCILL, 577
OSCTRL, 571
OSD, 437
OSE, 571
OSNSC, 571
OSOF, 437
OSP1, 571
OSP2, 571
OSS, 437
OSSE, 437
OST, 437
OST1, 571

OST2, 571
OTOL, 509

P

P..., 203

P_ACTFRAME, 320

Parameter

Aktual-, 172

Formal-, 172

-übergabe bei Unterprogrammaufruf, 201

-übergabe beim Unterprogrammaufruf, 173

Werkzeug-, 403

PCALL, 210

PDELAYOF, 585

PDELAYON, 585

Pendelbewegung

Umkehrbereich, 580

Umkehrpunkt, 580

Zustellung im Umkehrpunkt, 582

Zustellung unterdrücken, 580

Pendeln

Asynchrones, 571

Asynchrones Pendeln, 571

Synchrones Pendeln, 577

Teilstellung, 580

über Synchronaktion steuern, 577

PFRAME, 295

PHI

bei Orientierung entlang einer

Kegelmantelfläche, 351

Orientierungspolynome, 358

PHU, 39

PL

bei Polynom-Interpolation, 265

bei Spline-Interpolation, 250

Planfräsen - CYCLE61

extern programmieren, 683

PO, 265

PO[PHI]

bei Drehung der Werkzeugorientierung, 363

bei Orientierung entlang einer

Kegelmantelfläche, 351

Orientierungspolynome, 358

PO[PSI]

bei Drehung der Werkzeugorientierung, 363

bei Orientierung entlang einer

Kegelmantelfläche, 351

Orientierungspolynome, 358

PO[THT]

bei Drehung der Werkzeugorientierung, 363

Orientierungspolynome, 358

PO[XH]

bei Orientierungsvorgabe zweier Kontaktpunkte, 355

Orientierungspolynome, 358

PO[YH]

bei Orientierungsvorgabe zweier Kontaktpunkte, 355

Orientierungspolynome, 358

PO[ZH]

bei Orientierungsvorgabe zweier Kontaktpunkte, 355

Orientierungspolynome, 358

POCKET3

extern programmieren, 685

POCKET4

extern programmieren, 687

Polar-Transformation, 328

POLF

für NC-geführtes Rückziehen, 637

POLFA, 637

POLFMASK

für NC-geführtes Rückziehen, 637

POLFMLIN

für NC-geführtes Rückziehen, 637

POLY, 265

Polynom-Interpolation, 265

Polynomkoeffizient, 266

POLYPATH, 265

PON, 594

PONS, 585

POSFS, 554

Positionsattribute

indirekt programmieren, 69

Positionsmuster Gitter/Rahmen - CYCLE801

extern programmieren, 681

Positionsmuster Kreis - HOLES2

extern programmieren, 682

Positionsmuster Linie - HOLES1

extern programmieren, 680

POT, 73

PREPRO, 190

PRESETON, 309

Preset-Verschiebung, 309

PRIO, 132

PRLOC, 25

PROC, 176

Process DataShare, 623

Programm

Initialisierungs-, 230

-koordinierung, 124

-laufzeiten, 618

-speicher, 226

-sprünge, 103

-verzweigung, 106

-wiederholung, 203

Programmierbefehle
 Liste, 743
Programmschleife
 Endschleife, 118
 IF-Schleife, 116
 REPEAT-Schleife, 122
 WHILE-Schleife, 121
 Zählschleife, 119
Programmspeicher
 Dateitypen, 226
 Standard-Verzeichnisse, 226
Programmteil
 -wiederholung, 108
Programmteiwiederholung
 mit indirekter Programmierung CALL, 207
PSI
 bei Orientierung entlang einer
 Kegelmantelfläche, 351
 Orientierungspolynome, 358
PTP, 390
PTP bei TRANSMIT, 395
PTPG0, 395
PUD, 25
PUNCHACC, 585
PUTFTOC, 419
PUTFTOCF, 419
PW, 250

