SIEMENS

SINUMERIK

SINUMERIK 840D sl/ 828D
工作准备部分

编程手册

适用于

控制系统
SINUMERIK 840D sl / 840DE sl / 828D

数控软件　版本 4.8 SP3

08/2018
6FC5398-2BP40-6RA2

前言
基本安全说明 1
灵活的 NC 编程 2
文件和程序管理 3
保护区 4
特殊的位移指令 5
坐标转换（框架） 6
转换 7
运动链 8
含运动链的防撞 9
含运动链的转换 10
刀具补偿 11
轨迹特性 12
轴耦合 13
同步动作 14
摆动 15
冲裁和步冲 16
磨削 17
其它功能 18
自有切割程序 19
外部循环编程 20
表 21
附录 A

Промэлектроника, sales@prom-elec.com

法律资讯

警告提示系统

为了您的人身安全以及避免财产损失，必须注意本手册中的提示。人身安全的提示用一个警告三角表示，仅与财产损失有关的提示不带警告三角。警告提示根据危险等级由高到低如下表示。

⚠ 危险
表示如果不采取相应的小心措施，**将会**导致死亡或者严重的人身伤害。

⚠ 警告
表示如果不采取相应的小心措施，**可能**导致死亡或者严重的人身伤害。

⚠ 小心
表示如果不采取相应的小心措施，可能导致轻微的人身伤害。

注意
表示如果不采取相应的小心措施，可能导致财产损失。

当出现多个危险等级的情况下，每次总是使用最高等级的警告提示。如果在某个警告提示中带有警告可能导致人身伤害的警告三角，则可能在该警告提示中另外还附带有可能导致财产损失的警告。

合格的专业人员

本文件所属的产品/系统只允许由符合各项工作要求的**合格人员**进行操作。其操作必须遵照各自附带的文件说明，特别是其中的安全及警告提示。 由于具备相关培训及经验，合格人员可以察觉本产品/系统的风险，并避免可能的危险。

按规定使用 Siemens 产品

请注意下列说明：

⚠ 警告
Siemens 产品只允许用于目录和相关技术文件中规定的使用情况。如果要使用其他公司的产品和组件，必须得到 Siemens 推荐和允许。正确的运输、储存、组装、装配、安装、调试、操作和维护是产品安全、正常运行的前提。必须保证允许的环境条件。必须注意相关文件中的提示。

商标

所有带有标记符号 ® 的都是 Siemens AG 的注册商标。本印刷品中的其他符号可能是一些其他商标。若第三方出于自身目的使用这些商标，将侵害其所有者的权利。

责任免除

我们已对印刷品中所述内容与硬件和软件的一致性作过检查。然而不排除存在偏差的可能性，因此我们不保证印刷品中所述内容与硬件和软件完全一致。印刷品中的数据都按规定经过检测，必要的修正值包含在下一版本中。

Siemens AG
Division Digital Factory
Postfach 48 48
90026 NÜRNBERG
德国

文件订购号: 6FC5398-2BP40-6RA2
℗ 08/2018 本公司保留更改的权利

Copyright © Siemens AG 1995 - 2018.
保留所有权利

前言

SINUMERIK 文献

SINUMERIK 文档分为以下几个类别：

- 通用文档/产品样本
- 用户文档
- 制造商/服务文档

其它信息

访问下面的网址 (https://support.industry.siemens.com/cs/de/en/view/108464614)获取有关该主题的信息：

- 订购文档/查看印刷品一览
- 进入下载文档的链接
- 使用在线文档（查找搜索手册/信息）

如果您对技术文档有疑问（例如建议、修改），请发送一份电子邮件到以下地址 (mailto:docu.motioncontrol@siemens.com)。

mySupport/文档

您可以访问下面的网址 (https://support.industry.siemens.com/My/cn/zh/documentation)，了解如何随意组合西门子文档内容，再结合机器，创建自己的机器文档。

培训

通过以下地址 (http://www.siemens.com/sitrain)可获取有关 SITRAIN 的信息 - 西门子为驱动和自动化产品、系统和解决方案制定的培训。

常见问题

常见问题（FAQ）请参见产品支持 (https://support.industry.siemens.com/cs/de/en/ps/faq)下的服务&支持页面。

SINUMERIK

有关 SINUMERIK 的信息请访问以下网址 (http://www.siemens.com/sinumerik)。

目标客户

该手册供以下人员使用：

- 编程人员
- 设计人员

使用

利用该编程手册目标用户可以设计程序和软件界面、写入、测试和消除故障。

标准功能范畴

在该编程手册中描述了标准的功能范畴。 机床制造商增添或者更改的功能，由机床制造商资料进行说明。

控制系统有可能执行本文献中未描述的某些功能。 但是这并不意味着在提供系统时必须带有这些功能，或者为其提供有关的维修服务。

同样，因为只是概要，所以该文献不包括全部类型产品的所有详细信息，也无法考虑到安装、运行和维修中可能出现的各种情况。

技术支持

访问网址 (https://support.industry.siemens.com/sc/cn/zh/sc/-/oid2090)中的“联系”，您便可以获取各个国家技术支持的电话号码。

结构与内容的相关信息

编程手册之基本原理/工作准备分册

关于 NC 编程的说明分列在两本手册中：

1. **基本原理**
   编程手册“基本原理”供机床专业操作供使用，需要有相应的钻削、铣削和车削加工知识。 这里也利用一些简单的编程举例，说明常见的指令和语句（符合 DIN66025）。
2. **工作准备部分**
   编程手册“工作准备部分”供熟悉所有编程方法的工艺人员使用。 SINUMERIK 控制系统可利用一种专用编程语言对复杂的工件程序（例如自由成形曲面，通道坐标，.....）进行编程，并且可减轻工艺人员编程的负担。

NC 语言的可用性

本手册中所描述的全部 NC 语言都可用于 SINUMERIK 840D sl。有关 SINUMERIK 828D 的可用性见表格“指令：在 SINUMERIK 828D 上的可用性 (页 961)”。

目录

前言 3

1 基本安全说明 19

1.1 一般安全说明 19

1.2 应用示例的质保规定 20

1.3 工业安全 21

2 灵活的 NC 编程 23

2.1 变量 23
2.1.1 系统数据 23
2.1.2 预定义用户变量：计算参数 26
2.1.2.1 通道专用的计算参数（R） 26
2.1.2.2 全球计算参数（RG） 27
2.1.3 预定义用户变量：链接变量 29
2.1.4 定义用户变量（DEF） 31
2.1.5 重新定义系统变量、用户变量和 NC 语言指令 (REDEF) 38
2.1.6 属性：初始化值 42
2.1.7 属性： 极限值(LLI, ULI) 45
2.1.8 属性： 物理单位(PHU) 46
2.1.9 属性： 存取权限(APR, APW, APRP, APWP, APRB, APWB) 49
2.1.10 属性： 数据级（DCM、DCI、DCU）- 只适用于 SINUMERIK 828D 54
2.1.11 可定义和可重新定义的属性一览 55
2.1.12 定义和初始化数组变量（DEF，SET，REP） 56
2.1.13 定义和初始化数组变量（DEF，SET，REP）： 其它信息 61
2.1.14 数据类型 63
2.1.15 检查变量的存在性（ISVAR） 64
2.1.16 读取属性值/数据类型（GETVARPHU、GETVARAP、GETVARLIM、GETVARDIM、GETVARDFT、GETVARTYP） 65

2.2 间接编程 72
2.2.1 间接编程地址 72
2.2.2 间接编程 G 代码 74
2.2.3 间接编程位置属性（GP） 75
2.2.4 间接编程零件程序行（EXECSTRING） 78

2.3 运算功能 80

2.4 比较运算和逻辑运算 83

2.5 比较错误的精确度修正 (TRUNC) 85

2.6 参见“最大变量、最小变量和变量区域(MINVAL, MAXVAL, BOUND)” 87

2.7 运算的优先级....89
2.8 可能有的类型转换....90
2.9 字符串运算....91
2.9.1 类型转换到字符串 (AXSTRING)....91
2.9.2 从 STRING (NUMBER, ISNUMBER, AXNAME) 类型转换....92
2.9.3 字符串的链接 (<<)....93
2.9.4 大小写字母转换 (TOLOWER, TOUPPER)....95
2.9.5 确定一个字符串的长度 (STRLEN)....95
2.9.6 在字符串中查找字符/字符串 (INDEX, RINDEX, MINDEX, MATCH)....96
2.9.7 部分字符串的选择 (SUBSTR)....97
2.9.8 单个字符的读取和写入....98
2.9.9 格式化字符串 (SPRINT)....100
2.10 程序跳转和分支....110
2.10.1 跳回到程序开始 (GOTOS)....110
2.10.2 程序跳转到跳转标记处 (GOTOB, GOTOF, GOTO, GOTOC)....111
2.10.3 程序分支(CASE ... OF ... DEFAULT ...)....114
2.11 程序部分重复 (REPEAT, REPEATB, ENDLABEL, P)....116
2.12 控制结构....122
2.12.1 条件指令和分支（IF, ELSE, ENDIF）....123
2.12.2 无限程序循环（LOOP，ENDLOOP）....125
2.12.3 计数循环（FOR ... TO ...，ENDFOR）....125
2.12.4 在循环开始处带有条件的程序循环（WHILE，ENDWHILE）....127
2.12.5 在循环结束处带有条件的程序循环（REPEAT，UNTIL）....128
2.12.6 带层叠控制结构的程序示例....128
2.13 协调指令 (INIT、START、WAITM、WAITMC、WAITE、SETM、CLEARM)....130
2.14 中断程序（ASUP）....136
2.14.1 中断程序的功能....136
2.14.2 建立中断程序....137
2.14.3 中断程序赋值和启动(SETINT, PRIO, BLSYNC)....138
2.14.4 取消/再激活一个中断程序的赋值（DISABLE，ENABLE）....140
2.14.5 删除中断程序的赋值（CLRINT）....140
2.14.6 快速离开工件轮廓（SETINT LIFTFAST，ALF）....141
2.14.7 快速离开工件轮廓时的运行方向....144
2.14.8 中断程序下的运动过程....147
2.15 交换轴，交换主轴 (RELEASE, GET, GETD)....148
2.16 将轴移交到另一个通道中（AXTOCHAN）....153
2.17 有效设置机床数据（NEWCONF）....155
2.18 写入文件（WRITE）....156
2.19 删除文件（DELETE）....161

2.20 读取文件中的行（READ）...163
2.21 检查文件的存在性(ISFILE)...166
2.22 读取文件信息(FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO)...168
2.23 取整 (ROUNDUP)...171
2.24 子程序...172
2.24.1 概述...172
2.24.1.1 子程序...172
2.24.1.2 子程序名称...173
2.24.1.3 子程序的嵌套...174
2.24.1.4 查找路径...175
2.24.1.5 形式参数和实际参数...175
2.24.1.6 参数传递...176
2.24.2 定义子程序...178
2.24.2.1 没有参数传递的子程序...178
2.24.2.2 子程序，带 Call-by-Value 值调用式参数传递(PROC)...179
2.24.2.3 子程序，带 Call-by-Reference 引用调用式参数传递(PROC, VAR)...180
2.24.2.4 保存模态 G 功能（SAVE）...183
2.24.2.5 抑制单程序段处理 (SBLOF, SBLON)...184
2.24.2.6 抑制当前的程序段显示(DISPLOF, DISPLON, ACTBLOCNO)...189
2.24.2.7 标记子程序 “准备”(PREPRO)...193
2.24.2.8 子程序返回指令 M17...193
2.24.2.9 子程序返回指令 RET...194
2.24.2.10 可设定的子程序跳回 (RET ...)...196
2.24.2.11 可设定的子程序跳回 (RETB ...)...202
2.24.3 子程序调用...206
2.24.3.1 没有参数传递的子程序调用...206
2.24.3.2 带参数传递的子程序调用(EXTERN)...208
2.24.3.3 程序重复次数(P)...210
2.24.3.4 模态子程序调用 (MCALL)...212
2.24.3.5 间接子程序调用(CALL)...214
2.24.3.6 指定待执行部分的间接子程序调用(CALL BLOCK ... TO ...)...215
2.24.3.7 间接调用某个以 ISO 语言编程的程序 (ISOCALL)...216
2.24.3.8 调用带有路径说明和参数的子程序 (PCALL)...217
2.24.3.9 扩展调用子程序时的路径查找 (CALLPATH)...217
2.24.3.10 执行外部子程序 (840D sl) (EXTCALL)...219
2.24.3.11 执行外部子程序 (828D) (EXTCALL)...223
2.25 宏指令技术 (DEFINE ... AS)...227
3 文件和程序管理...231
3.1 程序存储器...231
3.1.1 NC 中的程序存储器...231
3.1.2 外部程序存储器...234

3.1.3 程序存储器文件的定址 ... 235
3.1.4 子程序调用时的查找路径 ... 240
3.1.5 查询路径和文件名 ... 242
3.2 工作存储器 (CHANDATA, COMPLETE, INITIAL) ... 244
4 保护区 ... 247
4.1 定义保护区 (CPROTDEF、NPROTDEF) ... 247
4.2 激活/取消激活保护区 (CPROT, NPROT) ... 252
4.3 检查保护区、工作区域限制和软件限位开关 (CALCPOSI) ... 256
5 特殊的位移指令 ... 269
5.1 逼近已经过编码处理的位置 (CAC, CIC, CDC, CACP, CACN) ... 269
5.2 样条插补 (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL) ... 271
5.3 样条组合(SPLINEPATH) ... 283
5.4 激活/关闭 NC 程序段压缩 (COMPON, COMPCURV, COMPCAD, COMPSURF, COMPOF) ... 285
5.5 多项式插补 (POLY, POLYPATH) ... 286
5.6 可设置的轨迹基准 (SPATH, UPATH) ... 292
5.7 用接触式探头测量 (MEAS, MEAW) ... 294
5.8 轴测量 (MEASA、MEAWA、MEAC)（选件） ... 297
5.9 OEM 专用函数(OMA1 ... OMA5, OEMIPO1, OEMIPO2, G810 ... G829) ... 308
5.10 带有角部减速的进给减速 (FENDNORM, G62, G621) ... 309
5.11 可编程的运动结束条件 (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA) ... 310
6 坐标转换（框架） ... 313
6.1 通过框架变量转换坐标 ... 313
6.1.1 预定义框架变量 ($P_CHBFRAME, $P_IFRAME, $P_PFRAME, $P_ACTFRAME) ... 315
6.2 给框架赋值 ... 320
6.2.1 直接赋值（轴值，角度，尺寸） ... 320
6.2.2 读取和修改框架组件 (TR, FI, RT, SC, MI) ... 322
6.2.3 通过框架计算 ... 323
6.2.4 定义框架变量 (DEF FRAME) ... 325
6.3 粗位移和精位移 (CTRANS, CFINE) ... 326
6.4 外部零点偏移 ($AA_ETRANS) ... 328
6.5 参考点状态的实际值设置和损失（PRESETON） ... 330
6.6 参考点状态的实际值设置，无损失（PRESETONS） ... 332

6.7 从空间中的三个测量点计算框架 (MEAFRAME)..334
6.8 NCU 全局框架..338
6.8.1 通道专用框架 ($P_CHBFR, $P_UBFR)..339
6.8.2 在通道中有效的框架..339
7 转换..345
7.1 转换方式的一般编程..345
7.1.1 转换时的定向运动..347
7.1.2 定向转换 TRAORI 概述..351
7.2 三轴、四轴和五轴转换 (TRAORI)..353
7.2.1 万向切削头的一般关系..353
7.2.2 三轴、四轴和五轴转换 (TRAORI)..356
7.2.3 定向编程变量和初始位置（ORIRESET）..357
7.2.4 编程刀具定向 (A..., B..., C..., LEAD, TILT)..359
7.2.5 端面铣削 (A4, B4, C4, A5, B5, C5)..365
7.2.6 定向轴的关系 (ORIWKS, ORIMKS)..366
7.2.7 定位轴编程(ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2)..368
7.2.8 沿一个圆锥表面定向编程(ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO)..371
7.2.9 两个接触点的定向预设值(ORICURVE, PO[XH]=, PO[YH]=, PO[ZH]=)..374
7.3 定向多项式(PO[角度], PO[坐标])..377
7.4 刀具定向旋转(ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA)..379
7.5 与轨迹相对的定向..382
7.5.1 定向方式相对于轨迹..382
7.5.2 轨迹相关的刀具定向旋转（ORIPATH、ORIPATHS、旋转角）..383
7.5.3 轨迹相关的刀具旋转插补（ORIROTC，THETA）..385
7.5.4 平滑定向变化(ORIPATHS A8=, B8=, C8=)..387
7.6 定向压缩 (COMPON, COMPCURV, COMPCAD, COMPSURF)..389
7.7 激活/取消定向曲线的平滑 (ORISON, ORISOF)..392
7.8 运动变换..394
7.8.1 激活端面转换 (TRANSMIT)..394
7.8.2 激活柱面转换 (TRACYL)..394
7.8.3 激活可编程角度的斜角转换 (TRAANG)..397
7.8.4 在磨床上斜向切入 (G5, G7)..398
7.9 激活级联转换 (TRACON)..401
7.10 直角坐标 PTP 运动..403
7.10.1 激活/取消坐标 PTP 运动 (PTP、PTPG0、PTPWOC、CP)..403
7.10.2 给定铰接位置 (STAT)..404
7.10.3 给定轴交角符号 (TU)..408

7.10.4 示例 1：带 ROBX 转换的 6 轴机械手的 PTP 运动 411
7.10.5 示例 2：生成 5 轴转换时的 PTP 运动 412
7.10.6 示例 3：PTPG0 和 TRANSMIT 413
7.11 在选择一个转换时的边界条件 415
7.12 取消转换 (TRAFOOF) 416
8 运动链 417
8.1 删除组件 (DELOBJ) 417
8.2 通过名称确定下标 (NAMETOINT) 422
9 含运动链的防撞 425
9.1 防撞对检查（COLLPAIR） 426
9.2 要求对碰撞监测的机床模型进行重新计算（PROTA） 427
9.3 设置保护区状态（PROTS） 428
9.4 确定保护区间距（PROTD） 429
10 含运动链的转换 431
10.1 激活转换 (TRAFOON) 431
10.2 根据机床测量修改定向转换（CORRTRAFO） 433
11 刀具补偿 441
11.1 补偿存储器 441
11.2 附加补偿 445
11.2.1 选择附加补偿（DL） 445
11.2.2 确定磨损值和设置值（$TC_SCPxy[t,d], $TC_ECPxy[t,d]） 446
11.2.3 清除附加补偿（DELDL） 447
11.3 刀具补偿 - 特殊操作 449
11.3.1 刀具长度镜像 451
11.3.2 磨损量的符号赋值 451
11.3.3 激活的加工的坐标系 (TOWSTD/TOWMCS/TOWWCS/TOWBCS/TOWTCS/TOWKCS) 452
11.3.4 刀具长度和平面更换 455
11.4 在线刀具补偿 457
11.4.1 定义多项式函数(FCTDEF) 457
11.4.2 连续写入在线刀具补偿(PUTFTOCF) 459
11.4.3 不连续写入在线刀具补偿(PUTFTOC) 460
11.4.4 激活/撤销在线刀具补偿(FTOCON/FTOCOF) 460
11.5 3D 刀具半径补偿 462
11.5.1 选择用于 3D 圆周铣削的 3D 刀具半径补偿（CUT3DC、CUT3DCD、ISD） 462
11.5.2 选择用于 3D 端面铣削的 3D 刀具半径补偿（CUT3DF、CUT3DFS、CUT3DFF、CUT3DFD） 466

11.5.3 考虑分界面的 3D 圆周铣削（CUT3DCC、CUT3DCCD）...471
11.6 刀具定向(ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST)...477
11.7 任意 D 编号赋值，切削刃编号...483
11.7.1 任意 D 编号赋值，切削刃编号 (地址 CE)...483
11.7.2 任意 D 编号赋值： 检查 D 号码(CHKDNO)...483
11.7.3 任意 D 编号赋值： 重命名 D 编号(GETDNO, SETDNO)...484
11.7.4 任意 D 编号赋值： 求得预先给出 D 编号刀具的 T 编号（GETACTTD）...485
11.7.5 任意 D 编号赋值：设定无效的 D 编号 (DZERO)...485
11.8 刀架的运动关系...487
11.9 用于可定向刀架的刀具长度补偿(TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ)...493
11.10 在线式刀具长度补偿 (TOFFON, TOFFOF)...496
11.11 可旋转刀具的补偿数据修改...499
11.11.1 计算定向（ORISOLH）...499
11.11.2 激活可旋转刀具的补偿数据修改（CUTMOD、CUTMODK）...508
11.12 使用刀具环境工作...517
11.12.1 保存刀具环境 (TOOLENV)...517
11.12.2 删除刀具环境 (DELTOOLENV)...520
11.12.3 读取 T 号、D 号和 DL 号 (GETTENV)...521
11.12.4 读取保存的刀具环境的信息 ($P_TOOLENVN, ($P_TOOLENV)...522
11.12.5 读取刀具长度或刀具长度分量 (GETTCOR)...523
11.12.6 修改刀具分量 (SETTCOR)...530
11.13 读取刀具长度 L1、 L2、 L3 对坐标轴的指定关系 (LENTOAX)...545
12 轨迹特性...549
12.1 切向控制...549
12.1.1 定义耦合 (TANG)...549
12.1.2 激活中间程序段生成 (TLIFT)...551
12.1.3 激活耦合 (TANGON)...552
12.1.4 取消耦合 (TANGOF)...553
12.1.5 删除耦合 (TANGDEL)...554
12.2 进给曲线 (FNORM, FLIN, FCUB, FPO)...556
12.3 加速性能...561
12.3.1 加速模式（BRISK，BRISKA，SOFT，SOFTA，DRIVE，DRIVEA）...561
12.3.2 跟随轴时的加速影响(VELOLIMA、ACCLIMA、JERKLIMA）...563
12.3.3 激活工艺专用动态值（DYNNORM，DYNPOS，DYNROUGH，DYNSEMIFIN，DYNFINISH）...565
12.4 带预控制运行（FFWON, FFWOF）...567
12.5 可编程轮廓精度（CPRECON, CPRECOF）...568

12.6 带有缓存的程序运行过程 (STOPFIFO, STARTFIFO, FIFOCTRL, STOPRE) 570
12.7 定义停止延迟区 (DELAYFSTON, DELAYFSTOF)..... 573
12.8 阻止 SERUPRO 的程序位置 (IPTRLOCK, IPTRUNLOCK)..... 576
12.9 返回轮廓（REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMIBL, RMBBL, RMEBL, RMNBL） 578
12.10 对运动控制的影响..... 587
12.10.1 百分比式急冲修正 (JERKLIM)..... 587
12.10.2 百分比式速度修正 (VELOLIM)..... 588
12.10.3 JERKLIM 和 VELOLIM 的程序举例..... 590
12.11 轮廓公差/定向公差的编程 (CTOL、OTOL、ATOL)..... 591
12.12 耦合生效时的程序段切换特性（CPBC）..... 596
13 轴耦合..... 597
13.1 联动 (TRAILON, TRAILOF)..... 597
13.2 曲线图表 (CTAB)..... 602
13.2.1 定义曲线图表(CTABDEF, CATBEND)..... 602
13.2.2 检查曲线图表的存在性(CTABEXISTS)..... 608
13.2.3 删除曲线图表(CTABDEL)..... 609
13.2.4 禁止删除和覆盖曲线图表(CTABLOCK, CTABUNLOCK)..... 610
13.2.5 曲线图表： 确定图表属性(CTABID, CTABISLOCK, CTABMEMTYP, CTABPERIOD)..... 612
13.2.6 读取曲线图表值 (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX)..... 613
13.2.7 曲线图表： 检查资源使用率(CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL)..... 618
13.3 轴向引导值耦合 (LEADON, LEADOF)..... 620
13.4 电子齿轮箱 (EG)..... 626
13.4.1 定义电子齿轮 (EGDEF)..... 626
13.4.2 接通电子齿轮（EGON，EGONSYN，EGONSYNE）..... 627
13.4.3 关闭电子齿轮（EGOFS, EGOFC）..... 631
13.4.4 删除某个电子齿轮箱的定义（EGDEL）..... 631
13.4.5 旋转进给（G95）/电子齿轮箱（FPR）..... 632
13.5 同步主轴..... 633
13.5.1 同步主轴： 编程（COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC）..... 633
13.6 同类耦合 (CP...)..... 644
13.7 主/从耦合 (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS)..... 651

14 同步动作 ..655
14.1 定义同步动作 ..655
15 摆动 ..657
15.1 异步摆动(OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB) ..657
15.2 由同步动作控制的摆动(OSCILL) ..663
16 冲裁和步冲 ..671
16.1 激活/取消 ..671
16.1.1 激活或取消冲压和步冲 (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC) ..671
16.2 自动划分位移 ..677
16.2.1 轨迹轴位移划分 ..679
16.2.2 在单个轴时的位移划分 ..681
17 磨削 ..683
17.1 激活/取消磨削专用的刀具监控 (TMON, TMOF) ..683
18 其它功能 ..685
18.1 轴功能 (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) ..685
18.2 可转换的几何轴 (GEOAX) ..688
18.3 轴容器 (AXCTSWE， AXCTSWED，AXCTSWEC) ..693
18.4 等待有效的轴位置（WAITENC） ..695
18.5 可编程参数组切换（SCPARA） ..697
18.6 检查现有的 NC 语言范围（STRINGIS） ..699
18.7 交互式调用零件程序 (MMC) 窗口 ..703
18.8 程序执行时间/工件计数器 ..709
18.8.1 程序运行时间 ..709
18.8.2 工件计数器 ..713
18.9 Process DataShare - 数据输出到外部设备/文件上（EXTOPEN，WRITE，EXTCLOSE） ..715
18.10 报警（SETAL） ..720
18.11 扩展停止和退回（ESR） ..722
18.11.1 NC 控制的 ESR ..723
18.11.1.1 NC 控制的退回 (POLF, POLFA, POLFMASK, POLFMLIN) ..723
18.11.1.2 NC 控制的停止 ..727
18.11.2 驱动自控 ESR ..728
18.11.2.1 配置驱动自控的停止（ESRS） ..728
18.11.2.2 配置驱动自控的退回（ESRR） ..729

18.12 定义毛坯 (WORKPIECE)....731
18.13 切换语言模式 (G290, G291)....736
19 自有切割程序....739
19.1 用于切割的支持性功能....739
19.2 设置轮廓表（CONTPRON）....740
19.3 设置轮廓表（CONTDCON）....747
19.4 计算两个轮廓元素之间的交点（INTERSEC）。....751
19.5 逐段执行某个图表的轮廓元素（EXECTAB）....753
19.6 计算圆的数据 (CALCDAT)....754
19.7 断开轮廓预处理（EXECUTE）....756
20 外部循环编程....757
20.1 工艺循环....757
20.1.1 引言....757
20.1.2 工艺专用一览....758
20.1.3 HOLES1 – 成排孔....760
20.1.4 HOLES2 - 圆周孔....760
20.1.5 POCKET3 - 铣削矩形腔....763
20.1.6 POCKET4 - 铣削圆形腔....767
20.1.7 SLOT1- 纵向槽....771
20.1.8 SLOT2 - 圆弧槽....774
20.1.9 LONGHOLE - 长孔....777
20.1.10 CYCLE60 - 雕刻循环....780
20.1.11 CYCLE61- 平面铣削....783
20.1.12 CYCLE62 - 轮廓调用....786
20.1.13 CYCLE63 - 轮廓腔铣削....787
20.1.14 CYCLE64 - 预钻轮廓腔....791
20.1.15 CYCLE70 - 螺纹铣削....792
20.1.16 CYCLE72 - 轨迹铣削....795
20.1.17 CYCLE76 - 铣削矩形凸台....799
20.1.18 CYCLE77 - 铣削圆形凸台 802
20.1.19 CYCLE78 - 螺纹铣削....805
20.1.20 CYCLE79 - 多边形....808
20.1.21 CYCLE81 - 钻削，钻中心孔....811
20.1.22 CYCL82 - 钻削, 锪平面....812
20.1.23 CYCLE83 - 深孔钻削....815
20.1.24 CYCLE84 – 刚性攻丝....819
20.1.25 CYCLE85 - 铰孔....823
20.1.26 CYCLE86 - 镗孔....825
20.1.27 CYCLE92 - 切断....826
20.1.28 CYCLE95 - 切削轮廓....828

20.1.29 CYCLE98 - 螺纹链 ..830
20.1.30 CYCLE99 - 螺纹车削 ..836
20.1.31 CYCLE435 - 设置修整器坐标 ..841
20.1.32 CYCLE495 - 成型 ..842
20.1.33 CYCLE800 – 回转 ..844
20.1.34 CYCLE801 - 方阵/框架 ..849
20.1.35 CYCLE802 - 任意位置 ..851
20.1.36 CYCLE830 - 深孔钻削 2 ..855
20.1.37 CYCLE832 - 快速设定 ..861
20.1.38 CYCLE840 - 带补偿夹具的攻丝 ..865
20.1.39 CYCLE899 - 铣削敞开槽 ..869
20.1.40 CYCLE930 - 凹槽 ..872
20.1.41 CYCLE940 - 退刀槽 ..875
20.1.42 CYCLE951- 切削 ..879
20.1.43 CYCLE952 - 槽式车削 ..882
20.1.44 CYCLE4071 - 反向点处带进给的纵向磨削 ..889
20.1.45 CYCLE4072 - 反向点处带进给的纵向磨削以及中断信号 ..891
20.1.46 CYCLE4073 - 带连续进给的纵向磨削 ..894
20.1.47 CYCLE4074 - 带连续进给的纵向磨削以及中断信号 ..895
20.1.48 CYCLE4075 - 反向点处带进给的平面磨削 ..899
20.1.49 CYCLE4077 - 反向点处带进给的平面磨削以及中断信号 ..901
20.1.50 CYCLE4078 - 带连续进给的平面磨削 ..904
20.1.51 CYCLE4079 - 带间歇进给的平面磨削 ..906
20.1.52 GROUP_BEGIN - 程序块开始 ..909
20.1.53 GROUP_END - 程序块结束 ..910
20.1.54 GROUP_ADDEND - 附加运行结束 ..910
20.1.55 前提条件 ..911
20.1.55.1 工艺对话框内的工艺调整 ..911
20.2 测量循环 ..913
21 表 ..915
21.1 指令 ..915
21.2 指令：在 SINUMERIK 828D 上的可用性 ..961
21.2.1 控制系统类型 铣削/车削 ..961
21.2.2 控制系统类型 磨削 ..1000
21.3 HMI 上的当前语言 ..1038
A 附录 ..1041
A.1 缩略语列表 ..1041
A.2 资料概览 ..1052
词汇表 ..1053
索引 ..1075

基本安全说明

1

1.1 一般安全说明

⚠ 警告

未遵循安全说明和遗留风险可引发生命危险

忽视随附硬件文档中的安全说明和遗留风险会导致重伤或死亡。

- 遵守硬件文档中的安全说明。
- 进行风险评估时应考虑到遗留风险。

⚠ 警告

因参数设置错误或修改参数设置引起机器误操作

参数设置错误可导致机器出现误操作，从而导致人员重伤或死亡。

- 防止恶意访问参数设置。
- 采取适当措施（如驻停或急停）应答可能的误操作。

1.2 应用示例的质保规定

应用示例在组态和配置以及各种突发事件方面对设备没有强制约束力，无需一一遵循。应用示例不会提供客户专用的解决方案，仅在典型任务设置中提供保护。

用户自行负责上述产品的规范运行事宜。应用示例并没有解除您在应用、安装、运行和维护时确保安全环境的责任。

1.3 工业安全

说明

工业安全

西门子为其产品及解决方案提供工业安全功能，以支持工厂、系统、机器和网络的安全运行。

为防止设备、系统、机器和电网受到网络攻击，需执行一个全面的工业安全方案（及持续维护），以符合最新的技术标准。西门子的产品和解决方案只是此类方案的一个组成部分。

用户有防止未经授权访问其设备、系统、机器和电网的责任。系统、机器和组件只能连接至企业网络或互联网并采取相应的保护措施（如使用防火墙和网络分段）。

此外，还须注意西门子针对相应保护措施的建议。更多有关工业安全的信息，请访问：

工业安全 (http://www.siemens.com/industrialsecurity)

有鉴于此，西门子不断对产品和解决方案进行开发和完善。西门子强烈推荐进行更新，从而始终使用最新的产品版本。使用过时或不再支持的版本可能会增大网络攻击的风险。

为了能始终获取产品更新信息，请通过以下链接订阅西门子工业安全 RSS Feed：

工业安全 (http://www.siemens.com/industrialsecurity)

其它信息请上网查找：

工业安全功能选型手册 (https://support.industry.siemens.com/cs/cn/zh/view/108862708/en)

⚠ 警告

篡改软件会引起不安全的驱动状态

篡改软件（如：病毒、木马、蠕虫、恶意软件）可使设备处于不安全的运行状态，从而可能导致死亡、重伤和财产损失。

- 请使用最新版软件。
- 根据当前技术版本，将自动化组件和驱动组件整合至设备或机器的整体工业安全机制中。
- 在整体工业安全机制中要注意所有使用的产品。
- 采取相应的保护措施（如杀毒软件）防止移动存储设备中的文件受到恶意软件的破坏。
- 请激活变频器功能“专有技术保护”，以防止对驱动进行未经授权的改装。

灵活的 NC 编程 2

2.1 变量

通过使用系统数据和用户数据中的变量，特别是计算功能和控制结构的相关变量，可以使 NC 程序和循环的编写更为灵活。

- 系统数据
  系统数据中有在系统中预定义的变量。这些变量有一个预定义含义。主要供系统软件使用。用户可在 NC 程序和循环中读取和写入这些变量。示例：机床数据、设定数据、系统变量。
  尽管系统数据的含义是预先给定的，但用户可通过在某个范围内重新定义修改其属性。
  参见“重新定义系统变量、用户变量和 NC 语言指令 (REDEF) (页 38)”
- 用户数据
  用户数据中有由用户定义的变量，其含义只可由用户确定。系统不会分析这些变量。
  用户数据划分为：
  - 预定义用户变量
    预定义用户变量是在系统中已经定义的变量，但还需通过机床数据对其数量进行设置。这些变量的属性可由用户进行调整。
    参见“重新定义系统变量、用户变量和 NC 语言指令 (REDEF) (页 38)”。
  - 用户定义变量
    用户定义变量是仅由用户定义的变量，直到运行时系统才会创建这些变量。它们的数量，数据类型，可见性和所有其它属性都完全由用户定义。
    参见“定义用户变量（DEF） (页 31)”

2.1.1 系统数据

系统数据包含在系统中预定义的变量，通过此变量可在 NC 程序与循环中存取当前控制系统的参数，例如机床、控制系统和加工步骤状态。

预处理变量

预处理变量是在预处理中，即在执行编程了变量的程序段时，读取和写入的系统变量。预处理变量不会触发预处理停止。

主处理变量

主处理变量是在主处理中，即在执行编程了变量的程序段时，读取和写入的系统变量。主处理变量有：

- 可在同步动作中编程的变量（读/写）
- 可在 NC 程序中编程、会触发预处理停止的变量（读/写）
- 可在 NC 程序中编程、在预处理中计算，但是只有在主处理中才写入的变量（主处理同步：只写）

前缀系统

系统变量的一个显著特点是其名称通常包含一个前缀，该前缀由一个 $ 字符、一个或两个字母以及一条下划线构成。

$ + 第 1 个字母	**含义：数据类型**
预处理数据（在预处理时读取/写入的系统数据）	
$M	机床数据 1)
$S	设定数据，保护区域 1)
$T	刀具管理参数
$P	程序数值
$C	ISO 包络循环的循环变量
$O	选项数据
R	R 参数（计算参数）2)
主处理数据（在主处理时读取/写入的系统数据）	
$$M	机床数据 1)
$$S	设定数据 1)
$A	当前主处理数据
$V	位置控制器数据
$R	R 参数（计算参数）2)
1)机床和设定数据被作为预处理变量还是主处理变量处理，取决于写入该数据时使用了一个还是两个 $ 符号。可针对应用自由选择写入方式。 2) 在零件程序/循环中使用 R 参数作为预处理变量时，不写入前缀，如 R10。在同步动作中用作主处理变量时，在前缀中写入一个 $ 字符，如 $R10。	

第 2 个字母	含义：变量显示
N	数控全局变量（NC）
C	通道专用变量（Channel）
A	轴专用变量（Axis）

边界条件

前缀系统特例

以下系统变量与上面说明的前缀系统有所不同：

- $TC_...：第 2 个字母 C 在这里表示的不是通道专用，而是刀架专用系统变量 (TC = Tool Carrier)
- $P_ ...：通道专用系统变量

在同步动作中使用机床数据和设定数据

在同步动作中使用机床数据和设定数据时，可通过前缀确定，机床数据或设定数据是和预处理同步还是和主处理同步地读取/写入。

如果数据在执行期间保持不变，则可以和预处理同步赌入，为此在机床数据或设定数据的前缀中写入一个 $ 字符：

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

如果数据在执行期间改变，则必须和主处理同步地读取/写入数据。为此在机床数据或设定数据的前缀中写入两个 $ 字符：

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

说明

写入机床数据和设定数据

在写入机床数据或设定数据时必须注意，在执行零件程序/循环时，生效的存取级允许写入操作，且数据的有效性为“IMMEDIATE”。

文档

全部系统变量的完整概览请参见：

参数手册 系统变量

参见

变量 (页 23)

2.1.2 预定义用户变量：计算参数

2.1.2.1 通道专用的计算参数（R）

通道专用计算参数或 R 参数是名称为 R 的预定义用户变量，定义为 REAL 数据类型的数组。由于历史原因，R 参数既可以带数组索引编写，如 `R[10]`，也可不带数组索引编写，如 `R10`。

在同步动作中使用计算参数时，必须写入 $ 字符作为前缀，如`$R10`。

句法

作为预处理变量使用时：

```
R<n>
R[<表达式>]
```

作为主运行变量使用时：

```
$R<n>
$R[<表达式>]
```

含义

R：	作为预处理变量使用时的名称，比如在零件程序中	
$R：	作为主运行变量使用时的名称，比如在同步动作中	
	类型：	REAL
	取值范围：	非指数的写入方式： ± (0.000 0001 ... 9999 9999) **提示：** 最多允许有 8 个小数位
		指数写入方式： ± ($1*10^{-300}$ ... $1*10^{+300}$) **提示：** ● 写入方式：<尾数>EX<指数>，如：8.2EX-3 ● 最多允许有 10 个字符（包括符号和小数点）。

<n>：	R 参数编号	
	类型：	INT
	取值范围：	0 - MAX_INDEX **提示** MAX_INDEX 由 R 参数中设置的数量得出： MAX_INDEX = (MD28050 $MN_MM_NUM_R_PARAM) - 1
<表达式>：	数组索引 只要可将表达式结果转换为数据类型 INT，则可设定任意表达式作为数组索引（INT，REAL，BOOL，CHAR）。	

示例

算术功能中 R 参数的赋值和应用：

程序代码	注释
R0=3.5678	; 在预处理中赋值
R[1]=-37.3	; 在预处理中赋值
R3=-7	; 在预处理中赋值
$R4=-0.1EX-5	; 在主处理中赋值：R4 = -0.1 * 10^-5
$R[6]=1.874EX8	; 在主处理中赋值：R6 = 1.874 * 10^8
R7=SIN(25.3)	; 在预处理中赋值
R[R2]=R10	; 通过 R 参数间接定址
R[(R1+R2)*R3]=5	; 通过算术表达式间接定址
X=(R1+R2)	; 将 X 轴运行至由 R1 和 R2 的和确定的位置
Z=SQRT(R1*R1+R2*R2)	; 将 Z 轴运行至通过 (R1^2 + R2^2) 的平方根确定的位置

参见

变量 (页 23)

2.1.2.2 全球计算参数（RG）

功能

除了通道专用的 R 参数以外，用户还有全球 R 参数可用。这些参数曾存在于控制系统中，并可从所有通道读/写。

例如全球 R 参数用于从一个通道获取信息至另一个通道。另一个示例是评估所有通道的全球设置，例如主轴毛坯件的突出部分。

通过操作界面或准备工作阶段的 NC 程序读取和写入全球 R 参数。不可用于同步动作或技术循环。

说明

在读取和写入全球 R 参数时，通道之间**没有**同步。

由于读取和写入在准备工作阶段进行，因此未定义从一个通道到另一个通道写入值有效的时间点。

示例：

通道 1 的全球 R 参数回路作为回路计数器运行。通道 2 在全球 R 参数写入一个值，使通道 1 的回路中断。所有截止到该时间点的在通道 1 处于准备工作阶段的回路仍将被执行。回路数量未定义，主要取决于通道的载荷。

通道之间的同步必须由用户通过例如 WAIT 标记自行完成！

句法

写入 NC 程序

```
RG[<n>]=<值>
RG[<表达式>]=<值>
```

读取 NC 程序

```
R...=RG[<n>]
R...=RG[<表达式>]
```

含义

RG:	**全球** R 参数的 NC 地址的默认名称 **提示：** NC 地址的名称可通过 MD15800 $MN_R_PARAM_NCK_NAME 设置	
<n>：	全球 R 参数编号	
	类型：	INT
	取值范围：	0 ... MAX_INDEX **注** MAX_INDEX 由全球 R 参数中设置的数量得出： MAX_INDEX = (MD18156 $MN_MM_NUM_R_PARAM_NCK) - 1

<表达式>:	只要可将表达式结果转换为数据类型 INT，则可设定任意表达式作为数组索引（INT，REAL，BOOL，CHAR）。	
<值>:	全球 R 参数值	
	类型：	REAL
	取值范围：	非指数的写入方式： ± (0.000 0001 ... 9999 9999) **提示：** 最多允许有 8 个小数位
		指数写入方式： ± ($1*10^{-300}$ ... $1*10^{+300}$) **提示：** ● 写入方式：<尾数>EX<指数>，例如：8.2EX-3 ● 最多允许有 10 个字符（包括符号和小数点）。

2.1.3 预定义用户变量：链接变量

通过链接变量，可在“NCU 链接”功能的范围内循环交换一个网络中相连的 NCU 之间的数据。 此时，可以访问链接变量存储器中特定格式的数据。 用户/机床制造商确定设备专用链接变量存储器时，既须考虑大小，也须考虑数据结构。

链接变量为系统全局用户变量，在设置了链接连接时这些变量能够从所有链接组的 NCU 中读取或写入到零件程序段和循环。 与全局用户数据（GUD）不同，链接变量也可用在同步动作中使用。

在无有效 NCU 链接的设备上，除了全局用户变量（GUD）外，可将链接变量作为控制系统本地全局用户变量使用。

句法

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

含义

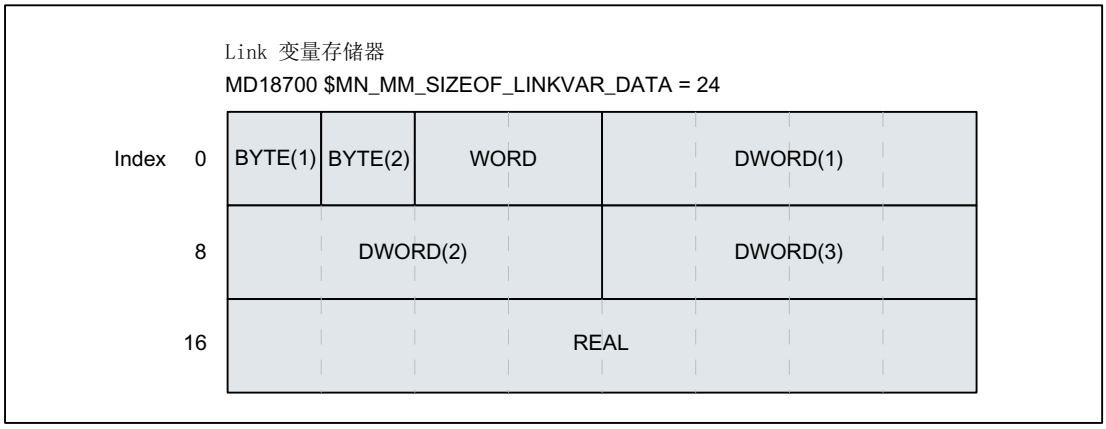
$A_DLB:	数据格式 BYTE（1 字节）的链接变量	
	数据类型：	UINT
	取值范围：	0 ... 255
$A_DLW:	数据格式 WORD（2 字节）的链接变量	
	数据类型：	INT
	取值范围：	-32768 ... 32767
$A_DLD:	数据格式 DWORD（4 字节）的链接变量	
	数据类型：	INT
	取值范围：	-2147483648 ... 2147483647
$A_DLR:	数据格式 REAL（8 字节）的链接变量	
	数据类型：	REAL
	取值范围：	±(2,2*10^{-308} … 1,8*10^{+308})
<索引>:	地址索引以字节，从链接变量存储器开始处计算	
	数据类型：	INT
	取值范围：	0 - MAX_INDEX **提示** • MAX_INDEX 由参数设置的链接变量存储器大小得出： MAX_INDEX = (MD18700 $MN_MM_SIZEOF_LINKVAR_DATA) - 1 • 只可对索引进行编程，从而可以使链接变量存储器中定址的字节位于数据格式限制内 ⇒ 索引= n * 字节，其中 n = 0，1，2，... – $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，...