R

R..., 21
Randbedingungen bei Transformationen, 399
READ, 158
REAL, 25
Rechenparameter
 -nummer n, 21
Rechenparameter (R), 21
Rechtecktasche - POCKET3
 extern programmieren, 685
Rechteckzapfen - CYCLE76
 extern programmieren, 690
REDEF, 31
Reiben - CYCLE85
 extern programmieren, 667
RELEASE, 143
REP, 47
REPEAT, 108
REPEATB, 108
REPOS, 130
REPOSA, 495
REPOSH, 495
REPOSHA, 495

REPOSL, 495
REPOSQ, 495
REPOSQA, 495
Restweglöschen, 283
Restzeit
 für ein Werkstück, 620
RET, 192
RET (parametrierbar), 193
Richtungsvektor, 343
RINDEX, 89
RMB, 495
RME, 495
RMI, 495
RMN, 495
ROUND, 73
ROUNDUP, 168
RPY-Winkel, 342
Ruck
 -begrenzung, 475
 -korrektur, 505
Rückziehen
 antriebsautarkes, 643
 NC-geführtes, 637

S

Satzanzeige, 209
 unterdrücken, 187
SAVE, 180
SBLOF, 181
SBLON, 181
Schachtelungstiefe
 von Kontrollstrukturen, 115
Schneidenummer, 443
Schnellabheben von der Kontur, 136
Schräge Achse (TRAANG), 385
Schutz
 -bereiche, 233
Schwenken – CYCLE800
 extern programmieren, 737
SCPARA, 610
SD, 250
SD42475, 369
SD42476, 369
SD42477, 369
SD42678, 371
SD42680, 371
SD42900, 413
SD42910, 413
SD42920, 414
SD42930, 414
SD42935, 417

- SD42940, 418
 SD42984, 459
 Seitwärtswinkel, 339
 SET, 47
 SETAL, 633
 SETDNO, 445
 SETINT, 132
 SETM, 124
 SIN, 73
 Singuläre Stellen, 347
 SLOT1
 extern programmieren, 696
 SLOT2
 extern programmieren, 699
 SOFT, 475
 SOFTA, 475
 Sollwertkopplung, 556
 SON, 585
 SONS, 585
 SPATH, 271
 Speicher
 Arbeits-, 230
 Programm-, 225
 Vorlauf-, 485
 SPF, 226
 SPI, 599
 SPIF1, 585
 SPIF2, 585
 Spindel
 -tausch, 143
 Spline
 -Interpolation, 250
 -Typen, 256
 SPLINEPATH, 261
 Spline-Verbund, 261
 SPN, 590
 SPOF, 585
 SPP, 590
 SPRINT, 93
 Sprung
 auf Programmanfang, 102
 auf Sprungmarken, 103
 Sprungmarke
 bei Programmsprüngen, 103
 bei Programmteiwiederholungen, 108
 SQRT, 73
 Stanzen
 aktivieren/deaktivieren, 585
 automatische Wegaufteilung, 590
 START, 124
 STARTFIFO, 485
 STAT, 390
 Stillsetzen
 antriebsautarkes, 642
 NC-geführtes, 641
 Stirnfräsen, 345
 STOLF, 512
 STOPFIFO, 485
 STOPRE, 485
 String
 formatieren, 93
 -länge, 89
 -operationen, 83
 -Verkettung, 86
 STRING, 25
 STRINGIS, 612
 STRLEN, 89
 SUBSTR, 91
 Suchpfad
 bei Unterprogrammaufruf, 228
 beim Unterprogrammaufruf, 172
 Programmierbarer Suchpfad, 211
 Suchunfähige Bereiche erfassen und suchen, 494
 Synchrones Pendeln
 Anhalten im Umkehrpunkt, 583
 Auswertung IPO-Takt, 584
 Nächste Teilzustellung, 584
 Synchronaktionen, 581
 Zuordnung von Pendel- und Zustellachse, 580
 Zustellbewegung, 582
 Zustellung im Umkehrbereich, 582
 Zustellungen festlegen, 580
 Synchronlauf
 fein, 556
 grob, 556
 Synchronspindel
 Kopplung, 553
 -paar festlegen, 559
 -Übersetzungsverhältnis (ÜFS / ÜLS), 555
 SYNRR, 25
 SYNRRW, 25
 SYNW, 25
 System
 -abhängige Verfügbarkeit, 5
 Systemframes, 317
 Systemvariablen
 Messtasterbegrenzung, 285
 Messtasterstatus, 285
- T**
- TAN, 73
 TANG, 463
 TANGDEL, 463

Tangentialsteuerung, 463
TANGOF, 463
TANGON, 463
TCARR, 452
TCOABS, 452
TCOFR, 452
TCOFRX, 452
TCOFRY, 452
TCOFRZ, 452
Teilstrecken - automatische Wegaufteilung, 590
THETA
 bei Drehung der Werkzeugorientierung, 359
 bei Interpolation der Werkzeugdrehung, 365
Tieflochbohren - CYCLE83
 extern programmieren, 668
TILT, 338
TLIFT, 463
TMOF, 597
TMON, 597
TOFFOF, 455
TOFFON, 455
Toleranz
 bei G0, 512
TOWER, 88
TOUPPER, 88
TOWBCS, 415
TOWKCS, 415
TOWMCS, 415
TOWSTD, 415
TOWTCS, 415
TOWWCS, 415
TRAANG, 385
TRACON, 401
TRACYL, 377
TRAFOOF, 400
TRAILOF, 515
TRAILON, 515
Transformation mit schwenkbarer Linearachse, 333
Transformationen
 Drei- Vier- und Fünf- Achstransformation, 334
 Kinematikunabhängige Grundstellung der
 Werkzeugorientierung, 324
 Kinematische Transformationen, 325
 Orientierungstransformation, 324
 verkettete, 401
 Verkettete Transformationen, 325
Transformationsarten
 Allgemeine Funktion, 323
TRANSMIT, 373
TRAORI, 334
Trigger-Ereignis
 beim Messen, 282

TRUE, 25
TRUNC, 73
TU, 390

U

ULI, 37
Umfangsfräsen
 mit Begrenzungsflächen, 433
Umkehr
 -punkt, 577
Umschaltbare Geometrieachsen, 602
Unterprogramm
 Anwendung, 169
 -aufruf indirekt, 206
 -aufruf mit Parameterübergabe, 201
 -aufruf modal, 204
 -aufruf ohne Parameterübergabe, 199
 -name, 170
 Programmierbarer Suchpfad, 211
 -rücksprung parametrierbar, 193
 -wiederholung, 203
Unterprogrammaufruf mit Pfadangabe und
Parametern, 210
UNTIL, 122
UPATH, 271

V

VAR, 178
Variable
 Typenkonvertierung, 82
Variablen
 anwenderdefiniert, 25
 Typkonvertierung, 84
VELOLIM, 506
VELOLIMA, 478
Verfügbarkeit
 System-abhängige, 5
Vergleichsoperatoren, 75
Verkettung
 von Strings, 86
Verschleißwert, 409
Voreilwinkel, 339
Vorlauf
 -speicher, 485

W

WAITC, 554
WAITE, 124

WAITENC, 608
 WAITM, 124
 Wegaufteilung, 594
 Wegaufteilung bei Bahnachsen, 593
 Werkstück
 -Hauptverzeichnis, 226
 -verzeichnisse, 226
 -zähler, 622
 Werkzeug
 -längenkorrektur, 452
 -orientierung, 338, 437
 -orientierung bei Framewechsel, 454
 -orientierung Glättung, 371
 -parameter, 403
 -radiuskorrektur, 411
 Werkzeugkorrektur
 Koordinatensystem für Verschleißwerte, 415
 Korrekturspeicher, 403
 Online-, 419
 Werkzeugkorrekturen
 additive, 407
 Werkzeugorientierung
 bahnrelativ, 362
 Werkzeugradiuskorrektur
 Eckenverzögerung, 288
 Werkzeugträger
 -kinematik, 447
 -Orientierbare, 452
 Werkzeugüberwachung
 schleifspezifische, 597
 WHEN, 569
 WHEN-DO, 581
 WHENEVER, 569
 WHENEVER-DO, 581
 WHILE, 121
 WRITE, 151

X

xe ye ze, 355
 XH YH ZH, 355
 xi yi zi, 355
 XOR, 75

Z

Zählschleife, 119
 Zentrieren - CYCLE81
 extern programmieren, 665
 Zustell
 -achse, 578