示例

在自动化设备中有 2 个 NCU（NCU1 和 NCU2）。 在 NCU1 上连接了加工轴 AX2，该轴作为 NCU2 的链接轴运行。

NCU1 将轴 AX2 的电流实际值（$VA_CURR）循环写入链接变量存储器。 NCU2 循环读取通过链接通讯传输的电流实际值，并在超出限值时显示报警 61000。

链接变量中的数据结构在下图中显示。 电流实际值以 REAL 值传输。



NCU1

NCU1 在静态同步动作的 IPO 周期中将轴 AX2 的电流实际值通过链接变量 $A_DLR[16] 循环写入链接变量存储器。

程序代码

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

NCU2

NCU2 在静态同步动作的 IPO 周期中通过链接变量 $A_DLR[16] 循环从链接变量存储器读取轴 AX2 的电流实际值。 如果电流实际值大于 23.0 A，则显示报警 61000。

程序代码

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

参见

变量 (页 23)

2.1.4 定义用户变量（DEF）

通过指令 DEF 可以定义用户专用变量或用户变量 (User Data) 并赋值。

根据变量的有效范围，即变量可见范围，用户变量可分为以下几个类别：

- 局部用户变量（LUD）
  局部用户变量 (LUD) 是在执行时不是主程序的 NC 程序中定义的变量。此指令在调用 NC 程序时创建，并在程序结束复位或下一次启动控制系统时删除。只能在定义 LUD 的 NC 程序中存取该 LUD。
- 程序全局用户变量（PUD）
  程序全局用户变量 (PUD) 是在作为主程序的 NC 程序中定义的变量。此指令在调用 NC 程序时创建，并在程序结束复位或下一次启动控制系统时删除。可在主程序及所有子程序中存取 PUD。

  说明

  程序全局用户变量的可用性 (PUD)

  当设置了以下机床数据时，在主程序中定义的程序全局用户变量 (PUD) 同样在子程序中可用。

  MD11120 $MN_LUD_EXTENDED_SCOPE = 1

  设置 MD11120 = 0 时，在主程序中定义的程序全局用户变量只在主程序中可用。
- 全局用户变量（GUD）
  全局用户变量 (GUD) 是在数据块 (SGUD、MGUD、UGUD、GUD4 ... GUD9) 中定义的 NC 或通道全局变量，此变量在程序结束复位或下一次启动控制系统后依然保留。在所有 NC 程序中都可访问 GUD。

在使用（读/写）用户变量前必须对其进行定义必须遵循以下规则：

- GUD 必须在定义文件如 _N_DEF_DIR/_N_UGUD_DEF 中定义 。
- PUD 和 LUD 必须在 NC 程序的定义段中定义。
- 必须在单独的程序段中进行数据定义。
- 每次数据定义只能使用一种数据类型。
- 每次数据定义可以定义多个相同数据类型的变量。

句法

LUD 和 PUD

```
DEF <类型> <物理单位> <限值> <名称>[<值_1>, <值_2>, <值_3>]=<初始值>
```

GUD

```
DEF <范围> <预处理停止> <存取权限> <数据级> <类型> <物理单位> <限值> <名称>[<值_1>, <值_2>, <值_3>]=<初始值>
```

含义

DEF:	用于定义用户变量 GUD，PUD，LUD 的指令	
<范围>：	有效范围，只和 GUD 相关：	
	NC:	NC 全局用户变量
	CHAN:	通道全局用户变量
<预处理停止>：	预处理停止，只 GUD 相关（可选）	
	SYNR:	在读时进刀停止
	SYNW:	在写入时进刀停止
	SYNRW:	在读取/写入时执行预处理停止
<存取权限>：	通过 NC 程序或 BTSS 读取/写入 GUD 的保护等级（可选）	
	APRP **<保护等级>**:	读取：NC 程序
	APWP **<保护等级>**:	写入：NC 程序
	APRB **<保护等级>**:	读取：BTSS
	APWB **<保护等级>**:	写入：BTSS
	<保护等级>:	取值范围：0 ... 7
	参见“属性： 存取权限(APR, APW, APRP, APWP, APRB, APWB) (页 49)”	
<数据级>：	数据级分配（仅 SINUMERIK 828D!）	
	DCM:	数据级 M（= Manufacturer）
	DCI:	数据级 I（= Individual）
	DCU:	数据级 U（= User）
	参见“属性： 数据级（DCM、DCI、DCU）- 只适用于 SINUMERIK 828D (页 54)”。	

<类型>：	数据类型：	
	INT:	带正负号的整数值
	REAL:	实数（根据 IEEE 为 LONG REAL）
	BOOL:	真值 TRUE (1) / FALSE (0)
	CHAR:	ASCII-字符
	STRING[<最大长度>]：	定义长度的字符串
	AXIS:	进给轴/主轴标识符
	FRAME：	静态坐标转换的几何设定
	参见“数据类型 (页 63)”	
<物理单位>：	物理单位（可选）	
	PHU <单位>：	物理单位
	参见“属性： 物理单位(PHU) (页 46)”	
<限值>：	上限/下限（可选）	
	LLI <限值>：	下限（lower limit）
	ULI <限值>：	上限（upper limit）
	参见“属性： 极限值(LLI, ULI) (页 45)”	
<名称>：	变量名称 **提示** ● 最多 31 个字符 ● 前两个字符必须为一个字母和/或一条下划线。 ● “$”字符预留给系统变量，不可使用。	
[<值_1>, <值_2>, <值_3>]：	设定 1 维至 3 维（最大）数组变量的数组长度（可选） 数组变量初始化请见“定义和初始化数组变量（DEF，SET，REP） (页 56)”	
<初始化值>：	初始化值（可选） 参见“属性：初始化值 (页 42)” 数组变量初始化请见“定义和初始化数组变量（DEF，SET，REP） (页 56)”	

示例

示例 1：在机床制造商数据块中定义用户变量

程序代码	注释
%_N_MGUD_DEF	; GUD 模块：机床制造商
$PATH=/_N_DEF_DIR	
DEF CHAN REAL PHU 24 LLI 0 ULI 10 STROM_1, STROM_2	
; 说明	
; 两个 GUD 的定义：STROM_1, STROM_2	
; 有效范围：整个通道	
; 数据类型：REAL	
; VL-停止：未编程 => 缺省值 = 无预处理停止	
; 物理单位：24 = [A]	
; 极限值：下限 = 0.0，上限 = 10.0	
; 存取权限：未编程 => 缺省值 = 7 = 钥匙开关位置 0	
; 初始化值：未编程 => 缺省值 = 0.0	
DEF NCK REAL PHU 13 LLI 10 APWP 3 APRP 3 APWB 0 APRB 2 ZEIT_1=12, ZEIT_2=45	
; 说明	
; 两个 GUD 的定义：ZEIT_1, ZEIT_2	
; 有效范围：NC 全局	
; 数据类型：REAL	
; VL-停止：未编程 => 缺省值 = 无预处理停止	
; 物理单位：13 = [s]	
; 极限值：下限 = 10.0，上限 = 未编程 => 定义范围上限	
; 存取权限：	
; NC 程序：写入/读取 = 3 = 最终用户	
; BTSS:写入 = 0 = 西门子，读取 = 3 = 最终用户	
; 初始化值：ZEIT_1 = 12.0, ZEIT_2 = 45.0	
DEF NCK APWP 3 APRP 3 APWB 0 APRB 3 STRING[5] GUD5_NAME = "COUNTER"	
; 说明	
; 一个 GUD 的定义：GUD5_NAME	
; 有效范围：NC 全局	
; 数据类型：STRING，最大 5 个字符	
; VL-停止：未编程 => 缺省值 = 无预处理停止	
; 物理单位：未编程 => 缺省值 = 0 = 无物理单位	
; 极限值：未编程 => 定义范围限制：下限 = 0，上限 = 255	
; 存取权限：	
; NC 程序：写入/读取 = 3 = 最终用户	
; BTSS:写入 = 0 = 西门子，读取 = 3 = 最终用户	
; 初始化值：“计数器”	
M30	

示例 2：程序全局和局部用户变量（PUD / LUD）

程序代码	注释
PROC MAIN	;主程序
DEF INT VAR1	;PUD-定义
...	
SUB2	; 子程序调用
...	
M30	

程序代码	注释
PROC SUB2	;子程序 SUB2
DEF INT VAR2	;LUD-定义
...	
IF(VAR1==1)	;PUD 读取
VAR1=VAR1+1	;PUD 读取和写入
VAR2=1	;LUD 写入
ENDIF	
SUB3	; 子程序调用
...	
M17	

程序代码	注释
PROC SUB3	;子程序 SUB3
...	
IF(VAR1==1)	;PUD 读取
VAR1=VAR1+1	;PUD 读取和写入
VAR2=1	;错误:SUB2 中的 LUD 未知
ENDIF	
...	
M17	

示例 3：数据类型为 AXIS 的用户变量的定义和应用

程序代码	注释
DEF AXIS ABSZISSE	; 第 1 几何轴
DEF AXIS SPINDLE	; 主轴
...	
IF ISAXIS(1) == FALSE GOTOF WEITER	
ABSZISSE = $P_AXN1	
继续:	
...	
SPINDLE=(S1)	; 第 1 主轴

程序代码	注释
OVRA[SPINDLE]=80	;主轴倍率 = 80%
SPINDLE=(S3)	; 第 3 主轴

边界条件

全局用户变量（GUD）

在定义全局用户变量（GUD）时须考虑以下机床数据：

编号	名称：$MN_	含义
11140	GUD_AREA_ SAVE_TAB	GUD 模块的附加备份
18118 1)	MM_NUM_GUD_MODULES	当前主动文件系统中 GUD 文件的数量
18120 1)	MM_NUM_GUD_NAMES_NCK	全局 GUD 名称数量
18130 1)	MM_NUM_GUD_NAMES_CHAN	通道特定的 GUD 名称数量
18140 1)	MM_NUM_GUD_NAMES_AXIS	轴特定的 GUD 名称数量
18150 1)	MM_GUD_VALUES_MEM	全局 GUD 值的存储空间
18660 1)	MM_NUM_SYNACT_GUD_REAL	可设置的数据类型为 REAL 的 GUD 数量
18661 1)	MM_NUM_SYNACT_GUD_INT	可设置的数据类型为 INT 的 GUD 数量
18662 1)	MM_NUM_SYNACT_GUD_BOOL	可设置的数据类型为 BOOL 的 GUD 数量
18663 1)	MM_NUM_SYNACT_GUD_AXIS	可设置的数据类型为 AXIS 的 GUD 数量
18664 1)	MM_NUM_SYNACT_GUD_CHAR	可设置的数据类型为 CHAR 的 GUD 数量
18665 1)	MM_NUM_SYNACT_GUD_STRING	可设置的数据类型为 STRING 的 GUD 数量

1) MD 在 SINUMERIK 828D 上为只读数据！

数据类型为 AXIS 的 NC 全局变量的跨通道应用

当通道中的轴的通道轴编号相同时，在数据块定义时使用轴名称初始化的，数据类型为 AXIS 的 NC 全局用户变量才可在 NC 的不同通道中使用。

如果不是这种情况，必须在 NC 程序开始处载入变量，或者象下面的例子一样使用 AXNAME(...) 功能（参见：“轴功能 (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL) (页 685)”）。

程序代码	注释
DEF NCK STRING[5] ACHSE="X"	; 在数据块中定义
...	
N100 AX[AXNAME(ACHSE)]=111 G00	; 在 NC 程序中使用

2.1.5 重新定义系统变量、用户变量和 NC 语言指令 (REDEF)

通过指令 REDEF 可对系统变量、用户变量和 NC 语言指令的属性进行修改。重新定义的前提条件是，必须在相应的定义后进行。

在重新定义中不能同时对多个属性进行更改。必须为每个需要更改的属性编程单独的 REDEF 指令。

如果编程的多个属性更改之间有冲突，则最后进行的更改生效。

属性值复位

通过 REDEF 修改的存取权限和初始化时间属性可通过重新编程 REDEF（后面是变量名称或 NC 语言指令名称）复位至缺省值：

- 存取权限：保护等级 7
- 初始化时间：未初始化或保留当前值

可重定义属性

参见“可定义和可重新定义的属性一览 (页 55)”。

局部用户变量（PUD / LUD）

不能对局部用户变量（PUD / LUD）进行重新定义。

句法

```
REDEF <名称> <预处理停止>
REDEF <名称> <物理单位>
REDEF <名称> <限值>
REDEF <名称> <存取权限>
REDEF <名称> <初始化时间>
REDEF <名称> <初始化时间> <初始化值>
REDEF <名称> <数据级>
REDEF <名称>
```

含义

REDEF：	用于重新定义系统变量、用户变量和 NC 语言指令的特定属性或复位属性“存取权限”和/或“初始化时间”的指令	
<名称>：	已定义的变量或 NC 语言指令的名称	
<预处理停止>：	预处理停止	
	SYNR:	在读时进刀停止
	SYNW：	在写入时进刀停止
	SYNRW：	在读取/写入时执行预处理停止
<物理单位>：	物理单位	
	PHU **<**单位**>**：	物理单位
	参见“属性： 物理单位(PHU) (页 46)”。 **提示** 不可进行重新定义： ● 系统变量 ● 以下数据类型的全局用户数据(GUD)：BOOL，AXIS，STRING，FRAME	
<限值>：	上限/下限	
	LLI **<**限值**>**：	下限（lower limit）
	ULI **<**限值**>**：	上限（upper limit）
	参见“属性： 极限值(LLI, ULI) (页 45)”。 **提示** 不可进行重新定义： ● 系统变量 ● 以下数据类型的全局用户数据(GUD)：BOOL，AXIS，STRING，FRAME	

<存取权限>：	通过零件程序或 BTSS 读取/写入的权限	
	APX <保护等级>：	执行：NC 语言元素
	APRP <保护等级>：	读取：零件程序
	APWP <保护等级>：	写入：零件程序
	APRB <保护等级>：	读取：BTSS
	APWB <保护等级>：	写入：BTSS
	<保护等级>：	取值范围：0 ... 7
	参见“属性： 存取权限(APR, APW, APRP, APWP, APRB, APWB) (页 49)”。	
<初始化时间>：	变量重新初始化的时间	
	INIPO：	上电
	INIRE：	主程序结束，NC 复位或上电
	INICF：	NEWCONF 或主程序结束，NC 复位或上电
	PRLOC：	主程序结束，本地更改后 NC 复位或上电
	参见“属性：初始化值 (页 42)”。	
<初始化值>：	初始化值 在定义初始化值时，必须设定初始化时间（参见<初始化时间>）。 参见“属性：初始化值 (页 42)”。 数组变量初始化请见“定义和初始化数组变量（DEF，SET，REP） (页 56)”。 **提示** 除了设定数据外，系统变量不可重新定义。	
<数据级>：	数据级分配（仅 SINUMERIK 828D!）	
	DCM：	数据级 M（= Manufacturer）
	DCI：	数据级 I（= Individual）
	DCU：	数据级 U（= User）
	参见“属性： 数据级（DCM、DCI、DCU）- 只适用于 SINUMERIK 828D (页 54)”。	

示例

重新定义系用于机床制造商的数据块的系统变量 $TC_DPCx

程序代码

```
%_N_MGUD_DEF                                    ; GUD 模块：机床制造商
N100 REDEF $TC_DPC1 APWB 2 APWP 3
N200 REDEF $TC_DPC2 PHU 21
N300 REDEF $TC_DPC3 LLI 0 ULI 200
N400 REDEF $TC_DPC4 INIPO (100, 101, 102, 103)
N800 REDEF $TC_DPC1
N900 REDEF $TC_DPC4
M30
```

对于 N100：　写入访问权限：BTSS = 保护等级 2，零件程序 = 保护等级 3

对于 N200：　物理单位 [%]

对于 N300：　下限 = 0，上限 = 200

对于 N400：　在上电时使用四个值初始化数组变量

对于 N800 / N900：　；复位属性值“存取权限”和/或“初始化时间”

说明

使用 ACCESS 文件

在使用 ACCESS 文件时，必须将存取权限的重新定义从 _N_MGUD_DEF 转移到 _N_MACCESS_DEF。

边界条件

粒度

重定义总是针对所有通过名称明确标识的变量。不能在数组变量中为单个的数组单元分配不同的属性值。

2.1.6 属性：初始化值

用户变量的定义(DEF)

在进行定义时可为以下变量预设一个初始化值：

- 全局用户变量（GUD）
- 程序全局用户变量（PUD）
- 局部用户变量（LUD）

重新定义（REDEF）系统和用户变量

在进行重定义时可为以下变量预设一个初始化值：

- 系统数据
  - 设定数据
- 用户数据
  - R 参数
  - 同步动作变量（$AC_MARKER，$AC_PARAM，$AC_TIMER）
  - 同步动作 GUD (SYG_xy[ ], 其中 x=R, I, B, A, C, S；y=S, M, U, 4, ..., 9)
  - EPS 参数
  - 刀具数据 OEM
  - 刀库数据 OEM
  - 全局用户变量（GUD）

重新初始化时间

在进行重新定义时可设定变量重新初始化的时间，即重新设置为初始化值的时间：

- INIPO（上电）
  在上电时重新初始化变量。
- INIRE（复位）
  在 NC 复位，BAG 复位，零件程序结束（M02 / M30）或上电时重新初始化变量。
- INICF (NEWCONF)
  在通过 HMI、零件程序指令 NEWCONF 启用功能“激活机床数据”时，或者 NC 复位，BAG 复位，零件程序结束（M02 / M30）或上电时重新初始化变量。
- PRLOC：（程序局部更改）
  只有在当前零件程序范围内进行修改变量时，才可在 NC 复位，BAG 复位或零件程序结束（M02 / M30）时进行重新初始化。
  PRLOC 属性必须与可编程设定数据（见下表）一起使用。

表格 2-1　可编程的设定数据

序号	名称	G 代码 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)使用此 G 代码对设定数据进行响应		

边界条件

初始化值：全局用户变量（GUD）

- 对于有效范围为 NC 的全局用户变量（GUD），只有 INIPO（上电）能预设为初始化时间。
- 对于有效范围为 CHAN 的全局用户变量（GUD），除了 INIPO（上电）外，INIRE（复位）或 INICF（NEWCONF）也可以预设为初始化时间。
- 对于有效范围为 CHAN、初始化时间为 INIRE（复位）或 INICF（NEWCONF）的全局用户变量(GUD)，只有在通道中触发了列举的事件时，才会在 NC 复位，BAG 复位和“激活机床数据”时在该通道中重新初始化变量。

初始化值：数据类型 FRAME

不能为数据类型为 FRAME 的变量设定初始化值。数据类型为 FRAME 的变量总是通过缺省框架隐式初始化。

初始化值：数据类型 CHAR

对于数据类型为 CHAR 的变量，也可通过带引号的 ASCII 符号编程，比如“A”，而不用 ASCII 码（0...255）。

初始化值：字符串数据类型

对于数据类型为 STRING 的变量，必须使用带引号的字符串进行编程，例如：...= "MASCHINE_1"

初始化值：数据类型 AXIS

对于数据类型为 AXIS 的变量，必须在具有扩展地址时将轴名称编写在括号中，例如：...=(X3)

初始化值：系统变量

对于系统变量，可通过重新定义来指定非用户专用的初始化值。系统变量的初始化值由系统固定预设。但是通过重新定义可以更改系统变量重新初始化的时间（INIRE，INICF）。

隐式初始化值：数据类型 AXIS

对于数据类型为 AXIS 的变量，使用以下隐式初始化值：

- 系统数据：“第一几何轴”
- 同步 GUD (名称：SYG_A*), PUD, LUD:
  机床数据中的轴名称：MD20082 $MC_AXCONF_CHANAX_DEFAULT_NAME

隐式初始化值：刀具和刀库数据

对于刀具和刀库数据，可通过以下机床数据预设隐式初始化值：MD17520 $MN_TOOL_DEFAULT_DATA_MASK

说明

同步

用户/机床制造商应全权负责实现同步，即触发全局数据重新初始化的事件和在别处读取该变量之间的同步。

参见

变量 (页 23)

2.1.7 属性： 极限值(LLI, ULI)

只允许为以下类型的数据确定定义范围的上限值和下限值。

- INT
- REAL
- CHAR

用户变量的定义(DEF)： 极限值和隐式初始化值

在定义上述某个数据类型的一个用户变量时，如果没有定义显式初始化值，该变量会设为该数据类型的隐式初始化值。

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

如果隐式初始化值超出了编程极限值构成的定义范围，该变量会按照隐式初始化值临近的极限值进行初始化：

- 隐式初始化值 < 下限值(LLI)⇒
  初始化值 = 下限值
- 隐式初始化值 > 上限值(LLI)⇒
  初始化值 = 上限值

示例：

程序代码	注释
DEF REAL GUD1	; 下限值 = 定义范围极限 ; 上限值 = 定义范围极限 ; 未编程初始化值 ; => 隐式初始化值 = 0.0
DEF REAL LLI 5.0 GUD2	; 下限值 = 5.0 ; 上限值 = 定义范围极限 ; => 初始化值 = 5.0
DEF REAL ULI -5 GUD3	; 下限值 = 定义范围极限 ; 上限值 = -5.0 ; => 初始化值 = -5.0

用户变量的重新定义(DEF)： 极限值和当前实际值

如果在重新定义一个用户变量的极限值时，当前实际值超出了新的定义范围，系统会输出报警，而不接收该极限值。

说明

用户变量的重新定义(DEF)

在重新定义用户变量的极限值时，请注意以下值的修改应保持一致：

- 极限值
- 实际值
- 初始化值，在重新定义和由于 INIPO、INIRE 或 INICF 自动重新初始化时

参见

变量 (页 23)

2.1.8 属性： 物理单位(PHU)

只允许为以下类型的变量设定物理单位：

- INT
- REAL

可编程的物理单位(PHU)

物理单位作为定点数设定： PHU <单位>

可编程以下物理单位：

<单位>	含义	物理单位
0	没有物理单位	-
1	线性位置或角度位置 1)2)	[mm], [inch], [deg]
2	线性位置 2)	[mm], [inch]
3	角度位置	[deg]
4	线性速度或角速度 1)2)	[mm/min], [inch/min], [rpm]
5	线性速度 2)	[mm/min]
6	角速度	[rpm]
7	线性加速度或角加速度 1)2)	[m/s^2], [$inch/s^2$], [rev/s^2]
8	线性加速度 2)	[m/s^2], [$inch/s^2$]
9	角加速度	[rev/s^2]
10	线性急动度或角急动度 1)2)	[m/s^3], [$inch/s^3$], [rev/s^3]
11	线性急动度 2)	[m/s^3], [$inch/s^3$]
12	角度加速度变化	[rev/s^3]
13	时间	[s]
14	位置控制器增益	[16.667/s]
15	旋转进给率 2)	[mm/rev], [inch/rev]
16	温度补偿 1)2)	[mm], [inch]
18	力	[N]
19	质量	[kg]
20	惯性矩 3)	[kgm^2]
21	百分比	[%]
22	频率	[Hz]
23	电压	[V]
24	电流	[A]
25	温度	[°C]
26	角度	[deg]
27	KV	[1000/min]
28	线性位置或角度位置 3)	[mm], [inch], [deg]
29	切削速度 2	[m/min], [feet/min]
30	圆周速度 2)	[m/s], [feet/s]

<单位>	含义	物理单位
31	电阻	[Ohm]
32	电感	[mH]
33	转矩 3)	[Nm]
34	转矩常量 3)	[Nm/A]
35	电流控制器增益	[V/A]
36	转速控制器增益 3)	[Nm/(rad*s)]
37	转速	[rpm]
42	功率	[kW]
43	弱电流	[µA]
46	低转矩 3)	[µNm]
48	千分比	-
49	-	[Hz/s]
65	流量	[l/min]
66	压力	[bar]
67	体积 3)	[cm^3]
68	距离增益 3)	[mm/(V*min)]
69	力控制器距离增益	[N/V]
155	螺距 3)	[mm/rev], [inch/rev]
156	螺距改变 3)	[mm/rev / rev], [inch/rev / rev]

1)这些物理单位取决于轴的类型： 线性轴和回转轴

2)单位制转换

G70/G71（英制/公制）
在使用 G70/G71 进行了基本单位制（MD10240 $MN_SCALING_SYSTEM_IS_METRIC）转换后， 在写入/读取长度相关系统变量和用户变量时**不换算**其数值（实际值、缺省值和限值）

G700/G710（英制/公制）
在使用 G700/G710 进行了基本单位制（MD10240 $MN_SCALING_SYSTEM_IS_METRIC）转换后， 在写入/读取长度相关系统变量和用户变量时**换算**其数值（实际值、缺省值和限值）

3)这些变量**不**会自动换算为 NC 的当前单位制（公制或英制） 换算功能只能由用户或机床制造商实现。

说明

单位换算导致的平面溢出

对于带和长度相关的物理单位的所有用户变量(GUD / PUD / LUD)，其内部存储格式为公制数据。 如果在 NCK 主运行中（如同步运行）过多地使用这些变量，在转换单位制时可能会导致插补平面的运算时间溢出，即报警 4240。

说明

单位之间的兼容性

在变量应用中（如赋值、比较和计算等），系统不会检查附属单位是否兼容。 必要的换算只能由用户或机床制造商实现。

参见

变量 (页 23)

2.1.9 属性： 存取权限(APR, APW, APRP, APWP, APRB, APWB)

名称

存取属性的名称 AP... 由以下组成：

1. A：**A**ccess
2. P：**P**rotection
3. R / W：**R**ead / **W**rite
4. P / B：**P**rogram / **B**TSS

存取权限/保护等级

存取权限对应了以下在编程时给定的保护等级：

存取权限	保护等级
系统口令	0
机床制造商口令	1
服务口令	2
最终用户口令	3
钥匙开关位置 3	4
钥匙开关位置 2	5

存取权限	保护等级
钥匙开关位置 1	6
钥匙开关位置 0	7

用户数据的定义 (DEF)

可以定义以下数据的存取权限 (APR... / APW...)：

- 全局用户数据（GUD）

系统数据和用户数据的重新定义 (REDEF)

可以重新定义以下数据的存取权限 (APR... / APW...)：

- 系统数据
  - 机床数据

    说明

    机床数据读取权限的重新定义

    只能通过关键字 APR 一同为零件程序和 OPI 设置读取机床数据的保护等级。

    重新定义读取权限后，系统不再支持关键字 APRP 和 APRB 并会显示报警 12490 “未设置存取权限 APRP/APRB<保护等级>”。
  - 设定数据
  - 系统变量
  - 过程数据
  - 刀库数据
  - 刀具数据
- 用户数据
  - R 参数
  - 同步动作变量（$AC_MARKER，$AC_PARAM，$AC_TIMER）
  - 同步动作 GUD (SYG_xy[ ], 其中 x=R, I, B, A, C, S；y=S, M, U, 4, ..., 9)
  - EPS 参数
  - 刀具数据 OEM
  - 刀库数据 OEM
  - 全局用户变量（GUD）

    说明

    在重新定义时可以自由确定变量的存取权限，这些变量处于最低保护等级 7 和自有保护等级如 1（机床制造商）之间。

NC 语言指令的重新定义(REDEF)

可以重新定义以下 NC 语言指令的存取权限或执行权限(APX)：

- G 代码/行程条件
  文档
  编程手册之基本原理；章节：G 代码/行程条件
- 预定义功能
  文档
  编程手册之基本原理；章节：预定义功能
- 预定义子程序调用
  文档
  编程手册之基本原理；章节：预定义子程序调用
- 执行同步动作时的 DO 指令
- 循环的程序名称
  循环必须保存在某一个循环目录中并且含有一个 PROC 指令。

NC 程序和循环的存取权限 (APRP, APWP)

不同的存取权限会对 NC 程序或循环的存取产生以下影响：

- APRP 0 / APWP 0
  - 执行 NC 程序时必须输入系统口令
  - 循环必须保存在目录 _N_CST_DIR（系统）中
  - 为使用目录 _N_CST_DIR，必须在机床数据 MD11160 $MN_ACCESS_EXEC_CST 中将执行权限设为“系统”
- APRP 1 / APWP 1 或 APRP 2 / APWP 2
  - 执行 NC 程序时必须输入机床制造商口令或服务口令
  - 循环必须保存在目录 _N_CMA_DIR（机床制造商）或 _N_CST_DIR 中
  - 为使用目录 _N_CMA_DIR 或 _N_CST_DIR，必须在以下机床数据中将执行权限至少设为“机床制造商”： MD11161 $MN_ACCESS_EXEC_CMA 或 MD11160 $MN_ACCESS_EXEC_CST。

- APRP 3 / APWP 3
  - 执行 NC 程序时必须输入最终用户口令
  - 循环必须保存在目录 N_CUS_DIR（用户）、 __N_CMA_DIR 或 _N_CST_DIR 中
  - 为使用目录 _N_CUS_DIR、_N_CMA_DIR 或 _N_CST_DIR，必须在以下机床数据中将执行权限至少设为“最终用户”：MD11162 $MN_ACCESS_EXEC_CUS、MD11161 $MN_ACCESS_EXEC_CMA 或 MD11160 $MN_ACCESS_EXEC_CST。
- APRP 4...7 / APWP 4...7
  - 在执行 NC 程序时必须设为钥匙开关位置 3 ... 0。
  - 循环必须保存在目录 N_CUS_DIR、 __N_CMA_DIR 或 _N_CST_DIR 中
  - 为使用目录 _N_CUS_DIR、_N_CMA_DIR 或 _N_CST_DIR，必须在以下机床数据中将执行权限至少设为相应的钥匙开关位置：MD11162 $MN_ACCESS_EXEC_CUS、MD11161 $MN_ACCESS_EXEC_CMA 或 MD11160 $MN_ACCESS_EXEC_CST。

OPI 的存取权限(APRB, APWB)

存取权限指令(APRB, APWB)以相同的方式限制所有系统组件（HMI、PLC、外部计算机、EPS 服务等）上、通过操作面板接口对系统变量和用户变量的存取。

说明

HMI 本地存取权限

在修改系统数据的存取权限时必须注意，该权限必须和 HMI 装置上定义的存取权限一致。

存取属性 APR / APW

由于兼容性的原因，属性 APR 和 APW 隐式映射至属性 APRP / APRB 和 APWP / APWB：

- APR x ⇒ APRP x APRB x
- APW y ⇒ APWP y APWB y

通过 ACCESS 文件分配存取权限

如果使用 ACCES 文件分配存取权限，则只能在该 ACCESS 文件中重新定义系统/用户数据和 NC 语言指令的存取权限。全局用户数据除外(GUD)。对于此类数据的存取权限，必要时应在相应的定义文件 *_DEF 中继续重新定义。

为保证存取保护的一致性，设置执行权限的机床数据和设置相应目录存取保护的机床数据必须相匹配。

请按照以下几个基本步骤执行：

1. 创建需要的定义文件：
   - _N_DEF_DIR/_N_SACCESS_DEF
   - _N_DEF_DIR/_N_MACCESS_DEF
   - _N_DEF_DIR/_N_UACCESS_DEF
2. 将此定义文件的写权限设为重新定义所需的值：
   - MD11170 $MN_ACCESS_WRITE_SACCESS = <保护等级>
   - MD11171 $MN_ACCESS_WRITE_MACCESS = <保护等级>
   - MD11172 $MN_ACCESS_WRITE_UACCESS = <保护等级>
3. 如果需要在循环中访问受保护单元，必须修改循环目录 _N_CST_DIR、_N_CMA_DIR 和 _N_CST_DIR 的执行权限和写权限。
   执行权限
   - MD11160 $MN_ACCESS_EXEC_CST = <保护等级>
   - MD11161 $MN_ACCESS_EXEC_CMA = <保护等级>
   - MD11162 $MN_ACCESS_EXEC_CUS = <保护等级>

   写权限
   - MD11165 $MN_ACCESS_WRITE_CST = <保护等级>
   - MD11166 $MN_ACCESS_WRITE_CMA = <保护等级>
   - MD11167 MN_ACCESS_WRITE_CUS = <保护等级>

   执行权限必须至少设为所用单元的最高保护等级。
   写权限必须至少设为和执行权限相同的保护等级。
4. HMI 本地循环目录的写权限必须设为和 NC 本地循环目录相同的保护等级。
   文档
   操作手册

ACCESS 文件中的子程序调用

也可以在 ACCESS 文件中调用子程序（SPF 或 MPF 标识），以继续分级存取保护。此时，子程序会采用待调用 ACCESS 文件的执行权限。

说明

在 ACCESS 文件中只能重新定义存取权限。所有其他属性必须继续在相应的定义文件中写入或重新定义。

参见

变量 (页 23)

2.1.10 属性： 数据级（DCM、DCI、DCU）- 只适用于 SINUMERIK 828D

为了简化调试、批量调试以及机床和批量机床升级过程中的数据处理，会将所有的 NC 系统数据和用户数据划分为不同的数据级。

数据级	数据
S = 系统	由西门子提供的系统数据，如机床数据和设定数据、标准循环和测量循环、定义(SGUD)和宏(SMAC)等
M = Manufacturer（机床制造商）	批量机床专用的调试数据如制造商循环、定义(MGUD)和宏(MMAC)以及确定机床功能范围的机床数据。
I = Individual（机床专用）	机床专用的调试数据，如补偿数据和参考点偏移。
U = User（用户）	机床运行中产生的机床专用数据，如刀具数据、设定数据、零件程序、用户循环、定义(UGUD)和宏(UMAC)。

文献：
SINUMERIK 828D 调试手册 车削和铣削；章节：“数据级的导入与应用”

用户数据的定义(DEF)

通过定义了用户数据的文件或目录的数据级来隐含地确定数据的数据级。 数据的数据级无法更改。

但在定义(DEF)用户数据时，可为**数据数值**预定义一个自己的并与数据的数据级不同的数据级。

数值的数据级必须适用：

数值数据级的优先级 ≤ 数据本身数据级的优先级

示例：

对测量头进行说明的 GUD 定义应保存到 M (= Manufacturer)数据级中，因为其对于制造商循环的执行是必需的。 因机床与机床的测量头型号可能有区别，所有数据的数值要划分到 I (= Individual)数据级。

MGUD.DEF （数据级 M）

```
...
DEF CHAN DCI INT CALIPER
...
```

系统数据的重新定义(REDEF)

系统数据的数据级可通过重新定义(REDEF)进行更改。 重新定义必须在数据级为 S 或 M 的定义文件中进行。

使用 ACCESS 文件时，重新定义只能在 ACCESS 文件中进行。

机床数据、设定数据和选件数据以及系统变量的各自数据级另见：

- 参数手册之详细的机床数据描述，参数：“类”
- 参数手册之系统变量

2.1.11 可定义和可重新定义的属性一览

下表展示了各个数据类型以及相应的可以定义(DEF)的属性或重新定义(REDEF)的属性。

系统数据

数据类型	初始化值	极限值	物理单位	存取权限	数据级（仅适用于 828D）
机床数据	---	---	---	REDEF	REDEF
设定数据	REDEF	---	---	REDEF	---
FRAME 数据	---	---	---	REDEF	---
过程数据	---	---	---	REDEF	---
主轴螺距误差补偿 (EEC)	---	---	---	REDEF	---
垂度补偿(CEC)	---	---	---	REDEF	---
象限误差补偿(QEC)	---	---	---	REDEF	---
刀库数据	---	---	---	REDEF	---
刀具数据	---	---	---	REDEF	---
保护区	---	---	---	REDEF	---
可定向刀架	---	---	---	REDEF	---
运动链	---	---	---	REDEF	---
3D 保护区	---	---	---	REDEF	---
工作区域限制	---	---	---	REDEF	---

用户数据

数据类型	初始化值	极限值	物理单位	存取权限	数据级
R 参数	REDEF	REDEF	REDEF	REDEF	---
同步变量($AC_...)	REDEF	REDEF	REDEF	REDEF	---
同步 GUD (SYG_...)	REDEF	REDEF	REDEF	REDEF	---
EPS 参数	REDEF	REDEF	REDEF	REDEF	---
刀具数据 OEM	REDEF	REDEF	REDEF	REDEF	---
刀库数据 OEM	REDEF	REDEF	REDEF	REDEF	---
全局用户变量（GUD）	DEF / REDEF	DEF	DEF	DEF / REDEF	DEF / REDEF
本地用户变量(PUD / LUD)	DEF	DEF	DEF	---	---

参见

变量 (页 23)

2.1.12 定义和初始化数组变量（DEF，SET，REP）

一个用户变量可以定义为 1 维~3 维数组(Array)。

- 1 维： `DEF <数据类型> <变量名称>[<n>]`
- 2 维： `DEF <数据类型> <变量名称>[<n>，<m>]`
- 3 维： `DEF <数据类型> <变量名称>[<n>,<m>,<o>]`

说明

STRING 数据类型的用户变量可以最大定义为 2 维数组。

数据类型

用户变量可以定义为以下类型的数组： BOOL, CHAR, INT, REAL, STRING, AXIS, FRAME

数组元素的赋值

可以在以下时间为数组元素赋值：

- 数组定义时（初始化值）
- 在程序执行过程中

可以通过以下方法赋值：

- 显式指定一个数组元素
- 显式指定一个数组元素为起始元素并给出值列表 (SET)
- 显式指定一个数组元素为起始元素并给出值列表以及重复的频率 (REP)

说明

不能向 FRAME 数据类型的用户变量分配初始化值。

句法(DEF)

```
DEF <数据类型> <变量名称>[<n>,<m>,<o>]
DEF  STRING[<字符串长度>] <变量名称>[<n>,<m>]
```

句法(DEF...=SET...)

使用值列表：

- 定义时：
  DEF <数据类型> <变量名称>[<n>,<m>,<o>]=SET(<值 1>,<值 2>,...)
  相同地：
  DEF <数据类型> <变量名称>[<n>，<m>，<o>]=（<值 1>，<值 2>，...）

说明

在通过值列表进行初始化时，可以选择给定 SET。

- 赋值时：
  <变量名称>[<n>，<m>，<o>]=SET（<值 1>，<值 2>，...）

句法(DEF...=REP...)

使用重复值

- 定义时：
  DEF<数据类型> <变量名称>[<n>，<m>，<o>]=REP（<值>）
  DEF<数据类型> <变量名称>[<n>，<m>，<o>]=REP（<值>,<数量_数组元素>）
- 赋值时：
  <变量名称>[<n>，<m>，<o>]=REP（<值>）
  <变量名称>[<n>，<m>，<o>]=REP（<值>，<数量_数组元素>）

含义

DEF:	变量定义指令	
<数据类型>:	变量数据类型	
	取值范围： • 对于系统变量： BOOL, CHAR, INT, REAL, STRING, AXIS • 对于 GUD 或 LUD 变量： BOOL, CHAR, INT, REAL, STRING, AXIS, FRAME	
<字符串长度>	STRING 数据类型下允许的最大字符数	
<变量名称>:	变量名称	
[<n>,<m>,<o>]:	数组长度或数组索引	
<n>:	1 维的数组长度或数组索引	
	类型：	INT（对于系统变量也为 AXIS）
	取值范围：	最大数组长度：65535 数组索引： 0 ≤ n ≤ 65534
<m>:	2 维的数组长度或数组索引	
	类型：	INT（对于系统变量也为 AXIS）
	取值范围：	最大数组长度：65535 数组索引： 0 ≤ m ≤ 65534
<o>:	3 维的数组长度或数组索引	
	类型：	INT（对于系统变量也为 AXIS）
	取值范围：	最大数组长度：65535 数组索引： 0 ≤ o ≤ 65534
SET:	通过给出的值列表赋值	
(<值 1>,<值 2>,...):	值列表	
REP:	通过给出的<值>赋值	

<值>：	数组元素在带 REP 的初始化时具有的数值。
<数量_数组元素>：	使用给定<值>的数组元素的数量。 其他的数组元素取决于不同时间点： ● 数组定义时初始化： → 剩下的数组元素赋值为零 ● 在程序运行过程中赋值： → 数组元素的当前值保持不变。 如果没有编程该参数，所有的数组元素都会分配到<值>。 如果参数为零，则取决于不同的时间点： ● 数组定义时初始化： → 所有元素预定为零 ● 在程序运行过程中赋值： → 数组元素的当前值保持不变。

数组索引

在如 SET 或 REP 的赋值中，通过数组索引从右向左的循环构成数组元素的隐式顺序。

示例： 3 维数组的初始化，数组具有 24 个元素：

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

相应地：

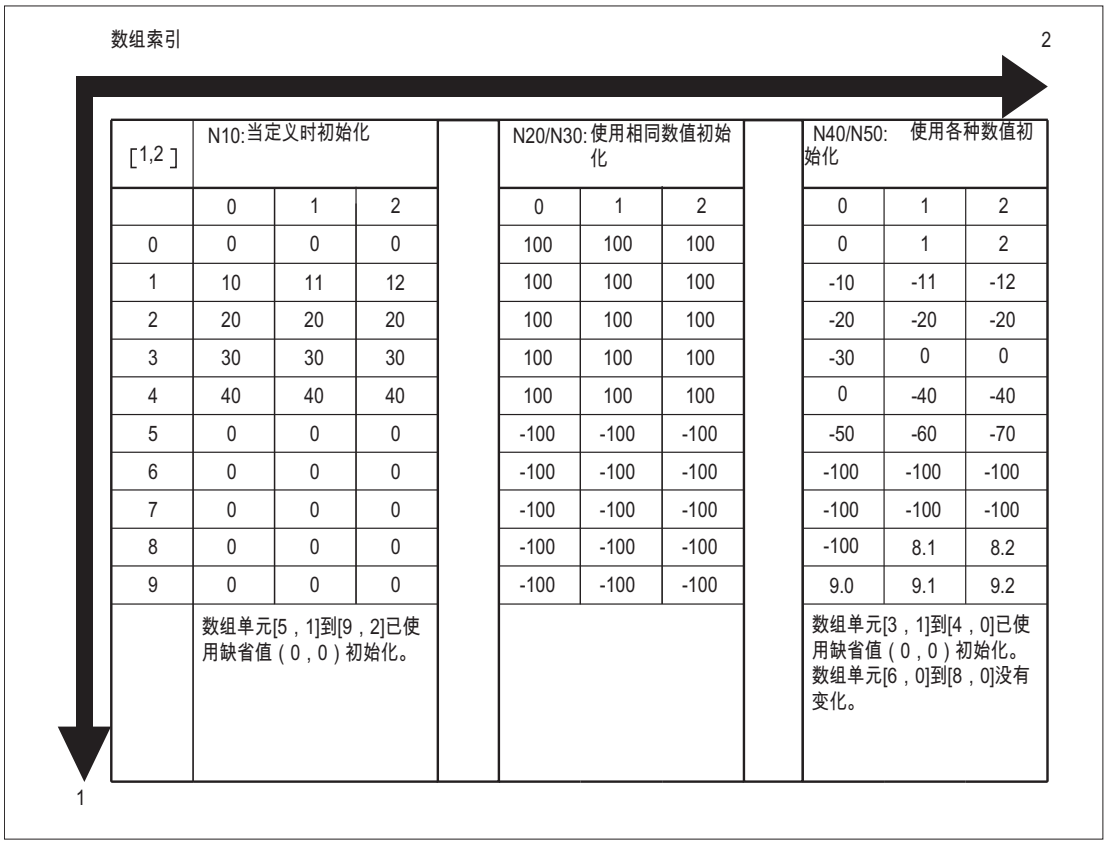
```
FOR n=0 TO 1
  FOR m=0 TO 2
    FOR o=0 TO 3
      FELD[n,m,o] = 1
    ENDFOR
  ENDFOR
ENDFOR
```

示例： 初始化整个变量数组

当前占用情况见插图。

程序代码

```
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)
```



[1,2]	N10:当定义时初始化			N20/N30: 使用相同数值初始化			N40/N50: 使用各种数值初始化		
	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
	数组单元[5，1]到[9，2]已使用缺省值（0，0）初始化。						数组单元[3，1]到[4，0]已使用缺省值（0，0）初始化。 数组单元[6，0]到[8，0]没有变化。		

参见

定义和初始化数组变量（DEF，SET，REP）： 其它信息 (页 61)

变量 (页 23)

2.1.13 定义和初始化数组变量（DEF，SET，REP）： 其它信息

更多信息(SET)

定义时进行初始化

- 从第 1 个数组元素开始，按照值列表中的值和写入的元素数量进行初始化。
- 值列表中没有显式指定值的数组元素(数值表中的空白)自动赋值 0。
- 对于 AXIS 数据类型的变量，值列表中不允许出现空白。
- 如果值列表包含的值大于数组元素的数量，则显示报警。

在程序执行中赋值

以上说明的定义规则同样适用于程序执行中的赋值。 此外，还有以下方法：

- 表达式也允许用作值列表的元素。
- 从编程的数组索引开始赋值。 从而根据需要赋值部分数组。

示例：

程序代码	注释
DEF INT FELD[5,5]	; 数组定义
FELD[0,0]=SET(1,2,3,4,5)	; 对前 5 个数组元素进行赋值[0,0] - [0,4]
FELD[0,0]=SET(1,2, , ,5)	; 对前 5 个数组元素[0,0] - [0,4]进行带空隙的赋值，数组元素 [0,2] 和 [0,3] = 0
FELD[2,3]=SET(VARIABLE,4*5.6)	; 带变量和表达式的赋值，自数组索引[2,3]起: [2,3] = VARIABLE [2,4] = 4 * 5.6 = 22.4

更多信息(REP)

定义时进行初始化

- 所有或指定数量的数组元素都会以给定值（常量）进行初始化。
- FRAME 数据类型的变量无法进行初始化。

示例：

程序代码	注释
DEF REAL varName[10]=REP(3.5,4)	; 数组定义，数组元素 0] ~[3] 以值 3.5 初始化

在程序执行中赋值

以上说明的定义规则同样适用于程序执行中的赋值。此外，还有以下方法：

- 表达式也允许用作值列表的元素。
- 从编程的数组索引开始赋值。 从而根据需要赋值部分数组。

示例：

程序代码	注释
DEF REAL varName[10]	; 数组定义
varName[5]=REP(4.5,3)	; 数组元素 [5] ~ [7] = 4.5
R10=REP(2.4,3)	; R-Parameter R10 ~ R12 = 2.4
DEF FRAME FRM[10]	; 数组定义
FRM[5]=REP(CTRANS(X,5))	; 数组元素[5] ~ [9] = CTRANS(X,5)

参见

定义和初始化数组变量（DEF，SET，REP） (页 56)

2.1.14 数据类型

NC 中提供了以下数据类型：

数据类型	含义	取值范围
INT	带正负号的整数值	-2147483648 ... +2147483647
REAL	实数（根据 IEEE 为 LONG REAL）	±(~2,2*10^{-308} … ~1,8*10^{+308})
BOOL	真值 真（1）和 假（0）	1, 0
CHAR	ASCII-字符	ASCII 代码 0...255
STRING	定义长度的字符串	最多 200 个字符（无特殊字符）
AXIS	进给轴/主轴标识符	通道轴名称
FRAME	用于静态坐标转换（平移、旋转、缩放和镜像）的几何参数	---

隐式数据类型转换

系统允许以下数据类型转换，并且在赋值和传递参数时隐式进行转换：

从 ↓/ 到 →	REAL	INT	BOOL
REAL	x	o	&
INT	x	x	&
BOOL	x	x	x
x： 无限制 o:由于超出取值范围可能会丢失数据 ⇒ 报警； 取整： 小数值 ≥ 0.5 ⇒ 向上取值, 小数值 < 0,5 ⇒ 略去 &: 值 ≠ 0 ⇒ TRUE, 值 == 0 ⇒ FALSE			

参见

变量 (页 23)

2.1.15 检查变量的存在性（ISVAR）

使用预定义的 ISVAR 功能，可检查一个系统变量/用户变量（例如机床数据、设定数据、系统变量、通用变量如 GUD）在 NC 中是否为已知状态。

变量

待查询的变量如下构建：

无维变量：	**<变量>**
一维变量，无数组索引：	**<变量>[ ]**
一维变量，带数组索引：	**<变量>[<n>]**
二维变量，无数组索引：	**<变量>[,]**
二维变量，带数组索引 n 和 m：	**<变量>[<n>,<m>]**

句法

```
<结果>=ISVAR(<变量>[<n>,<m>])
```

含义

<结果>：	返回值		
	数据类型：	BOOL	
	取值范围：	1	变量存在
		0	变量未知
ISVAR：	检查给定的系统变量/用户变量在 NC 中是否为已知状态。		
<变量>：	系统变量/用户变量的名称		
	数据类型：	STRING	
<n>：	第一维的数组索引（**可选**）		
	数据类型：	INT	
<m>：	第二维的数组索引（**可选**）		
	数据类型：	INT	

根据传送参数进行下列检查：

- 名称是否为已知
- 变量是否为数组

- 是否是一维或二维数组
- 相应的数组索引是否在允许的范围内

当所有检查结果均为“正”时，会反馈 TRUE (1)。

若一项检查结果为“负”或出现句法错误，则会反馈 FALSE (0)。

示例

程序代码	注释
DEF INT VAR1	
DEF BOOL IS_VAR=FALSE	
N10 IS_VAR=ISVAR（“VAR1”）	; IS_VAR 在此状况下为 TRUE。

程序代码	注释
DEF REAL VARARRAY[10,10]	
DEF BOOL IS_VAR=FALSE	
N10 IS_VAR=ISVAR("VARARRAY[,]")	; IS_VAR 在此状况下为 TRUE，且为二维数组。
N20 IS_VAR=ISVAR("VARARRAY")	; IS_VAR 为 TRUE，变量存在。
N30 IS_VAR=ISVAR("VARARRAY[8,11]")	; IS_VAR 为 FALSE，不允许此数组索引。
N40 IS_VAR=ISVAR("VARARRAY[8,8")	; IS_VAR 为 FALSE，缺少“]”（句法错误）。
N50 IS_VAR=ISVAR("VARARRAY[,8]")	; IS_VAR 为 TRUE，允许此数组索引。
N60 IS_VAR=ISVAR("VARARRAY[8,]")	; IS_VAR 为 TRUE，允许此数组索引。

程序代码	注释
DEF BOOL IS_VAR=FALSE	
N100 IS_VAR=ISVAR("$MC_GCODE_RESET_VALUES[1]"	；传送参数为机床数据，IS_VAR 为 TRUE。

程序代码	注释
DEF BOOL IS_VAR=FALSE	
N10 IS_VAR=ISVAR（“$P_EP”）	; IS_VAR 在此状况下为 TRUE。
N20 IS_VAR=ISVAR("$P_EP[X]")	; IS_VAR 在此状况下为 TRUE。

2.1.16 读取属性值/数据类型（GETVARPHU、GETVARAP、GETVARLIM、GETVARDIM、GETVARDFT、GETVARTYP）

适用预定义功能 GETVARPHU、GETVARAP、GETVARLIM、GETVARDFT 可读取系统变量/用户变量的属性值，使用 GETVARTYP 可读取系统变量/用户变量的数据类型。

读取物理单位

句法：
<结果>=GETVARPHU(<名称>)

含义：

<结果>：	物理单位的数值		
	数据类型：	INT	
	取值范围：	参见“属性： 物理单位(PHU) (页 46)”中的表格	
		故障情况：	
		- 2	所给定的<名称>未被分配至系统参数或用户变量。
GETVARPHU：	读取系统变量/用户变量的物理单位		
<名称>：	系统变量/用户变量的名称		
	数据类型：	STRING	

示例：

NC 包含以下 GUD 变量：

DEF CHAN REAL PHU 42 LLI 0 ULI 10000 electric

程序代码	注释
DEF INT result=0	
result=GETVARPHU("electric")	；获取该 GUD 变量的物理单位。
IF (result < 0) GOTOF error	

所反馈的结果为值 42。此值对应的物理单位为 [kW]。

说明

GETVARPHU 例如可用于在变量分配 a = b 时确认这两个变量的单位。

读取存取权限

句法：
<结果>=GETVARAP(<名称>,<存取>)

含义：

<结果>：	所给定 **<**存取**>** 的保护等级		
	数据类型：	INT	
	取值范围：	0 ... 7	参见“属性： 存取权限(APR, APW, APRP, APWP, APRB, APWB) (页 49)”。
		故障情况：	
		- 1	不可写入（仅与**<**存取**>**"WP"和"WB"相关）
		- 2	所给定的<名称>未被分配至系统参数或用户变量。
		- 3	<存取>赋值错误
GETVARAP：	读取对一个系统变量/用户变量的存取权限		
<名称>：	系统变量/用户变量的名称		
	数据类型：	STRING	
<存取>：	存取类型		
	数据类型：	STRING	
	取值范围：	"RP"	通过零件程序读取
		"WP"	通过零件程序写入
		"RB"	通过 OPI 读取
		"WB"	通过 OPI 写入

示例：

程序代码	**注释**
DEF INT result=0	
result=GETVARAP("$TC_MAP8","WB")	；获取通过 OPI 对系统参数“刀库位置”进行写入的存取保护等级。
IF (result < 0) GOTOF error	

所反馈的结果为值 7。这一结果对应的是钥匙开关位置 0（= 无存取保护等级）。

说明

GETVARAP 例如可用于编写一段检查程序，以检查是否具有特定应用所需的存取权限。

读取限值

句法：
<状态>=GETVARLIM(<名称>,<限值>,<结果>)

含义：

<状态>：	功能状态		
	数据类型：	INT	
	取值范围：	1	正常
		-1	未定义限值 （对于 AXIS、STRING、FRAME 类型的变量）
		-2	所给定的<名称>未被分配至系统参数或用户变量。
		-3	<限值>赋值错误
GETVARLIM：	读取系统变量/用户变量的下限/上限值		
<名称>：	系统变量/用户变量的名称		
	数据类型：	STRING	
<限值>：	给定需要读取哪一个限值		
	数据类型：	CHAR	
	取值范围：	"L"：	= 下限值
		"U"：	= 上限值
<结果>：	限值的返回方式		
	数据类型：	VAR REAL	

示例：

程序代码	注释
DEF INT state=0	
DEF REAL result=0	
state=GETVARLIM("$MA_MAX_AX_VELO","L",result)	；获取 MD32000 $MA_MAX_AX_VELO 的下限值。
IF (result < 0) GOTOF error	

读取属性/数据类型

句法：
<结果>=GETVARDIM(<名称>, 索引)

含义：

<结果>：	数组<索引>的维度/数量	
	数据类型：	INT
GETVARDIM：	读取系统变量/用户变量的下限/上限值	
<名称>：	读取数组元素的数量	
	数据类型：	STRING
<索引>：	数组的编号，最大 3	
	数据类型：	INT

示例：

程序代码	注释
N5 DEF REAL myReal[5,4]	
N10 R1 = GETVATDIM(„myReal“,1)	R1 = 5
N15 R2 = GETVATDIM(„myReal“,2)	R2 = 4

读取缺省值

句法：
<状态>=GETVARDFT(<名称>,<结果>[,<索引_1>,<索引_2>,<索引_3>])

含义：

<状态>：	功能状态		
	数据类型：	INT	
	取值范围：	1	正常
		-1	无可用缺省值 （例如由于 <结果> 的类型与 <名称> 不符）
		-2	所给定的<名称>未被分配至系统参数或用户变量。
		-3	<索引_1>赋值错误，维度小于一（= 无数组 = 标度）
		-4	<索引_2>赋值错误
		-5	<索引_3>赋值错误
GETVARDFT：	读取系统变量/用户变量的缺省值		
<名称>：	系统变量/用户变量的名称		
	数据类型：	STRING	

<结果>：	缺省值的返回方式	
	数据类型：	VAR REAL （在读取 INT、REAL、BOOL、AXIS 类型变量的缺省值时）
		VAR STRING （在读取 STRING 和 CHAR 类型变量的缺省值时）
		VAR FRAME （在读取 FRAME 类型变量的缺省值时）
<索引_1>：	至第一维的索引（可选）	
	数据类型：	INT
	不编写表示 = 0	
<索引_2>：	至第二维的索引（可选）	
	数据类型：	INT
	不编写表示 = 0	
<索引_3>：	至第三维的索引（可选）	
	数据类型：	INT
	不编写表示 = 0	

示例：

程序代码	注释
DEF INT state=0	
DEF REAL resultR=0	；变量用于接收 INT、REAL、BOOL、AXIS 类型的缺省值。
DEF FRAME resultF=0	；变量用于接收 FRAME 类型的缺省值
IF (GETVARTYP("$MA_MAX_AX_VELO") <> 4) GOTOF error	
state=GETVARDFT("$MA_MAX_AX_VELO", resultR, AXTOINT(X))	；获取 “X” 轴的缺省值
IF (resultR < 0) GOTOF error	
IF (GETVARTYP("$TC_TP8") <> 3) GOTOF error	
state=GETVARDFT("$TC_TP8", resultR)	
IF (GETVARTYP("$P_UBFR") <> 7) GOTOF error	
state=GETVARDFT("$P_UBFR", resultF)	

读取数据类型

句法：
<结果>=GETVARTYP(<名称>)

含义：

<结果>：	所给定系统变量/用户变量的数据类型		
	数据类型：	INT	
	取值范围：	1	= BOOL
		2	= CHAR
		3	= INT
		4	= REAL
		5	= STRING
		6	= AXIS
		7	= FRAME
		故障情况：	
		< 0	所给定的<名称>未被分配至系统参数或用户变量。
GETVARTYP：	读取系统变量/用户变量的数据类型		
<名称>：	系统变量/用户变量的名称		
	数据类型：	STRING	

示例：

程序代码	注释
DEF INT result=0	
DEF STRING name="R"	
result=GETVARTYP(name)	; 获取 R 参数的数据类型。
IF (result < 0) GOTOF error	

所反馈的结果为值 4。该数值对应的是 REAL 数据类型。

2.2 间接编程

2.2.1 间接编程地址

在间接编程地址时，扩展的地址（<索引>）由一个合适的变量类型替代。

说明

在下列情况下，不能间接编程地址：

- N（程序段编号）
- L（子程序）
- 可调地址
  （例如，X[1] 代替 X1 是不允许的）

句法

<地址>[<索引>]

含义

<地址>[...]:	带扩展名（索引）的固定地址
<索引>:	变量，例如主轴编号，轴，....

示例

示例 1： 间接编程一个主轴编号

直接编程：

程序代码	注释
S1=300	；主轴转速 300 转/分钟，编号为 1。

间接编程

程序代码	注释
DEF INT SPINU=1	；定义 INT 型变量和赋值
S[SPINU]=300	；主轴转速 300 转/分钟，其编号保存在变量 SPINU 中（在该示例中，主轴编号为 1）。

示例 2： 间接编程一个轴

直接编程：

程序代码	注释
FA[U]=300	；“U”轴的进给量为 300。

间接编程

程序代码	注释
DEF AXIS AXVAR2=U	；定义一个 AXIS 型变量和赋值。
FA[AXVAR2]=300	；轴的进给量为 300，其地址名称保存在名称为 AXVAR2 的变量中。

示例 3： 间接编程一个轴

直接编程：

程序代码	注释
$AA_MM[X]	；读取轴的测量探头一测量值（MKS）“X。

间接编程

程序代码	注释
DEF AXIS AXVAR3=X	；定义一个 AXIS 型变量和赋值。
$AA_MM[AXVAR3]	；为轴读取测量探头一测量值（MKS），其名称保存在变量 AXVAR3 中。

示例 4： 间接编程一个轴

直接编程：

程序代码
X1=100 X2=200

间接编程

程序代码	注释
DEF AXIS AXVAR1 AXVAR2	；定义两个 AXIS 型变量。
AXVAR1=(X1) AXVAR2=(X2)	；分配轴名称。
AX[AXVAR1]=100 AX[AXVAR2]=200	；运行轴，其地址名称保存在名为 AXVAR1 和 AXVAR2 的变量中。

示例 5： 间接编程一个轴

直接编程：

程序代码
G2 X100 I20

间接编程

程序代码	注释
DEF AXIS AXVAR1=X	；定义一个 AXIS 型变量和赋值。
G2 X100 IP[AXVAR1]=20	；间接编程轴的中点数据，其地址名称保存在名为 AXVAR1 的变量中。

示例 6： 间接编程数组元素

直接编程：

程序代码	注释
DEF INT 数组 1[4,5]	；定义数组 1。

间接编程

程序代码	注释
DEFINE DIM1 AS 4	；对于数组维，必须将数组大小设定为固定值。
DEFINE DIM2 AS 5	
DEF INT 数组[DIM1,DIM2]	
数组[DIM1-1,DIM2-1]=5	

示例 7：间接子程序调用

程序代码	注释
CALL "L" << R10	；调用其编号在 R10 中的程序（字符串级联）。

2.2.2 间接编程 G 代码

通过间接编程 G 代码，可以进行有效的循环编程。

句法

G[<组]=<编号>

含义

G[...]:	带扩展名（索引）的 G 代码	
<组>:	索引参数 G 代码组	
	类型：	INT

<编号>:	G 代码编号变量	
	类型：	INT 或 REAL

说明

通常只能间接编程非句法定义的 G 代码。

句法定义的 G 代码中只有 G 代码组 1 可采用间接编程。
而 G 代码组 2、3 和 4 中的句法定义 G 代码则不可以。

说明

在间接 G 代码编程中不允许进行算术计算。必须在 G 代码间接编程前，在一个自身的零件程序行中进行必要的 G 代码编号计算。

示例

示例 1：可设定零点偏移（G 代码组 8）

程序代码	注释
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 代码计算
N1110 G[8]=INT_VAR G1 X0 Y0	;G55

示例 2：平面选择（G 代码组 6）

程序代码	注释
N2010 R10=$P_GG[6]	; 读取 G 代码组 6 中生效的 G 代码
...	
N2090 G[6]=R10	

文档

有关 G 代码组信息参见：
编程手册之基本原理分册；“G 代码组”一章

2.2.3 间接编程位置属性（GP）

位置属性，例如轴位置的增量或绝对编程可以连同关键字 GP 一起间接编程为变量。

应用

位置属性的间接编程在 **替换循环**中有应用，因为，在这里，与将位置属性编程为关键字（例如 IC，AC，...）相比，有下列优点：

通过间接编程为变量，**无需**通过可能的位置属性转到的 CASE 指令。

句法

```
<定位指令>[<轴/主轴>]=
GP（<位置>，<位置属性）
<轴/主轴>=GP（<位置>，<位置属性）
```

含义

<定位指令>[]:	下列定位指令可以与关键字 GP 一起进行编程： POS, POSA, SPOS, SPOSA 此外，还可以编程： ● 所有通道中现有的轴/主轴名称： <轴/主轴> ● 可变的轴/主轴名称 AX
<轴/主轴>：	需要定位的轴/主轴
GP():	用于定位的关键字
<位置>:	参数 1 轴/主轴作为常量或变量
<位置属性>：	参数 2 位置属性（例如位置开始模式）作为变量（例如 $P_SUB_SPOSMODE）或作为关键字（IC，AC ...）

由变量提供的值有下列含义：

值	含义	允许：
0	不改变位置属性	
1	AC	POS, POSA, SPOS, SPOSA, AX, 轴地址
2	IC	POS, POSA, SPOS, SPOSA, AX, 轴地址
3	DC	POS, POSA, SPOS, SPOSA, AX, 轴地址
4	ACP	POS, POSA, SPOS, SPOSA, AX, 轴地址
5	ACN	POS, POSA, SPOS, SPOSA, AX, 轴地址

值	含义	允许：
6	OC	-
7	PC	-
8	DAC	POS, POSA, AX, 轴地址
9	DIC	POS, POSA, AX, 轴地址
10	RAC	POS, POSA, AX, 轴地址
11	RIC	POS, POSA, AX, 轴地址
12	CAC	POS, POSA
13	CIC	POS, POSA
14	CDC	POS, POSA
15	CACP	POS, POSA
16	CACN	POS, POSA

示例

对于一个有效的同步主轴耦合，在主动主轴 S1 和随动主轴 S2 之间通过 SPOS 指令在下面的用于主轴定位的替换循环的主程序中调用。

通过 N2230 中的指令进行定位：

```
SPOS[1]=GP($P_SUB_SPOSIT,$P_SUB_SPOSMODE)
SPOS[2]=GP($P_SUB_SPOSIT,$P_SUB_SPOSMODE)
```

从系统变量 $P_SUB_SPOSIT 中读取开始位置，从系统变量 $P_SUB_SPOSMODE 中读取位置开始模式。

```
N1000 PROC LANG_SUB DISPLOF SBLOF
...
N2100 IF($P_SUB_AXFCT==2)
N2110                                   ; 对于有效的同步主轴耦合，替换
                                        SPOS / SPOSA / M19 指令
N2185 DELAYFSTON                        ; 开始停止延时段
N2190 COUPOF(S2,S1)                     ; 取消同步主轴耦合
N2200                                   ; 定位引导主轴和随动主轴
N2210 IF($P_SUB_SPOS==TRUE) OR
($P_SUB_SPOSA==TRUE)
N2220                                   ; 用 SPOS 定位主轴：
N2230    SPOS[1]=GP($P_SUB_SPOSIT,
$P_SUB_SPOSMODE)
         SPOS[2]=GP($P_SUB_SPOSIT,
$P_SUB_SPOSMODE)
```

程序代码	注释
N2250 ELSE	
N2260	; 用 M19 定位主轴：
N2270 M1=19 M2=19	; 定位引导主轴和随动主轴
N2280 ENDIF	
N2285 DELAYFSTOF	; 结束停止延时段
N2290 COUPON(S2,S1)	; 激活同步主轴耦合
N2410 ELSE	
N2420	; 查询其它替换
...	
N3300 ENDIF	
...	
N9999 RET	

边界条件

- 不能够在同步动作中间接编程位置属性。

文档

功能手册 基本功能；BAG，通道，程序运行，复位特性（K1），
章节： 由子程序替换 NC 功能

2.2.4 间接编程零件程序行（EXECSTRING）

使用零件程序指令 EXECSTRING 可将之前生成的 String 变量作为零件程序行执行。

句法

EXECSTRING 须编程在一个单独的零件程序行中：
EXECSTRING (<字符串变量>)

含义

EXECSTRING:	将 String 变量作为零件程序行执行的指令
<字符串变量>：	包含了原本需要执行的零件程序的 STRING 变量

说明

通过 EXECSTRING 可取消除控制结构 (页 122)以外的所有可**在零件程序段落中**编程的零件程序结构。 PROC 和 DEF 指令除外，在 INI 和 DEF 文件中的应用通常也不可。

示例

程序代码	注释
N100 DEF STRING[100] MY_BLOCK	; 定义包含需要执行的零件程序行的 String 变量。
N110 DEF STRING[10] MFCT1="M7"	
...	
N200 EXECSTRING(MFCT1 << "M4711")	; 执行零件程序行 “M7 M4711”。
...	
N300 R10=1	
N310 MY_BLOCK="M3"	
N320 IF(R10)	
N330 MY_BLOCK = MY_BLOCK << MFCT1	
N340 ENDIF	
N350 EXECSTRING(MY_BLOCK)	; 执行零件程序行 “M3 M7”。

2.3 运算功能

运算符/计算功能	含义
+	加法
-	减法
*	乘法
/ [1]	除法 [1]
DIV [1]	整除 [1]
MOD [1]	取模除法（提供整除法的余数）[1]
:	框架变量的串运算符
SIN()	正弦
COS()	余弦
TAN()	正切
ASIN()	反正弦
ACOS( )	反余弦
ATAN2(,) [1]	反正切 2 [1]
SQRT()	平方根
ABS()	绝对值
POT()	2 次幂（平方）
TRUNC()	整数部分 比较命令时的精确度，可使用 TRUNC 设置（参见“比较错误的精确度修正 (TRUNC) (页 85)”）
ROUND()	取整
LN()	自然对数
EXP()	指数函数
MINVAL ()	两变量中的较小值 （参见“参见“最大变量、最小变量和变量区域(MINVAL, MAXVAL, BOUND)” (页 87)”）
MAXVAL ()	两变量中的较大值 （参见“参见“最大变量、最小变量和变量区域(MINVAL, MAXVAL, BOUND)” (页 87)”）

BOUND ()	已定义值域中的变量值 （参见“参见“最大变量、最小变量和变量区域(MINVAL, MAXVAL, BOUND)” (页 87)”）
CTRANS()	偏移
CROT ()	旋转
CSCALE ()	比例修改
CMIRROR ()	镜像
1) 参见“示例”一段	

编程

计算功能采用通常使用的数学运算法则。在处理中需优先处理的用圆括号给出。对于三角函数和它的反函数其单位是度(直角=90）。

示例

除法：/

（REAL 型）=（INT 型或 REAL 型）/（INT 型或 REAL 型）；

示例：3 / 4 = 0.75

整除法：DIV

（INT 型）=（INT 型或 REAL 型）/（INT 型或 REAL 型）；

示例：7 DIV 4.1 = 1

取模除法（提供整除法的余数）：MOD

（REAL 型）=（INT 型或 REAL 型）MOD（INT 型或 REAL 型）；

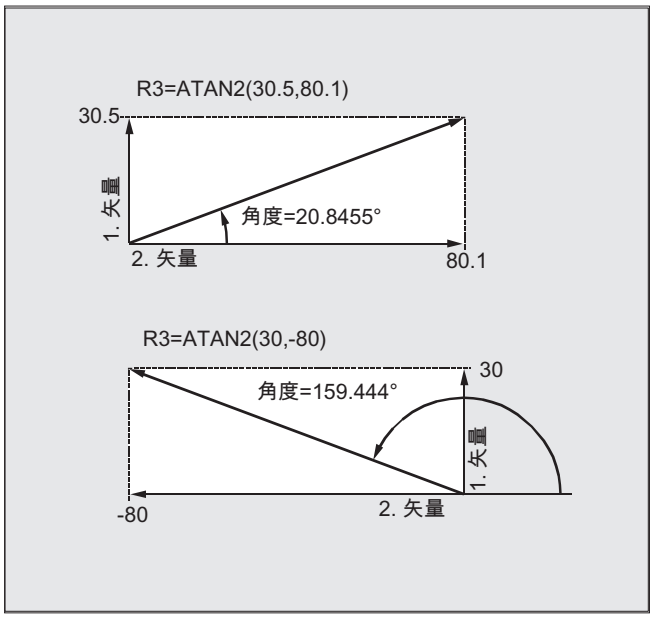
示例：7 MOD 4.1 = 2.9

反正切 2：ATAN2

ATAN2 计算功能可以从两个互相垂直的矢量计算出总矢量的角度。

结果位于四个象限的范围内 (-180°< 0 < +180°)。

角度是指在正方向的第 2 个数值。



编程示例

程序代码	注释
R1=R1+1	; 新的 R1 = 旧的 R1 + 1
R1=R2+R3 R4=R5-R6 R7=R8*R9	
R10=R11/R12 R13=SIN(25.3)	
R14=R1*R2+R3	; 四则运算法则。
R14=(R1+R2)*R3	; 首先运算括号中的表达式。
R15=SQRT(POT(R1)+POT(R2))	; 首先计算内括号： R15 = ((R1^2 + R2^2)) 的平方根
RESFRAME=FRAME1:FRAME2	; 带串运算符的框架连接
FRAME3=CTRANS(…):CROT(…)	赋值给一个框架组件

2.4 比较运算和逻辑运算

比较运算 例如可以用来表达某个跳转条件。 完整的表达式也可以进行比较。

比较函数可用于 CHAR、INT、 REAL 和 BOOL 型的变量。 对于 CHAR 型变量，比较代码值。
对于 STRING、AXIS 和 FRAME 可以为： **==** 和 **<>**，也可在同步动作中用于运算 STRING 型的变量。

比较运算的结果始终为 BOOL 型。

逻辑运算 用来将真值联系起来。

逻辑运算只能用于 BOOL 型变量。 通过内部类型转换也可将其用于 CHAR、INT 和 REAL 数据类型。

对于逻辑（布尔）运算而言，适用数据类型为 BOOL, CHAR, INT 和 REAL:

- 0 表示： FALSE
- 等于 0 相当于： TRUE

逐位逻辑运算符

使用 CHAR 和 INT 型变量也可进行逐位逻辑运算。 如果有这种情况，类型转换自动进行。

编程

比较运算符	含义
==	相同于
<>	不等
>	大于
<	小于
>=	大于等于
<=	小于等于

逻辑运算符	含义
AND	与
OR	或者

NOT	非
XOR	异－或

逐位逻辑运算符	含义
B_AND	位方式“与”
B_OR	位方式“或”
B_NOT	位方式“非”
B_XOR	位方式“异－或”

说明

在算术表达式中可以通过圆括号确定所有运算的顺序并且由此脱离原来普通的优先计算规则。

说明

在布尔的操作数和运算符之间必须加入空格。

说明

运算符 B_NOT 只与一个运算域有关。 它位于运算符之后。

示例

示例 1： 比较运算符

```
IF R10>=100 GOTOF 目标
```

或者

```
R11=R10>=100
IF R11 GOTOF 目标
```

R10>=100 的比较结果首先存储在 R11 中。

示例 2： 逻辑运算符

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

或者

```
IF NOT R10 GOTOB START
```

NOT 只与一个运算域有关。

示例 3： 逐位逻辑运算符

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

2.5 比较错误的精确度修正 (TRUNC)

TRUNC 指令用来截断与一个精度系数相乘后的运算数。

比较操作时的可设定精度

实数型零件程序参数内部用 64 位的 IEEE 格式描述。 这种显示形式不能构成精确的十进制数，在与理想计算的数值进行比较时可能会带来不好的结果。

相对相等性

为了使这种描述所带来的不精确性不影响程序流程，在比较指令中不检测绝对奇偶性，而是检测一个相对相等性。

句法

比较错误时的精度补偿

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

含义

TRUNC:	去除小数点后位数

所考虑的相对相等性为 10^{-12} 当

- 相等性： (==)
- 不相等性： (<>)
- 大于等于： (>=)
- 小于等于： (<=)
- 大于/小于： (><)绝对相等
- 大于： (>)
- 小于： (<)

兼容性

出于兼容性考虑，在 (>) 和 (<) 时通过设置机床数据 MD10280 $MN_PROG_FUNCTION_MASK Bit0 = 1 可以取消相对相等性的检测。

说明

与实数型数据比较时，由于以上原因一般会出现一定的误差。 当出现不可接受的偏差时，必须另选 INTEGER 型计算，方法是将运算数和一个精度系数相乘，然后再使用 TRUNC 截断。

同步动作

所描述的比较指令性能也适用于同步动作。

示例

示例 1： 精度检查

程序代码	注释
N40 R1=61.01 R2=61.02 R3=0.01	; 初始值分配
N41 IF ABS(R2-R1) > R3 GOTOF FEHLER	;到此为止将可执行跳转
N42 M30	; 程序结束
N43 出错： SETAL(66000)	
R1=61.01 R2=61.02 R3=0.01	; 初始值分配
R11=TRUNC(R1*1000) R12=TRUNC(R2*1000) R13=TRUNC(R3*1000)	;精度修正
IF ABS(R12-R11) > R13 GOTOF 出错	;不再执行跳转
M30	; 程序结束
FEHLER:SETAL(66000)	

示例 2： 得出并且分析两个运算数的商

程序代码	注释
R1=61.01 R2=61.02 R3=0.01	; 初始值分配
IF ABS((R2-R1)/R3)-1) > 10EX-5 GOTOF FEHLER	;不执行跳转
M30	; 程序结束
FEHLER:SETAL(66000)	

2.6 参见“最大变量、最小变量和变量区域(MINVAL, MAXVAL, BOUND)”

使用指令 MINVAL 和 MAXVAL 可以比较两个变量的值。 其中的较小值（采用 MINVAL 时）或较大值（采用 MAXVAL 时）会作为结果返回。

使用指令功能 BOUND 可以检查，待检变量的值是否在定义的值域内。

句法

```
<较小值>=MINVAL(<变量 1>,<变量 2>)
<较大值>=MAXVAL(<变量 1>,<变量 2>)
<返回值>=<BOUND>(<最小>,<最大>,<待检变量>)
```

含义

MINVAL:	确定两个变量即(<变量 1>, <变量 2>)中的**较小**值
<较小值>:	指令 MINVAL 的结果变量 设为较小的变量值。
MAXVAL:	确定两个变量即(<变量 1>, <变量 2>)中的**较大**值
<较大值>:	指令 MAXVAL 的结果变量 设为较大的变量值。
BOUND:	检查（<待检变量>）是否在已定义的值域中。
<最小>:	用于定义取值范围中最小值的变量
<最大>:	用于定义取值范围中最大值的变量
<返回值>:	指令 BOUND 的结果变量 如果待检变量的值在定义的取值范围内，结果变量会设为此待检变量。 如果待检变量的值大于最大值，结果变量会设为取值范围的最大值。 如果待检变量的值小于最小值，结果变量会设为取值范围的最小值。

说明

MINVAL、MAXVAL 和 BOUND 也可以在同步动作中编程。

说明

值相等时的属性

两值相等时，MINVAL/MAXVAL 返回该等值。 BOUND 再次返回待检变量的值。

示例

程序代码	注释
DEF REAL rVar1=10.5, rVar2=33.7, rVar3, rVar4,	rVar5, rValMin, rValMax, rRetVar
rValMin=MINVAL(rVar1,rVar2)	; rValMin 设为值 10.5。
rValMax=MAXVAL(rVar1,rVar2)	; rValMax 设为值 33.7。
rVar3=19.7	
rRetVar=BOUND(rVar1,rVar2,rVar3)	; rVar3 在极限值范围内，rRetVar 设为 19.7。
rVar3=1.8	
rRetVar=BOUND(rVar1,rVar2,rVar3)	; rVar3 小于最小值，rRetVar 设为 10.5。
rVar3=45.2	
rRetVar=BOUND(rVar1,rVar2,rVar3)	; rVar3 大于最大值，rRetVar 设为 33.7。

2.7 运算的优先级

每个运算符都被赋予一个优先级。 在计算一个表达式时，有高一级优先权的运算总是首先被执行。 在优先级相同的运算中，运算由左到右进行。

在算术表达式中可以通过圆括号确定所有运算的顺序并且由此脱离原来普通的优先计算规则。

运算的顺序

从最高到最低优先级

1.	NOT, B_NOT	非，位方式非
2.	*, /, DIV, MOD	乘除
3.	+, –	加减
4.	B_AND	位方式“与”
5.	B_XOR	位方式“异或”
6.	B_OR	位方式“或”
7.	AND	与
8.	XOR	异或
9.	OR	或者
10.	<<	字符串的链接,结果类型 STRING
11.	==, <>, >, <, >=, <=	比较运算符

说明

级联运算符“：”在表达式中不能与其它的运算符同时出现。 因此这种运算符不要求划分优先级。

如果-语句举例

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

2.8 可能有的类型转换

功能

赋值时的类型转换

数值常量、变量或者给某个变量赋值的表达式必须与该变量的类型相容。 一旦变量给出，在赋值时类型自动转换。

可能的类型转换

到	REAL	INT	BOOL	CHAR	STRING	AXIS	FRAME
从							
REAL	是	是*	是 1)	是*	–	–	–
INT	是	是	是 1)	是 2)	–	–	–
BOOL	是	是	是	是	是	–	–
CHAR	是	是	是 1)	是	是	–	–
STRING	–	–	是 4)	是 3)	是	–	–
AXIS	–	–	–	–	–	是	–
FRAME	–	–	–	–	–	–	是

说明

* 从实数型到整数型的转换中，小数值>= 0.5 时向上园整，否则舍去(ROUND 功能)。

1) 值<> 0 对应于 TRUE,值== 0 对应于 FALSE

2) 如果数值在允许的值范围内

3) 如果只有 1 个字符

4) 字符串长度 0 = >FALSE,否则 TRUE

说明

如果在转换中一个值大于目标范围，就会出现出错提示。

如果表达式中出现混合类型，系统会自动进行类型匹配。 类型转换也可用于同步动作中，见章节“同步运行动作，隐式类型转换”。

2.9 字符串运算

字符串运算符

除了典型的运算符“赋值”和“比较”之外，可以有下列字符串运算符：

- 类型转换到字符串 (AXSTRING) (页 91)
- 从 STRING (NUMBER, ISNUMBER, AXNAME) 类型转换 (页 92)
- 字符串的链接 (<<) (页 93)
- 大小写字母转换 (TOLOWER, TOUPPER) (页 95)
- 确定一个字符串的长度 (STRLEN) (页 95)
- 在字符串中查找字符/字符串 (INDEX, RINDEX, MINDEX, MATCH) (页 96)
- 部分字符串的选择 (SUBSTR) (页 97)
- 单个字符的读取和写入 (页 98)
- 格式化字符串 (SPRINT) (页 100)

字符 0 的特别意义

在系统内部，字符 0 被视为一个字符串的结束标志。 如果一个字符被一个 0 字符代替，那么这个字符串就被缩短了。

示例：

程序代码	注释
`DEF STRING[20] STRG="轴静止"`	
`STRG[6] = "X"`	
`MSG(STRG)`	；返回提示信息“X 轴静止”。
`STRG[6] = 0`	
`MSG(STRG)`	；返回提示信息“轴”。

2.9.1 类型转换到字符串 (AXSTRING)

通过功能“向字符串类型转换”，不同类型的变量可以用作一个信息（MSG）的组成部分。

使用运算符<<时，隐含适用于数据类型 INT，REAL，CHAR 和 BOOL（参见“字符串的链接 (<<) (页 93)”）。

一个 INT 值会被转换为可读形式。 在显示实数值时会给出小数点后 10 位。

通过指令 AXSTRING 可以将变量由类型 AXIS 转换到 STRING。

句法

```
<STRING_ERG> = << <bel._Typ>
<STRING_ERG> = AXSTRING(<轴名称>)
```

含义

<STRING_ERG>:	用于类型转换结果的变量	
	类型：	STRING
<bel._Typ>:	变量类型 INT, REAL, CHAR, STRING 和 BOOL	
AXSTRING:	指令 AXSTRING 作为字符串提供所给出的轴名称。	
<轴名称>:	轴名称变量	
	类型：	AXIS

说明

FRAME 变量不能被转换。

2.9.2 从 STRING (NUMBER, ISNUMBER, AXNAME) 类型转换

通过指令 NUMBER 可以实现从 STRING 到 REAL 的转换。 转换可行性可以通过指令 ISNUMBER 检测。

通过指令 AXNAME 转换一个字符串到数据类型 AXIS。

句法

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

含义

NUMBER:	指令 NUMBER 将由<字符串>格式显示的数值作为实数值送回。
<字符串>:	要转换的类型 STRING 的变量

<REAL_ERG>:	通过 NUMBER 类型转换结果变量		
	类型：	REAL	
ISNUMBER:	通过指令 ISNUMBER 可以检测这个<字符串>是否能被转换为一个有效的数字。		
<BOOL_ERG>:	通过 ISNUMBER 查询结果变量		
	类型：	BOOL	
	值：	TRUE	当<字符串>显示一个根据语言规则有效的实数时，ISNUMBER 显示值为 TRUE。
		FALSE	当 ISNUMBER 给出值 FALSE 时，就会在调用带有相同<字符串> 的 NUMBER 时触发报警。
AXNAME:	指令 AXNAME 转换规定的 <字符串> 到一个轴名称内。 **提示：** 如果该 <字符串> 没有分配到设计的轴名称，则触发报警。		
<AXIS_ERG>:	通过 AXNAME 类型转换结果变量		
	类型：	AXIS	

示例

```
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
```

2.9.3 字符串的链接 (<<)

功能“字符串链接”使单个的字符串可以组合在一起。

通过运算符“**<<**”实现链接。 这个运算符适用于所有基本类型 CHAR，BOOL，INT，REAL 和 STRING 的组合，变成目标类型 STRING。 必需的数据类型转换按照现有的规则进行。

句法

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

含义

<bel._Typ>:	类型 CHAR, BOOL, INT, REAL 或 STRING 的变量
<< :	变量链接运算 (<bel._Typ>) 可以得到一个合并的字符串 (类型 STRING)。 该运算符也可单独作为“一元”变量使用。 这样可以执行一个明确的、到字符串类型的转换（FRAME 和 AXIS 不可用）： << <bel._Typ>

比如，自文本列表组成一个信息或一个命令，并且插入参数（如一个模块名）：
MSG(STRG_TAB[LOAD_IDX]<<BAUSTEIN_NAME)

说明

在字符串级联时，中间结果不可以超过最大字符串长度。

说明

类型 FRAME 和 AXIS 类型不能使与运算符“**<<**”一起使用。

示例

示例 1：字符串的链接

程序代码	注释
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)	; 显示: “下标：2/值：9.654”
NO_MSG:	

示例 2： 明确的类型转换通过 <<

程序代码	注释
DEF REAL VALUE=3.5	
<<VALUE	; 将指定变量从类型 REAL 转换到类型 STRING。

2.9.4 大小写字母转换 (TOLOWER, TOUPPER)

功能“大小写字母转换”允许一个字符串的全部字母一起转换到另一种统一的表示方式。

句法

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

含义

TOUPPER:	通过指令 TOUPPER 将一个字符串的全部字母都转换为 **大写**字母。	
TOLOWER:	通过指令 TOLOWER 将一个字符串的全部字母都转换为 **小写**字母。	
<字符串>:	要转换的字符串	
	类型：	STRING
<STRING_ERG>:	类型转换结果变量	
	类型：	STRING

示例

因为也有可能与操作界面上的用户输入发生冲突，可以使用大写或者小写字母来统一显示结果：

程序代码

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

2.9.5 确定一个字符串的长度 (STRLEN)

通过指令 STRLEN 可以确定字符串的长度。

句法

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

含义

STRLEN:	通过指令 STRLEN 可以确定指定字符串的长度。 它可以返回字符的数量，即从字符串开头数起，但不包括 0 的字符数量。	
<字符串>:	要确定长度的字符串	
	类型：	STRING
<INT_ERG>:	类型转换结果变量	
	类型：	INT

示例

该功能连同单字符访问一起可以确定一个字符串的末尾：

程序代码

```
IF(STRLEN (模块名称) > 10) GOTOF 出错
```

2.9.6 在字符串中查找字符/字符串 (INDEX, RINDEX, MINDEX, MATCH)

利用此功能，可以在后面一个字符串中查找单个字符或者一个字符串。 查找结果说明：在字符串的一个位置找到需要查找的字符/字符串。

句法

INT_ERG=INDEX(STRING,CHAR)；**结果类型： INT**

INT_ERG=RINDEX(STRING,CHAR)；**结果类型： INT**

INT_ERG=MINDEX(STRING,STRING)；**结果类型： INT**

INT_ERG=MATCH(STRING,STRING)；**结果类型： INT**

符号语义

查找功能： 它会把所查找字符串（第一个参数）中的位置送回。 如果找不到字符或字符串，就会送回数值-1。 第一个字符位置为 0。

含义

INDEX:	在第一个参数中寻找作为第二个参数的字符（从前面）。
RINDEX:	在第一个参数中寻找作为第二个参数的字符（从后面）。
MINDEX:	相当于 INDEX 功能，不同之处在于，它传送一个字符列表作为字符串，其中传送了第一个被发现的字符的索引。
MATCH:	在一个字符串中寻找一个字符串。

这样字符串可以按照一定的标准进行分解，大约是在空格或路径分隔符的位置("/")。

示例

将一个输入分解成路径名称和模块名称

程序代码	注释
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	；将 3 作为 LISTIDX 中的值送回；因为 "N" 为参数 EINGABE 中的第一个字符（从选择列表前面数起）。
PFADIDX = INDEX (EINGABE, "/") +1	由此 PFADIDX = 1 有效
PROGIDX = RINDEX (EINGABE, "/") +1	由此 PROGIDX = 12 有效
	；借助下一段落中引用的功能 SUBSTR 将变量 EINGABE 分解成“路径”和“模块”：
VARIABLE = SUBSTR (EINGABE, PFADIDX, PROGIDX-PFADIDX-1)	;然后给出 "_N_MPF_DIR"
VARIABLE = SUBSTR (EINGABE, PROGIDX)	;然后给出 "_N_EXECUTE_MPF"

2.9.7 部分字符串的选择 (SUBSTR)

使用 SUBSTRING 功能可读取一个字符串中的任意部分。

句法

<STRING_ERG>=SUBSTR(<字符串>,<索引>,<长度>)

<STRING_ERG>=SUBSTR(<字符串>,<索引>)

含义

SUBSTR：	此功能可从 <索引> 位置起根据所给定的 <长度> 从 <字符串> 中输出一个子字符串。 若未给定 <长度> 参数，则此功能输出的子字符串长度为给定 <索引> 位置到原字符串末尾。
<索引>：	设定字符串中子字符串的起始位置。 若起始位置位于字符串末尾之后，则会反馈一个空字符串（ ""）。 字符串的首个字符： 索引 = 0 取值范围： 0 ... （字符串长 - 1）
<长度>：	子字符串的长度。 若此参数设定过长，则反馈的子字符串只会到原字符串末尾。 取值范围： 1 ... （字符串长 - 1）

示例

程序代码	注释
DEF STRING [29] ERG	
; 1	
; 0123456789012345678	
ERG = SUBSTR ("应答: 10 ～ 99", 10, 2)	; ERG == "10"有效
ERG = SUBSTR ("应答: 10 ～ 99", 10)	; ERG == "10 ～ 99"

2.9.8 单个字符的读取和写入

在一个字符串中可读取和写入单个字符。

此时必须遵循下列边界条件：

- 此功能仅适用于用户定义变量，不可用于系统变量
- 在子程序调用中，字符串的单个字符仅可采用“值调用（call by value）”传输

句法

```
<字符>=<字符串>[<索引>]
<字符>=<字符串_数组>[<数组_索引>,<索引>]
<字符串>[<索引>]=<字符>
<字符串_数组>[<数组_索引>,<索引>]=<字符>
```

含义

<字符串>：	任意字符串
<字符>：	CHAR 类型变量
<索引>：	字符串中字符的位置。 字符串的首个字符： 索引 = 0 取值范围： 0 ...（字符串长 - 1）

示例

示例 1： 变量信息

```
;                          0123456789
DEF STRING [50] MELDUNG = "轴 n 已经到达位置"
MELDUNG [6] = "X"
MSG (MELDUNG)                          ; “轴 x 已经到达位置”
```

示例 2： 分析系统变量

```
DEF STRING [50] STRG                   ; 系统变量缓存
...
STRG = $P_MMCA                         ; 载入系统变量
IF STRG[0] == "E" GOTO ...             ; 分析系统变量
```

示例 3： "值调用（call by value）“和”引用调用(call by reference)"参数传输

```
;                       0123456
DEF STRING[50] STRG = "轴 X"
DEF CHAR CHR
...
EXTERN UP_VAL(ACHSE)                   ; 通过“值调用（call by value）”参数定义子程序
EXTERN UP_REF(VAR ACHSE)               ; 通过“引用调用（call by ref.）”参数定义子程序
...
UP_VAL(STRG[6])                        ; “值调用”参数传输
...
CHR = STRG [6]                         ; 缓存
UP_REF(CHR)                            ; “引用调用”参数传输
```

2.9.9 格式化字符串 (SPRINT)

通过预定义函数 SPRINT 可以对字符串进行格式化，然后输出给外部设备（可参见“Process DataShare - 数据输出到外部设备/文件上（EXTOPEN，WRITE，EXTCLOSE） (页 715)”）。

句法

“<结果_字符串>”=SPRINT（“<格式_字符串>”，<值_1>，<值_2>，...，<值_n>）

含义

SPRINT：	预定义函数的标识，提供一个字符串类型的值。
“<格式_字符串>”：	包含固定以及可变部分的字符串。 可变部分，包含格式控制符 % 以及格式说明符。
< 值_1>,< 值_2>,…,< 值_n>:	这些值都是常量或 NC 变量，位于第 n 个格式控制符%旁，根据格式说明符，插入到<格式_字符串> 中。
“<结果_字符串>”：	格式化的字符串（最多 400 个字节）

提供的格式说明符

%B：	将以下数值转换为字符串“TRUE”： ● 不为 0 的值 ● 非空字符串（针对字符串型数值） 将以下数值转换为字符串“FALSE”： ● 为 0 的值 ● 空字符串 **示例：** N10 DEF BOOL BOOL_VAR=1 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT（“CONTENT OF BOOL_VAR：%B”，BOOL_VAR） 结果： 字符串变量 RESULT 用字符串“CONTENT OF BOOL_VAR：TRUE”描述。
%C：	转换为 ASCII 字符。 **示例：** N10 DEF CHAR CHAR_VAR=“X” N20 DEF STRING[80] RESULT N30 RESULT=SPRINT（“CONTENT OF CHAR_VAR：%C”，CHAR_VAR） 结果： 字符串变量 RESULT 用字符串“CONTENT OF CHAR_VAR：X”描述。
%D：	转换为整型字符串（INTEGER）。 **示例：** N10 DEF INT INT_VAR=123 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT（“CONTENT OF INT_VAR：%D”，INT_VAR） 结果： 字符串变量 RESULT 用字符串“CONTENT OF INT_VAR：123”描述。
%<m>D：	转换为整型字符串（INTEGER）。 此字符串最小长度为 <m> 个字符。空缺位置用空格符填充在左侧。 **示例：** N10 DEF INT INT_VAR=-123 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT（“CONTENT OF INT_VAR：%6D”，INT_VAR） 结果： 字符串变量 RESULT 用字符串“CONTENT OF INT_VAR:xx-123”描述（其中“x”代表空格符）。

%F:	转换为小数字符串（小数点后 6 位）。 必要时，会删除多余的小数位，或用 0 填充空缺的小数位。 **示例：** N10 DEF REAL REAL_VAR=-1.2341234EX+03 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT（"CONTENT OF REAL_VAR: %F"'REAL_VAR） 结果： 字符串变量 RESULT 用字符串"CONTENT OF REAL_VAR: -1234.123400”描述。
%<m>F:	转换为小数字符串（小数点后 6 位），并且最小总长度为 **<m>** 个字符。 必要时，会删除多余的小数位，或用 0 填充空缺的小数位。 如果总长不够 **<m>** 个字符，空缺位置用空格符填充在左侧。 **示例：** N10 DEF REAL REAL_VAR=-1.23412345678EX+03 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT（"CONTENT OF REAL_VAR: %15F”，REAL_VAR） 结果： 字符串变量 RESULT 用字符串"CONTENT OF REAL_VAR: xxx-1234.123457"描述（“X”在示例中表示空格）。
%.<n>F:	转换为小数字符串，小数点后 **<n>** 位。 必要时，会删除多余的小数位，或用 0 填充空缺的小数位。 **示例：** N10 DEF REAL REAL_VAR=-1.2345678EX+03 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT（"CONTENT OF REAL_VAR: %.3F”，REAL_VAR） 结果： 字符串变量 RESULT 用字符串"CONTENT OF REAL_VAR: -1234.568”描述。
%<m>.<n>F:	转换为小数字符串，小数点后 **<n>** 位，并且最小总长度为 **<m>** 个字符。 必要时，会删除多余的小数位，或用 0 填充空缺的小数位。 如果总长不够 **<m>** 个字符，空缺位置用空格符填充在左侧。 **示例：** N10 DEF REAL REAL_VAR=-1.2341234567890EX+03 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT（"CONTENT OF REAL_VAR: %10.2F”，REAL_VAR） 结果： 字符串变量 RESULT 用字符串"CONTENT OF REAL_VAR:xx-1234.12”描述（其中"x”代表空格符）。

%E:	转换为指数形式的十进制值字符串。 小数点前总有一个数字，保留小数点后 6 位。 必要时，会删除多余的小数位，或用 0 填充空缺的小数位。 指数以指令字“EX”开始， 后面带有符号“+”或“-”、两位或三位数字。 **示例：** N10 DEF REAL REAL_VAR=-1234.567890 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT（“CONTENT OF REAL_VAR: %E”，REAL_VAR） 结果： 字符串变量 RESULT 用字符串“CONTENT OF REAL_VAR:-1.234568EX+03”描述。
%<m>E:	转换为指数形式的十进制值字符串，最小总长度为 <m> 个字符。 空缺的字符用空格符填充在左侧。 小数点前总有一个数字，保留小数点后 6 位。 必要时，会删除多余的小数位，或用 0 填充空缺的小数位。 指数以指令字“EX”开始， 后面带有符号“+”或“-”、两位或三位数字。 **示例：** N10 DEF REAL REAL_VAR=-1234.5 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT（“CONTENT OF REAL_VAR: %20E”，REAL_VAR） 结果： 字符串变量 RESULT 用字符串“CONTENT OF REAL_VAR:xxxxxx-1.234500EX+03”描述（其中“x”代表空格符）。
%.<n>E:	转换为指数形式的十进制值字符串。 小数点前总有一个数字，保留小数点后 <n> 位。 必要时，会删除多余的小数位，或用 0 填充空缺的小数位。 指数以指令字“EX”开始， 后面带有符号“+”或“-”、两位或三位数字。 **示例：** N10 DEF REAL REAL_VAR=-1234.5678 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT（“CONTENT OF REAL_VAR: %.2E”，REAL_VAR） 结果： 字符串变量 RESULT 用字符串“CONTENT OF REAL_VAR:-1.23EX+03”描述。
%<m>.<n>E:	转换为指数形式的十进制值字符串，最小总长度为 <m> 个字符。 空缺的字符用空格符填充在左侧。 小数点前总有一个数字，保留小数点后 <n> 位。 必要时，会删除多余的小数位，或用 0 填充空缺的小数位。 指数以指令字“EX”开始， 后面带有符号“+”或“-”、两位或三位数字。 **示例：** N10 DEF REAL REAL_VAR=-1234.5678 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT（“CONTENT OF REAL_VAR: %12.2E”，REAL_VAR） 结果： 字符串变量 RESULT 用字符串“CONTENT OF REAL_VAR:xx-1.23EX+03”描述（其中“x”代表空格符）。

%G:	根据数值范围，转换为小数形式或指数形式的十进制值字符串： 如果原始值的绝对值小于 1.0EX-04 或者大于等于 1.0EX+06，则采用指数形式，否则就采用小数形式。 最多会显示六个有效数字位，必要时四舍五入。 **小数形式示例：** N10 DEF REAL REAL_VAR=1.234567890123456EX-04 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT（"CONTENT OF REAL_VAR: %G"，REAL_VAR） 结果： 字符串变量 RESULT 用字符串"CONTENT OF REAL_VAR: 0.000123457"描述。 **指数形式示例：** N10 DEF REAL REAL_VAR=1.234567890123456EX+06 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT（"CONTENT OF REAL_VAR: %G"，REAL_VAR） 结果： 字符串变量 RESULT 用字符串"CONTENT OF REAL_VAR: 1.23457EX+06"描述。
%<m>G:	根据数值范围，转换为小数形式或指数形式的十进制值字符串（同%G）： 此字符串最小总长为 **<m>** 个字符。 空缺的字符用空格符填充在左侧。 **小数形式示例：** N10 DEF REAL REAL_VAR=1.234567890123456EX-04 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT（"CONTENT OF REAL_VAR: %15G"，REAL_VAR） 结果： 字符串变量 RESULT 用字符串"CONTENT OF REAL_VAR:xxxx0.000123457"描述（其中"x"代表空格符）。 **指数形式示例：** N10 DEF REAL REAL_VAR=1.234567890123456EX+06 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT（"CONTENT OF REAL_VAR: %15G"，REAL_VAR） 结果： 字符串变量 RESULT 用字符串"CONTENT OF REAL_VAR:xxx1.23457EX+06"描述（其中"x"代表空格符）。

%.<n>G:	根据数值范围，转换为小数形式或指数形式的十进制值字符串。 最多显示 **<n>** 个有效位，必要时舍入多余位数。 如果原始值的绝对值小于 1.0EX-04 或者大于等于 1.0EX(+<n>，则采用指数形式，否则就采用小数形式。 **小数形式示例：** N10 DEF REAL REAL_VAR=1.234567890123456EX-04 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT（"CONTENT OF REAL_VAR: %.3G"，REAL_VAR） 结果： 字符串变量 RESULT 用字符串"CONTENT OF REAL_VAR: 0.000123"描述。 **指数形式示例：** N10 DEF REAL REAL_VAR=1.234567890123456EX+03 N20 DEF STRING[80] RESULT N30 RESULT = SPRINT（"CONTENT OF REAL_VAR: %.3G"，REAL_VAR） 结果： 字符串变量 RESULT 用字符串"CONTENT OF REAL_VAR:1.23EX +03"描述。
%<m>.<n>G:	根据数值范围，转换为小数形式或指数形式的十进制值字符串（同 %.<n>G）： 此字符串最小总长为 **<m>** 个字符。 空缺的字符用空格符填充在左侧。 **小数形式示例：** N10 DEF REAL REAL_VAR=1.234567890123456EX-04 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%12.4G",REAL_VAR) 结果： 字符串变量 RESULT 用字符串"CONTENT OF REAL_VAR:xxx0.0001235"描述（其中"x"代表空格符）。 **指数形式示例：** N10 DEF REAL REAL_VAR=1.234567890123456EX+04 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF REAL_VAR:%12.4G",REAL_VAR) 结果： 字符串变量 RESULT 用字符串"CONTENT OF REAL_VAR:xx1.235EX+06"描述（其中"x"代表空格符）。

%.<n>P:	从实型值转换为整型值，小数点后保留 <n> 位。 输出的整型值是 32 位值。 如果初始值不能以 32 位表示，转换会中断，并输出报警。 %.<n>P 格式指令生成的字节串有可能也包含二进制零，因此，生成的整个字符串不再符合 NC 数据类型“字符串”的形式。 所以，它既也不能保存为“字符串”，也不能用 NC 语言的“STRING”指令继续转换。 它唯一的用途是，传送到 WRITE 指令，输出给外部设备，参见下面的示例。 如果<格式_字符串>包含%P 类型的格式说明符，就会按照 MD10750 $MN_SPRINT_FORMAT_P_CODE ，以 ASCII、ISO （DIN6024） 或 EIA （RS244）的字符代码输出整个字符串，但%.<n>P 生成的字符例外。 如果编写了一个不可转换的字符，转换会中断，并发出报警。 **示例：** 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 ... ； 处理错误 N200 WRITE（ERROR,EXT_DEVICE,SPRINT（“INTEGER BINARY CODED：%.3P”，REAL_VAR） N210 IF ERROR <> 0 … ； 处理错误 结果： 字符串“INTEGER BINARY CODED： 'H0001E23A'" 传输到输出设备/ext/dev/1。 十六进制值 0x0001E23A 相当于十进制值 123450。

%<m>.<n>P :	按照机床数据 MD10751 $MN_SPRINT_FORMAT_P_DECIMAL 的设定，将实型值转换为以下字符串： • **<m> + <n>**位的整数字符串**或** • 小数点前最多**<m>** 位，小数点后精确 **<n>**位的小数字符串。 例如，在写入格式说明符%.<n>P 时，整个字符串保存为 MD10750 $MN_SPRINT_FORMAT_P_CODE 定义的字符代码。 **MD10751 = 0 的转换：** 实型值转换为**<m> + <n>**位的整数字符串，必要时删除多余的小数位或用 0 填充缺少的小数位。 缺少的整数位用空格填充。 负号在左侧插入，正号用空格占位。 **示例：** N10 DEF REAL REAL_VAR=-123.45 N20 DEF STRING[80] RESULT N30 RESULT=SPRINT(“PUNCHED TAPE FORMAT: %5.3P”, REAL_VAR) 结果： 字符串变量 RESULT 用字符串“PUNCHED TAPE FORMAT:-xx123450”描述（其中“x”代表空格符）。 **MD10751 = 1 的转换：** 实型值转换为小数点前最多 **<m>** 位，小数点后精确 **<n>**位的小数字符串，必要时会删除多余的小数位和整数位，或用 0 填充缺少的位数。 如果 **<n>** 为 0，也可以舍去小数点。 **示例：** 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） 结果： 字符串变量 RESULT 用字符串“PUNCHED TAPE FORMAT:-123.450 VAR2:23”描述。
%S:	插入字符串。 **示例：** N10 DEF STRING[16] STRING_VAR=“ABCDEFG” N20 DEF STRING[80] RESULT N30 RESULT=SPRINT（“CONTENT OF STRING_VAR: %S”, STRING_VAR） 结果： 字符串变量 RESULT 用字符串“CONTENT OF STRING_VAR:ABCDEFG”描述。

%<m>S:	插入最少<m>位的字符串。 缺少的位数用空格填充。 **示例：** N10 DEF STRING[16] STRING_VAR="ABCDEFG" N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF STRING_VAR: %10S", STRING_VAR) 结果： 字符串变量 RESULT 用字符串"CONTENT OF STRING_VAR:xxxABCDEFG"描述（其中"x"代表空格符）。
%.<n>S:	输入包含<n>个字符的字符串（从第一个字符开始计数） **示例：** N10 DEF STRING[16] STRING_VAR="ABCDEFG" N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF STRING_VAR: %.3S", STRING_VAR) 结果： 字符串变量 RESULT 用字符串"CONTENT OF STRING_VAR:ABC"描述。
%<m>.<n>S:	输入包含<n>个字符的字符串（从第一个字符开始计数） 生成的字符串最小总长为 < m> 个字符。 缺少的位数用空格填充。 **示例：** N10 DEF STRING[16] STRING_VAR="ABCDEFG" N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("CONTENT OF STRING_VAR: %10.5S", STRING_VAR) 结果： 字符串变量 RESULT 用字符串"CONTENT OF STRING_VAR:xxxxxABCDE"描述（其中"x"代表空格符）。
%X:	整型值转换为十六进制的字符串。 **示例：** N10 DEF INT INT_VAR='HA5B8' N20 DEF STRING[80] RESULT N30 RESULT=SPRINT("INTEGER HEXADECIMAL: %X", INT_VAR) 结果： 字符串变量 RESULT 用字符串"INTEGER HEXADECIMAL:A5B8"描述。

说明

NC 语言的函数名称和指令字不区分大小写，格式说明符也是如此。 因此，大小写对功能没有影响。

组合方法

下表列出了各个格式说明符可以使用的 NC 数据类型。同样，针对数据类型转换的固定规则也适用，参见“数据类型 (页 63)”。

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

说明

从该表可以看出，在“SPRINT”函数中不能直接使用“AXIS”和“FRAME”。 但是可以：

- 通过“AXSTRING”函数首先将“AXIS”转换为字符串，然后用“SPRINT”函数继续转换。
- 可以通过访问 Frame 的分量来读取“FRAME”的各个值， 这样便可获得实型数据，用“SPRINT”函数继续转换。

2.10 程序跳转和分支

2.10.1 跳回到程序开始 (GOTOS)

通过指令 GOTOS 可以用于程序重复时跳回到某个主程序或者子程序的开始处。

通过机床数据可以设置在每次跳回时在程序开始处：

- 程序运行时间设置为“0”。
- 工件计数提高值“1”。

句法

```
GOTOS
```

含义

GOTOS:	带有程序开始跳转目标的跳转指令。	
	该执行通过 NC/PLC 接口信号控制： DB21, ... DBX384.0 (调节程序分支)	
	值：	含义：
	0	没有跳回到程序开始。 程序加工在 GOTOS 后继续进行下一个零件程序段。
	1	跳回到程序开始。 重复零件程序。

边界条件

- GOTOS 触发内部一个 STOPRE（预处理停止）。
- 对于一个带有数据定义（LUD 变量）的零件程序，通过 GOTOS 根据定义段跳转到第一个程序段，即不重新执行数据定义。 为此定义的变量保持了 GOTOS 程序段中达到的值，并且不跳回到定义段中编程的标准值。
- 在同步动作和工艺周期中不提供指令 GOTOS 。

示例

程序代码	注释
N10 ...	；程序开始。

程序代码	注释
...	
N90 GOTOS	；跳转到程序开始。
...	

2.10.2 程序跳转到跳转标记处 (GOTOB, GOTOF, GOTO, GOTOC)

在一个程序中可以设置跳转标记（标签），通过指令 GOTOF, GOTOB, GOTO 或 GOTOC 可以在同一个程序内从其他位置跳转到跳转标记处。然后通过该指令继续程序加工，该指令直接跟随在跳转标记后。因此可以在程序内实现分支。

除了跳转标记外，主程序段号码和旁支程序段号码也可以作为跳转目标。

如果在跳转指令前存在跳转条件（IF ...），则仅在满足跳转条件情况下才进行程序跳转。

句法

```
GOTOB <跳转目标>
IF <跳转条件> == TRUE GOTOB <跳转目标>

GOTOF <跳转目标>
IF <跳转条件> == TRUE GOTOF <跳转目标>

GOTO <跳转目标>
IF <跳转条件> == TRUE GOTO <跳转目标>

GOTOC <跳转目标>
IF <跳转条件> == TRUE GOTOC <跳转目标>
```

含义

GOTOB:	以程序开始方向的带跳转目标的跳转指令。
GOTOF:	以程序末尾方向的带跳转目标的跳转指令。
GOTO:	带跳转目标查找的跳转指令。查找先向程序末尾方向进行，然后再从程序开始处进行查找。
GOTOC:	与 GOTO 有区别的是，报警 14080“跳转目标未找到”被抑制。 这表示，在跳转目标查找没有结果情况下不中断程序加工，而以指令 GOTOC 下面的程序行继续进行。

<跳转目标>:	跳转目标参数 允许的说明有：	
	<跳转标记**>**:	跳转目标是程序中带有用户定义名称的跳转标记：**<**跳转标记>：
	<程序段号码**>**:	主程序段号或者分支程序段号作为跳转目标（比如：200, N300)
	STRING 类型变量：	跳转目标变量。变量提供一个跳转标记或者一个程序段号。
IF:	用于编制跳转条件的关键字。 跳转条件允许使用所有的比较运算和逻辑运算（结果：TRUE 或者 FALSE)。如果这种运算的结果为 TRUE，则执行程序跳转。	

说明

跳转标记（标签）

跳转标记总是位于一个程序段的起始处。如果有程序号，则跳转标记紧跟在程序段号之后。

跳转标记名称有下列规定：

- 字符数：
  - 至少 2 个
  - 最多 32 个
- 允许使用的字符有：
  - 字母
  - 数字
  - 下划线
- 开始的两个字符必须是字母或者下划线。
- 在跳转标记名之后为一个冒号（“：”）。

边界条件

- 跳转目标可能仅仅是一个带跳转标记或者程序段号的程序段，它们位于程序**内**。
- 不带跳转条件的跳转指令必须在一个独立的程序段中编程。带跳转条件的跳转指令不适用于这类限制。在一个程序段中可以编制几个跳转指令。
- 在不带跳转条件的跳转指令的程序中，程序结束 M2/M30 并不一定必须位于程序结束处。

示例

示例 1：跳转到跳转标记

程序代码	注释
N10 …	
N20 GOTOF Label_1	；向程序末尾方向 跳转到跳转标记 “Label_1”。
N30 …	
N40 Label_0:R1=R2+R3	；设置了跳转标记“Label_0”。
N50 …	
N60 Label_1:	；设置了跳转标记“Label_1”。
N70 …	
N80 GOTOB Label_0	；向程序开始方向 跳转到跳转标记 “Label_0”。
N90 …	

示例 2：间接跳转到程序段号

程序代码	注释
IF <条件> == TRUE	
R10=100	；分配跳转目标
ELSE	
R10=110	；分配跳转目标
ENDIF	
；向程序末尾方向跳转到程序段编号为 R10 的程序段	
N10 GOTOF "N"<<R10	
...	
N90 ...	
N100 ...	；跳转目标
N110 ...	
...	

示例 3：跳转到可变的跳转目标

程序代码	注释
DEF STRING[20] ZIEL	
IF <条件> == TRUE	
目标 = “标记 1”	；分配跳转目标
ELSE	
ZIEL = "Marke2"	；分配跳转目标
ENDIF	
；向程序末尾方向跳转到可变的跳转目标“目标内容”	
GOTOF ZIEL	
标记 1：T="孔 1"	；跳转目标 1
...	

程序代码	注释
标记 2：T="孔 2"	；跳转目标 2
...	

示例 4：带跳转条件的跳转

程序代码	注释
N40 R1=30 R2=60 R3=10 R4=11 R5=50 R6=20	；初始值分配
N41 LA1:G0 X=R2*COS(R1)+R5 Y=R2*SIN(R1)+R6	；跳转标记 LA1
N42 R1=R1+R3 R4=R4-1	
；IF 跳转条件 == TRUE	
；THEN 向程序开始方向跳转到跳转标记 LA1	
N43 IF R4>0 GOTOB LA1	
N44 M30	；程序结束

2.10.3 程序分支(CASE ... OF ... DEFAULT ...)

CASE 功能可以检测一个变量或者一个计算函数当前值 (类型： INT），根据结果跳转到程序中的不同位置。

句法

```
CASE(<表达式>) OF <常量_1> GOTOF <跳转目标_1> <常量_2> GOTOF <跳转目标_2> ... DEFAULT GOTOF <跳转目标_n>
```

含义

CASE:	跳转指令	
<表达式>:	变量或计算函数	
OF:	用于编制有条件程序分支的关键字	
<常量_1>:	变量或者计算函数首先规定的恒定值	
	类型：	INT
<常量_2>:	变量或者计算函数第二个规定的恒定值	
	类型：	INT
DEFAULT:	对于变量或者计算函数没有采用规定值的情况,可以用 DEFAULT 指令确定跳转目标。 **提示：** 如果 DEFAULT 指令没有被编程，在这些情况中紧跟在 CASE 指令之后程序段将成为跳转目标。	

GOTOF:	以程序末尾方向的带跳转目标的跳转指令。 代替 GOTOF 可编程所有其他的 GOTO 指令（参见主题“程序跳转到跳转标记”）。	
<跳转目标_1>:	当变量值或者计算函数值符合第一个规定的常量，程序分支到跳转目标。 可以如下规定跳转目标：	
	<跳转标记**>**:	跳转目标是程序中带有用户定义名称的跳转标记： **<**跳转标记**>**：
	<程序段号码**>**:	主程序段号或者分支程序段号作为跳转目标（比如： 200, N300)
	STRING 类型变量：	跳转目标变量。 变量提供一个跳转标记或者一个程序段号。
<跳转目标_2>:	当变量值或者计算函数值符合第二个规定的常量，程序分支到跳转目标。	
<跳转目标_n>:	当变量值不符合规定的常量，程序分支到跳转目标。	

示例

程序代码

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

CASE 指令由 N30 定义下列程序分支可行性：

1. 如果计算函数值 VAR1+VAR2-VAR3 = 7，则跳转到带有跳转标记定义的程序段 "Label_1" (→ N40)。
2. 如果计算函数值 VAR1+VAR2-VAR3 = 9，则跳转到带有跳转标记定义的程序段 "Label_2" (→ N50)。
3. 如果计算函数 VAR1+VAR2-VAR3 的值既不等于 7 也不等于 9，则跳转到带有跳转标记定义的程序段“Label_3”(→ N60)。

2.11 程序部分重复 (REPEAT, REPEATB, ENDLABEL, P)

程序部分重复是指在一个程序中，可以任意组合重复已经编写的程序部分。

需要重复的程序行或程序段落带有跳转标记（标签）。

说明

跳转标记（标签）

跳转标记总是位于一个程序段的起始处。 如果有程序号，则跳转标记紧跟在程序段号之后。

跳转标记名称有下列规定：

- 字符数：
  - 至少 2 个
  - 最多 32 个
- 允许使用的字符有：
  - 字母
  - 数字
  - 下划线
- 开始的两个字符必须是字母或者下划线。
- 在跳转标记名之后为一个冒号（“：”）。

句法

1. 重复单个程序行：

```
<跳转标记>: ...
...
REPEATB <跳转标记> P=<n>
...
```

2. 重复跳转标记和 REPEAT 指令之间的程序段落：

```
<跳转标记>: ...
...
REPEAT <跳转标记> P=<n>
...
```

3. 重复两个跳转标记间的段落：

```
<起始跳转标记>: ...
...
<结束跳转标记>: ...
...
```

```
REPEAT <起始跳转标记> <结束跳转标记> P=<n>
...
```

说明

REPEAT 指令不能被括在这两个跳转标记之间。 如果在 REPEAT 指令前找到了<起始跳转标记>，但在 REPEAT 指令前没有找到<结束跳转标记>，则重复<起始跳转标记>和 REPEAT 指令之间的程序段落。

4. 重复跳转标记和 ENDLABEL 间的段落：

```
<跳转标记>: ...
...
ENDLABEL:...
...
REPEAT <跳转标记> P=<n>
...
```

说明

REPEAT 指令不能被括在<跳转标记>和 ENDLABEL 之间。 如果在 REPEAT 指令前找到了<跳转标记>，但在 REPEAT 指令前没有找到 ENDLABEL，则重复<跳转标记>和 REPEAT 指令之间的程序段落。

含义

REPEATB:	重复程序行的指令
REPEAT:	重复程序段落的指令

<跳转标记>:	<跳转标记>标出了： • 需要重复的程序行（REPEATB） 或者 • 需要重复的程序段落的开始（REPEAT）
	标有<跳转标记>的程序行可以位于 REPEAT-/REPEATB 的前面或后面。 首先在向程序起始的方向搜索。 如果在这个方向没有找到跳转标记，则向程序末尾方向搜索。 **例外：** 如果需要重复跳转标记和 REPEAT 指令之间的程序段落（参见句法下的第 2 点），带有<跳转标记>的程序行必须位于 REPEAT 指令**之前**，因为此时**只**向程序起始的方向搜索。 如果带<跳转标记>的程序行中还有其它的指令，在每次重复时都会重新执行这些指令。
ENDLABEL:	标出需要重复的程序段落结尾的关键字 如果带 ENDLABEL 的程序行中还有其它的指令，在每次重复时都会重新执行这些指令。 在程序中可以多次使用 ENDLABEL。
P:	指定重复数量的地址
<n>:	程序部分重复的次数
	类型： INT
	程序部分会重复<n>次。 在重复最后一次之后，继续执行 REPEAT-/REPEATB 行之后的语句。 **提示：** 如果没有指定 P=<n>，则程序部分仅重复一次。

示例

示例 1： 重复单个程序行

程序代码	注释
N10 POSITION1: X10 Y20	
N20 POSITION2: CYCLE(0,,9,8)	; 位置循环
N30 ...	
N40 REPEATB POSITION1 P=5	; 执行程序段 SATZ N10 五次。
N50 REPEATB POSITION2	; 执行程序段 N20 一次。
N60 ...	
N70 M30	

示例 2： 重复跳转标记和 REPEAT 指令之间的程序段落

程序代码	注释
N5 R10=15	
N10 Begin: R10=R10+1	; 宽度
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	; 执行 N10 到 N70 程序部分四次。
N90 Z10	
N100 M30	

示例 3： 重复两个跳转标记间的段落

程序代码	注释
N5 R10=15	
N10 Begin:R10=R10+1	; 宽度
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	; 执行 N10 到 N70 程序部分三次。
N110 Z10	
N120 M30	

示例 4： 重复跳转标记和 ENDLABEL 间的段落

程序代码	注释
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	

程序代码	注释
N120 Y30	
N130 REPEAT BEGIN3 P=3	; 执行 N110 到 N120 程序部分三次。
N140 REPEAT BEGIN2 P=2	; 执行 N50 到 N80 之间的程序部分两次。
N150 M100	
N160 REPEAT BEGIN1 P=2	; 执行 N20 到 N80 之间的程序部分两次。
N170 Z10	
N180 X0 Y0	
N190 M30	

示例 5： 铣削加工、采用不同的工艺加工钻孔位置

程序代码	注释
N10 ZENTRIERBOHRER()	; 换上定中钻头。
N20 POS_1:	; 钻孔位置 1
N30 X1 Y1	
N40 X2	
N50 Y2	
N60 X3 Y3	
N70 ENDLABEL:	
N80 POS_2:	; 钻孔位置 2
N90 X10 Y5	
N100 X9 Y-5	
N110 X3 Y3	
N120 ENDLABEL:	
N130 BOHRER()	; 更换钻头和钻孔循环。
N140 GEWINDE(6)	; 换上螺纹钻 M6 和螺纹循环。
N150 REPEAT POS_1	; 重复程序部分一次，自 POS_1 到 ENDLABEL。
N160 BOHRER()	; 更换钻头和钻孔循环。
N170 GEWINDE(8)	; 换上螺纹钻 M8 和螺纹循环。
N180 REPEAT POS_2	; 重复程序部分一次，自 POS_2 到 ENDLABEL。
N190 M30	

其它信息

- 程序部分重复可以嵌套调用。 每次调用占用一个子程序级。
- 如果在执行程序重复过程中编程了 M17 或者 RET，则程序重复被停止。 程序接着从 REPEAT 指令行之后的语句开始运行。
- 在当前的程序显示中，程序重复部分作为单独的子程序级显示。
- 如果在执行程序部分重复过程中取消该级别，则在调用程序部分执行之后，继续加工该程序。
  示例：

程序代码	注释
N5 R10=15	
N10 BEGIN: R10=R10+1	; 宽度
N20 Z=10-R10	
N30 G1 X=R10 F200	
N40 Y=R10	; 级别取消
N50 X=-R10	
N60 Y=-R10	
N70 END: Z10	
N80 Z10	
N90 CYCLE(10,20,30)	
N100 REPEAT BEGIN END P=3	
N120 Z10	; 继续程序加工。
N130 M30	

- 控制结构和程序部分重复可以组合使用。 但是，两者之间不得产生重叠。 一个程序部分重复应该位于一个控制结构分支之内，或者一个控制结构位于一个程序部分重复部分之内。
- 如果跳转和程序重复部分交织在一起，则程序段按次序执行。 比如说，程序重复部分有一个跳跃，则一直进行加工，直至找到编程的程序结束部分。
  示例：

程序代码
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

说明

REPEAT 指令应位于运行程序段之后。

2.12 控制结构

控制系统自动按编程顺序处理各个程序段。

该顺序可以通过编程可选的程序块和程序循环改变。该控制结构的编程通过关键字 IF、ELSE、ENDIF、LOOP、FOR、WHILE 和 REPEAT 实现。

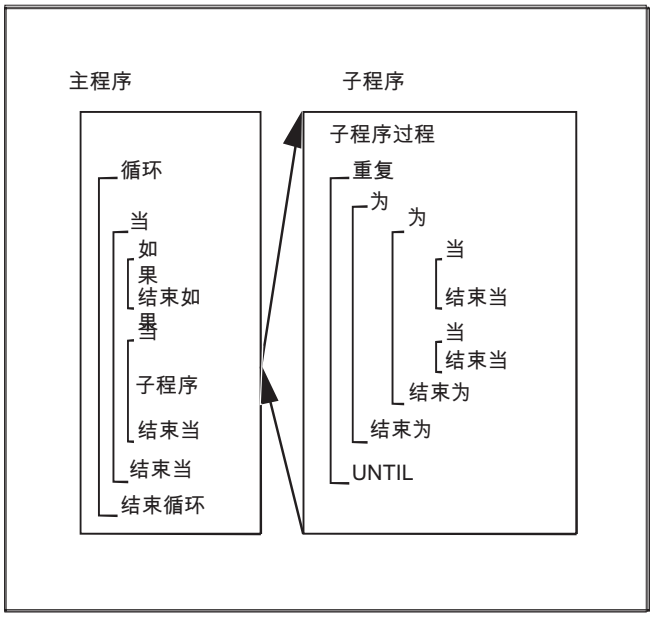
注意
编程错误 控制结构只有在一个程序的指令部分才可能。程序头的定义不能有条件或重复执行。 标准控制结构的关键词和跳转目标一样不能和宏叠加。宏定义时不能进行检测

生效方式

控制结构不可以跨程序使用。

嵌套深度

在每个子程序之内,嵌套的层数可以达到 16 个标准控制结构。



操作时间的实现

在标准有效的翻译操作中,可以通过程序跳转的运用达到比标准控制结构快的程序操作。

在前面汇编的循环中,程序跳转和标准控制结构没有实际的区别。

程序循环时的当前程序段显示

如果在一个程序循环中执行的都只是预处理程序段，则在当前程序段显示中显示的是程序循环**之前**的最后一条主处理程序段。

例如，如果想在当前程序段显示中也一同显示执行完的预处理程序段，以便进行诊断，则必须激活译码单程序段 SBL2。

文档

功能手册 基本功能，章节：BAG，通道，程序运行（K1）> 单程序段 > 译码单程序段 SBL2 带预处理停止

无主处理程序段的循环

如果在程序循环中没有编程主处理程序段，则循环会始终在预处理阶段运行，直至循环条件满足。

这样会导致较高的负载，从而影响显示。

解决方法是，**在循环中**添加指令 STOPRE 或 0 秒的暂停时间 G04。

边界条件

- 带有标准控制结构数组元的程序段不能被跳过。
- 跳转标记（标签）不允许在带控制结构单元的程序段中。
- 标准控制结构被翻译。在识别一个循环结尾时,考虑到所找到的标准控制结构，会寻找循环开头。之后在翻译过程中,模块结构不会完全被检测。
- 建议不要混合使用标准控制结构和程序跳转。
- 在循环的预处理中,会检查控制结构的正确嵌套。

2.12.1 条件指令和分支（IF, ELSE, ENDIF）

条件指令： IF - 程序块 - ENDIF

编写了条件指令时，只有满足特定条件，系统才会执行 `IF` 和 `ENDIF` 之间的程序块。

分支： IF-程序块_1-ELSE-程序块_2-ENDIF

编写了分支指令时，系统总是会执行两个程序块中的一个。

条件满足时，执行 IF 和 ELSE 之间的程序块_1。

条件**未**满足时，执行 ELSE 和 ENDIF 之间的程序块_2。

句法

条件指令

```
IF <条件>
程序块                                    ; 执行： <条件>== TRUE
ENDIF
```

分支

```
IF <条件>
程序块_1                                  ; 执行： <条件>== TRUE
ELSE
程序块_2                                  ; 执行： <条件>== FALSE
ENDIF
```

含义

IF:	引导条件指令或分支。
ELSE:	导入可选的程序块。
ENDIF:	标记条件指令或分支的结束。
<条件>:	逻辑表达式（结果为 TRUE 或 FALSE）。

示例： 刀具更换子程序

程序代码	注释
PROC L6	刀具更换路线
N500 DEF INT TNR_AKTUELL	有效 T 号码变量
N510 DEF INT TNR_VORWAHL	预选 T 号码变量
	确定当前刀具
N520 STOPRE	
N530 IF $P_ISTEST	正在运行程序测试...
N540 TNR_AKTUELL = $P_TOOLNO	... 从程序文本中读取“当前”刀具。
N550 ELSE	否则...
N560 TNR_AKTUELL = $TC_MPP6[9998,1]	... 已读取主轴刀具。
N570 ENDIF	
N580 GETSELT(TNR_VORWAHL)	读取主轴上预选刀具的 T 号码。

程序代码	注释
N590 IF TNR_AKTUELL <> TNR_VORWAHL	如果预选刀具还不是当前刀具，则...
N600 G0 G40 G60 G90 SUPA X450 Y300 Z300 D0	... 回到刀具更换点...
N610 M206	... 并进行刀具更换。
N620 ENDIF	
N630 M17	

2.12.2 无限程序循环（LOOP，ENDLOOP）

无限循环在无限程序中被应用。 在循环结尾总是跳转到循环开头重新进行。

句法

```
LOOP
...
ENDLOOP
```

含义

LOOP:	引入无限循环。
ENDLOOP:	标记循环结束处并跳转到循环开头。

示例

程序代码
...
LOOP
MSG（“无刀沿有效”）
M0
STOPRE
ENDLOOP
...

2.12.3 计数循环（FOR ... TO ...，ENDFOR）

当一个带有一个确定值的操作程序被循环重复，计数循环就会被运行。

句法

```
FOR <变量> = <初值> TO <终值>
...
ENDFOR
```

含义

FOR:	引入计数循环。	
ENDFOR:	一旦还没有得到计数终值，则标记循环结束处并跳转到循环开头。	
<变量>:	计数变量从初值开始向上计数，直到终值且在每次运行时提高值“1”。	
	类型	INT 或 REAL **提示：** 如果为计数循环编程了例如：R 参数， 则采用实数型变量。 如果计数变量为实数变量，则将四舍五入该变量值。
<初值>:	计数的初值 条件 初值必须小于终值。	
<终值>:	计数的终值	

示例

示例 1： 整数变量或 R 参数作为计数变量

整数变量作为计数变量：

程序代码	注释
DEF INT iVARIABLE1	
R10=R12-R20*R1 R11=6	
FOR iVARIABLE1 = R10 TO R11	; 计数变量 = 整数变量
R20=R21*R22+R33	
ENDFOR	
M30	

R 参数作为计数变量：

程序代码	注释
R11=6	
FOR R10=R12-R20*R1 TO R11	; 计数变量 = R 参数（实数变量）

程序代码	注释
R20=R21*R22+R33	
ENDFOR	
M30	

示例 2： 加工一个固定的零件数

程序代码	注释
DEF INT STUECKZAHL	；用名称"STUECKZAHL"定义的 INT 型变量。
FOR STUECKZAHL = 0 TO 100	；引入计数循环。 变量"STUECKZAHL"从初值"0"向上计数，直到终值"100"。
G01 …	
ENDFOR	；计数循环结束。
M30	

2.12.4 在循环开始处带有条件的程序循环（WHILE，ENDWHILE）

WHILE 循环的开始是有条件的。 一旦满足条件，WHILE 循环即开始运行。

句法

```
WHILE <条件>
...
ENDWHILE
```

含义

WHILE:	引入程序循环。
ENDWHILE:	标记循环结束处并跳转到循环开头。
<条件>:	必须满足条件，只有这样 WHILE 循环才能运行。

示例

程序代码	注释
...	
WHILE $AA_IW[钻削轴] > -10	；在下列条件下调用 WHILE 循环： 当前的钻削轴 WKS 额定值必须大于 -10。
G1 G91 F250 AX[钻削轴] = -1	
ENDWHILE	

程序代码	注释
...	

2.12.5 在循环结束处带有条件的程序循环（REPEAT，UNTIL）

REPEAT 循环的结束是有条件的。 REPEAT 循环一旦被执行会不断重复,直到满足条件为止。

句法

```
REPEAT
...
UNTIL <条件>
```

含义

REPEAT:	引入程序循环。
UNTIL:	标记循环结束处并跳转到循环开头。
<条件>:	必须满足条件，只有这样 REPEAT 循环才能运行。

示例

程序代码	注释
...	
REPEAT	; 调用 REPEAT 循环。
...	
UNTIL ...	; 检查是否已满足条件。
...	

2.12.6 带层叠控制结构的程序示例

程序代码	注释
LOOP	
IF NOT $P_SEARCH	; 如果没有程序段搜索
G1 G90 X0 Z10 F1000	
WHILE $AA_IM[X] <= 100	; 当 X 轴 设定值 <= 100 时
G1 G91 X10 F500	;钻孔图

程序代码	注释
Z-5 F100	
Z5	
ENDWHILE	
ELSE	; 否则执行程序段搜索
MSG（“在搜索过程中不钻孔”）	
ENDIF	; ENDIF
$A_OUT[1]=1	;下一个钻孔板
G4 F2	
ENDLOOP	
M30	

2.13 协调指令 (INIT、START、WAITM、WAITMC、WAITE、SETM、CLEARM)

一个 NC 通道原则上能够独立于同一运行方式组（BAG）中之其他通道执行在这个通道中启动的程序。但若一个 BAG 的多个通道中的多个程序同时参与工件加工，则必须在不同的通道中通过以下协调指令实现程序过程的协调。

前提条件

参与程序协调的所有通道必须属于同一运行方式组（BAG）。

MD10010 $MC_ASSIGN_CHAN_TO_MODE_GROUP[<通道>] = <BAG 编号>

通道名称替代通道编号

亦可用在 MD20000 $MC_CHAN_NAME[<通道索引>] 中记录的通道名称来替代通道编号，作为程序协调的预定义程序的参数。必须首先使能“在 NC 程序中使用通道编号”：

MD10280 $MN_PROG_FUNCTION_MASK，位 1 = TRUE

句法

```
INIT(<ChanNr>, <Prog>, <AckMode>)
START(<ChanNr>, <ChanNr>, ...)
WAITM(<MarkNr>, <ChanNr>, <ChanNr>, ...)
WAITE(<ChanNr>, <ChanNr>, ...)
WAITMC(<MarkNr>, <ChanNr>, <ChanNr>, ...)
SETM(<MarkNr>, <MarkNr>, ...)
CLEARM(<MarkNr>, <MarkNr>, ...)
```

含义

INIT():	预定义步骤，用于选择需要在所给定通道中执行的 NC 程序
START():	预定义步骤，用于启动在各个通道中选择的程序
WAITM():	预定义步骤，用于等待到达给定通道中的等待标记 在自身通道中会通过 WAITM 设置给定的等待标记。前一个程序段通过准停结束。等待标记会在同步后被清除。 每个通道最多可同时设置 10 个标记。
WAITE():	预定义步骤，用于等待一个或多个其它通道中的程序结束。
WAITMC(): [1)]	预定义步骤，用于等待到达给定通道中的等待标记 与 WAITM 的不同之处在于，在其它通道尚未到达等待标记的情况下，才会将轴制动至准停状态。

SETM(): [1]	预定义步骤，用于针对通道协调设置一个或多个等待标记 自身通道中的程序执行不受此影响。 SETM 在通道复位和 NC 启动后仍保持其有效性。		
CLEARM(): [1]	预定义步骤，用于清除通道协调的一个或多个等待标记 自身通道中的程序执行不受此影响。 CLEARM() 清除通道中的所有等待标记。 CLEARM(0) 仅清除等待标记“0”。 CLEARM 在通道复位和 NC 启动后仍保持其有效性。		
<ChanNr>:	通道编号 不必给定自身通道的编号。		
	类型：	INT	
<Prog>:	绝对或相对路径说明（**可选**）+ 程序名称		
	类型：	STRING	
	路径说明的相关信息参见： **文档** 编程手册之工作准备分册；章节：“文件和程序管理”>“程序存储器”>“程序存储器文件定址” (页 235)		
<AckMode>:	应答模式（**可选**）		
	类型：	CHAR	
	数值：	"N"	无应答 指令发送后，程序将继续执行。若无法成功执行指令，系统不会通知发送者。
		"S"	同步应答 在接收组件对指令进行应答前，程序将暂停执行。应答为正时会执行下一个指令。应答为负时会输出故障信息。
<MarkNr>:	等待标记编号 **提示** 在一个多通道系统中最多可设置 100 个等待标记（等待标记 0 ... 99）。 在单通道系统中则只有等待标记 0 可供使用。		
1) 为进行用户专用通讯和/或通道协调，借助 SETM / CLEARM 可在不使用有条件等待指令 WAITMC 的情况下使用等待标记。此时，等待标记保留通道复位和 NC 启动及其值。			

示例

通过 MD20000 中的通道名称启动

- 参数设置

```
MD10280 $MN_PROG_FUNCTION_MASK，位 1 = TRUE
$MC_CHAN_NAME[ 0 ] = "BEARBEITUNG"  ；通道 1 的名称
$MC_CHAN_NAME[ 1 ] = "ZUFUEHRUNG"   ；通道 2 的名称
```

- 编程

程序代码	注释
START(BEARBEITUNG)	；通道 1 启动
START(ZUFUEHRUNG)	；通道 2 启动

通过本地“通道名称”和用户变量启动

程序代码	注释
DEF INT MASCHINE = 1	；定义通道 1 的用户变量
DEF INT LADER = 2	；定义通道 2 的用户变量
...	
START(MASCHINE)	；通道 1 启动
START(LADER)	；通道 2 启动

通过本地“通道名称”、用户变量和设置的通道名称启动

程序代码	注释
DEF INT chanNo1	；定义通道 1 的用户变量
DEF INT chanNo2	；定义通道 2 的用户变量
chanNo1 = CHAN_1	；将设置的通道名称分配给通道 1
chanNo2 = CHAN_2	；将设置的通道名称分配给通道 2
...	
START(chanNo1)	；通道 1 启动
START(chanNo2)	；通道 2 启动

带绝对路径说明的 INIT 指令

在通道 2 中选择程序 /_N_MPF_DIR/_N_ABSPAN1_MPF。

程序代码

```
INIT(2,"/_N_WKS_DIR/_N_WELLE1_WPD/_N_ABSPAN1_MPF")
```

带程序名的 INIT 指令

通过名称“MYPROG”选择程序。控制系统通过搜索路径搜索程序。

程序代码

```
INIT(2,"MYPROG")
```

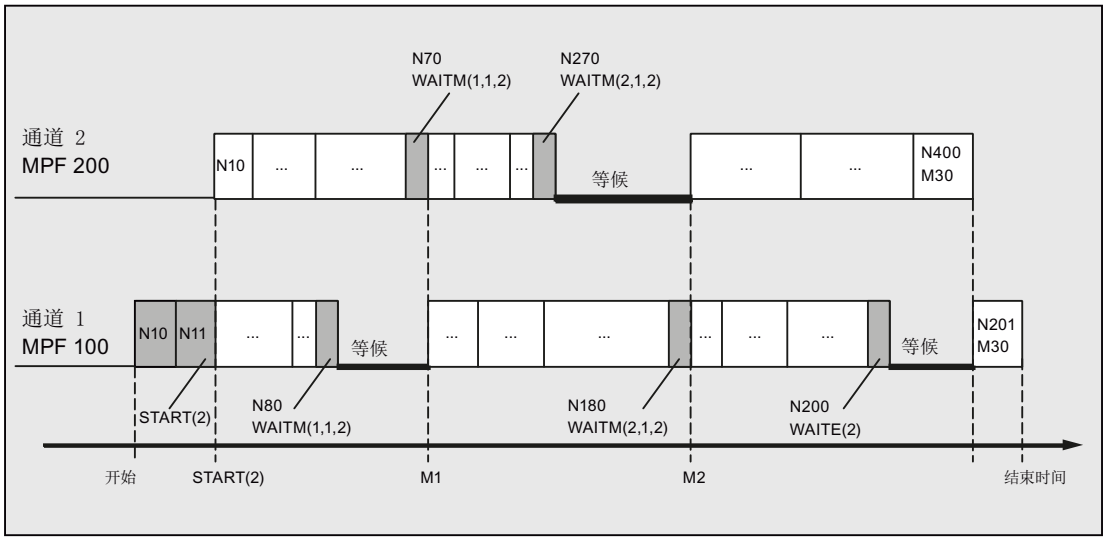
带 WAITM 的程序协调

- 通道 1：已选择并启动程序 /_N_MPF_DIR/_N_MPF100_MPF。

程序代码	注释
	; 程序 MPF100
N10 INIT(2,"MPF200","N")	; 选择程序 MPF200，通道 2
N11 START(2)	; 通道 2 启动
...	
N80 WAITM(1,1,2)	; 等待通道 1 和 2 中的 WAIT 标记 1
N81 ...	; 通道 1，N81 和通道 2，N71 ; 同步开始
...	
N180 WAITM(2,1,2)	; 等待通道 1 和 2 中的 WAIT 标记 2
N181 ...	; 通道 1，N181 和通道 2，N271 ; 同步开始
...	
N200 WAITE(2)	; 等待通道 2 中的程序结束
N201 ...	; N201 在 ; 通道 2 中的 MPF200 程序结束后才开始
N201 M30	; 通道 1 中的程序结束

- 通道 2：在通道 1 中通过程序段 N10 和 N20 选择并启动通道 2 中的程序 MPF200_MPF。

程序代码	注释
;$PATH=/_N_MPF_DIR	; 程序 MPF200
...	
N70 WAITM(1,1,2)	等待通道 1 和 2 中的 WAIT 标记 1
N71 ...	; 通道 1，N81 和通道 2，N71 ; 同步开始
...	
N270 WAITM(2,1,2)	等待通道 1 和 2 中的 WAIT 标记 2
N271 ...	; 通道 1，N181 和通道 2，N271 ; 同步开始
...	
N400 M30	通道 2 中的程序结束



边界条件

在 WAIT 标记后非同步开始执行后续程序段

在通道协调时，通过 WAIT 标记可导致非同步开始执行后续程序段。在待同步之通道中的一个中到达共同的 WAIT 标记前，触发在此通道中导致带有隐性重新定位（REPOSA）的剩余行程删除的动作的情况下，便会出现此特性。

假设：通道 1 和 2 中的当前轴分配

- 通道 1：轴 X1 和 U
- 通道 2：轴 X2

表格 2-2　通道 1 和 2 中的时间顺序

通道 1	通道 2	说明
...	...	通道 1 和 2 中的任意执行
N100 WAITM(20,1,2)		通道 1：到达 WAIT 标记并等待与通道 2 同步
开始执行 GETD(U)： • *跨通道取轴* • *剩余行程删除* • *REPOSA* *结束*	N200 GETD(U)	通道 2：请求通道 1 中的轴 U 通道 1：在后台执行 GET(U)
	N210 WAITM(20,1,2)	通道 2：到达 WAIT 标记。⇒ 为此，通道 1 和 2 的同步已结束
	N220 G0 X2=100	通道 2：开始执行 N220
N110 G0 X1=100		通道 1：时间错开地开始执行 N110

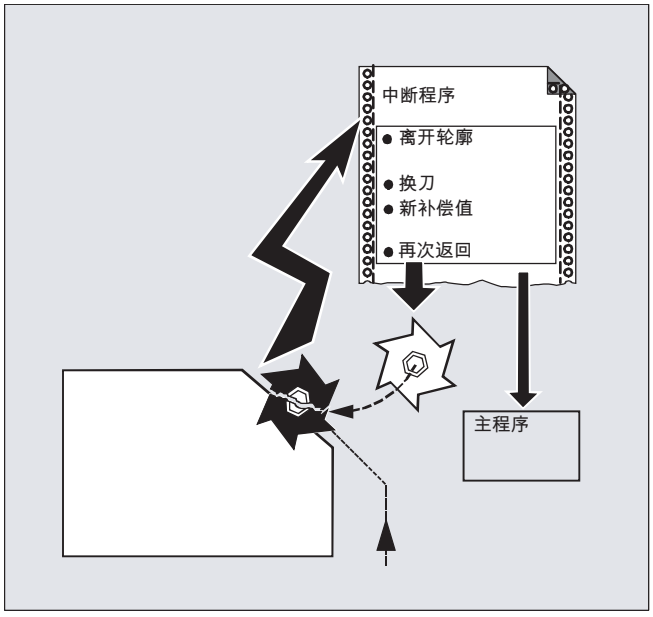
2.14 中断程序（ASUP）

2.14.1 中断程序的功能

说明

在以下说明中交替出现的“异步子程序” (ASUP)和“中断程序”表示同一种功能。

应该依据某个典型示例来阐述中断程序的功能：



在加工过程中工具折断。 由此触发一个信号，这个信号中止正在运行的处理过程并同时开始一个子程序，也就是那个所谓的中断程序。 在这个子程序中有所有在这种情况下应当被执行的指令。

如果子程序已执行完毕（并且因此而恢复运行就绪状态），控制系统就会跳回到主程序中，并且 － 根据 REPOS 指令－ 在中断点继续执行加工。（见“返回轮廓 (页 578)”）。

⚠ 小心

碰撞危险

如果在子程序中没有编程任何 REPOS 指令，则向着程序段的结束点定位，该结束点跟随中断的程序段。

文档

功能手册 基本功能；BAG，通道，程序运行，复位特性（K1），章节：“异步子程序 (ASUP)、中断程序”

2.14.2 建立中断程序

建立作为子程序的中断程序

这个中断程序在定义时和一个子程序一样被标识。

示例：

程序代码	注释
PROC ABHEB_Z	；程序名称“ABHEB_Z”
N10 ...	；紧接着的是 NC 程序段。
...	
N50 M17	；最后结束程序并返回到主程序

备份模态 G 代码 (SAVE)

进行定义时，可使用 SAVE 来标识中断程序。

属性 SAVE 发挥下列作用：在调用中断程序之前保存有效的模态 G 代码，并在结束中断程序之后再次激活（见“带 SAVE 机制（SAVE）的子程序 (页 183)”）。

由此，可以在结束中断程序之后，在中断点继续进行加工。

示例：

程序代码
PROC ABHEB_Z SAVE
N10 ...
...
N50 M17

赋值其它中断程序（SETINT）

可以在中断程序内部编程 SETINT 指令（见 “赋值并启动中断程序（SETINT）” (页 138)），并由此立即接通其它的中断程序。只有通过输入端才可以触发。

文档

有关建立子程序的其它信息可参阅“子程序技术，宏技术”一章。

2.14.3 中断程序赋值和启动(SETINT, PRIO, BLSYNC)

控制系统有多个快速输入（输入 1... 8），每一个输入都触发一个中断（1... 8）。每个中断可以通过指令 SETINT 分配优先级和中断程序。如果中断通过设置快速输入触发，在通道中的当前处理被停止，并启动中断程序。

中断优先级

如果一个零件程序中赋值了多个输入端/中断，则必须为这些中断赋值不同的优先级。

中断时可分配优先级值 1 ... 128。1 是最高优先级值，128 是最低优先级。

句法

```
SETINT(<n>) <名称>
SETINT(<n>) PRIO=<值> <名称>
SETINT(<n>) PRIO=<值> <名称> BLSYNC
SETINT(<n>) PRIO=<值> <名称> LIFTFAST
```

含义

SETINT(<n>):	将 NC 程序 (ASUB) <名称>分配给中断信号 <n>。一旦检测到中断信号 <n> == 1，则启动分配的中断程序。 **注：** 如果一个中断信号 <n> 被另一个中断程序赋值，那么之前的赋值会自动失效。	
<n>:	中断信号编号	
	类型：	INT
	取值范围：	1 ... 32
PRIO= :	中断优先级 （可选）	
<值>:	（**可选**） 优先级值	
	类型：	INT
	取值范围：	1 ... 128（1 ⇒ 最高优先级）
<名称>:	NC 程序 (ASUB) 的名称	

BLSYNC:	（**可选**）BLSYNC 会导致触发中断后首先等待，直至当前程序段执行完毕。之后才执行中断程序。
LIFTFAST:	（**可选**）LIFTFAST 会导致触发中断后首先进行快速退刀（参见章节“快速离开工件轮廓（SETINT LIFTFAST，ALF） (页 141)”）。之后才执行中断程序。

边界条件

中断规则

1. 对于每个不能被立即处理的中断或当前已在处理的中断，另一个中断请求将被存储。 该中断的任何进一步中断请求丢失。
2. 如果当前在处理一个中断，并触发一个更高优先级的中断，则优先级较低的中断停止。较高优先级的中断结束后，低优先级的中断将继续。在处理高优先级的中断时，如果出现低优先级中断的进一步要求，则该要求被存储。其他请求丢失。
3. 如果当前在处理一个中断，并触发一个更高优先级的中断，则优先级较低的中断停止。较高优先级的中断被处理。如果一个较高优先级的中断再次被触发，则当前中断被停止，处理较高优先级的中断。最多六个激活的中断等级。当前处理的中断等级和五个等待的中断等级。对于每个激活的中断等级最多可存储一个进一步的中断请求。所有其他的中断请求都将丢失。同样，如果为其他中断等级请求（中断等级≥ 7），则该中断请求丢失。

示例

示例 1：赋值中断程序和确定优先级

程序代码	注释
N20 SETINT(3) PRIO=1 ABHEB_Z	; 如果接通了输入端 3 == 1
	，则启动中断程序 “ABHEB_Z”
N30 SETINT(2) PRIO=2 ABHEB_X	; 如果接通了输入端 2 == 1
	，则启动中断程序 “ABHEB_X”。

如果多个输入端同时保留，则中断程序会根据级别数的顺序进行处理。首先是“ABHEB_Z”，然后是“ABHEB_X”。

示例 2：重新赋值中断程序

程序代码	注释
N20 SETINT(3) PRIO=2 ABHEB_Z	; 如果接通了输入端 3 == 1
	，则启动中断程序 **“ABHEB_Z”**
...	
N80 SETINT(3) PRIO=1 ABHEB_X	; 如果接通了输入端 3 == 1
	，则启动中断程序 **“ABHEB_X”**

2.14.4 取消/再激活一个中断程序的赋值（DISABLE，ENABLE）

SETINT 指令可以通过 DISABLE 取消，并通过 ENABLE 再次激活，不会丢失输入端 → 中断程序的赋值。

句法

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

含义

DISABLE(<n>):	指令：**取消** 中断程序输入端的赋值 **<n>**	
ENABLE(<n>):	指令：**再次激活** 中断程序输入端的赋值 **<n>**	
<n>:	参数：中断信号编号	
	类型：	INT
	取值范围：	1 ... 32

示例

程序代码	注释
N20 SETINT(3) PRIO=1 ABHEB_Z	；如果接通了输入端 3，则应该 启动中断程序 “ABHEB_Z”。
...	
N90 DISABLE(3)	；取消 N20 中的 SETINT 指令。
...	
N130 ENABLE(3)	；再次激活 N20 中的 SETINT 指令。
...	

2.14.5 删除中断程序的赋值（CLRINT）

用 SETINT 定义的将中断信号分配给 NC 程序 (ASUB) 可通过 CLRINT 删除。

句法

```
CLRINT(<n>)
```

含义

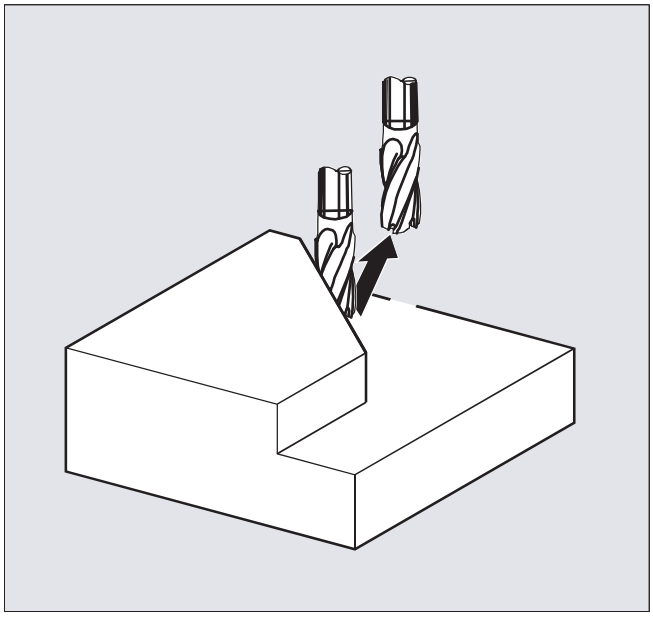
CLRINT(<n>):	指令：删除将中断信号 **<n>** 分配给通过 SETINT 定义的 NC 程序 (ASUB) **<n>**	
<n>:	参数：中断信号编号	
	类型：	INT
	取值范围：	1 ... 32

示例

程序代码	注释
N20 SETINT(3) PRIO=2 ABHEB_Z	
...	
N50 CLRINT(3)	；输入端“3”和程序“ABHEB_Z”之间的赋值被删除。

2.14.6 快速离开工件轮廓（SETINT LIFTFAST，ALF）

如果 SETINT 指令带 LIFTFAST，在接通输入端时，通过快速从工件轮廓离开的方式使刀具离开。



其它的过程与 LIFTFAST 旁的 SETINT 指令是否包含一个中断程序有关：

带中断程序： 在快速离开**之后**执行中断程序。

不带中断程序： 在快速离开之后加工停止并发出报警。

句法

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

含义

SETINT(<n>):	指令： 赋值中断程序输入端 **<n>**。 当接通输入端 **<n>**时，启动赋值的中断程序。	
<n>:	参数： 输入端编号	
	类型：	INT
	取值范围：	1 ... 8
PRIO= ：	确定优先级	
<值>:	优先级值	
	取值范围：	1 ... 128
	优先级 1 相当于最高优先级。	
<名称>:	需要处理的子程序（中断程序）名称。	
LIFTFAST:	指令： 快速离开工件轮廓	
ALF=… :	指令： 可编程的运动方向（在运动程序段中） 有关用 ALF 编程的方法，见主题“快速离开工件轮廓时的运行方向 (页 144)”。	

边界条件

带镜像的有效框架的性能

在确定离开方向时会检测,是否有一个框架带镜像被激活。 在这种情况下，刀具沿正切线离开，左右相间。 在刀具方向的方向分量没有镜像。 通过 MD 设置激活该性能：

MD21202 $MC_LIFTFAST_WITH_MIRROR = TRUE

示例

折断的刀具自动地被另一个刀具替代。 加工以新的刀具继续进行。

主程序：

主程序	注释
N10 SETINT(1) PRIO=1 W_WECHS LIFTFAST	；如果接通输入端 1 ，刀具会立刻以快速离开（代码 7 对应工具半径补偿 G41）的方式离开工件轮廓。 然后中断程序"W_WECHS"被执行。
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	

子程序：

子程序	注释
PROC W_WECHS SAVE	；带当前运行状态储存的子程序
N10 G0 Z100 M5	；换刀位置，主轴停止
N20 T11 M6 D1 G41	;更换刀具
N30 REPOSL RMBBL M3	；返回轮廓并跳转到主程序中（在一个程序段中编程）

2.14.7 快速离开工件轮廓时的运行方向

后退运行

退回运动的平面由下列 G 代码确定：

- LFTXT
  由轨迹切线和刀具方向来确定退回运动的平面（标准设置）。
- LFWP
  退回运动的平面是用 G 代码 G17，G18 或 G19 选择的、已激活的工作平面。撤回运动的方向不由轨道切线决定。由此可以编程一个与轴并行的快速离开。
- LFPOS
  使通过 POLFMASK / POLFMLIN 指明的轴回到用 POLF 编程的绝对轴位置。
  ALF 在多个轴以及多个线性相关轴上时对退刀方向没有影响。
  文档:
  编程手册 基本原理；章节：“螺纹切削时的快速返回”

可编程的运行方向（ALF=...）

在退回平面中，用 ALF 以 45 度的不连续步骤对方向进行编程。

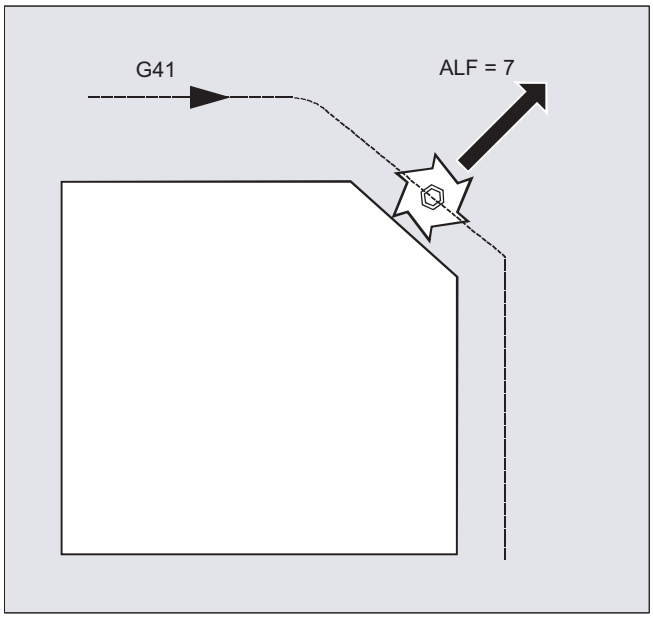
可能的运行方向存储在控制系统中，带专门的代码号，并可以在这个代码下调用。

示例：

程序代码

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

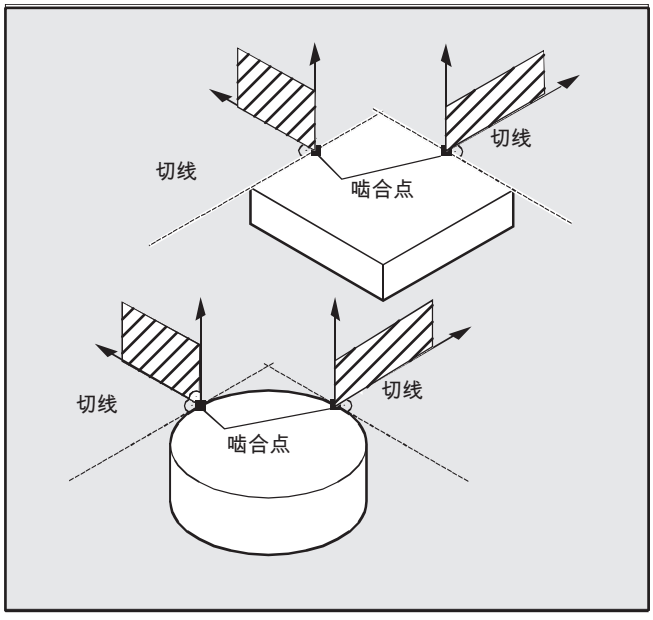
刀具在启用了 G41 的情况下（从轮廓左侧加工方向）垂直从轮廓上离开。



LFTXT 下用于描述运行方向的基准面

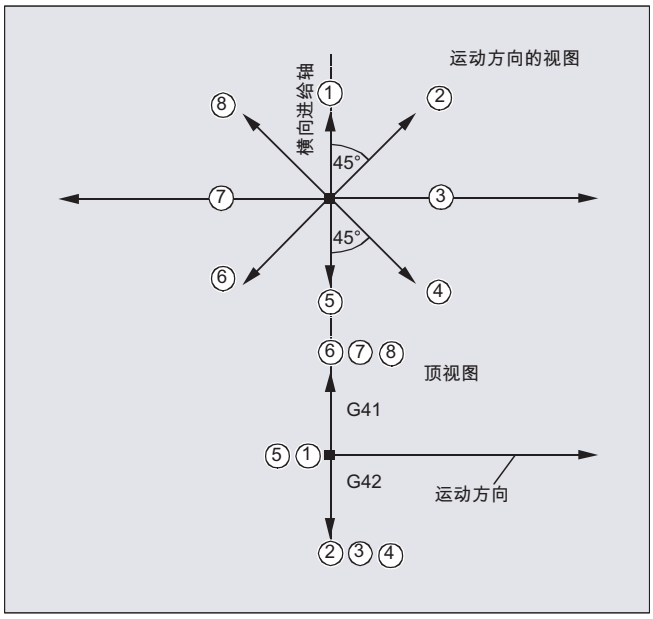
工具在编程的轮廓上的切入点有一个平面,它作为带相应代码离开运动的参数说明的基准面。

这个基准面由工具径向轴(进刀方向)和一个矢量组成,这个矢量与这个平面相对并与工具在轮廓上的切入点的切线垂直。



LFTXT 下带运行方向的代码编号

从这个基准面出发您可以在下面的插图里找到带运行方向的代码编号。



对于 ALF=1，后退在刀具方向中确定。

用 ALF=0 取消“快速离开”功能。

⚠ 小心

碰撞危险

对于已接通的刀具半径补偿，应该：

- 对于 G41，编码 2，3，4
- 对于 G42，编码 6，7，8

不会 被使用，因为在这些情况下刀具驶向轮廓并会与工件相撞。

LFWP 下带运行方向的代码编号

使用 LFWP 时，工作平面中的方向被分配如下：

- G17:X/Y 平面
  ALF=1：在 X 方向后退
  ALF=3：在 Y 方向后退
- G18:Z/X 平面
  ALF=1：在 Z 方向后退
  ALF=3：在 X 方向后退
- G19:Y/Z 平面
  ALF=1：在 Y 方向后退
  ALF=3：在 Z 方向后退

2.14.8 中断程序下的运动过程

没有 LIFTFAST 的中断程序

轴在轨迹上运动，直至在停止状态中停止。 接着启动中断程序。

停止状态位置被保存为中断位置，并且在 REPOS 下，用 RMIBL 在中断程序结束时向该位置逼近。

带 LIFTFAST 的中断程序

轴运动在轨迹上停止。 同时，LIFTFAST 运动作为叠加运动被执行。 如果轨迹运动和 LIFTFAST 运动停止，则启动中断程序。

轮廓上的位置作为中断位置被保存, 在这个位置上开始 LIFTFAST 运动并由此离开轨迹。

带有 LIFTFAST 和 ALF=0 的中断程序与没有 LIFTFAST 的中断程序有一样的特性。

说明

几何轴快速离开工件轮廓时所移动的距离，可以通过机床数据设定。

2.15 交换轴，交换主轴 (RELEASE, GET, GETD)

一个或多个进给轴和主轴总是只能在一个通道中进行插补。 如果某个轴必须在两个通道中交替工作（例如：换刀器），则必须首先在当前通道中将其释放，然后将其传送到另一个通道中。 轴会在两个通道之间来回切换。

扩展取轴

一个进给轴/主轴可以通过预处理停止和同步动作在预运行和主运行之间切换，或者也可以不通过预处理停止进行切换。 此外，也可以通过下列方式取轴

- 轴容器旋转指令 AXCTSWE 或者 AXCTWED，指令隐含有 GET/GETD。
- 含旋转的框架指令，如果该轴还可其他轴关联。
- 同步动作，参见运动同步动作“跨通道取轴 RELEASE，GET”。

机床制造商

请注意机床制造商说明。 轴必须事先通过机床数据明确定义，才可以跨通道取轴。取轴方式也可以通过机床数据设置。

句法

RELEASE(轴名称，轴名称，...) 或者 RELEASE(S1)

GET (轴名称，轴名称，...) 或者 GET (S2)

GETD（轴名称，轴名称 ...）或者 GETD（S3）

用 GETD(GET Directly) 将一个轴从另一个通道中直接取出。 这就是说，在另一个通道中不必为该 GETD 指令编程配套的 RELEASE。 不过这也意味着，现在必须建立另一个通道通讯（例如等待标记）。

含义

RELEASE (轴名称, 轴名称, …):	释放轴
GET (轴名称, 轴名称, …):	传送轴
GETD (轴名称, 轴名称, …):	直接传送轴
轴名称:	系统中的轴指定： AX1, AX2, … 或者机床轴名称
RELEASE(S1)：	释放主轴 S1,S2,...
GET(S2)：	传送主轴 S1,S2,…
GETD(S3)：	直接传送主轴 S1,S2,...

无预处理停止的 GET 指令

如果在一个**无**预处理停止的 GET 指令后通过 RELEASE(轴)或 WAITP(轴)再次释放轴，则接下来的 GET 指令会变为**带**预处理停止的 GET 指令。

> ⚠ 小心
>
> **轴分配改变**
>
> 即使在按键复位或者程序复位之后，使用 GET 传送的进给轴和主轴也会保留在当前的通道中。
>
> 重新启动程序后，如果在基本通道中需要使用轴，就必须通过编程来取回轴。
>
> 在重新上电后，轴分配给机床数据中指定的通道。

示例

示例 1： 在两个通道之间取轴

6 个轴在通道 1 中用于加工的为：1., 2., 3. 和第 4 个轴。
第 5 和第 6 个轴用于通道 2 中的工件切换。

现在 ，第 2 轴要换到另一个通道，并且在重新上电后再次分给通道 1。

通道 1 中的程序“MAIN”：

程序代码	注释
INIT (2,"GET2")	; 在通道 2 中选择程序“GET2”。
N… START (2)	; 在通道 2 中启动程序。
N… GET (AX2)	; 传送轴 AX2。
...	
N… RELEASE (AX2)	; 释放轴 AX2。
N… WAITM (1,1,2)	; 等待通道 1 和通道 2 中的 WAIT 标记，以便两个通道同步。
...	; 取轴之后的后续流程。
N… M30	

通道 2 中的程序“GET2”：

编程	注释
N… RELEASE(AX2)	
N160 WAITM (1,1,2)	; 等待通道 1 和通道 2 中的 WAIT 标记，以便两个通道同步。
N150 GET (AX2)	; 传送轴 AX2。
...	; 取轴之后的后续流程。

编程	注释
N… M30	

示例 2： 没有同步的取轴

如果轴无须同步，则 GET 不会产生预处理停止。

编程	注释
N01 G0 X0	
N02 RELEASE(AX5)	
N03 G64 X10	
N04 X20	
N05 GET(AX5)	；当不需要同步时，该程序段变为不可执行的程序段。
N06 G01 F5000	；不可执行的程序段。
N07 X20	；不可执行的程序段，因为 X 轴位置与 N04 中的一样。
N08 X30	；在 N05 之后的第一个可执行的程序段。
...	

示例 3： 激活无预处理停止的取轴

前提条件： 无预处理停止的取轴必须通过机床数据定义。

编程	注释
N010 M4 S100	
N011 G4 F2	
N020 M5	
N021 SPOS=0	
N022 POS[B]=1	
N023 WAITP[B]	；轴 B 变成中立轴。
N030 X1 F10	
N031 X100 F500	
N032 X200	
N040 M3 S500	；轴不触发预处理停止/重组。
N041 G4 F2	
N050 M5	
N099 M30	

如果主轴或者 B 轴在程序段 N023 后立即变为 **PLC 轴**，例如运行到 180 度且返回到 1 度，则该轴会重新变为中立轴，在程序段 N40 中不触发预处理停止。

其它信息

轴交换的前提

- 轴必须已经通过机床数据在所有要使用该轴的通道中定义好。
- 必须通过 **achs** 特定的机床数据确定在 POWER ON 之后将轴分配给哪个通道。

说明

释放轴： RELEASE

在轴使能时必须要注意:

1. 轴不可以参加转换。
2. 在轴耦合时(正切控制),所有相关轴都必须使能。
3. 一个参与的定位轴在这种状态下不能交换。
4. 在龙门架主轴机床中，所有跟随轴也被交换。
5. 在轴耦合时(联动,引导轴耦合,电子齿轮)只有相连的引导轴被使能。

接受轴： GET

用这个命令执行原来的轴交换。 完全由已在其中编程了该指令的通道来负责轴。

GET 的作用

带同步的轴变换:

当某个轴临时处在另外一个通道中或者分配给了 PLC、且在 GET 之前没有通过 "WAITP", G74 或者删除剩余行程的方式进行同步时，才必须对该轴进行同步。

- 进给停止（与 STOPRE 相同）。
- 在交换完全执行之前，加工始终保持中断状态。

自动的"GET"

如果一个轴在通道中原则上可用,但是当时实际上不是作为挨 如果这个（些）轴已经被同步，就不会产生进给停止。

设置可修改的轴交换属性。

轴的交换时刻可通过机床数据如下设置：

- 如果轴通过 WAITP 处于一个中性状态(与前面的性能一样),那么也可以在两个通道之间进行自动的轴变换。
- 当某个轴容器旋转请求可由执行的通道分配的轴容器所有轴通过隐式 GET 和 GETD 指令取出放入通道中。 随后的轴交换仅允许在结束轴容器旋转后进行。
- 在主程序中插入一个临时程序段之后，检查是否已成功进行了重新编组。 只有当该程序段的轴状态与当前的轴状态不 一致时，才有必要进行重新编组。
- 也可以在不停止进给的情况下进行轴交换，而无需带进给停止和进给与主程序同步的 GET 程序段。 然后只生成带 GET 指令的临时程序段。 在主程序中处理该程序段时，检查程序段中的轴状态是否与当前轴状态一致。

轴或主轴交换功能的其它信息参见
功能手册扩展功能；BAGs、通道、轴交换（K5）。

2.16 将轴移交到另一个通道中（AXTOCHAN）

用语言指令 AXTOCHAN 可以把轴指定给一个特定通道，以此把轴移到另一个通道。 该轴可以从 NC 零件程序以及同步动作中移到相应的通道。

句法

```
AXTOCHAN（轴名称，通道名称，[，轴名称，通道名称[，...]]）
```

含义

元素	说明
AXTOCHAN:	指定轴为某一特定通道
轴名称:	系统中的轴指定： X，Y，... 或者参与的加工轴名称的数据。 待执行的通道不必是其自身通道，也不必是当前具有该轴插补权的通道。
通道编号：	要给轴分配的通道号

说明

参与的定位轴和仅由 PLC 控制的轴

一个作为参与定位轴的 PLC 轴不能更换通道。 一个仅由 PLC 控制的轴不能分配给 NC 程序。

文档：
功能手册 扩展功能；定位轴（P2）

示例

NC 程序中的 AXTOCHAN

轴 X 和 Y 在通道 1 和 2 中已知。 当前通道 1 具有插补权且将在通道 1 中启动下列程序：

程序代码	注释
N110 AXTOCHAN(Y,2)	; Y 轴移向通道 2。
N111 M0	
N120 AXTOCHAN(Y,1)	; 重新取回 Y 轴（中性）。
N121 M0	
N130 AXTOCHAN(Y,2,X,2)	; Y 轴和 X 轴移到通道 2（轴中性）。
N131 M0	
N140 AXTOCHAN(Y,2)	; Y 轴移向通道 2(NC 程序)。
N141 M0	

其它信息

NC 程序中的 AXTOCHAN

对于在自身通道中的 NC 程序，仅当轴请求时，执行 GET，并由此等待真正的状态改变。如果轴被要求用于另一个通道或者要变成自身通道中的中性轴时，取消相应指令。

同步动作的 AXTOCHAN

如果要求轴用于自身通道时，则将来自同步动作的 AXTOCHAN 映像到同步动作的 GET。 在这种情况下，轴在首个用于自身通道的请求时成为中性轴。 第二个请求时，把轴分配给 NC 程序，与 NC 程序中的 GET 指令类似。 关于同步动作的 GET 指令参见章节“运动同步动作”。

2.17 有效设置机床数据（NEWCONF）

使用指令 NEWCONF 可使所有机床数据生效。也可在操作界面 HMI 中通过按下软键“激活机床数据”的方式来激活该功能。

当执行功能 NEWCONF 时，会出现隐式预处理停止，即轨迹运动会被中断。

句法

```
NEWCONF
```

含义

NEWCONF:	激活所有生效级为“NEW_CONFIG”的机床数据的指令

跨通道执行零件程序中的 NEWCONF

如果改变了零件程序的轴机床数据，并随即用 NEWCONF 激活，则 NEWCONF 仅激活会导致零件程序通道改变的机床数据。

说明

为了确保所有的更改有效，必须在每个通道中执行 NEWCONF 指令，在这些通道中和机床数据更改相关的轴或者功能也被即时计算。

对于 NEWCONF 无轴向机床数据设置为有效。

由 PLC 控制的轴必须执行轴复位。

示例

铣削加工:用不同的工艺加工钻孔的位置。

程序代码	注释
N10 $MA_CONTOUR_TOL[AX]=1.0	; 更改机床数据。
N20 NEWCONF	;有效设置机床数据。
...	

2.18 写入文件（WRITE）

通过 WRITE 指令，可以把 NC 程序中的程序段/数据读写到被动文件系统中或位于一个外部程序存储器上的文件（记录文件）的结尾。 正在处理的程序也可以执行该指令。

说明

如果需要通过 WRITE 指令说明的文件不在程序存储器中，则应先新建该文件。

前提条件

当前所设置的保护级别必须等于或者大于文件的 WRITE 权限。 否则系统会拒绝访问并且显示出错提示（出错变量的返回值 = 13）。

句法

```
DEF INT <错误>
...
WRITE(<错误>,"<文件名称>"/"<外部设备>", "<程序段/数据>")
```

含义

WRITE:	将一个程序段/数据插入到指定文件末尾的指令		
<错误>:	**参数 1：** 返回错误值的变量		
	类型：	INT	
	值：	0	无错误
		1	非法路径
		2	路径未找到
		3	文件未找到
		4	错误的文件类型
		10	文件已满
		11	文件正被使用
		12	没有空余的存储量
		13	无访问权限
		14	输出设备缺少 EXTOPEN 指令或指令出错
		15	写入外部设备出错
		16	写入了无效的外部路径
<文件名称>:	**参数 2：** 插入指定程序段/数据的文件名称		
	类型：	STRING	
	可在原文件名前指定绝对路径。 如果没有路径说明，系统会在当前的目录（=选中程序的目录）中查找文件。 路径说明规定参见“程序存储器文件的定址 (页 235)”。		
<外部设备>：	如果数据需要通过 “Process DataShare” 功能输出到外部设备/文件上，必须输入一个对应外部设备/文件的标识，而不是文件名称。		
	类型：	STRING	
	更多信息参见“Process DataShare - 数据输出到外部设备/文件上（EXTOPEN，WRITE， EXTCLOSE） (页 715)”。 **注：** 标识必须与 EXTOPEN 中输入的标识一致。		

<程序段/数据>:	**参数 3：** 插入指定文件的段/数据。	
	类型：	STRING

说明

在数据输出到被动文件系统中或一个外部程序存储器上时，WRITE 指令会自动在输出字符串的结尾处插入一个 “LF” 字符（LINE FEED = 换行）。

但该方法不适用于通过 “Process DataShare” 功能将数据输出到一个外部设备/文件上。 如果需要“LF”一同输出，必须在输出字符串中明确指定。

→ 参见示例 3： 包含/不包含“LF”！

边界条件

- **最大文件大小(→ 机床制造商!)**
  被动文件系统中允许的最大记录文件在以下机床数据中定义：
  MD11420 $MN_LEN_PROTOCOL_FILE
  最大文件大小的限制适用于所有被动文件系统中通过 WRITE 指令创建的文件。 一旦超出该限制，系统就会输出出错信息，并不再保存程序段或数据。 如果存储器够用，就可以新建一个文件。

示例

示例 1： WRITE 指令，将数据写入被动文件系统，无绝对路径

程序代码	注释
N10 DEF INT ERROR	; 出错变量的定义。
N20 WRITE(ERROR,"PROT","97 年 2 月 7 号的记录")	; 将文本“97 年 2 月 7 号的记录”写入到文件“_N_PROT_MPF” 中
N30 IF ERROR	; 出错分析。
N40 MSG ("执行 WRITE 指令时出错:" <<ERROR)	
N50 M0	
N60 ENDIF	
...	

示例 2： WRITE 指令，将数据写入被动文件系统，有绝对路径

程序代码
...
WRITE(ERROR,"/_N_WKS_DIR/_N_PROT_WPD/_N_PROT_MPF","97 年 2 月 7 号的记录")

程序代码

```
...
```

示例 3： 自动生成的/精确编程的“LF”

a，将数据写入被动文件系统，有自动生成的“LF”

程序代码

```
...
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
```

输出结果：

MY_STRING

MY_STRING

b，将数据写入外部文件，无自动生成的“LF”

程序代码

```
...
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
```

输出结果：

MY_STRINGMY_STRING

c，将数据写入外部文件，有精确编程的“LF”

按照如下编程，就可以得到和“a,”一样的结果：

程序代码

```
...
N200 DEF STRING[30] DEV_1
```

程序代码

```
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
```

输出结果：

MY_STRING

MY_STRING

2.19 删除文件（DELETE）

用 DELETE 指令可以删除所有的文件，无论它是否通过 WRITE 指令产生。 通过更高存取级别产生的文件也可以用 DELETE 删除。

句法

```
DEF INT <错误>
DELETE(<错误>,"<文件名称>")
```

含义

DELETE:	删除指定文件的指令		
<错误>:	返回错误值的变量		
	类型：	INT	
	值：	0	无错误
		1	非法路径
		2	路径未找到
		3	文件未找到
		4	错误的文件类型
		11	文件正被使用
		12	没有空余的存储量
		20	其它错误
<文件名称>:	需要删除的文件的名称		
	类型：	STRING	
	可在原文件名前指定绝对路径。 如果没有路径说明，系统会在当前的目录（=选中程序的目录）中查找文件。 路径说明规定参见“程序存储器文件的定址 (页 235)”。		

示例

程序代码	注释
N10 DEF INT ERROR	; 出错变量的定义。
N15 STOPRE	; 预处理停止。
N20 DELETE(ERROR,"/_N_SPF_DIR/_N_TEST1_SPF")	; 删除子程序目录中的文件 TEST1。
N30 IF ERROR	; 出错分析。
N40 MSG（“执行 DELETE 指令时出错：” <<ERROR)	

程序代码	注释
N50 M0	
N60 ENDIF	

2.20 读取文件中的行（READ）

READ 指令用来在指定文件中读取一个或者多个行，并且将所读取的信息保存在一个 STRING 型数组中。 每个读入的文件行都占用数组中的一个数组元素。

前提条件

当前所设置的保护级别必须等于或者大于文件的 READ 权限。 否则系统会拒绝访问并且显示出错提示（出错变量的返回值 = 13）。

句法

```
DEF INT <错误>
DEF  STRING[<字符串长度>] <结果>[<n>,<m>]
READ(<错误>,"<文件名称>",<起始行>,<行数>,<结果>)
```

含义

READ:	指令，用于读取指定文件中的某些数据行并将它保存到变量数组中。		
<错误>:	返回错误值的变量（Call-By-Reference 引用调用参数）		
	类型：	INT	
	值：	0	无错误
		1	非法路径
		2	路径未找到
		3	文件未找到
		4	错误的文件类型
		11	文件正被使用
		13	访问权限不够
		21	行不存在（参数 <起始行>或<行数>大于指定文件中的总行数）。
		22	结果变量（<结果>）的数组长度太短。
		23	行范围太大（选择的参数<行数>太大，已超出了文件末尾）。

<文件名称>:	待读取的文件的名称（Call-By-Value 值调用参数）		
	类型：	STRING	
	可在原文件名前指定绝对路径。如果没有路径说明，系统会在当前的目录（=选中程序的目录）中查找文件。 路径说明规定参见“程序存储器文件的定址 (页 235)”。		
<起始行>:	待读取的文件范围的起始行（Call-By-Value 值调用参数）		
	类型：	INT	
	值：	0	根据参数<行数>指定的数量，读取文件末尾前的数据行。
		1 … n	第一个要需读取行的编号。
<行数>:	待读取的文件行的数量（Call-By-Value 值调用参数）		
	类型：	INT	
<结果>:	结果变量（Call-By-Reference 引用调用参数） 存有读入的文本的变量数组。		
	类型：	STRING（最大长度： 255)	
	如果参数<行数>中指定的数量小于结果变量中的数组长度 [<n>,<m>]，则剩余的数组元素保持不变。 通过控制符“LF”（换行）或者“CR LF”（回车换行）表示的行尾将不保存在结果变量中。 如果文件行比定义的字符串长度长，读入的文件行会被隔断。 不会出现错误提示。		

说明

二进制文件不能被读入。 系统会输出错误提示“错误的文件类型”（出错变量的返回值 = 4）。 以下的文件类型不可读: _BIN, _EXE, _OBJ, _LIB, _BOT, _TRC, _ACC, _CYC, _NCK.

示例

程序代码	注释
N10 DEF INT ERROR	; 出错变量的定义。
N20 DEF STRING[255] RESULT[5]	; 结果变量的定义。
N30 READ(ERROR,"/_N_CST_DIR/_N_TESTFILE_MPF", 1,5,RESULT)	; 带域标识符和文件标识符的文件名和路径指定。
N40 IF ERROR <>0	; 出错分析。
N50 MSG("错误"<<ERROR<<"READ 指令")	
N60 M0	

程序代码	注释
N70 ENDIF	
...	

2.21 检查文件的存在性(ISFILE)

借助 ISFILE 指令可检查文件是否位于程序存储器中。

句法

<结果>=ISFILE("<文件名>")

含义

ISFILE:	用于检查文件是否存在的指令		
<文件名称>:	应被检查是否存在的文件的名称。		
	类型：	STRING	
	可在原文件名前指定绝对路径。如果没有路径说明，系统会在当前的目录（=选中程序的目录）中查找文件。 路径说明规定参见“程序存储器文件的定址 (页 235)”。		
<结果>:	用于接收检查结果的变量		
	类型：	BOOL	
	值：	TRUE	文件存在
		FALSE	文件不存在

示例

示例 1

程序代码	注释
N10 DEF BOOL RESULT	；结果变量的定义。
N20 RESULT=ISFILE("TESTFILE")	
N30 IF(RESULT==FALSE)	
N40 MSG("文件不存在")	
N50 M0	
N60 ENDIF	
...	

示例 2

程序代码	注释
N10 DEF BOOL RESULT	；结果变量的定义。
N20 RESULT=ISFILE("TESTFILE")	
N30 IF(NOT ISFILE("TESTFILE"))	

程序代码	注释
N40 MSG("文件不存在")	
N50 M0	
N60 ENDIF	
...	

2.22 读取文件信息(FILEDATE, FILETIME, FILESIZE, FILESTAT, FILEINFO)

借助指令 FILEDATE、FILETIME、FILESIZE、FILESTAT 和 FILEINFO 可以读取特定文件的信息，如：上次读访问时的日期/时间、当前文件大小、文件状态或信息总和。

前提条件

当前所设置的保护级别必须等于或者大于上一级目录的 Show（显示）权限。否则系统会拒绝访问并且显示出错提示（出错变量的返回值 = 13）。

句法

```
FILE....(<错误>,"<文件名>",<结果>)
```

含义

FILEDATE:	提供最近一次对文件进行写入访问的**日期**
FILETIME:	提供最近一次对文件进行写入访问的**时间**
FILESIZE:	提供当前文件的**大小**
FILESTAT:	提供文件与下列**权限**相关的**状态**信息： • 读取 (r: read) • 写入 (w: write) • 执行 (x: execute) • 显示 (s: show) • 删除 (d: delete) **注：** 该保护级别是被动文件系统的特征。 访问外部程序存储器时返回 FILESTAT，因此只有默认的访问权限（77777）。
FILEINFO:	提供可通过 FILEDATE, FILETIME, FILESIZE 和 FILESTAT 读取的文件的**信息汇总**

<错误>:	返回错误值的变量（Call-By-Reference 引用调用参数）			
	类型：	VAR INT		
	值：	0	无错误	
		1	非法路径	
		2	路径未找到	
		3	文件未找到	
		4	错误的文件类型	
		13	访问权限不够	
		22	结果变量（<结果>）的字符串长度太短。	
<文件名称>:	需要读取信息的文件的名称			
	类型：	CHAR[160]		
	可在原文件名前指定绝对路径。如果没有路径说明，系统会在当前的目录（=选中程序的目录）中查找文件。 路径说明规定参见“程序存储器文件的定址 (页 235)”。			
<结果>:	结果变量 (Call-By-Reference 参数) 保存所获取的文件信息的变量。			
	类型：	VAR CHAR[8]	对于	FILEDATE 格式："dd.mm.yy"
		VAR CHAR[8]	对于	FILETIME 格式："hh:mm.ss"
		VAR INT	对于	FILESIZE 文件大小以字节表示。
		VAR CHAR[5]	对于	FILESTAT 格式："rwxsd" (r: **r**ead, w: **w**rite, x: e**x**ecute, s: **s**how, d: **d**elete)
		VAR CHAR[32]	对于	FILEINFO 格式："rwxsd nnnnnnnn dd.mm.yy hh:mm:ss"

示例

程序代码	注释
N10 DEF INT ERROR	；出错变量的定义。

程序代码	注释
N20 STRING[32] RESULT	；结果变量的定义。
N30 FILEINFO(ERROR,"/_N_MPF_DIR/ _N_TESTFILE_MPF",RESULT)	；有域标识符、文件标识符和路径说明的文件名称。
N40 IF ERROR <> 0	；错误计值
N50 MSG("错误"<<ERROR<<"发出 FILEINFO 指令时")	
N60 M0	
N70 ENDIF	
...	

在本示例中，结果变量 RESULT 会提供以下结果：

"77777 12345678 26.05.00 13:51:30"

2.23 取整 (ROUNDUP)

通过功能“ROUNDUP”可以将 REAL 型的输入值（带小数点的数字）取整为一个较大的整数值。

句法

```
ROUNDUP(<值>)
```

含义

ROUNDUP:	用于取整输入值的指令
<值>：	REAL 型的输入值

说明

原样返回一个 INTEGER 型的输入值（一个整数）。

示例

示例 1： 不同的输入值及其取整结果

示例	取整结果
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

示例 2： NC 程序中的 ROUNDUP

程序代码

```
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])
...
```

2.24 子程序

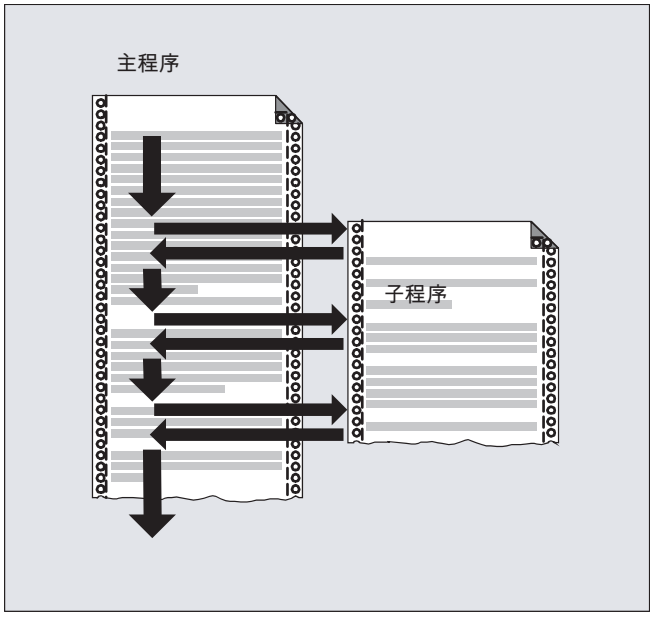
2.24.1 概述

2.24.1.1 子程序

在零件程序之前还固定区分为“主程序”和“子程序”的时候，就出现了“子程序”的概念。 其中，主程序指在控制系统上选择加以处理，随后启动的零件程序。 而子程序指由主程序调用的零件程序。

在目前的 SINUMERIK NC 语言中，这种固定的划分已不再存在。 原则上，每个零件程序既可以作为主程序选择并启动；也可以作为子程序由另一个零件程序调用。

因此，随着子程序定义的演变，零件程序指可以由另一个零件程序调用的程序。



应用

如同所有高级的编程语言一样，在 NC 语言中也使用了子程序，以便将一些多次应用的程序部分保存为独立、封闭的程序。

子程序具有以下优点：

- 提高了清晰性和程序可读性
- 通过重复使用测试的程序部分提高了质量

- 可以提供建立专门的加工库
- 节省了存储空间

2.24.1.2 子程序名称

命名规定

子程序名称在遵守以下规定的前提下可以自由选择：

- 许可字符：
  - 字母：A ... Z, a ... z
  - 数字：0 ... 9
  - 下划线：_
- 前两个字符应该是两个字母或是下划线加一个字母。

  说明

  当满足该条件时，才能够仅仅通过输入程序名称将一个 NC 程序作为子程序从其他程序中进行调用。反之，如果程序名称使用数字开头，那么子程序调用就只能通过 CALL 指令进行。

- 最大长度：24 个字符

说明

大/小写字母

在 SINUMERIK NC 语言中不区分大小写字母。

说明

不允许的程序名称

为避免与 Windows 应用程序冲突，不允许使用以下程序名称：

- CON, PRN, AUX, NUL
- COM1, COM2, COM3, COM4, COM5, COM6, COM7, COM8, COM9
- LPT1, LPT2, LPT3, LPT4, LPT5, LPT6, LPT7, LPT8, LPT9

控制系统内部的扩展

在控制系统内部会为创建程序时给定的子程序名称添加前缀名和后缀名：

- 前缀名：_N_
- 后缀名：_SPF

程序名称的使用

在使用程序名称时，如调用子程序时，可以组合所有的前缀名、程序名称和后缀名。

示例：

名为 SUB_PROG 的子程序可以通过以下名称调用：

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

主程序和子程序的名称相同

如果一个主程序（.MPF）和子程序（.SPF）具有相同的程序名，则在 NC 程序中使用程序名时必须注明可以清楚识别的相应文件扩展名。在其他情况下则只使用沿着搜索路径作为第一个根据该指定程序名可找到的程序。

2.24.1.3 子程序的嵌套

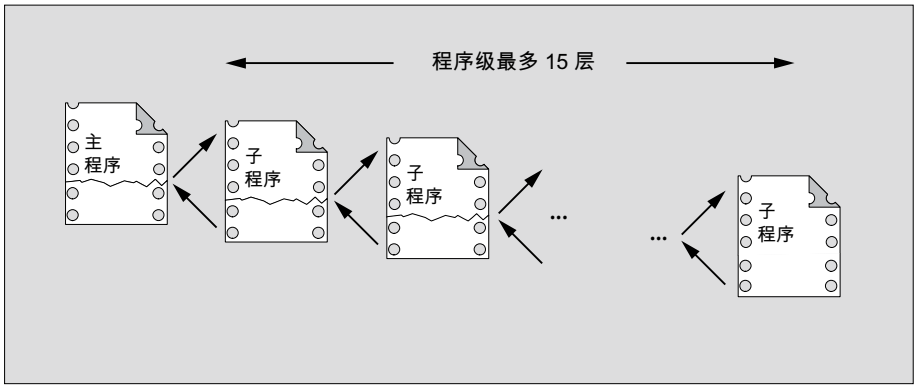
一个主程序可以调用子程序，而这个子程序又能继续调用一个子程序。 因此各个程序以相互嵌套的方式运行。 此时，每个程序都在各自的程序级上运行。

嵌套深度

NC 语言目前提供 16 个程序级。 主程序始终在最高的程序级上运行，即 0。而子程序始终在下一个更低级别的程序级上运行。 因此，程序级 1 是第一个子程序级。

程序级的划分：

- 程序级 0： 主程序级
- 程序级 1 - 15： 子程序级 1 - 15



中断程序（ASUP）

如果在中断程序的范围内调用了子程序，该程序将不会在通道中当前生效的程序级(n)上执行，而是在下一个更低级别的程序级(n+1)上执行。 考虑到中断程序，为了在最低的程序级上也能执行上述步骤，还另外提供了 2 个程序级（16 和 17）。

如果为此需要的程序级大于 2，必须在构建通道中处理的零件程序时加以考虑。即：应为中断程序的处理预留足够多的程序级。

如果中断程序处理需要 4 个程序级，那么零件程序最多只能占用 13 个程序级。 在进行中断时，这 4 个程序级（14~17）将发挥作用。

西门子循环

西门子循环为此需要使用 3 个程序级。 因此必须最迟在以下程序级中调用西门子循环：

- 零件程序处理： 程序级 12：
- 中断程序： 程序级 14：

2.24.1.4 查找路径

在调用没有路径说明的子程序时，控制系统根据预先定义的搜索顺序（参见“子程序调用时的查找路径 (页 240)”）查找现有的程序存储器。

2.24.1.5 形式参数和实际参数

形式参数和实际参数通常与带参数传递的子程序的定义和调用相关。

形式参数

在定义子程序时必须定义需要传递给子程序的参数（即形式参数）的类型和名称。

形式参数由此定义了子程序的接口。

示例：

程序代码	注释
`PROC KONTUR (REAL X, REAL Y)`	`; 形式参数： X 和 Y，都是 REAL 型`
`N20 X1=X Y1=Y`	`; 将轴 X1 运行到位置 X 上；轴 Y1 运行到位置 Y 上`
`...`	
`N100 RET`	

实际参数

在调用子程序时，必须将绝对值或变量，即实际参数传递给子程序。

在调用时，实际参数由此为子程序接口填充实际值。

示例：

程序代码	注释
N10 DEF REAL BREITE	; 变量定义
N20 BREITE=20.0	; 变量赋值
N30 KONTUR(5.5, BREITE)	; 子程序调用，带实际参数： 5.5 和 BREITE
...	
N100 M30	

2.24.1.6 参数传递

定义一个带参数传递的子程序

通过关键字 PROC、一张包含了所有子程序需要的参数的完整列表可以定义一个带参数传递的子程序。

不完整的参数传递

在调用子程序时，不需要总是显式传递所有在子程序接口中定义的参数。 如果省略了一个参数，会传递缺省值“0”给该参数。

但为了明确区分参数的顺序，必须始终用逗号隔开参数。 最后一个参数后面不需要逗号。 如果在调用时略去该参数，最后一个逗号也可以省略。

示例：

子程序：

程序代码	注释
PROC SUB_PROG (REAL X, REAL Y, REAL Z)	; 形式参数： X, Y 和 Z
...	
N100 RET	

主程序：

程序代码	注释
PROC MAIN_PROG	

程序代码	注释
...	
N30 SUB_PROG(1.0,2.0,3.0)	；子程序调用，带完整的参数传递： X=1.0, Y=2.0, Z=3.0
…	
N100 M30	

示例：在 N30 中调用子程序，带完整的参数传递：

```
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
```

注意

Call-by-Reference 引用调用式参数传递

在调用子程序时不应省略引用调用方式传递的参数。

注意

数据类型 AXIS

在调用子程序时不应省略 AXIS 数据类型的参数。

检查传递参数

借助系统变量“P_SUBPAR [n]”（其中 n = 1, 2, ...）可以检查子程序中是否显式传递或省略了某个参数。 索引 n 指形式参数的顺序。 索引 n = 1 表示第 1 个形式参数；索引 n = 2 表示第 2 个形式参数，依此类推。

下面的程序段落举例说明了如何检查第 1 个形式参数。

编程	注释
PROC SUB_PROG (REAL X, REAL Y, REAL Z)	；形式参数： X, Y 和 Z
N20 IF $P_SUBPAR[1]==TRUE	；检查第 1 个形式参数 X。
...	；如果显式传递了形式参数 X，执行此动作。
N40 ELSE	
...	；如果没有显式传递形式参数 X，执行此动作。
N60 ENDIF	
...	；通用动作
N100 RET	

2.24.2 定义子程序

2.24.2.1 没有参数传递的子程序

在定义没有参数传递的子程序时，可以省略程序头的定义行。

句法

```
[PROC <程序名称>]
...
```

含义

PROC:	程序开头的定义指令
<程序名称>:	程序的名称

示例

示例 1： 子程序，带 PROC 指令

程序代码	注释
PROC SUB_PROG	; 定义行
N10 G01 G90 G64 F1000	
N20 X10 Y20	
...	
N100 RET	; 子程序返回

示例 2： 子程序，不带 PROC 指令

程序代码	注释
N10 G01 G90 G64 F1000	
N20 X10 Y20	
...	
N100 RET	; 子程序返回

参见

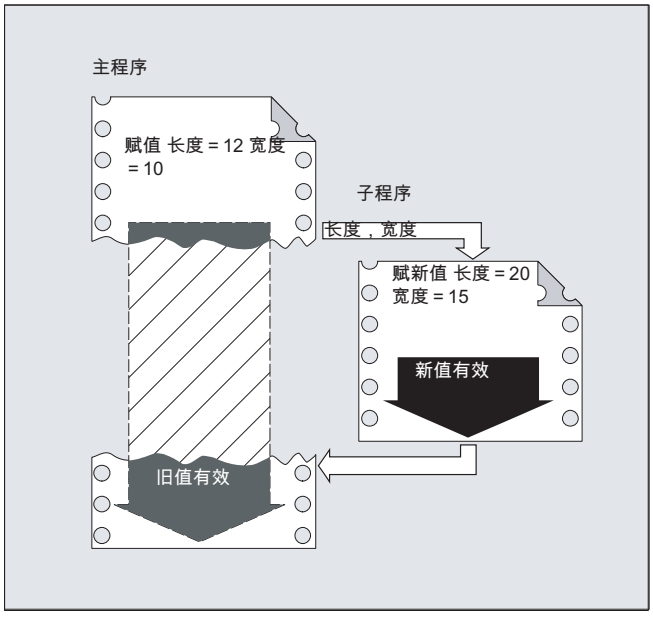
没有参数传递的子程序调用 (页 206)

2.24.2.2 子程序，带 Call-by-Value 值调用式参数传递(PROC)

通过关键字 PROC、程序名称、一张包含了所有参数及其类型和名称的完整列表，就可以定义一个按值调用参数的子程序。定义指令必须位于第一个程序行中。

按值调用

主调程序在按值调用参数时只向子程序传递变量值，因此，子程序无法直接访问变量。因此，在修改参数值时，只能修改子程序中显示的值。在主调程序中定义的变量的值保持不变。按值调用参数不会对主调程序产生影响。



句法

```
PROC <程序名称> (<参数类型> <参数名称>=<初始化值>, ...)
```

说明

最多可以传递 127 个参数。

含义

PROC:	程序开头的定义指令
<程序名称>:	程序的名称
<参数类型>:	参数的数据类型，如 REAL, INT, BOOL

<参数名称>：	参数名称
<初始化值>：	用于参数初始化的最优值（**可选**） 若在调用子程序时未设定参数，则会向参数分配初始值。

示例

示例 1

通过三个 REAL 类型的参数以缺省值定义子程序 SUB_PROG。

程序代码

```
PROC SUB_PROG(REAL LENGTH=10.0, REAL WIDTH=20.0, REAL HIGHT=30.0)
```

示例 2

不同的调用方式

程序代码

```
PROC MAIN_PROG
  REAL PAR_1 = 100
  REAL PAR_2 = 200
  REAL PAR_3 = 300
  ; 调用方案
  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
```

参见

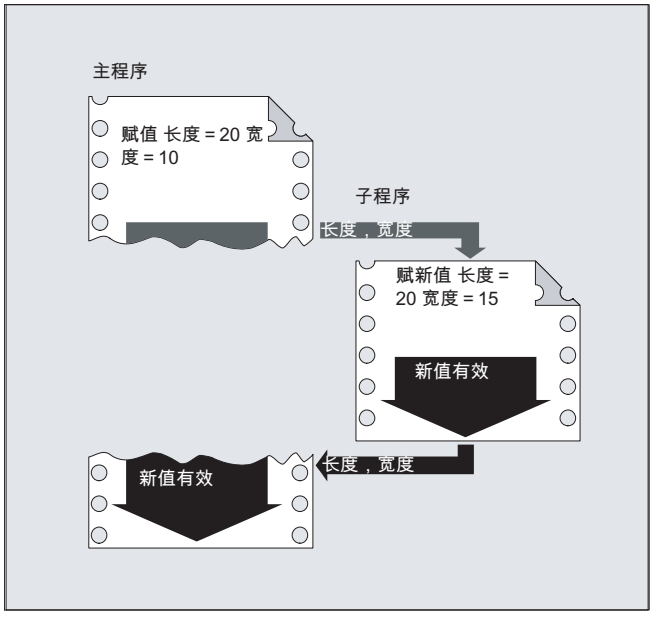
带参数传递的子程序调用(EXTERN) (页 208)

2.24.2.3 子程序，带 Call-by-Reference 引用调用式参数传递(PROC, VAR)

通过关键字 PROC、程序名称、一张包含了所有参数关键字 VAR、类型和名称的完整列表，就可以定义一个带 Call-by-Reference 引用调用式参数传递的子程序。定义指令必须位于第一个程序行中。作为参数，也可以传递数组的引用。

引用调用

主调程序在引用调用式参数传递时向子程序传递的不是变量值，而是变量的引用（矢量）。为此，子程序可以直接访问变量。因此，在修改参数值时，不仅可以修改子程序中显示的值，也可以修改在主调程序中定义的变量的值。子程序结束后，引用调用式参数传递仍会对主调程序产生影响。



说明

因此只有当在主调程序中定义了传递变量 (LUD) 时，才需要按照引用调用的方法传递参数。而通道全局变量或 NC 全局变量无需传递，因为子程序也能够直接访问这些变量。

句法

```
PROC <程序名称> (VAR <参数类型> <参数名称>, ...)
PROC <程序名称> (VAR <数组类型> <数组名称> [<m>,<n>,<o>], ...)
```

说明

最多可以传递 127 个参数。

含义

PROC:	程序开头的定义指令
VAR:	按照引用进行的参数传递的关键字

<程序名称>:	程序的名称	
<参数类型>:	参数的数据类型，如 REAL, INT, BOOL	
<参数名称>:	参数名称	
<数组类型>:	数组元素的数据类型，如 REAL, INT, BOOL	
<数组名称>:	数组的名称	
[<m>,<n>,<o>]:	数组长度 目前最多可以为 3 维数组：	
	<m>:	1 维数组长度
	<n>:	2 维数组长度
	<o>:	3 维数组长度

说明

- 关键字 PROC 后指定的程序名称必须和操作界面上指定的程序名称一致。
- 子程序可以将不确定长度的数组用作形式参数，来处理可变长度的数组。为此在定义一个形式参数的二维数组时，不规定 1 维的长度。但是必须写上逗号。
  示例：PROC <程序名称> (VAR REAL FELD[,5])

示例

定义带两个参数（作为 REAL 型的引用）的子程序：

程序代码

```
; 参数 1：参考类型：REAL，名称：LAENGE
; 参数 2：参考类型：REAL，名称：BREITE
PROC SUB_PROG(VAR REAL LAENGE, VAR REAL BREITE)
```

参见

带参数传递的子程序调用(EXTERN) (页 208)

2.24.2.4 保存模态 G 功能（SAVE）

属性 SAVE 用于保存子程序调用前激活的模态 G 代码，在子程序结束后再次激活。

注意
连续路径运行的中断 如果在连续路径运行生效时调用了含 SAVE 属性的子程序，则在此子程序结束（返回）时连续路径运行会中断。

句法

```
PROC <子程序名称> SAVE
```

含义

SAVE:	保存子程序调用前激活的模态 G 代码，并使功能在子程序结束后再次生效

示例

在子程序 KONTUR 中，模态 G 代码 G91（增量尺寸）生效。在主程序中，模态 G 代码 G90（绝对尺寸）生效。通过带 SAVE 的子程序定义，G90 在主程序中的子程序结束后再次生效。

子程序定义：

程序代码	注释
PROC KONTUR (REAL WERT1) SAVE	; 带参数 SAVE 的子程序定义
N10 G91 ...	; 模态 G 代码 G91：增量尺寸
N100 M17	;子程序结束

主程序：

程序代码	注释
N10 G0 X...Y...G90	; 模态 G 代码 G90：绝对尺寸
N20 ...	
...	
N50 KONTUR (12.4)	; 子程序调用
N60 X...Y...	; 模态 G 代码 G90 通过 SAVE 再次激活

边界条件

框架

与带属性 SAVE 的子程序相关的框架特性取决于框架类型，并可以通过机床数据设置。

文档

功能手册基本功能；轴、坐标系,框架(K2)，
章节："SAVE 子程序返回"

2.24.2.5 抑制单程序段处理 (SBLOF, SBLON)

全部程序的单程序段抑制

带有 SBLOF 标记的程序，在有效单程序段处理时如同一个程序段一样进行完整处理，即对于整个程序，抑制单程序段处理。

SBLOF 位于 PROC 行，并且一直有效，直至子程序结束或者中断。 使用返回指令判断在子程序结束处是否被停止：

通过 M17 跳回：	停止于子程序末尾处
通过 RET 跳回：	在子程序末尾处不停止

程序内的单程序段抑制

SBLOF 必须单独在程序段中。 从这个程序段起，关闭单段至：

- 下一个 SBLON
  或者
- 生效子程序级的结束处

句法

全部程序的单程序段抑制：

```
PROC ... SBLOF
```

程序内的单程序段抑制：

```
SBLOF
...
SBLON
```

含义

PROC:	一个程序的第一个指令
SBLOF:	用于关闭单程序段处理的指令 SBLOF 可以位于一个 PROC 程序段中，或者单独位于程序段中。
SBLON:	用于打开单程序段处理的指令 SBLON 必须位于一个独立的程序段中。

边界条件

- **单程序段抑制和程序段显示**
  可以在循环/子程序中使用 DISPLOF 抑制当前的程序段显示。 如果 DISPLOF 连同 SBLOF 一起编程，则在循环/子程序之内在单程序段停止时，如同在调用循环/子程序之前一样显示。
- **系统 ASUP 或用户 ASUP 中的单程序段抑制**
  如果系统或用户 ASUP 中的单程序段停止通过在机床数据 MD10702 $MN_IGNORE_SINGLEBLOCK_MASK 中设置进行抑制，（Bit0 = 1 或 Bit1 = 1），则单程序段停止可以通过在 ASUP 中编程 SBLON 再次激活。
  如果用户 ASUP 中的单程序段停止通过在机床数据 MD20117 $MC_IGNORE_SINGLEBLOCK_ASUP 中设置进行抑制，则单程序段停止通过在 ASUP 中编程 SBLON **无法** 再次激活。
- **不同的单程序段处理类型，单程序段抑制的特性**
  在激活的单程序段处理 SBL2（在零件程序段后停止）时，当在 MD10702 $MN_IGNORE_SINGLEBLOCK_MASK（避免单程序段停止）设置 Bit 12 为“1”时，在 SBLON 程序段中 **不** 停止。
  在激活的单程序段处理 SBL3（也在循环中零件程序段后停止）时，指令 SBLOF 被抑制。

示例

示例 1：程序内的单程序段抑制

程序代码	注释
N10 G1 X100 F1000	
N20 SBLOF	;关闭单程序段
N30 Y20	
N40 M100	
N50 R10=90	

程序代码	注释
N60 SBLON	;再次启用单程序段
N70 M110	
N80 ...	

N20 和 N60 之间的区域，在单程序段运行时作为一步处理。

示例 2： 循环对于用户而言就如同一个指令

主程序：

程序代码
N10 G1 X10 G90 F200
N20 X-4 Y6
N30 CYCLE1
N40 G1 X0
N50 M30

循环 CYCLE1：

程序代码	注释
N100 PROC CYCLE1 DISPLOF SBLOF	; 抑制单段
N110 R10=3*SIN(R20)+5	
N120 IF (R11 <= 0)	
N130 SETAL(61000)	
N140 ENDIF	
N150 G1 G91 Z=R10 F=R11	
N160 M17	

当激活单程序段时执行循环 CYCLE1，即处理 CYCLE1 时，必须按一次“启动”按钮。

示例 3： 为激活已修改的零点偏移和刀具补偿而被 PLC 启动的 ASUP 应该被隐藏。

程序代码
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

程序代码
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

示例 4： 通过 MD10702 Bit 12 = 1 不停止

初始情况：

- 单程序段处理激活。
- MD10702 $MN_IGNORE_SINGLEBLOCK_MASK Bit12 = 1

主程序：

程序代码	注释
N10 G0 X0	；在该零件程序行中停止。
N20 X10	；在该零件程序行中停止。
N30 CYCLE	；由循环产生的运行程序段。
N50 G90 X20	；在该零件程序行中停止。
M30	

循环 CYCLE：

程序代码	注释
PROC CYCLE SBLOF	；抑制单段停止
N100 R0 = 1	
N110 SBLON	；由于 MD10702 Bit12=1，在该零件程序行中不停止。
N120 X1	；在该零件程序行中停止。
N140 SBLOF	
N150 R0 = 2	
RET	

示例 5： 程序嵌套时单段抑制

初始情况：

单程序段处理激活。

程序嵌套：

程序代码	注释
N10 X0 F1000	; 在该程序段中停止。
N20 UP1(0)	
PROC UP1(INT _NR) SBLOF	; 抑制单程序段停止。
N100 X10	
N110 UP2(0)	
PROC UP2(INT _NR)	
N200 X20	
N210 SBLON	; 激活单程序段停止。
N220 X22	; 在该程序段中停止。
N230 UP3(0)	
PROC UP3(INT _NR)	
N300 SBLOF	; 抑制单程序段停止。
N305 X30	
N310 SBLON	; 激活单程序段停止。
N320 X32	; 在该程序段中停止。
N330 SBLOF	; 抑制单程序段停止。
N340 X34	
N350 M17	; SBLOF 生效。
N240 X24	; 在该程序段中停止。 SBLON 激活。
N250 M17	; 在该程序段中停止。SBLON 激活。
N120 X12	
N130 M17	; 在该返回程序段中停止。 PROC 语句的 SBLOF 激活。
N30 X0	; 在该程序段中停止。
N40 M30	; 在该程序段中停止。

其它信息

异步子程序单段禁止

为了在某个步骤中执行单程序段中的 ASUP，必须在 ASUP 中编程一个带有 SBLOF 的 PROC 指令。 这也适用于功能“可编辑的系统 ASUP”（MD11610 $MN_ASUP_EDITABLE）。

可编辑系统 ASUP 举例：

程序代码	注释
N10 PROC ASUP1 SBLOF DISPLOF	
N20 IF $AC_ASUP=='H200'	

程序代码	注释
N30 RET	; 运行方式转换时不 REPOS。
N40 ELSE	
N50 REPOSA	; 所有其他情形下 REPOS。
N60 ENDIF	

在单段中的程序影响

在单程序段处理中，用户可以按程序段方式执行零件程序。 有下列设置类型：

- SBL1： 在每个机床功能程序段后面带有停止的 IPO 单程序段。
- SBL2: 单段，在每个程序段之后停顿
- SBL3: 在循环中停顿（通过选择 SBL 3 抑制 SBLOF 指令）。

程序嵌套时单段抑制

如果在一个子程序中编程 SBLOF 在 PROC 语句中，则用 M17 停止到子程序跳回。 由此防止在调用的程序中已经执行下一个程序段。 如果在某个子程序中使用 SBLOF（PROC 语句中没有 SBLOF）激活某个单程序段抑制，就只有在调用程序的下一个机床功能程序段之后停止。 如果不希望如此，则在子程序中在跳回之前（M17）必须再次编程 SBLON。 在一个上一级的程序中，在用 RET 跳回时，不停止。

2.24.2.6 抑制当前的程序段显示(DISPLOF, DISPLON, ACTBLOCNO)

在标准情况下，程序段显示画面中会显示当前的程序段。 在循环或子程序中可以通过指令 DISPLOF 抑制当前程序段的显示。 显示循环的调用或者子程序的调用，而不显示当前的程序段。 借助指令 DISPLON 可以再次恢复程序段显示。

DISPLOF 或 DISPLON 应写入包含 PROC 指令的程序行中，它作用于整个子程序，并会隐式影响所有该子程序调用的其他子程序，这些子程序中不包含 DISPLON 或 DISPLOF 指令。 这个属性同样针对 ASUP。

句法

```
PROC … DISPLOF
PROC … DISPLOF ACTBLOCNO
PROC … DISPLON
```

含义

DISPLOF:	用于抑制当前程序段显示的指令。	
	位置：	在包含 PROC 指令的程序行的结尾
	生效方式：	直至从子程序返回或者程序结束。
	提示： 如果从带 DISPLOF 指令的子程序调用其他子程序，而在这些子程序中没有显式编程 DISPLON，则也抑制其中的当前程序段显示。	
DISPLON:	用于恢复当前程序段显示的指令	
	位置：	在包含 PROC 指令的程序行的结尾
	生效方式：	直至从子程序返回或者程序结束。
	提示： 如果从带 DISPLON 指令的子程序调用其他子程序，而在这些子程序中没有显式编程 DISPLOF，则也激活其中的当前程序段显示。	
ACTBLOCNO:	DISPLOF 连同属性 ACTBLOCNO 一起作用，在报警情况下会输出出现报警的当前程序段的号码。 同样，如果在较低等级的程序级中只编程了 DISPLOF，也会输出相应号码。 与此相反，DISPLOF 不带 ACTBLOCNO 时，循环或子程序调用号码由上一个不带 DISPLOF 标记的程序级显示。	

示例

示例 1： 在循环中抑制当前程序段显示

程序代码	注释
PROC CYCLE (AXIS TOMOV, REAL POSITION) SAVE DISPLOF	；抑制当前的程序段显示。 换而言之，如果要显示循环调用，例如： CYCLE(X,100.0)
DEF REAL DIFF	；循环内容
G01 ...	
...	
RET	；子程序跳回。 在程序段显示中在循环调用上显示下列程序段。

示例 2： 发出报警时程序段显示

子程序 SUBPROG1（带有 ACTBLOCNO）：

程序代码	注释
PROC SUBPROG1 DISPLOF ACTBLOCNO	
N8000 R10 = R33 + R44	
...	
N9040 R10 = 66 X100	; 触发报警 12080
...	
N10000 M17	

子程序 SUBPROG2（不带 ACTBLOCNO）：

程序代码	注释
PROC SUBPROG2 DISPLOF	
N5000 R10 = R33 + R44	
...	
N6040 R10 = 66 X100	; 触发报警 12080
...	
N7000 M17	

主程序：

程序代码	注释
N1000 G0 X0 Y0 Z0	
N1010 ...	
...	
N2050 SUBPROG1	; 发出报警 = "12080 通道 K1 程序段 N9040 同步错误 对于文本 R10="
N2060 ...	
N2350 SUBPROG2	; 发出报警 = "12080 通道 K1 程序段 N2350 同步错误 对于文本 R10="
...	
N3000 M30	

示例 3：恢复当前程序段显示

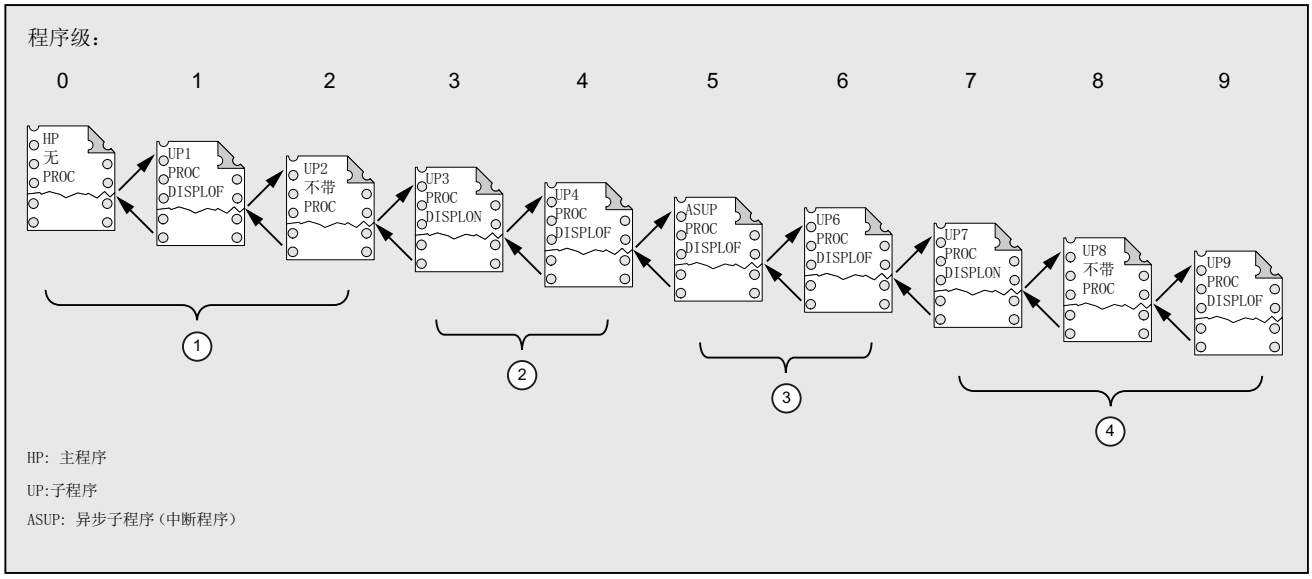
子程序 SUB1 带抑制：

程序代码	注释
PROC SUB1 DISPLOF	；抑制子程序 SUB1 中当前程序段显示。 而应显示带 SUB1 调用的程序段。
...	
N300 SUB2	；调用 子程序 SUB2。
...	
N500 M17	

子程序 SUB2 不带抑制：

程序代码	注释
PROC SUB2 DISPLON	；激活子程序 SUB2 中的当前程序段显示。
...	
N200 M17	；返回到子程序 SUB1。 在 SUB1 中再次抑制当前程序段显示。

示例 4： 不同的 DISPLON/DISPLOF 组合下的显示属性



① 在当前程序段显示中，会显示从程序级 0 开始的零件程序行。

② 在当前程序段显示中，会显示从程序级 3 开始的零件程序行。

③ 在当前程序段显示中，会显示从程序级 3 开始的零件程序行。

④ 在当前程序段显示中会显示从程序级 7/8 开始的零件程序行。

2.24.2.7 标记子程序“准备”(PREPRO)

关键字 PREPRO 可以在引导启动中 PROC 指令行结尾处标记所有文件。

说明

程序预处理的方式取决于相应设置的机床数据。 参见机床制造商说明。

文档：
功能手册 特殊功能；预处理 (V2)

句法

```
PROC … PREPRO
```

含义

PREPRO:	关键字，用于标记引导启动中经过预处理的文件以及循环目录中保存的 NC 程序

读入经过预处理的子程序和子程序调用

不管是在启动中经过处理的、带参数的子程序，还是子程序调用，循环目录的处理顺序都相同：

1. _N_CUS_DIR 用户循环
2. _N_CMA_DIR 制造商循环
3. _N_CST_DIR 标准循环

如果带相同名称的 NC 程序有不同的特征，则首先激活找到的 PROC 指令而忽略其它 PROC 指令，而不输出报警提示。

2.24.2.8 子程序返回指令 M17

返回指令 M17 或零件程序结束指令 M30 位于子程序的末尾。 它使得程序执行返回到主调程序中、子程序调用指令后的零件程序段上。

说明

M17 和 M30 在 NC 语言中视为同等的指令。

句法

```
PROC <程序名称>
...
M17/M30
```

边界条件

子程序返回对连续路径运行的影响

如果 M17 或 M30 位于单独的零件程序段中，则通道中激活的连续路径运行被中断。

为避免此类中断，应在最后一个运行程序段中写入 M17 或 M30。 此外，还必须将以下机床数据设为 0：

MD20800 $MC_SPF_END_TO_VDI = 0 （没有 M30/M17 输出给 NC/PLC 接口 ）

示例

1. M17 位于单独程序段中的子程序

程序代码	注释
N10 G64 F2000 G91 X10 Y10	
N20 X10 Z10	
N30 M17	；返回，中断连续路径运行。

2. M17 位于最后一个运行程序段中的子程序

程序代码	注释
N10 G64 F2000 G91 X10 Y10	
N20 X10 Z10 M17	；返回，不中断连续路径运行。

2.24.2.9 子程序返回指令 RET

在子程序中也可以代替 M17 而编程指令 RET。 RET 必须在一个单独的零件程序段中编程。 和 M17 类似，RET 使得程序执行返回到主调程序中、子程序调用指令之后的零件程序段上。

说明

通过编程参数可以修改 RET 的跳回属性（参见“可设定的子程序跳回 (RET ...) (页 196)”）。

应用

如果不希望因为返回而中断 G64 连续路径运行（G641 ... G645），则必须使用 RET 指令。

前提条件

只能在未定义 SAVE 属性的子程序中使用 RET 指令。

句法

```
PROC <程序名称>
...
RET
```

示例

主程序：

程序代码	注释
PROC MAIN_PROGRAM	; 程序开始
...	
N50 SUB_PROG	; 子程序调用： SUB_PROG
N60 ...	
...	
N100 M30	; 程序结束

子程序：

程序代码	注释
PROC SUB_PROG	
...	
N100 RET	; 跳回到主程序的程序段 N60。

2.24.2.10 可设定的子程序跳回 (RET ...)

通常使用命令 RET 会从子程序跳回当前调用的程序中。之后会从子程序调用处紧接着的程序行继续执行。如要从其他位置继续执行程序，可通过以下方式进行：

- 在 ISO 语言方式下调用切削循环之后，会根据轮廓说明继续程序处理。
- 在故障处理时，从任意一个子程序级（也在 ASUP 之后）跳回到主程序。
- 跳回需越过几个程序级，用于在编译循环和 ISO 语言方式中的特殊应用。

为此，须为指令 RET 编程其它参数。

查找方向

声明参数<目标程序段>时，会首先跳回调用程序段之后的程序段。接着，向所跳回的程序的**末尾**方向查找目标。如未找到目标，则会接着向程序开始方向进行查找。

句法

```
RET("<目标程序段>")
RET("<目标程序段>",<目标程序段后的程序段>)
RET("<目标程序段>",<目标程序段后的程序段>,<返回级的数量>)
RET("<目标程序段>", ,<返回级的数量>)
RET("<目标程序段>",<目标程序段后的程序段>,<返回级的数量>,
<返回程序头>)
RET( , ,<返回级的数量>,<返回程序头>)
```

含义

RET:	子程序结束		
<目标程序段>：	在参数中将需要继续程序处理的程序段指定为跳转目标。 如果未编程参数**<**返回级的数量**>**，跳转目标则会位于主调程序中，当前子程序从该主调程序中调用。 允许的说明有：		
	<程序段编号**>**	目标程序段编号。 在跳回到的主调程序中先按程序结束的方向查找程序段编号。	
	<跳转标记**>**	跳回到的主调程序中的跳转标记必须存在。 在跳回到的主调程序中先按程序结束的方向查找跳转标记。	
	<字符串**>**	跳回到的主调程序中的字符串必须存在（例如程序或者变量名称）。 在跳回到的主调程序中先按程序结束的方向查找字符串。 对于目标程序段中的字符串编程，有下列规则： • **末尾处空格**（不同于末尾处用冒号“:”表示的跳转标记） • **字符串前** 仅允许设置一个程序段号码和/或一个跳转标记，**没有程序指令**。	
<目标程序段后的程序段>:	在参数中指定是否在在参数**<**目标程序段**>**下指定的程序段中或后续程序段中继续程序处理。		
	类型：	INT	
	值：	0	跳回到在参数**<**目标程序段**>**中指定的程序段。
		> 0	跳回到在参数**<**目标程序段**>**中指定的后续程序段。

<返回级的数量>:	在参数中指定须跳回的程序级的数量，以查找目标程序段并继续程序处理。		
	类型：	INT	
	值：	1	程序就在“当前程序级 －1”中继续执行 (如同不带参数的 RET)。
		2	程序在“当前程序级－2”中继续执行，即越过一级。
		3	程序在“当前程序级－3”中继续执行，即越过两级。
		...	
	取值范围：	1 ... 15	
<返回到程序头>:	在参数中指定是否在跳回到主程序且主程序中的 **ISO 语言模式**激活时继续回到程序头。		
	类型：	BOOL	
	值：	1	当跳回到主程序中且主程序中已有一个 **ISO 语言模式**激活时，就回到程序头。

说明

如果一个子程序返回中指定了一个字符串用于目标程序段查找，则始终首先在主调程序中查找跳转标记。

如果要通过一个字符串明确定义返回目标，该字符串不允许与跳转标记同名，否则子程序总是返回该跳转标记，而不会返回到该字符串（参见示例 2）。

边界条件

在越过几个程序级返回时，会分析各个程序级的 SAVE 指令。

如果在越过几个程序级返回时已有一个模态子程序激活，且如果在某个被跳过的子程序中已经为该模态子程序编程了取消指令 MCALL，那么该模态子程序将继续保持激活状态。

注意
编程错误 用户需要注意：在越过几个程序级返回时使用正确的模态设置继续执行。例如，通过编程一个相应的主程序段就可做到这一点。

示例

示例 1：在 ASUP 加工之后，在主程序中继续。

编程	注释
N10010 CALL "UP1"	; 程序级 0（主程序）
N11000 PROC UP1	; 程序级 1
N11010 CALL "UP2"	
N12000 PROC UP2	; 程序级 2
...	
N19000 PROC ASUP	; 程序级 3（ASUP 加工）
...	
N19100 RET("N10900", ,$P_STACK)	; 子程序返回到主程序中
	; $P_STACK:当前程序级
N10900	; 主程序中的目标程序段
N10910 MCALL	; 关闭模态子程序调用
N10920 G0 G60 G40 M5	; 初始化其它模态设置

示例 2：字符串 (<String>) 作为目标程序段查找数据

主程序：

程序代码	注释
PROC MAIN_PROGRAM	
N1000 DEF INT iVar1=1, iVar2=4	
N1010 ...	
N1200 subProg1	; 调用子程序“subProg1”
N1210 M2 S1000 X10 F1000	
N1220	
N1400 subProg2	; 调用子程序“subProg2”
N1410 M3 S500 Y20	
N1420 ..	
N1500 lab1:iVar1=R10*44	
N1510 F500 X5	
N1520 ...	
N1550 subprog1:G1 X30	; "subProg1" 这里定义为跳转标记。
N1560 ...	
N1600 subProg3	; 调用子程序“subProg3”
N1610 ...	
N1900 M30	

子程序 subProg1:

程序代码	注释
PROC subProg1	
N2000 R10=R20+100	
N2010 ...	
N2200 RET("subProg2")	; 返回到主程序中的程序段 N1400

子程序 subProg2:

程序代码	注释
PROC subProg2	
N2000 R10=R20+100	
N2010 ...	
N2200 RET("iVar1")	; 返回到主程序中的程序段 N1500

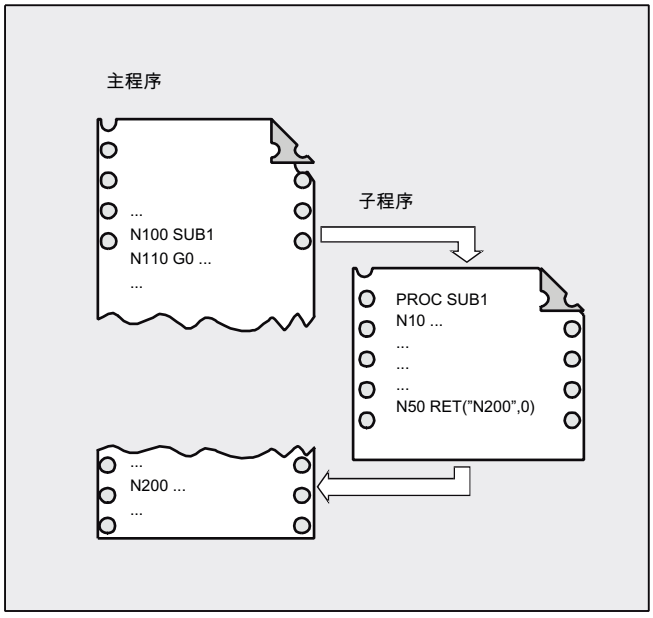
子程序 subProg3:

程序代码	注释
PROC subProg3	
N2000 R10=R20+100	
N2010 ...	
N2200 RET("subProg1")	; 返回到主程序中的程序段 N1550

其它信息

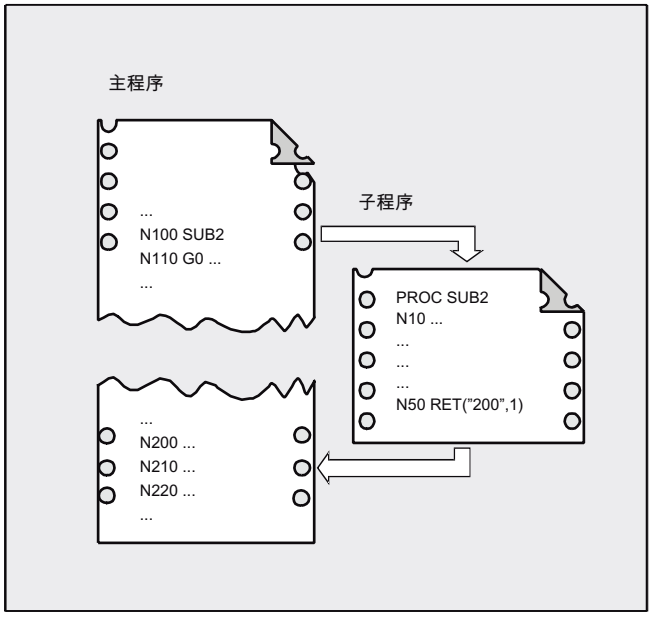
下列图形显示了返回参数的不同效用

1. <目标程序段> = “N200”，<目标程序段后的程序段> = 0



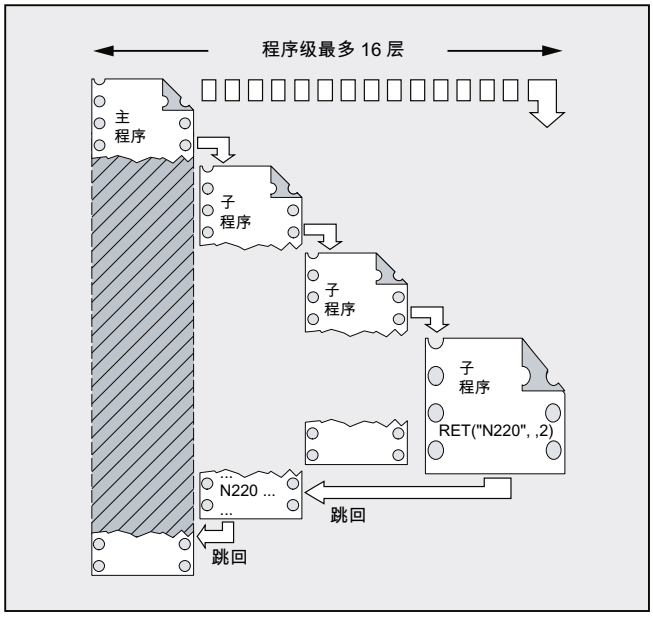
根据 RET 指令，程序处理继续通过主程序中程序段 N200 进行。

2. <目标程序段> = “N200”，<目标程序段后的程序段> = 1



根据 RET 指令，程序处理继续通过程序段 （N210 ）进行，该程序段紧跟在主程序中的程序段 N200 之后。

3. <目标程序段> = “N220”，<返回级的数量> = 2



根据 RET 指令，跳回到两个程序级，程序处理继续通过程序段 N220 进行。

2.24.2.11 可设定的子程序跳回 (RETB ...)

通常使用命令 RETB 会从子程序跳回当前调用的程序中。之后会从子程序调用处紧接着的程序行继续执行。如要从其他位置继续执行程序，可通过以下方式进行：

- 在 ISO 语言方式下调用切削循环之后，会根据轮廓说明继续程序处理。
- 在故障处理时，从任意一个子程序级（也在 ASUP 之后）跳回到主程序。
- 跳回需越过几个程序级，用于在编译循环和 ISO 语言方式中的特殊应用。

为此，须为指令 RETB 编程其它参数。

查找方向

声明参数<目标程序段>时，会首先跳回调用程序段之后的程序段。接着，向所跳回的程序的**开头**方向查找目标。如未找到目标，则会接着向程序末尾方向进行查找。

句法

```
RETB(“<目标程序段>”)
RETB(“<目标程序段>”,<目标程序段后的程序段>)
RETB(“<目标程序段>”,<目标程序段后的程序段>,<返回级的数量>)
RETB(“<目标程序段>”, ,<返回级的数量>)
RETB(“<目标程序段>”,<目标程序段后的程序段>,<返回级的数量>,
<返回程序头>)
RETB( , ,<返回级的数量>,<返回程序头>)
```

含义

RETB：	子程序结束		
<目标程序段>：	在参数中将需要继续程序处理的程序段指定为跳转目标。 如果未编程参数**<**返回级的数量**>**，跳转目标则会位于主调程序中，当前子程序从该主调程序中调用。 允许的说明有：		
	<程序段编号**>**	目标程序段编号。 在跳回到的主调程序中先按程序开始的方向查找程序段编号。	
	<跳转标记**>**	跳回到的主调程序中的跳转标记必须存在。 在跳回到的主调程序中先按程序开始的方向查找跳转标记。	
	<字符串**>**	跳回到的主调程序中的字符串必须存在（例如程序或者变量名称）。 在跳回到的主调程序中先按程序开始的方向查找字符串。 对于目标程序段中的字符串编程，有下列规则： • **末尾处空格**（不同于末尾处用冒号“:”表示的跳转标记） • **字符串前** 仅允许设置一个程序段号码和/或一个跳转标记，**没有程序指令**。	
<目标程序段后的程序段>:	在参数中指定是否在在参数**<**目标程序段**>**下指定的程序段中或后续程序段中继续程序处理。		
	类型：	INT	
	值：	0	跳回到在参数**<**目标程序段**>**中指定的程序段。
		> 0	跳回到在参数**<**目标程序段**>**中指定的后续程序段。

<返回级的数量>:	在参数中指定须跳回的程序级的数量，以查找目标程序段并继续程序处理。		
	类型：	INT	
	值：	1	程序就在“当前程序级 －1”中继续执行 (如同不带参数的 RET)。
		2	程序在“当前程序级－2”中继续执行，即越过一级。
		3	程序在“当前程序级－3”中继续执行，即越过两级。
		...	
	取值范围：	1 ... 15	
<返回到程序头>:	在参数中指定是否在跳回到主程序且主程序中的 **ISO 语言模式**激活时继续回到程序头。		
	类型：	BOOL	
	值：	1	当跳回到主程序中且主程序中已有一个 **ISO 语言模式**激活时，就回到程序头。

说明

如果一个子程序返回中指定了一个字符串用于目标程序段查找，则始终首先在主调程序中查找跳转标记。

如果要通过一个字符串明确定义返回目标，该字符串不允许与跳转标记同名，否则子程序总是返回该跳转标记，而不会返回到该字符串（参见示例 2）。

边界条件

在越过几个程序级返回时，会分析各个程序级的 SAVE 指令。

如果在越过几个程序级返回时已有一个模态子程序激活，且如果在某个被跳过的子程序中已经为该模态子程序编程了取消指令 MCALL，那么该模态子程序将继续保持激活状态。

注意
编程错误 用户需要注意：在越过几个程序级返回时使用正确的模态设置继续执行。例如，通过编程一个相应的主程序段就可做到这一点。

示例

```
BEISPIEL.MPF
    …
   N3000 START_CYC(parm1, param2, …)
   N3010 TECH_CYC1(param1, param2, …)
   N3020 TECH_CYC2(param1, param2, …)
   N3030 TECH_CYC3(param1, param2, …)
   N3040 END_CYC(param1, param2, …)
   N3040 END_CYC(param1, param2, …)
   N3050 …
   N4500 START_CYC(param11, param12, …)
   N4510 …
   N4590 END_CYC(param11, param12, ..)
   N5000 …
    …
N6000 M30
```

程序代码	注释
PROC END_CYC(…)	; 在主程序中调用行 N3040
N10000 …	
N15000 if status == 1	
N15010 RETB("START_CYC")	; 跳回到主调程序 BEISPIEL.MPF
	; 查找字符串 "START_CYC"
	; 查找方向：向后方向程序开始
	; 通过行 N3000 继续程序处理
N15020 endif	
N15030 if status == 0	
N15040 RET	; 跳回到主调程序 BEISPIEL.MPF
	; 通过行 N3050 继续程序处理
N15050 endif	
N16000 RET("START_CYC")	; 跳回到主调程序 BEISPIEL.MPF
	; 查找字符串 "START_CYC"
	; 查找方向：向前方向程序末尾
	; 通过行 N4500 继续程序处理
N17060 RETB	; 跳回到主调程序 BEISPIEL.MPF
	; 通过行 N3050 继续程序处理
	; 无参数的 RETB 等同于 RET

2.24.3 子程序调用

2.24.3.1 没有参数传递的子程序调用

调用子程序时，可以使用地址 L 加子程序号，或者直接使用程序名称。

一个主程序也可以作为子程序调用。 此时，主程序中设置的程序结束指令 M2 或 M30 视作 M17（返回到主调程序的程序结束）处理。

说明

同样，一个子程序也可以作为主程序启动。

控制系统的查找方法：

是否有 *_MPF ?

是否有 *_SPF ?

接着： 如果被调子程序的名称和主程序的名称相同，则再次调用主调主程序。 一般这种情况不应发生，所以主程序和子程序的名称必须相互区别，不得相同。

说明

从一个初始化文件中可以调用无需参数传递的子程序。

句法

L<编号>/<程序名称>

说明

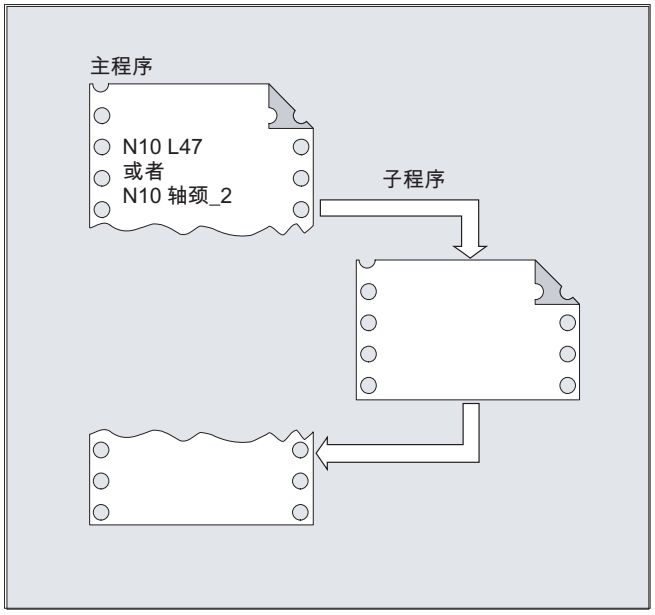
子程序调用必须在独立的 NC 程序段中编程。

含义

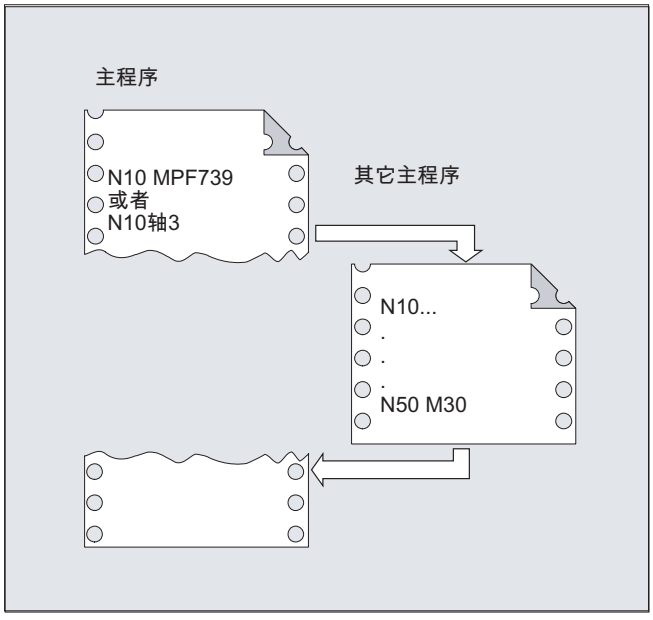
L:	子程序调用地址	
<编号>:	子程序号码	
	类型：	INT
	值：	最多 7 位数 **注意：** 数值中开始的零在命名时具有不同的含义（⇒L123, L0123 和 L00123 表示三个不同的子程序）。
<程序名称>:	子程序或主程序的名称	

示例

示例 1： 调用一个不带参数传递的子程序



示例 2： 作为子程序调用主程序



参见

没有参数传递的子程序 (页 178)

2.24.3.2 带参数传递的子程序调用(EXTERN)

在带参数传递的子程序调用时，可以直接传递变量或者数值（不针对 VAR 参数）。

必须在调用之前在主程序中使用 EXTERN 声明带参数传递的子程序，例如，在程序头。 其中应给出子程序的名称以及传递顺序中的变量类型。

注意
混淆危险 不管是变量类型还是传递的顺序，均必须和子程序中 PROC 所约定的定义相符。 参数名称可以在主程序和子程序中不一样。

句法

```
EXTERN <程序名称>(<类型_参数 1>,<类型_参数 2>,<类型_参数 3>)
...
<程序名称>(<数值_参数 1>,<数值_参数 2>,<数值_参数 3>)
```

说明

子程序调用必须在独立的 NC 程序段中编程。

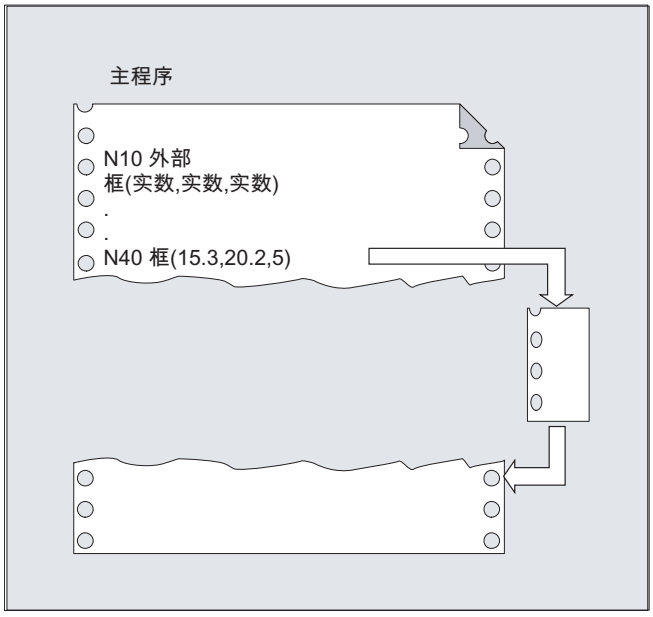
含义

<程序名称>:	子程序名称
EXTERN:	关键字，用于带有参数传递的子程序声明 **提示：** 必须仅当子程序在工件中或者在全局子程序目录中时，才可指定 EXTERN。 循环不必声明为 EXTERN 。
<类型_参数 1>,<类型_参数 2>,<类型_参数 3>:	传递序列中要传递的参数变量类型
<数值_参数 1>,<数值_参数 2>,<数值_参数 3>:	要传递的参数变量值

示例

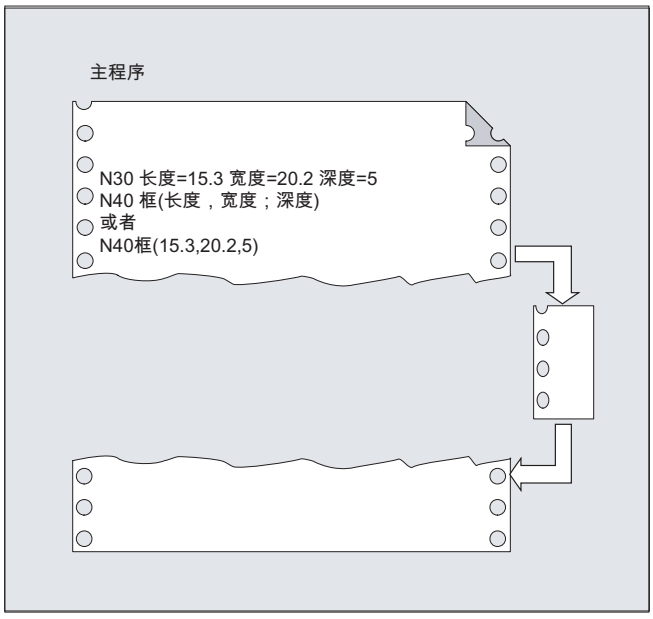
示例 1： 子程序调用，事先声明

程序代码	注释
N10 EXTERN RAHMEN(REAL,REAL,REAL)	；子程序说明。
...	
N40 RAHMEN(15.3,20.2,5)	；调用带参数传递的子程序。



示例 2： 子程序调用， 无声明

程序代码	注释
N10 DEF REAL LAENGE, BREITE, TIEFE	
N20 …	
N30 LAENGE=15.3 BREITE=20.2 TIEFE=5	
N40 RAHMEN(LAENGE,BREITE,TIEFE)	；或者： N40 RAHMEN(15.3,20.2,5)



参见

子程序，带 Call-by-Value 值调用式参数传递(PROC) (页 179)

子程序，带 Call-by-Reference 引用调用式参数传递(PROC, VAR) (页 180)

2.24.3.3 程序重复次数(P)

如果一个子程序需要多次连续执行，则可以在该程序段中在地址 P 下编程重复调用的次数。

⚠ 小心

带程序重复和参数传递的子程序调用

参数仅在程序调用时或者第一次执行时传送。 在后续重复过程中，这些参数保持不变。如果您在程序重复时要修改参数，则您必须在子程序中确定相应的协议。

句法

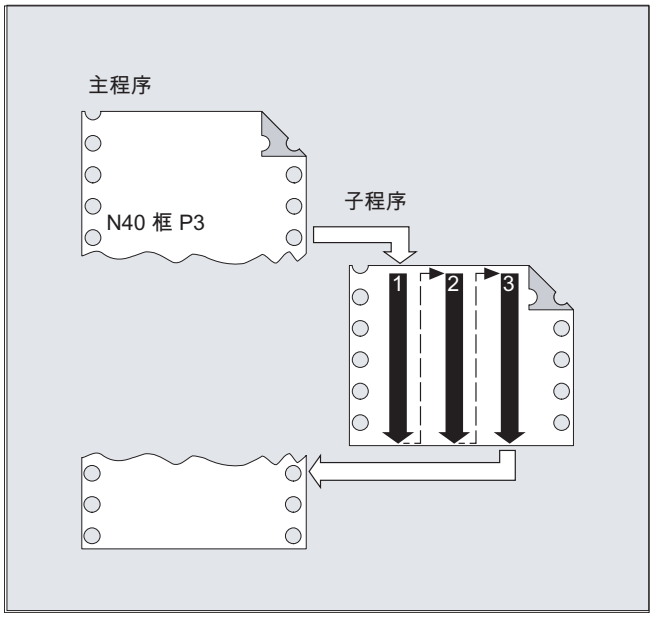
<程序名称> P<值>

含义

<程序名称>:	子程序调用	
P:	程序重复的编程地址	
<值>:	程序重复次数	
	类型:	INT
	取值范围:	1 … 9999 (不带正负号)

示例

程序代码	注释
...	
N40 RAHMEN P3	; 子程序"RAHMEN"应被连续执行三次。
...	



2.24.3.4 模态子程序调用 (MCALL)

通过模态子程序调用 MCALL(<程序名称>)，已声明的子程序不会被立即调用。而是从该时刻起在零件程序中每个带有轨迹运动的运行程序段后自动调用。同时是跨程序级的。

说明

在一个程序运行中，总是只有最后一个模态子程序调用有效 MCALL(<程序名>)。当前的模态子程序调用替代了之前有效的模态子程序调用。

如果要将参数传送给子程序，则只能在调用 MCALL(<程序名称>(参数 1、参数 2、...))时进行传送。

注意
模态子程序调用，无需轨迹运动 在以下情况中也可以调用模态子程序，无需编程一个轨迹运动： • 编程地址 S 或 F，当 G0 或 G1 有效时 • G0 或 G1 单独在程序段中编程或与其他 G 代码一起编程。

句法

```
MCALL <程序名称>
...
MCALL
```

含义

MCALL [<程序名称>]:	打开功能“模态子程序调用”
<程序名称>:	子程序名称
MCALL:	如在使用 MCALL 时未指定程序名称，系统会关闭“模态子程序调用”功能。

边界条件

ASUP

如果通过一个 ASUP（参见章节“中断程序（ASUP） (页 136)”）中断零件程序加工，那么在该 ASUP 中**不会**执行模态子程序调用。

如果一个 ASUP 是在“复位”通道状态下启动的，那么该 ASUP 的特性相比模态子程序调用更类似于一个普通零件程序。

换刀循环

如在换刀循环中取消功能“模态子程序调用”，则应注意在程序段查找之后隐含地通过查找 ASUP 或手动执行重写来调用换刀循环。此时不允许取消功能“模态子程序调用”，否则查找结果会不真实。因此建议在换刀循环中按以下方式对功能“模态子程序调用”的取消进行编程：

程序代码	注释
...	
IF $AC_ASUP == 0	; 不通过查找 ASUP 或覆盖进行调用。
MCALL	; 关闭功能“模态子程序调用”。
ENDIF	
...	

示例

示例 1

程序代码	注释
N10 G0 X0 Y0	
N20 MCALL L70	; 打开 L70 的模态子程序调用。
N30 X10 Y10	; X10 Y10 逼近，接着调用 L70。
N40 X20 Y20	; X20 Y20 逼近，接着调用 L70。
...	
N100 MCALL	; 关闭功能“模态子程序调用”。
N110 X0 Y0	; X0 Y0 逼近，不调用 L70。

示例 2

程序代码
N10 G0 X0 Y0
N20 MCALL L70
N30 L80

在这个例子中，子程序 L80 中有编程的轨迹轴和后续的 NC 程序段。L70 通过 L80 调用。

2.24.3.5 间接子程序调用(CALL)

根据所给定的条件，可以在一个地点调用不同的子程序。 这里子程序名称存放在一个字符串类型的变量中。 子程序调用通过 CALL 和变量名进行。

说明

间接调用子程序仅可以用于没有参数传递的子程序。 直接调用某个子程序时，可将名称保存在一个字符串常量中

句法

```
CALL <程序名称>
```

含义

CALL:	用于间接子程序调用的指令	
<程序名称>:	子程序的名称（变量或常量）	
	类型：	STRING

示例

使用字符串常量直接调用：

程序代码	注释
…	
CALL "/_N_WKS_DIR/_N_SUBPROG_WPD/_N_TEIL1_SPF"	; 使用 CALL 直接调用子程序 TEIL1。
…	

使用变量间接调用：

程序代码	注释
…	
DEF STRING[100] PROGNAME	; 定义变量。
PROGNAME="/_N_WKS_DIR/_N_SUBPROG_WPD/_N_TEIL1_SPF"	; 将变量 PROGNAME 指定给子程序 TEIL1。
CALL PROGNAME	; 通过 CALL 和变量 PROGNAME 间接调用子程序 TEIL1。
…	

2.24.3.6 指定待执行部分的间接子程序调用(CALL BLOCK ... TO ...)

通过 CALL 和关键字组合 BLOCK ... TO 可以间接调用一个子程序，并执行用起始标签和结束标签标记的程序部分。

句法

```
CALL <程序名称> BLOCK <起始标签> TO <结束标签>
CALL BLOCK <起始标签> TO <结束标签>
```

含义

CALL:	用于间接子程序调用的指令	
<程序名称>:	子程序名称（变量或常量），其中包含了要处理的程序部分**(指定 可选)**。	
	类型：	STRING
	提示： 如果没有编程<程序名称>，则在当前程序中查找带有<起始标签>和<结束标签>标记的程序部分并执行此部分。	
BLOCK ... TO ... :	用于间接执行程序部分的关键字组合	
<起始标记>:	表明要处理的程序部分开头的变量	
	类型：	STRING
<结束标记>:	表明要处理的程序部分末尾的变量。	
	类型：	STRING

示例

主程序：

程序代码	注释
...	
DEF STRING[20] STARTLABEL, ENDLABEL	；起始标记和结束标记的变量定义。
STARTLABEL="LABEL_1"	
ENDLABEL="LABEL_2"	
...	
CALL "CONTUR_1" BLOCK STARTLABEL TO ENDLABEL	；间接的子程序调用并标记待执行的程序部分。

程序代码	注释
...	

子程序：

程序代码	注释
PROC CONTUR_1 ...	
LABEL_1	；起始标记： 执行程序部分的开始
N1000 G1 ...	
...	
LABEL_2	；结束标记： 执行程序部分的结束
...	

2.24.3.7 间接调用某个以 ISO 语言编程的程序 (ISOCALL)

利用间接程序调用 ISOCALL，可以调用一个用 ISO 语言编程的程序。由此激活机床数据中设定的 ISO 模式。在程序结束处，原先的加工方式再次生效。如果在机床数据中没有设定 ISO 方式，则子程序调用以西门子方式进行。

有关 ISO 模式的其它信息，参见：
文档：
功能说明 ISO 语言

句法

```
ISOCALL <程序名称>
```

含义

ISOCALL:	子程序调用关键字，由此激活机床数据中设定的 ISO 模式。
<程序名称>:	ISO 语言编程的程序名称（STRING 型的变量和常量）

示例：使用 ISO 模式的循环编程调用轮廓

程序代码	注释
0122_SPF	；以 ISO 模式描述轮廓
N1010 G1 X10 Z20	
N1020 X30 R5	
N1030 Z50 C10	
N1040 X50	
N1050 M99	
N0010 DEF STRING[5] PROGNAME = "0122"	;西门子零件程序 (-循环)

程序代码	注释
...	
N2000 R11 = $AA_IW[X]	
N2010 ISOCALL PROGNAME	
N2020 R10 = R10+1	；以 ISO 模式编辑程序 0122.spf
...	
N2400 M30	

2.24.3.8 调用带有路径说明和参数的子程序 (PCALL)

利用 PCALL 可以调用带绝对路径说明和参数传送的子程序。

句法

PCALL <路径/程序名称>(<参数 1>,…,<参数 n>)

含义

PCALL:	关键字，用于带绝对路径说明的子程序调用
<路径/程序名称>:	包含子程序名的绝对路径说明。 路径说明规定参见“程序存储器文件的定址 (页 235)”。 如果没有说明绝对路径，则 PCALL 表现如同一个带程序名的标准子程序调用。 程序名无前缀和文件标识。 如果要给程序名编写前缀和标识，就必须通过指令 EXTERN 明确说明前缀和标识。
<参数 1>, ...:	实际参数符合子程序的 PROC 指令。

示例

程序代码
PCALL/_N_WKS_DIR/_N_WELLE_WPD/WELLE(参数 1,参数 2,…)

2.24.3.9 扩展调用子程序时的路径查找 (CALLPATH)

使用指令 CALLPATH 可以扩展查找路径用于子程序调用。 这样就可以从某个未选中的工件目录中调用子程序，而无需指定完整、绝对的子程序路径名。

如果外部程序存储器的目录用于存储全球子程序，则 EES 操作模式“EES 无 GDIR"提供更多的应用可能性。 在这种情况下可通过查找路径的 CALLPATH 扩展子程序目录。

在输入用户循环之前扩展查找路径（_N_CUS_DIR）。

通过下列结果再次选择查找路径扩展：

- CALLPATH 带空格
- CALLPATH 不带参数
- 零件程序结束
- Reset

句法

```
CALLPATH("<路径名称>")
```

含义

CALLPATH:	关键字，用于可编程的查找路径扩展。 在一个自有零件程序行中编程。
<路径名称>:	字符串型常量或变量。 包含某个目录的绝对路径说明，以此来扩展查找路径。 路径说明规定参见“程序存储器文件的定址 (页 235)”。

示例

应围绕某个确定的工件目录扩展查找路径：

程序代码

```
...
CALLPATH ("/_N_WKS_DIR/_N_MYWPD_WPD")
...
```

以此来设置下列查找路径（位置 5 是新建的）：

1. 当前目录/*name*
2. 当前目录/*name_SPF*
3. 当前目录/*name_MPF*
4. //NC:/_N_SPF_DIR / *name_SPF*
5. **/_N_WKS_DIR/_N_MYWPD_WPD/name_SPF**
6. /N_CUS_DIR/*name_SPF*

7. /_N_CMA_DIR/*name_SPF*
8. /_N_CST_DIR/*name_SPF*

边界条件

- CALLPATH 用来检查所编写的路径名是否存在。 在故障情况下，零件程序加工带补偿程序段报警 14009 中断。
- CALLPATH 也可以在 INI 文件中编程。 然后就会对 INI 文件的处理时间产生影响 (WPD-INI-文件或者用于 NC-活动文件的初始化程序，例如第 1 个通道中的框架 _N_CH1_UFR_INI). 然后再次复位查找路径。

2.24.3.10 执行外部子程序 (840D sl) (EXTCALL)

使用指令 EXTCALL 可从外部存储器中回装和执行零件程序。

可用的外部存储器包括：

- 本地驱动器
- 网络驱动器
- USB 驱动器

说明

执行 USB 驱动器上的外部程序仅可使用操作面板前端或 TCU 上的 USB 接口。

注意

USB 闪存可损坏刀具/工件

建议在执行外部子程序时不要使用 USB 闪存。如在执行零件程序的过程中由于接触不良、脱落以及因碰撞或误拔出而中断与 USB 闪存的通讯，会导致加工立即停止。这可能会损坏刀具或/和工件。

外部程序路径的预设

在以下设定数据中可以预设至外部子程序目录的路径：

SD42700 $SC_EXT_PROG_PATH

此路径和 EXTCALL 中指定的程序路径及标识共同组成待调用零件程序的完整路径。

说明

参数

在调用外部程序时，无法向该程序传送参数。

句法

```
EXTCALL("<路径/><程序名称>")
```

含义

EXTCALL:	调用一个外部子程序的指令	
"<路径/><程序名称>":	字符串型常量/变量	
	<路径/>:	绝对或相对路径说明（**可选**）
	<程序名称>:	设定程序名称时不添加“_N_”前缀。 可使用字符“_”或“.”将后缀名（“MPF”、“SPF”）添加在程序名上（**可选**）。 示例： "WELLE" "WELLE_SPF" "WELLE.SPF"

路径说明：缩写

进行路径说明时可采用以下缩写：

- 本地驱动器：**"LOCAL_DRIVE:"**
- CF 卡：**"CF_CARD:"**
- USB 驱动器（操作面板）：**"USB:"**

"CF_CARD:"和"LOCAL_DRIVE:"可交替使用。

示例

从本地驱动执行

主程序“Main.mpf”位于 NC 存储器中，并已选择执行该程序：

子程序“SP_1”

外部子程序“SP_1.SPF”及“SP_1.MPF”位于本地驱动器的目录“/user/sinumerik/data/prog/WKS.DIR/WST1.WPD”下。

外部程序目录的路径应设置为：

SD42700 $SC_EXT_PROG_PATH = LOCAL_DRIVE:WKS.DIR/WST1.WPD

说明

用于调用外部子程序的路径说明：

- 不使用默认设置："LOCAL_DRIVE:WKS.DIR/WST1.WPD/SP_1"
- 使用默认设置：“SP_1”

子程序“SP_2”

外部子程序“SP_2.SPF”及“SP_2.MPF”位于 USB 驱动器的目录 WKS.DIR/WST1.WPD 下。到外部程序目录的路径预设功能已对子程序 “SP_1” 使用，并且在主程序中不会对预设进行改写。因此在调用子程序“SP_2”时必须给定完整的路径。

主程序“MAIN”

程序代码

```
N010 PROC MAIN
N020 ...
N030 EXTCALL("SP_1")
N030 EXTCALL("USB:WKS.DIR/WST1.WPD/SP_2")
N050 ...
N060 M30
```

其它信息

EXTCALL 调用，带绝对路径说明

如果在给定的路径下存在子程序，则通过 EXTCALL 调用执行该序。 如果在指定的路径下不存在子程序，那么调用 EXTCALL 的程序执行过程将中断。

EXTCALL 调用，带相对路径说明/不带路径说明

在进行带相对路径说明/不带路径说明的 EXTCALL 调用时，系统根据下列模式查找存在的程序存储器：

1. 如果在 SD42700 $SC_EXT_PROG_PATH 中预设了路径说明，则首先从此路径出发查找 EXTCALL 中的设定（程序名或者相对路径说明）。 而绝对路径由字符串组成：
   - SD42700 $SC_EXT_PROG_PATH 中预设的路径说明
   - 分隔符“/”
   - 指令 EXTCALL 中的路径说明和子程序名称
2. 若在第 1 步中未找到子程序，则会在用户存储器目录中进行搜索。

一旦找到子程序，查找结束。 如果未找到子程序，那么调用 EXTCALL 的程序执行过程将中断。

可设定的回装存储器（FIFO 缓存器）

执行外部子程序需要一个回装存储器。 回装存储器的大小预设为 30 KB，且仅可由机床制造商进行修改（通过 MD18360 MM_EXT_PROG_BUFFER_SIZE 修改）。

说明

子程序，带跳转语句

对于含有跳转语句(GOTOF，GOTOB，CASE，FOR，LOOP，WHILE，REPEAT，IF，ELSE，ENDIF 等)的外部子程序，在回装存储器中应存在跳转目标。

说明

ShopMill/ShopTurn 程序

由于文件末尾附带了轮廓描述，ShopMill/ShopTurn 程序必须完整地保存在回装存储器中。

平行执行多个外部子程序时，各子程序均需要一个独立的回装存储器。

复位/程序结束/上电

通过复位和上电，可以中断外部的子程序调用，并且清除各自的加载存储器。

为“外部执行”选择的程序在复位/程序结束或上电后仍保持“选择用于外部执行”状态。只要外部程序存储器仍然可用，该特性就等同于在内部选择的程序。

文档

有关“外部执行”的更多相关信息请参见：

功能手册之基本功能；BAG、通道、程序运行、复位特性 (K1)

2.24.3.11 执行外部子程序 (828D) (EXTCALL)

使用指令 EXTCALL 可从外部存储器中回装和执行零件程序。

可用的外部存储器包括：

- 用户 CF 卡
- 网络驱动器
- USB 驱动器

说明

执行存在于 USB 驱动器上的外部程序时，只能使用前操作面板(PPU)上的 USB 接口。

注意

USB 闪存可损坏刀具/工件

建议在执行外部子程序时不要使用 USB 闪存。如在执行零件程序的过程中由于接触不良、脱落以及因碰撞或误拔出而中断与 USB 闪存的通讯，会导致加工立即停止。这可能会损坏刀具或/和工件。

外部程序路径的预设

在以下设定数据中可以预设至外部子程序目录的路径：

SD42700 $SC_EXT_PROG_PATH

此路径和 EXTCALL 中指定的程序路径及标识共同组成待调用零件程序的完整路径。

说明

参数

在调用外部程序时，无法向该程序传送参数。

句法

EXTCALL("<路径/><程序名称>")

含义

EXTCALL:	调用一个外部子程序的指令	
"<路径/><程序名称>"：	字符串型常量/变量	
	<路径/>：	绝对或相对路径说明（**可选**）
	<程序名称>：	设定程序名称时不添加"_N_"前缀。 可使用字符"_"或"."将后缀名（"MPF"、"SPF"）添加在程序名上（**可选**）。 示例： "WELLE" "WELLE_SPF" "WELLE.SPF"

路径说明：缩写

进行路径说明时可采用以下缩写：

- 用户 CF 卡："**CF_CARD:**"
- USB 驱动器（操作面板）："**USB:**"

示例

主程序"Main.mpf"位于 NC 存储器中，并已选择执行该程序：

子程序"SP_1"

外部子程序"SP_1.SPF"及"SP_1.MPF"位于用户存储卡的目录"/WKS.DIR/WST1.WPD"下。

外部程序目录的路径应设置为：

SD42700 $SC_EXT_PROG_PATH = CF_CARD:WKS.DIR/WST1.WPD

说明

用于调用外部子程序的路径说明：

- 不使用默认设置："CF_CARD:WKS.DIR/WST1.WPD/SP_1"
- 使用默认设置："SP_1"

子程序“SP_2”

外部子程序“SP_2.SPF”及“SP_2.MPF”位于 USB 驱动器的目录 WKS.DIR/WST1.WPD 下。到外部程序目录的路径预设功能已对子程序 “SP_1” 使用，并且在主程序中不会对预设进行改写。因此在调用子程序“SP_2”时必须给定完整的路径。

主程序“MAIN”

程序代码

```
N010 PROC MAIN
N020 ...
N030 EXTCALL("SP_1")
N030 EXTCALL("USB:WKS.DIR/WST1.WPD/SP_2")
N050 ...
N060 M30
```

其它信息

EXTCALL 调用，带绝对路径说明

如果在给定的路径下存在子程序，则通过 EXTCALL 调用执行该序。如果在指定的路径下不存在子程序，那么调用 EXTCALL 的程序执行过程将中断。

EXTCALL 调用，带相对路径说明/不带路径说明

在进行带相对路径说明/不带路径说明的 EXTCALL 调用时，系统根据下列模式查找存在的程序存储器：

1. 如果在 SD42700 $SC_EXT_PROG_PATH 中预设了路径说明，则首先从此路径出发查找 EXTCALL 中的设定（程序名或者相对路径说明）。而绝对路径由字符串组成：
   - SD42700 $SC_EXT_PROG_PATH 中预设的路径说明
   - 分隔符 “/”
   - 指令 EXTCALL 中的路径说明和子程序名称
2. 若在第 1 步中未找到子程序，则会在用户存储器目录中进行搜索。

一旦找到子程序，查找结束。如果未找到子程序，那么调用 EXTCALL 的程序执行过程将中断。

可设定的回装存储器（FIFO 缓存器）

执行外部子程序需要一个回装存储器。 回装存储器的大小已经预设（参见 MD18360 MM_EXT_PROG_BUFFER_SIZE）。

说明

子程序，带跳转语句

对于含有跳转语句(GOTOF，GOTOB，CASE，FOR，LOOP，WHILE，REPEAT，IF，ELSE，ENDIF 等)的外部子程序，在回装存储器中应存在跳转目标。

说明

ShopMill/ShopTurn 程序

由于文件末尾附带了轮廓描述，ShopMill/ShopTurn 程序必须完整地保存在回装存储器中。

平行执行多个外部子程序时，各子程序均需要一个独立的回装存储器。

复位/程序结束/上电

通过复位和上电可以中断外部子程序调用，并且清空各自的回装存储器。

为“外部执行”选择的程序在复位/程序结束或上电后仍保持“选择用于外部执行”状态。只要外部程序存储器仍然可用，该特性就等同于在内部选择的程序。

文献

有关“外部执行”的更多相关信息请参见：

功能手册之基本功能；BAG、通道、程序运行、复位特性 (K1)

2.25 宏指令技术 (DEFINE ... AS)

注意
宏指令技术加大了编程的复杂性 使用宏指令技术可能会使控制系统的编程语言发生巨大变化。宏指令技术使用时必须特别小心。

作为宏指令，是指单个的指令组合成一个新的总指令，带自己的名称。在程序运行中调用该宏指令时，可以在该宏指令名下一个接一个地执行编程的指令。

根据变量的有效范围，即宏指令定义生效范围，宏指令可分为以下几个类别：

- 局部宏指令
  局部宏指令是在执行 NC 程序开始处定义的不是主程序定义的宏指令。此指令在调用 NC 程序时创建，并在程序结束复位或下一次启动控制系统时删除。只能在定义局部宏指令的 NC 程序中存取该指令。
- 程序全局宏指令
  局部宏指令是在其中一个用作主程序的 NC 程序的开始处定义的指令。此指令在调用 NC 程序时创建，并在程序结束复位或下一次启动控制系统时删除。可在主程序及所有子程序中存取程序全局宏指令。

  说明

  程序全局宏指令的可用性

  当设置了以下机床数据时，在主程序中定义的程序全局宏指令同样在子程序中可用：

  MD11120 $MN_LUD_EXTENDED_SCOPE = 1

  设置 MD11120 = 0 时，在主程序中定义的程序全局宏指令只在主程序中可用。

- 全局宏指令
  全局宏指令是在定义文件（宏指令文件）中定义的 NC 或通道全局宏指令，此指令在程序结束复位或下一次启动控制系统后依然保留。全局宏指令可以在任意一个主程序或者子程序中调用和执行。

  说明

  为了可以在 NC 程序中使用**外部**宏文件的宏指令，必须将宏文件装载到 NC 中。

使用宏指令前必须对其进行定义。必须遵循以下规则：

- 在宏中可以定义任意的命名符、G 功能、M 功能、H 功能和 L 子程序名。
- 可在程序开始处或一个单独的定义文件（宏指令文件）中定义宏指令。
- 局部和程序全局宏指令在程序开始处定义。

- 全局宏指令必须在一个宏指令文件，如 _N_DEF_DIR/_N_UMAC_DEF 中定义。
- G 代码宏指令只可定义为全局宏指令。
- H 功能和 L 功能可以 2 位编程。
- M 和 G 代码可以进行 3 位编程。

说明

不得使用宏指令对关键字和备用名称进行覆写。

句法

宏指令定义：

```
DEFINE <Macro_Name> AS <Operation_1> <Operation_2> ...
```

在 NC 程序中调用：

```
<Macro_Name>
```

含义

DEFINE ... AS：	关键字组合用于定义一个宏指令
<Macro_Name>：	宏名称 只有命名符才允许用作宏指令名称。 通过宏名称可以从 NC 程序中调用宏。
<Operation_1>：	宏指令中的第一个编程指令
<Operation_2>：	宏指令中的第二个编程指令

边界条件

嵌套

不可以嵌套宏指令。

示例

示例 1：程序开始处的宏定义

程序代码	注释
DEFINE LINIE AS G1 G94 F300	；宏指令定义
...	

程序代码	注释
N70 LINIE X10 Y20	；宏指令调用
...	

示例 2：一个宏文件中的宏定义

程序代码	注释
DEFINE M6 AS L6	；当换刀时调用接收所需数据传送的某个子程序。在子程序中输出实际的换刀 M 功能（例如 M106）.
DEFINE G81 AS DRILL(81)	；模拟 DIN G 代码。
DEFINE G33 AS M333 G333	；在切削螺纹时要求与 PLC 同步。原始的 G 代码 G33 通过机床数据改名为 G333，编程方式对于用户而言仍一样。

示例 3：外部宏文件

在控制系统中读入外部宏文件后，必须将宏文件装载到 NC 中。然后才可以使用 NC 程序中的宏。

程序代码	注释
%_N_UMAC_DEF	
;$PATH=/_N_DEF_DIR	;用户特有的宏
DEFINE PI AS 3.14	
DEFINE TC1 AS M3 S1000	
DEFINE M13 AS M3 M7	；主轴右转，冷却液开
DEFINE M14 AS M4 M7	；主轴左转，冷却液开
DEFINE M15 AS M5 M9	；主轴停止，冷却液关
DEFINE M6 AS L6	；调用刀具更换程序
DEFINE G80 AS MCALL	；撤销选择钻削循环
M30	

文件和程序管理

3

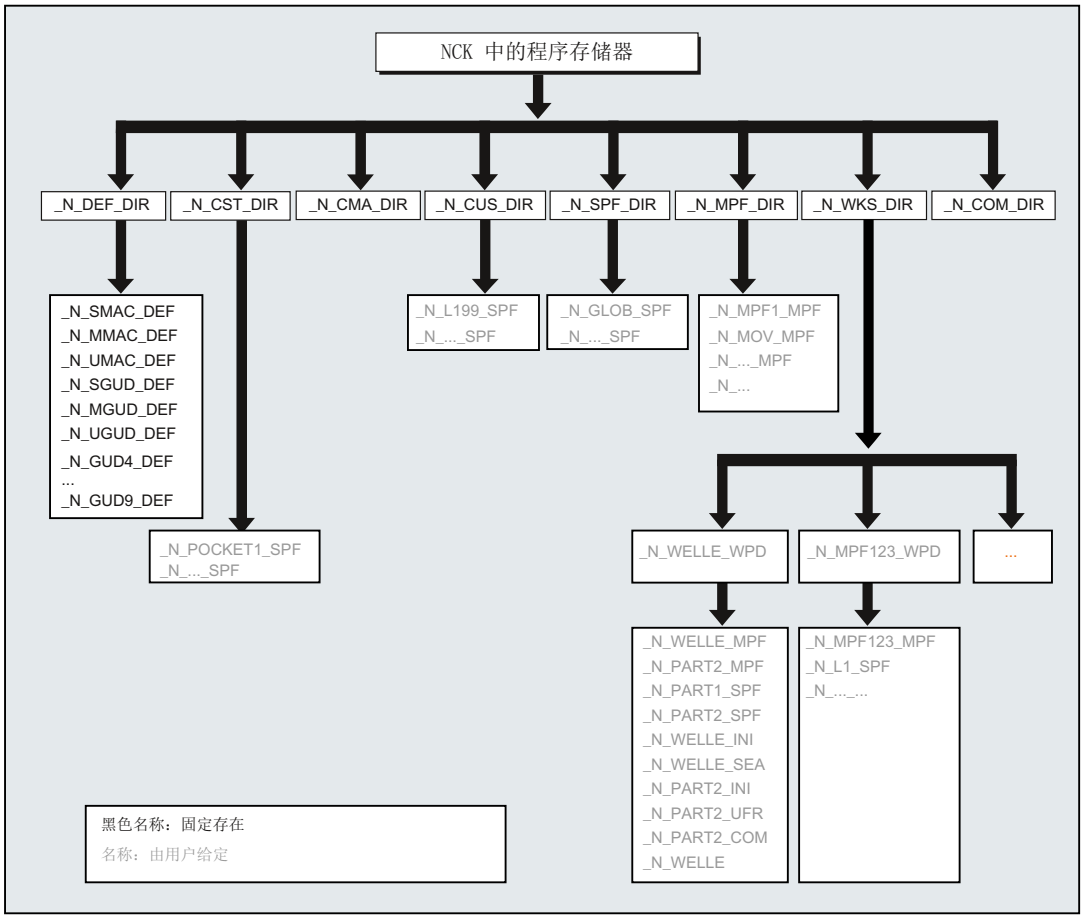
3.1 程序存储器

3.1.1 NC 中的程序存储器

文件和程序（例如：主程序和子程序、宏指令定义）将永久保存在程序存储器中（→ 被动文件系统）。

文档：
功能手册 扩展功能；存储器功能(S7)

此外，还有一些文件类型可以临时保存在这里并且在需要时（例如当加工某个特定的工件时）传送到工作存储器中（例如用于初始化目的）。



标准目录

正常情况下有以下目录：

目录	内容
_N_DEF_DIR	数据块和宏指令块
_N_CST_DIR	标准循环
_N_CMA_DIR	机床制造商循环
_N_CUS_DIR	用户循环
_N_WKS_DIR	工件
_N_SPF_DIR	全局子程序
_N_MPF_DIR	主程序
_N_COM_DIR	注释

文件类型

在程序存储器中可以有以下文件类型：

文件类型	说明
<名称>_MPF	主程序
<名称>_SPF	子程序
<名称>_TEA	机床数据
<名称>_SEA	设定数据
<名称>_TOA	刀具补偿
<名称>_UFR	零点偏移/框架
<名称>_INI	初始化文件
<名称>_GUD	全局用户数据
<名称>_RPA	R 参数
<名称>_COM	注释
<名称>_DEF	全局用户数据和宏指令定义

工件主目录 (_N_WKS_DIR)

工件主目录以默认名称_N_WKS_DIR 建立在程序存储器中。工件主目录包含所有编程工件的相应工件目录。

工件主目录(..._WPD)

工件目录包含加工工件时所需的所有文件。它可以是主程序，子程序，任意初始化程序和注释文件。

在选中程序后，第一次零件程序开始时一次性执行初始化程序（根据机床数据 MD11280 $MN_WPD_INI_MODE) 。

示例：

工件目录 _N_WELLE_WPD, 为工件 WELLE 所建立，包含有下列文件：

文件	说明
_N_WELLE_MPF	主程序
_N_PART2_MPF	主程序
_N_PART1_SPF	子程序
_N_PART2_SPF	子程序
_N_WELLE_INI	工件数据的常规初始化程序
_N_WELLE_SEA	设定数据初始化程序
_N_PART2_INI	程序第 2 部分数据的常规初始化程序
_N_PART2_UFR	用于程序第 2 部分框架文件的初始化程序
_N_WELLE_COM	注释文件

此外也可以在工件目录中保存另一些 NC 不直接用于加工的数据。这些数据除了是 ASCII 文件外也可以是二进制文件，如：JPG 格式的图片或 PDF 格式的说明。NC 必须能够识别文件扩展名（文件扩展名在调试时通过 MD17000 $MN_ EXTENSIONS_OF_BIN_FILES 设置；缺省文件扩展名为：JPG, GIF, PNG, BMP, PDF, ICO, HTM），才能将这些数据译为二进制文件。

选择用于加工的工件

可以为一个通道中的加工选择一个工件目录。如果在该目录中有一个**同名** 主程序或者只有一个唯一的主程序(_MPF)，就自动选择该程序来执行。

文档：

操作手册

3.1.2 外部程序存储器

除了 NC 中的被动文件系统，也可以在设备上使用外部程序存储器（例如：在本地驱动器上或者在一个网络驱动器上）。

通过“从外部执行”或“EES（从外部存储器执行）”功能可以**直接**从外部程序存储器执行零件程序。

文档：
功能手册之基本功能分册；K1：BAG、通道、程序运行、复位特性

全局零件程序存储器 (GDIR)

在确定驱动器时，可将其中一个驱动器定义为全局零件程序存储器（GDIR）。

文献:
操作手册，章节：“管理程序” > “设置驱动器”

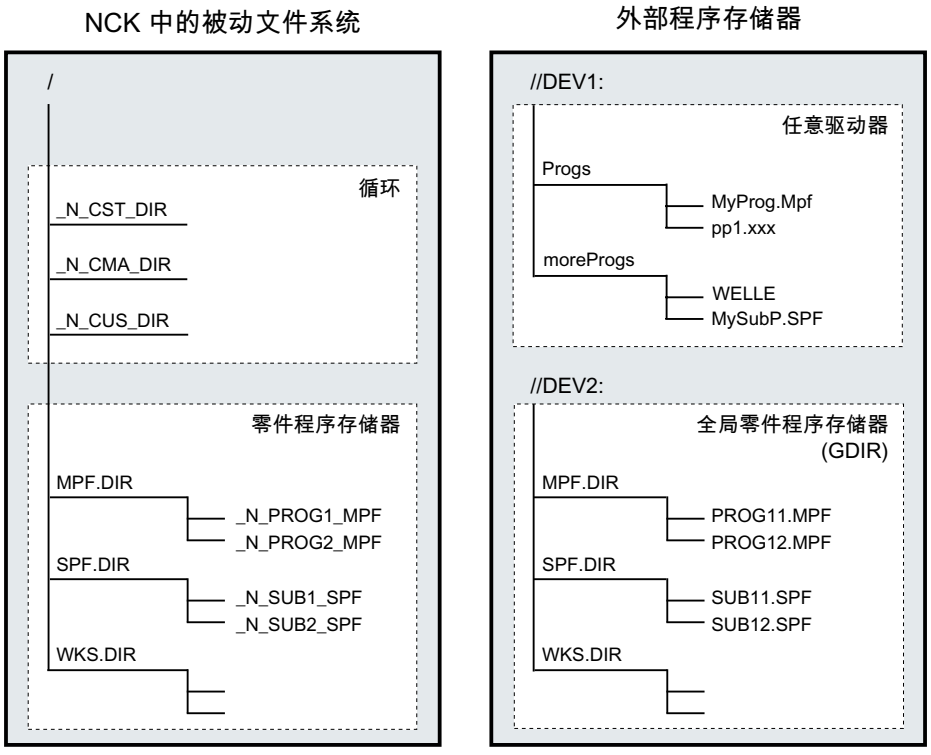
系统会在该驱动器上自动创建 MPF.DIR、SPF.DIR 和 WKS.DIR 目录。这三个目录共同组成 GDIR。

GDIR 只用于 EES 功能。根据驱动器配置，GDIR 或取代 NC 零件程序存储器，或作为对其的扩展。GDIR 设置对于 EES 操作并不是必要的。

GDIR 目录和文件可以和在被动文件系统一样，在零件程序中以相同的方式写上地址。这样就可以将 NC 程序和路径信息从被动文件系统兼容转移到 GDIR。GDIR 的 SPF.DIR 目录保存在子程序的查找路径中。

程序组织

下图说明了外部程序存储器上的程序结构。



不区分大小写的文件系统

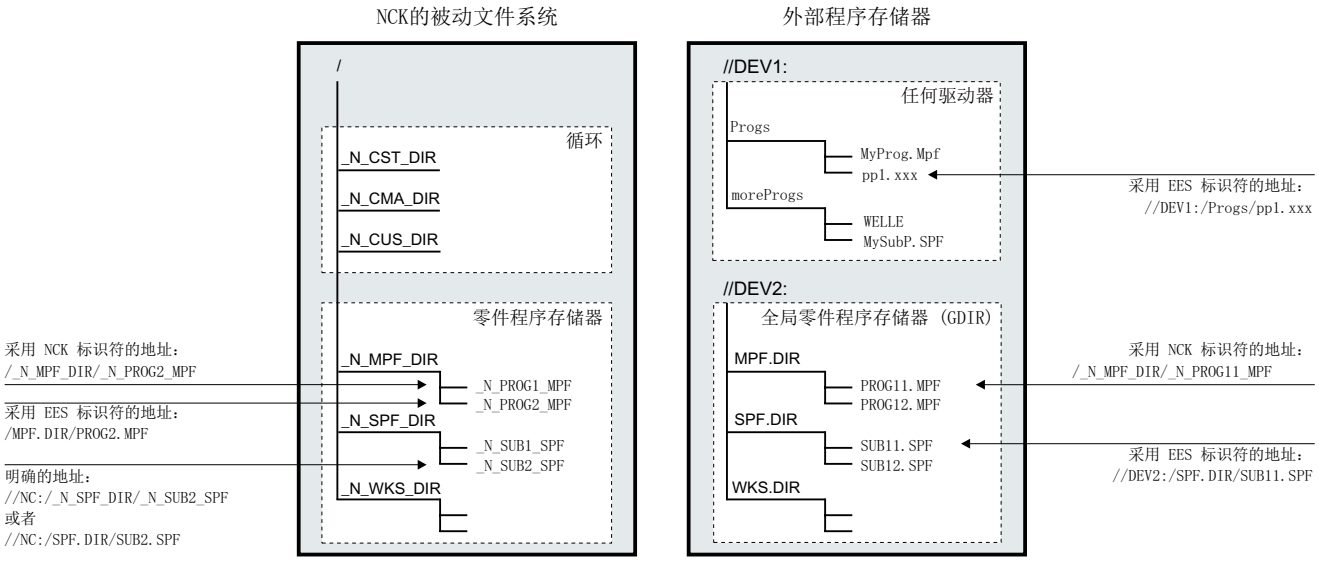
说明

为了避免文件地址大小写的问题（参见“程序存储器文件的定址 (页 235)”），必须使用**不区分大小写**的文件系统作为外部程序存储器。

3.1.3 程序存储器文件的定址

程序存储器中通过一个文件处理指令（如：WRITE、DELETE、READ、ISFILE、FILEDATE、FILETIME、FILESIZE、FILESTAT、FILEINFO）响应的文件是通过一个包含文件名的绝对路径或只是通过文件名单独回参考点的。另一种情况是所选程序的路径被用作文件路径。

NC/EES 标识符定址



被动文件系统文件的定址

被动文件系统文件的定址通常未指定驱动器名称，采用的是 **NC 标识符**（目录和文件名以域标识符 “_N_” 开头，目录/文件扩展名的分隔符为 “_” ）。**EES 标识符**定址也是允许的（无域标识符 “_N_” ，目录/文件扩展名的分隔符为 “.” ）。

示例：

- NC 标识符: "/_N_SPF_DIR/_N_SUB1_SPF"
- EES 标识符: "/SPF.DIR/SUB1.SPF"

说明

被动文件系统中文件的 EES 标识符定址在内部根据以下规定转换成 NC 标识符：

- 目录和文件名通过域标识符 “_N_” 扩展。
- 如果目录或文件名中倒数第四位字符是一个点（"."），则转换成下划线（"_"）。

通过预定义的驱动器名称 “//NC:” 也可以对被动文件系统进行精确的定址。

示例：

- NC 标识符: "//NC:/_N_SPF_DIR/_N_SUB1_SPF"
- EES 标识符: "//NC:/SPF.DIR/SUB1.SPF"

外部程序存储器文件的定址

不作为 GDIR 记录的外部程序存储器中的文件，其定址必须以 EES 标识符进行。在定址路径的开头就必须指定驱动器名称（例如"//DEV1:"）。所有在 “/user/sinumerik/hmi/cfg/logdrive.ini” 中配置的设备名称都是允许的。

示例：

- EES 标识符: "//DEV1:/MyProgDir/pp1.xxx"
- NC 格式：不允许

全局零件程序存储器 (GDIR) 文件的定址

在对 GDIR 文件进行定址时，既可以是采用 EES 标识符的路径说明，也可以是采用 NC 标识符的路径说明。

示例：

- EES 标识符: "//DEV2:/MPF.DIR/PROG11.MPF"
- NC 标识符: "/_N_MPF_DIR/_N_PROG11_MPF"

说明

GDIR 文件的 NC 标识符定址在内部根据以下规定转换成 EES 标识符：
- 删除目录和文件名中的域标识符 “_N_” 。
- 如果目录或文件名中倒数第四位字符是一个下划线（"_"），则转换成一个点（"."）。

路径说明规定

一个完整的路径说明包括驱动器名称、目录路径和文件名。

驱动器名称

驱动器名称的说明适用以下规定：

- 所有在 “/user/sinumerik/hmi/cfg/logdrive.ini” 中配置的设备名称都是允许的。
- 以字符 “//” 开头，后面至少跟一个字母或一个数字。
- 后面的字符可以是字母、数字、"_"和空格的任意组合。
- 名称的结尾可以是一个字母或一个数字，后面跟一个字符 “:” 。
- 禁止使用其它特殊字符。

说明

为被动文件系统预定义了驱动器名称"//NC:"。

示例：

- 外部程序存储器：
  - //Drive1:
  - //Drive_1:
  - //Drive 1:
  - //A B:
  - //1 B C 2:

目录路径

目录路径的说明适用以下规定：

- 目录路径的开始和末尾以及单个路径部分的分隔符使用"/"。

  说明

  目录路径中不允许使用双斜杠（"//"）！

- 目录名：
  - 目录名必须以字母或数字开始。只有在 NC 标识符定址时目录名才能使用域标识符“_N_”开始。
  - 后面的字符可以是字母、数字和 “_” 的任意组合。

    说明

    外部程序存储器也允许在目录名中使用空格。但如果外部程序存储器设置为全球零件程序存储器（GDIR），则不适用。

  - 禁止使用其它特殊字符。
- 目录扩展：
  - 目录扩展必须由三个字母/数字组成。
  - 使用“_”（NC 标识符）或“.”（EES 标识符）与目录名称分隔开来。

  说明

  在被动文件系统中只有目录扩展_DIR 和_WPD。

示例：

- 被动文件系统或 GDIR：
  - NC 标识符: _N_WKS_DIR/_N_MYNCPROGS_WPD/...
  - EES 标识符：WKS.DIR/MYPROGS.WPD/...
- 外部程序存储器：
  - /abc
  - /ab_c.def
  - /ab c1.def
  - /a b c .d11
  - /abc.def/ghi.klm

文件名

文件名须遵循以下规定：

- 只有在 NC 标识符定址时文件名才能使用域标识符 “_N_” 开始。
- 接下来的两个字符应该是两个字母或是下划线加一个字母。

  说明

  当满足该条件时，才能够仅仅通过输入程序名称将一个 NC 程序作为子程序从其他程序中进行调用。反之，如果程序名称使用数字开头，那么子程序调用就只能通过 CALL 指令进行。

- 后面的字符可以是字母、数字和 “_” 的任意组合。
- 文件扩展名：
  - 文件扩展名必须由三个字母/数字组成。

    说明

    被动文件系统中允许的文件扩展名请参见“NC 中的程序存储器 (页 231)”。

  - 使用“_”（NC 标识符）或“.”（EES 标识符）与文件名分隔开来。

示例：

- 被动文件系统或 GDIR：
  - NC 标识符: _N_SUB1_SPF
  - EES 标识符：SUB1.SPF
- 外部程序存储器：
  - 部分 1
  - _Teil1
  - Teil_1.spf
  - Teil1.mpf

DIN 子程序名称

DIN 子程序名称须遵循以下规定：

- 第一个字符必须是字母 “L”。
- 后面的字符是数字（至少要有一个数字）。
- 文件扩展名：
  - 文件扩展名必须由三个字母组成。
  - 使用“_”（NC 标识符）或“.”（EES 标识符）与文件名分隔开来。

示例：

- L123
- L1_SPF (NC 标识符) 或 L1.SPF (EES 标识符)

最大路径长度

驱动器名称和目录路径说明最大不超过 128 字节，文件名的长度最长不得超过 31 字节。整个路径的最大长度为 159 字节。

3.1.4 子程序调用时的查找路径

在进行无路径说明的子程序调用时，绝对路径是通过默认查找路径确定的。

此时，系统会按如下顺序查找程序存储器：

		目录	说明
1		当前目录/*name*	当前目录是指在其中选择程序的目录。 它们可以是： ● NC 零件程序存储器或全局零件程序存储器中的工件目录或标准目录 _N_MPF_DIR 或者 ● 一个外部程序存储器的任意一个目录
2		当前目录/*name_SPF*	
3		当前目录/*name_MPF*	
4	a	//NC:/_N_SPF_DIR / *name_SPF*	NC 零件程序存储器中的子程序目录
	b	//DEV2:/_N_SPF_DIR / name_SPF [1]	全局零件程序存储器中的子程序目录 **注：** 如果没有设置全局零件程序存储器或者在 NC 零件程序存储器中选择程序，则可以省略这一步。
5		借助 CALLPATH 编写的查找路径扩展（参见 “扩展调用子程序时的路径查找 (CALLPATH) (页 217)”）。 **注：** 如果没有编写 CALLPATH，则可以省略这一步。	
6		/_N_CUS_DIR / *名称_SPF*	用户循环目录
7		/_N_CMA_DIR / *名称_SPF*	制造商循环目录
8		/_N_CST_DIR / *名称_SPF*	标准循环目录

[1] 例如 //DEV2:" 用于设置了全局零件程序存储器的驱动器。

查找须遵循以下规定：

- 查找路径会贯穿每个子程序调用，即：与上级程序的位置无关。
- 根据目录采用不同的文件类型。
- 原则上，系统不会在一个目录的子目录或次级目录中查找。

3.1.5 查询路径和文件名

以下在零件程序中可读的系统变量可用于查询 NC 程序的路径和文件名：

系统变量	类型	含义
$P_STACK	INT	提供在其内执行当前 NC 程序的程序级。
$P_PATH[<n>]	STRING	提供在通过数组索引 <n> 选择的程序级内执行的 NC 程序的路径。 示例： $P_PATH[0] 提供主程序的路径，如：“/_N_WKS_DIR/_N_WELLE_WPD/”。 $P_PATH[$P_STACK - 1] 提供主调程序的路径。 如果该路径针对的是保存在 NC 的被动文件系统或全局零件程序存储器 (GDIR) 中的 NC 程序，系统则会提供采用 NC 标识符的路径。 如果该路径针对的是一个由另一个外部程序存储器（而非全局零件程序存储器）执行的 NC 程序，$P_PATH 则会提供采用 EES 标识符的路径。
$P_PROG[<n>]	STRING	提供在通过数组索引 <n> 选择的程序级内执行的 NC 程序的名称。 如果该 NC 程序保存在 NC 的被动文件系统或全局零件程序存储器中，系统则会提供采用 NC 标识符的程序名称。 如果该 NC 程序是由另一个外部驱动器（而非全局零件程序存储器）执行的，$P_PROG 则会提供采用 EES 标识符的名称。
$P_PROGPATH	STRING	提供正在执行的 NC 程序的路径。 $P_PROGPATH 的调用方式与 $P_PATH[$P_STACK] 的相同。

系统变量	类型	含义	
$P_IS_EES_PATH[<n>]	BOOL	询问 $P_PATH[<n>] 提供的路径或 $P_PROG[<n>] 提供的程序名称是否与 NC 标识符或 EES 标识符相符。	
		= FALSE	$P_PATH[<n>] 和 $P_PROG[<n>] 提供的数据都采用 NC 标识符。也就是说，每个名称前面都有一个前缀 “_N_” 。文件标识符的分隔符为 “_” 。 示例： • 采用 NC 标识符的路径："/_N_WKS_DIR/_N_MYWPD_WPD/" • 采用 NC 标识符的程序名称："_N_MYPROG_MPF" 采用 NC 标识符的路径既可以针对 NC 中的被动文件系统，也可以针对全局零件程序存储器。
		= TRUE	$P_PATH[<n>] 和 $P_PROG[<n>] 提供的数据都采用 EES 标识符。也就是说，名称前面不可以有前缀 “_N_” 。文件标识符的分隔符为 “.” 。 示例： • 采用 EES 标识符的路径：“//DEV1:/WKS.DIR/MYWPD.WPD/” • 采用 EES 标识符的程序名称："MYPROG.MPF"

<n>：索引 **<n>** 定义的是从其中读取路径信息的程序级（值域：0 ... 17）

说明

在 EES 模式中，除全局零件程序存储器 (GDIR) 外，系统变量 $P_PROG、$P_PATH 和 $P_PROGPATH 提供的都是采用 EES 标识符的路径名称。因此，必须对 EES 模式中用于分析和进一步处理这些路径名称的用户程序进行扩展，使其也能处理采用 EES 标识符的路径名称。

3.2 工作存储器 (CHANDATA, COMPLETE, INITIAL)

功能

工作存储器包含当前的系统数据和用户数据，控制系统以此数据运行（有源文件系统），例如

- 激活的机床数据
- 刀具补偿数据
- 零点偏移
- ...

初始化程序

这里讨论工作存储器数据可以预置（初始化）时如何编程。可以使用以下的文件类型：

文件类型	说明
name_TEA	机床数据
name_SEA	设定数据
name_TOA	刀具补偿
name_UFR	零点偏移/框架
name_INI	初始化文件
name_GUD	全局用户数据
name_RPA	R 参数

数据区

数据可以划分为不同的区。例如，某个控制系统可以具有多个通道，通常也可拥有多个轴。

有：

标记	数据区
NC	NC 专用数据
CH<n>	通道特有的数据（<n> 用来指定通道号）
AX<n>	轴特有的数据（<n> 用来指定机床轴的编号）
TO	刀具数据
COMPLETE	所有数据

在外部计算机上生成初始化程序

利用数据区标志和数据类型标志，可以确定数据保护时视作数组的数据区：

_N_AX5_TEA_INI	用于第 5 轴的机床数据
_N_CH2_UFR_INI	通道 2 框架
_N_COMPLETE_TEA_INI	所有机床数据

在系统开机调试之后在工作存储器中有一个数据组，它保证控制系统正常运行。

多通道控制系统的工作步骤 (CHANDATA)

用于多个通道的 CHANDATA(<通道号>) 仅在文件 N_INITIAL_INI 中允许。这是调试文件，用来初始化控制系统的所有数据。

程序代码	注释
%_N_INITIAL_INI	
CHANDATA(1)	
	; 机床轴分配通道 1：
$MC_AXCONF_MACHAX_USED[0]=1	
$MC_AXCONF_MACHAX_USED[1]=2	
$MC_AXCONF_MACHAX_USED[2]=3	
CHANDATA(2)	
	; 机床轴分配通道 2：
$MC_AXCONF_MACHAX_USED[0]=4	
$MC_AXCONF_MACHAX_USED[1]=5	
CHANDATA(1)	
	; 轴机床数据：
	; 粗准停窗口：
$MA_STOP_LIMIT_COARSE[AX1]=0.2	; 轴 1
$MA_STOP_LIMIT_COARSE[AX2]=0.2	; 轴 2
	; 精准停窗口：
$MA_STOP_LIMIT_FINE[AX1]=0.01	; 轴 1
$MA_STOP_LIMIT_FINE[AX1]=0.01	; 轴 2

注意

CHANDATA-语句

在零件程序中，只可以将 CHANDATA 指令设置给执行 NC 程序的通道。也就是说，可以将该语句用来防止 NC 程序在非配置的通道上执行。

在故障时停止程序执行。

说明

在工作表中的 INI 文件不含 CHANDATA 指令。

保存初始化程序 (COMPLETE, INITIAL)

工作存储器的文件可以保护到一个外部 PC 中，并可以从那儿再次读入。

- 使用 COMPLETE 备份文件。
- 使用 INITIAL 通过所有范围生成一个 INI(_N_INITIAL_INI) 文件。

读入初始化程序

注意
数据丢失 如果读入名称“INITIAL_INI”的文件，则对所有文件中未提供的数据用标准数据进行初始化。只有机床数据除外。也提供**设置数据，刀具数据，NPV, GUD 值, ...** 与标准数据（一般情况下“零”)。

为了读入单独的机床数据，例如适用于文件 COMPLETE_TEA_INI。在该文件中控制系统仅等待机床数据。为此在这种情况下其他数据范围保持不变。

加载初始化程序

如果 INI 程序仅使用一个通道的数据，则它也可以作为零件程序选择并调用。因此也就可以初始化程序控制的文件。

保护区 4

4.1 定义保护区 (CPROTDEF、NPROTDEF)

防止机床部件碰撞的保护区分别在零件程序的程序块中定义。其中包含以下元素：

1. 确定工作平面
   定义保护区前须选择保护区轮廓描述的工作平面。
2. 定义开始指令
   根据 NC 指令，创建一个通道专用或机床专用保护区。
3. 保护区轮廓描述
   保护区的轮廓是借助运行动作来描述的。这些运行动作不会被执行，与之前的或后续的几何尺寸描述也没有任何关系。它们只用于定义保护区。
4. 定义结束指令

句法

```
DEF INT <Var>
G17/G18/G19
CPROTDEF/NPROTDEF(<n>,<t>,<AppLim>,<AppPlus>,<AppMinus>)
G0/G1/...X/Y/Z...
...
EXECUTE(<Var>)
```

含义

DEF INT <Var>:	定义数据类型为 INTEGER 的本地辅助变量
<Var>:	辅助变量名称
G17/G18/G19:	工作平面 **说明：** 定义结束前不可以更改工作平面。在定义开始和结束之间不允许编程应用。
CPROTDEF():	用于定义**通道**专用保护区的预定义程序
NPROTDEF():	用于定义**机床**专用保护区的预定义程序

<n>:	定义的保护区序号		
	数据类型:	INT	
<t>:	保护区类型		
	数据类型:	BOOL	
	值:	TRUE	与**刀具**有关的保护区
		FALSE	与**工件**有关的保护区
<AppLim>:	第 3 个维度的限制类型		
	数据类型:	INT	
	值:	0	无限制
		1	正向限制
		2	负向限制
		3	正负方向的限制
<AppPlus>:	第 3 个维度正向的限制值		
	数据类型:	实数	
<AppMinus>:	第 3 个维度负向的限制值		
	数据类型:	实数	

G0/G1/...X/Y/Z... ...:	保护区的轮廓可以最多通过所选工作平面中的 11 次运行动作来描述。其中第一次运行为向轮廓的运动。轮廓描述的最后一个点必须始终与第一个点重合。 轮廓左侧的区域作为保护区： ● 内部保护区 必须逆时针方向描述内部保护区的轮廓。 ● 外部保护区（只允许用于与工件相关的保护区） 必须顺时针方向描述外部保护区的轮廓。 允许以下轮廓单元： ● G0、G1 用于直线轮廓单元 ● G2 用于顺时针方向的圆弧段 只允许用于与工件相关的保护区！由于保护区只允许是凸面的，因此只能用于与刀具相关的保护区。 ● G3 用于逆时针圆弧段 **说明：** 无法通过一个整圆来描述保护区。一个整圆必须划分为两个半圆。 **说明：** 不允许使用顺序 G2 → G3 或 G3 → G2。必须在两个圆弧程序段之间插入一个较短的 G1 程序段。
EXECUTE(<Var>):	标记定义结束的预定义程序 通过 EXECUTE 切回正常程序加工。

示例

参见“激活/取消激活保护区 (CPROT, NPROT) (页 252)”一章中的示例。

更多信息

机床专用保护区

机床专用的保护区及其轮廓是借助几何轴，即参照一个通道的基准坐标系 (BCS) 来定义的。为了能在所有通道（含激活的机床专用保护区的通道）中进行正确的保护区监控，所有相关通道的基准坐标系 (BCS) 必须是一致的。

- 即坐标原点相对于机床零点和坐标轴定向的位置必须是一致的。
- 坐标轴定向

轮廓描述中的基准点

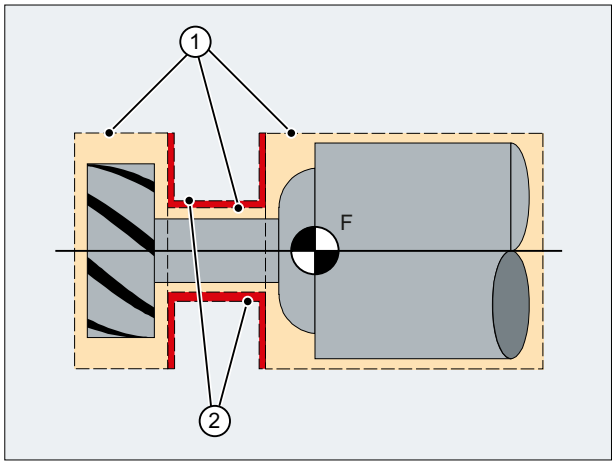
- 与刀具有关的保护区：
  与**刀具**相关的保护区的坐标必须**以参照刀架参考点 F 的绝对值的形式**给出。
- 与工件相关的保护区
  与**工件**相关的保护区的坐标必须**以参照基准坐标系零点的绝对值的形式**给出。

旋转对称保护区

对于旋转对称的保护区（例如主轴卡盘），必须定义全部轮廓（不仅仅到旋转中心为止）。

与刀具有关的保护区：

与刀具有关的保护区必须始终为凸面 。如果希望有一个凹面保护区，则可以把它分成多个凸面保护区。



① 凸面保护区

② 凹面保护区（**不允许！**）

F 刀架参考点

边界条件

定义保护区时允许以下功能未生效或未被使用：

- 刀具半径补偿（铣刀半径补偿、刀沿半径补偿）
- 转换
- 回参考点 (G74)
- 回固定点 (G75)
- 暂停时间 (G4)
- 程序段预处理停止 (STOPRE)

- 程序结束 (M17、M30)
- M 功能：M0、M1、M2

4.2 激活/取消激活保护区 (CPROT, NPROT)

可随时激活或之后通过 PLC 用户程序预激活之前在零件程序中定义的保护区。可随时再次取消激活生效的保护区。

此外，还可通过移动保护区的参考点激活或预激活。

说明

只有在通道中的所有几何轴都回参考点后才需考虑保护区。

说明

保护区监控

如果没有刀具相关的保护区有效，则按照工件相关的保护区对刀具轨迹进行检查。
如果没有工件相关的保护区有效，则不进行保护区监控。

句法

```
CPROT(<n>,<Status>,<XMov>,<YMov>,<ZMov>)
NPROT(<n>,<Status>,<XMov>,<YMov>,<ZMov>)
```

含义

CPROT:	用于激活**通道**专用保护区的预定义程序		
NPROT:	用于激活**机床**专用保护区的预定义程序		
<n>:	保护区序号		
	数据类型:	INT	
<Status>:	通过该参数设置通道专用激活状态		
	数据类型:	INT	
	值:	0	取消激活保护区
		1	预先激活保护区
		2	激活保护区
		3	预先激活保护区，有条件停止

<XMov>,<YMov>,<ZMov>:	X/Y/Z 方向上的附加偏移值 可以用 1、2 或 3 维尺寸偏移。偏移值与以下相关： ● 与**工件**相关的保护区中的机床零点 ● 与**刀具**相关的保护区中的刀架基准点 F	
	数据类型：	实数

示例

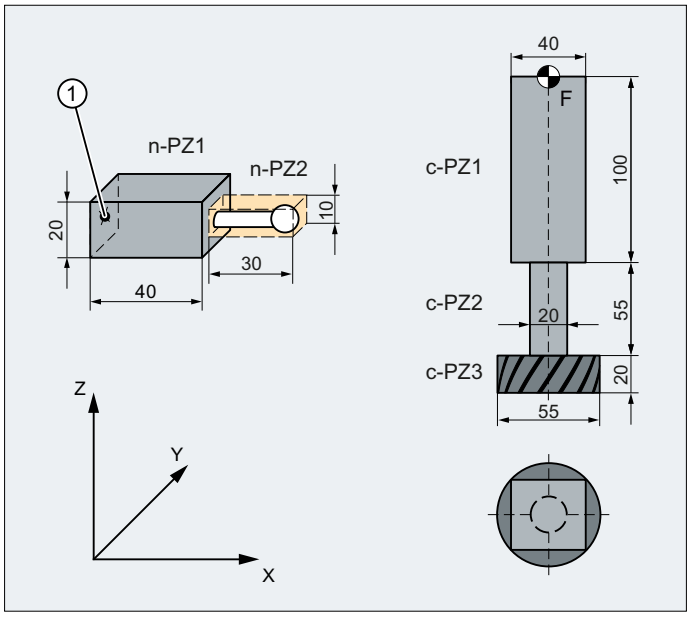
对于铣床而言，应对铣刀与探头可能会有的碰撞进行监控。探头的位置应在激活时通过位移来设定。

为此，定义以下的保护区：

- 探头支架 (n-PZ1) 和探头自身 (n-PZ2) 各有一个机床专用的保护区、一个与工件有关的保护区。
- 铣刀刀夹 (c-PZ1)、铣刀柄 (c-PZ2) 和铣刀自身 (c-PZ3) 各有一个通道专用的保护区、一个和刀具相关的保护区。

所有保护区的定向均在 Z 方向中。

当激活时，探头的参考点位置应为 X = -120, Y = 60 和 Z = 80 。



① 测头保护区的参考点

F 刀架参考点

程序代码	注释
DEF INT PROTZONE	;定义一个辅助变量
G17	; 工作平面 XY
; 定义保护区：	
NPROTDEF(1,FALSE,3,10,-10)	; 保护区 n-PZ1
G01 X0 Y-10	
X40	
Y10	
X0	
Y-10	
EXECUTE(PROTZONE)	
NPROTDEF(2,FALSE,3,5,-5)	; 保护区 n-PZ2
G01 X40 Y-5	
X70	
Y5	
X40	
Y-5	
EXECUTE(PROTZONE)	
CPROTDEF(1,TRUE,3,0,-100)	; 保护区 c-PZ1
G01 X-20 Y-20	
X20	
Y20	
X-20	
Y-20	
EXECUTE(PROTZONE)	
CPROTDEF(2,TRUE,3,-100,-150)	; 保护区 c-PZ2
G01 X0 Y-10	
G03 X0 Y10 J10	
X0 Y-10 J-10	
EXECUTE(PROTZONE)	
CPROTDEF(3,TRUE,3,-150,-170)	; 保护区 c-PZ3
G01 X0 Y-27.5	
G03 X0 Y27.5 J27.5	
X0 Y27.5 J-27.5	
EXECUTE(PROTZONE)	
; 激活保护区	
NPROT(1,2,-120,60,80)	; 激活带偏移的保护区 n-PZ1
NPROT(2.2,-120,60,80)	; 激活带偏移的保护区 n-PZ2
CPROT(1,2,0,0,0)	; 激活保护区 c-PZ1
CPROT(2,2,0,0,0)	; 激活保护区 c-PZ2
CPROT(3,2,0,0,0)	; 激活保护区 c-PZ3

其它信息

控制系统启动后的激活状态

保护区在控制系统启动以及轴回参考点后生效。这是当以下系统变量的保护区被设为 TRUE 时的情况：

- $SN_PA_ACTIV_IMMED[<n>]（用于机床专用的保护区）或
- $SC_PA_ACTIV_IMMED[<n>]（用于通道专用的保护区）
  索引 “<n>” 对应保护区的编号：0 = 1。保护区

通过状态 = 2 激活保护区且不带偏移。

多次激活保护区

机床专用保护区同时可以在多个通道中生效（例如两个相对滑板的顶尖套筒）。只有当所有的几何轴都回参考点之后，才可以监控保护区。

在一个通道中，保护区不能同时激活不同的偏移。

刀具半径补偿激活时的保护区监控

在刀具半径补偿激活时，仅在刀具半径补偿的平面和保护区定义的平面相同时才能进行有效的保护区监控。

4.3 检查保护区、工作区域限制和软件限位开关 (CALCPOSI)

功能

CALCPOSI 函数可检查在工件坐标系 (WCS) 中**几何轴**从起点起按指定行程运行是否会超出当前激活的各个限位。如果因为限位几何轴不能完成指定行程，则系统会反馈一个状态值（十进制正值）和允许的最大行程。

定义

```
INT CALCPOSI(VAR REAL[3] <Start>, VAR REAL[3] <Dist>, VAR REAL[5]
<Limit>, VAR REAL[3] <MaxDist>, BOOL <MeasSys>, INT <TestLim>)
```

句法

```
<Status> = CALCPOSI(VAR <Start>, VAR <Dist>, VAR <Limit>, VAR
<MaxDist>, <MeasSys>, <TestLim>)
```

含义

CALCPOSI(...):	用于检查几何轴是否超限的预定义功能	
	预处理停止：	否
	在单独程序段中编程：	是

<Status>： （第 1 部分）	函数的返回值。负值代表故障状态。		
	数据类型：		INT
	取值范围：		-8 ≤ x ≤ 100000
	值：	0	可完成整段行程
		-1	在<Limit>中至少有一个分量为负。
		-2	坐标转换计算中出错。 示例：轴穿过奇点，以至于无法确定轴位置。
		-3	指定的行程<Dist>和允许的最大行程<MaxDist>呈线性关系。 **提示** 只能与 <TestLim>，位 4 == 1 同时出现。
		-4	<Dist>包含的运行方向投影到限位面上为零矢量或运行方向垂直于被超出的限位面。 **提示** 只能与 <TestLim>，位 5 == 1 同时出现。
		-5	<TestLim>中，位 4 == 1 且位 5 == 1
		-6	至少有一个需要检测其是否超限的机床轴没有回参考点。
		-7	防撞功能：运动链或保护区定义无效。
		-8	防撞功能：该功能可能因内存不足而无法执行。

<Status>: （第 2 部分）	个位 **提示** 如果同时出现多个超限错误，则个位上显示的是导致行程缩减幅度最大的限位。		
	值：	1	软件限位开关限制了行程
		2	工作区域限制了行程
		3	保护区限制了行程
		4	防撞功能：保护区限制了行程
	十位		
	值：	1x	起点超限
		2x	指定的直线超限。 当终点自身没有超限，但是在从起点到终点的行程中却有可能超限时（例如穿过保护区，进行诸如 Transmit 非线性转换时 WCS 中的软件限位开关弯曲），也会返回该值。
<Status>: （第 3 部分）	百位		
	值：	1xx	个位 == 1 或 2： 超出正限值。
			个位 == 3 [1]： 超出 NC 专用的保护区。
		2xx	个位 == 1 或 2： 超出负限值。
			个位 == 3 [1]： 超出通道专用的保护区。
<Status>: （第 4 部分）	千位		
	值：	1xxx	个位 == 1 或 2： 因子，与轴号相乘，超出限值。轴从 1 开始计数。 基准： ● 软件限位开关：机床轴 ● 工作区域限制：几何轴 个位 == 3 [1]： 因子，与超出保护范围的号码相乘。

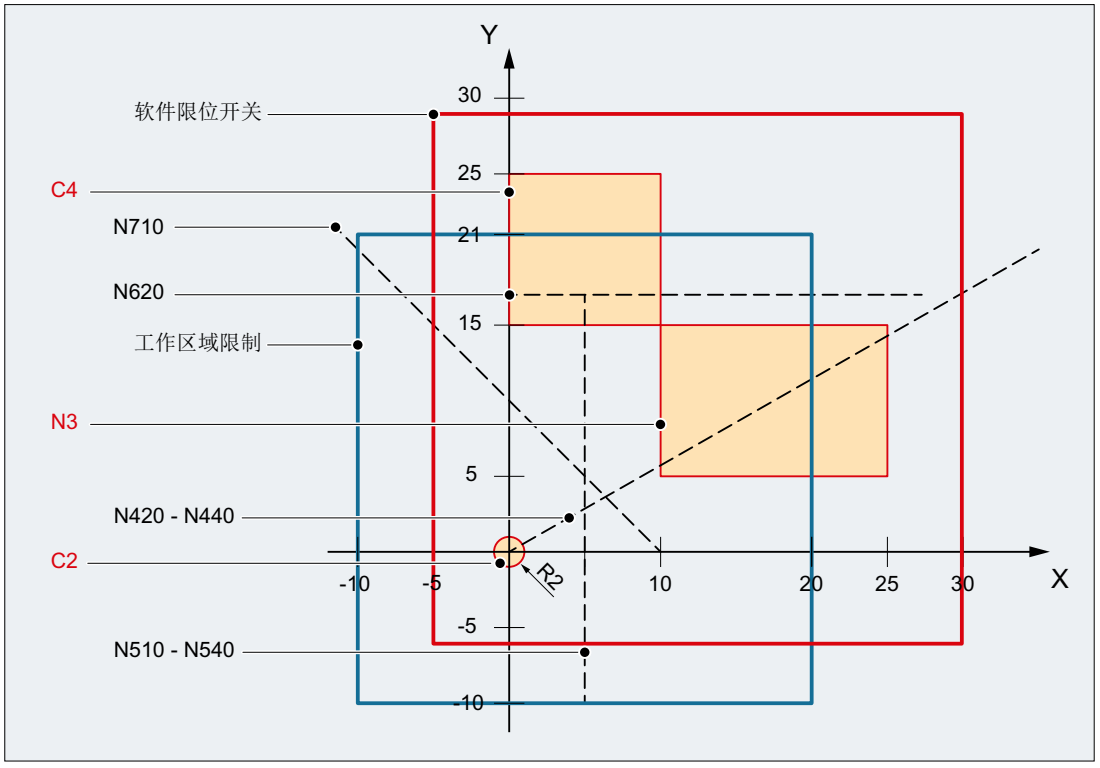
<Status>: （第 5 部分）	十万位		
	值：	0xxxxx	十万位 == 0：<Dist> 保持不变
		1xxxxx	<Dist>中返回一个方向矢量，该矢量定义了限位面上的后续运动方向。 只允许在以下边界条件中出现： • 超出软件限位开关或工作区域限制（不在起点） • 坐标转换**未**激活 • <TestID>，位 4 或 5 == 1
<Start>:	含起始位置的矢量： • <Start> [0]:第 1 几何轴 • <Start> [1]:第 2 几何轴 • <Start> [2]:第 3 几何轴		
	参数类型：		输入
	数据类型：		VAR REAL [3]
	取值范围：		最大负实数值 ≤ x[<n>] ≤ 最大正实数值
<Dist>:	参照矢量。		
	输入：增量行程 • <Dist> [0]:第 1 几何轴 • <Dist> [1]:第 2 几何轴 • <Dist> [2]:第 3 几何轴		
	输出（仅限<Status>中的十万位置位时）：		
	<Dist> 的输出值为 WCS 中后续运行方向的单位矢量 **v**。		
	情形 1：<TestID> 位 4 == 1 时矢量 **v** 的计算 输入矢量 <Dist> 和 <MaxDist> 构成了运动面。该平面与受侵犯的限位面相交。两平面的相交线定义了矢量 **v** 的方向。此时系统会选择合适的定位方向（符号），以确保输入矢量 <MaxDist> 和 **v** 之间的夹角不超过 90 度。		
	情形 2：<TestID> 位 5 == 1 时矢量 **v** 的计算 矢量 **v** 是一个方向矢量，即 <Dist> 包含的方向矢量在限位面上的投影。如果该投影为零矢量，则返回错误。		
	参数类型：		输入/输出
	数据类型：		VAR REAL [3]
	取值范围：		最大负实数值 ≤ x[<n>] ≤ 最大正实数值

<Limit>:	参照长度为 5 的数组。 • **<Limit> [0 - 2]**：几何轴到限位的最小间距： – **<Limit> [0]**:第 1 几何轴 – **<Limit> [1]**:第 2 几何轴 – **<Limit> [2]**:第 3 几何轴 以下情况下需要遵守最小间距： – 工作区域限制：没有限制 – 软件限位开关：坐标转换未激活，或某个几何轴可明确指定为线性机床轴的坐标转换激活（比如 5 轴转换）。 • **<Limit> [3]**:包含线性机床轴的最小间距，例如：机床轴由于非线性坐标转换没有明确的几何轴。此外，该值还用作传统保护区和防撞保护区的限值。 • **<Limit> [4]**:包含旋转机械轴的最小距离，例如由于非线性变换没有几何轴可以分配。 **注** 该值只在监控特殊转换软件限位开关时有效。	
	参数类型：	输入
	数据类型：	VAR REAL [5]
	取值范围：	最大负实数值 ≤ x[n] ≤ 最大正实数值
<MaxDist>:	含增量行程的矢量，即所有机床轴的最大行程，在该行程内，所有机床轴都可以和限位保持指定的最小间距。 • **<Dist> [0]**:第 1 几何轴 • **<Dist> [1]**:第 2 几何轴 • **<Dist> [2]**:第 3 几何轴 如果行程没有受限，则该返回参数的内容等同于**<Dist>**的内容。	
	<TestID>，位 4 == 1：**<Dist>** 和 **<MaxDist>** **<MaxDist>** 和 **<Dist>** 必须包含构成运动面的输入矢量。两个矢量之间不允许呈线性关系。**<MaxDist>** 值任意。运动方向的计算参见 **<Dist>** 说明。	
	参数类型：	输出
	数据类型：	VAR REAL [3]
	取值范围：	最大负实数值 ≤ x[<n>] ≤ 最大正实数值

<table>
<tr><td rowspan="4"><MeasSys>:</td><td colspan="3">位置和长度的单位（英制/公制）（可选）</td></tr>
<tr><td colspan="2">数据类型：</td><td>BOOL</td></tr>
<tr><td rowspan="2">值：</td><td>FALSE（默认）</td><td>单位根据 G 代码组 13 中当前激活的 G 代码 (G70, G71, G700, G710)。
提示
G70 激活、基本单位制为公制或 G71 激活、基本单位制为英制时，则基本系统中返回系统变量 $AA_IW 和 $AA_MW，有可能必须经过换算以供 CALCPOSI 使用。</td></tr>
<tr><td>TRUE</td><td>符合所设基本单位制的单位制：
MD52806 $MN_ISO_SCALING_SYSTEM</td></tr>
<tr><td rowspan="11"><TestLim>:</td><td colspan="3">选择需要监控的限位，位编码（可选）</td></tr>
<tr><td colspan="2">数据类型：</td><td>INT</td></tr>
<tr><td colspan="2">默认值：</td><td>位 0、1、2、3、6、7 == 1 (207)</td></tr>
<tr><td>位</td><td>十进制</td><td>含义</td></tr>
<tr><td>0</td><td>1</td><td>软件限位开关</td></tr>
<tr><td>1</td><td>2</td><td>工作区域限制</td></tr>
<tr><td>2</td><td>4</td><td>激活的传统保护区</td></tr>
<tr><td>3</td><td>8</td><td>预激活的传统保护区</td></tr>
<tr><td>4</td><td>16</td><td>越过软件限位开关或工作区域限制时，在<Dist>中返回运行方向，同上文情形 1。</td></tr>
<tr><td>5</td><td>32</td><td>越过软件限位开关或工作区域限制时，在<Dist>中返回运行方向，同上文情形 2。</td></tr>
<tr><td>6</td><td>64</td><td>激活的防撞保护区</td></tr>
<tr><td>7</td><td>128</td><td>预激活的防撞保护区</td></tr>
<tr><td>8</td><td>256</td><td>激活的防撞保护区和预激活的防撞保护区</td></tr>
<tr><td colspan="4">1) 如果同时出现多个保护区超限错误，则显示的是对设定行程约束力最强的保护区。</td></tr>
</table>

示例

限制



示例中显示了 XY 平面上有效的软件限位开关和工作区域限制及以下三个保护区：

- C2:与刀具有关的通道专用的保护区，激活、圆弧形、半径 = 2 mm
- C4:与工件有关的通道专用的保护区，预激活、正方形、长度 = 10 mm
- N3：机床专用的保护区，激活、矩形、长宽 = 10 mm x 15 mm

NC 程序

首先在 NC 程序中定义保护区和加工区域限制。接着调用含多个参数的函数 `CALCPOSI()`。

程序代码

```
N10 DEF REAL _START[3]
N20 DEF REAL _DIST[3]
N30 DEF REAL _LIMIT[5]
N40 DEF REAL _MAXDIST[3]
N50 DEF INT _PA
N60 DEF INT _STATUS
```

程序代码

```
; 刀具相关的保护区 C2
N70 CPROTDEF(2, TRUE, 0)
N80 G17 G1 X-2 Y0
N90 G3 I2 X2
N100 I-2 X-2
N110 EXECUTE(_PA)
; 工件相关的保护区 C4
N120 CPROTDEF(4, FALSE, 0)
N130 G17 G1 X0 Y15
N140 X10
N150 Y25
N160 X0
N170 Y15
N180 EXECUTE(_PA)
; 机床专用的保护区 N3
N190 NPROTDEF(3, FALSE, 0)
N200 G17 G1 X10 Y5
N210 X25
N220 Y15
N230 X10
N240 Y5
N250 EXECUTE(_PA)
; 激活或者预激活保护区
N260 CPROT(2, 2, 0, 0, 0)
N270 CPROT(4, 1, 0, 0, 0)
N280 NPROT(3, 2, 0, 0, 0)
; 定义工作区域限制
N290 G25 XX=-10 YY=-10
N300 G26 XX=20 YY=21
N310 _START[0] = 0.
N320 _START[1] = 0.
N330 _START[2] = 0.
N340 _DIST[0] = 35.
N350 _DIST[1] = 20.
N360 _DIST[2] = 0.
N370 _LIMIT[0] = 0.
N380 _LIMIT[1] = 0.
N390 _LIMIT[2] = 0.
N400 _LIMIT[3] = 0.
N410 _LIMIT[4] = 0.
N420 _STATUS = CALCPOSI(_START, _DIST, _LIMIT, _MAXDIST)
N430 _STATUS = CALCPOSI(_START, _DIST, _LIMIT, _MAXDIST,,3)
N440 _STATUS = CALCPOSI(_START, _DIST, _LIMIT, _MAXDIST,,1)
N450 _START[0] = 5.
N460 _START[1] = 17.
N470 _START[2] = 0.
```

程序代码

```
N480 _DIST[0] = 0.
N490 _DIST[1] =-27.
N500 _DIST[2] = 0.
N510 _STATUS = CALCPOSI(_START, _DIST, _LIMIT, _MAXDIST,,14)
N520 _STATUS = CALCPOSI(_START, _DIST, _LIMIT, _MAXDIST,, 6)
N530 _LIMIT[1] = 2.
N540 _STATUS = CALCPOSI(_START, _DIST, _LIMIT, _MAXDIST,, 6)
N550 _START[0] = 27.
N560 _START[1] = 17.1
N570 _START[2] = 0.
N580 _DIST[0] =-27.
N590 _DIST[1] = 0.
N600 _DIST[2] = 0.
N610 _LIMIT[3] = 2.
N620 _STATUS = CALCPOSI(_START, _DIST, _LIMIT, _MAXDIST,,12)
N630 _START[0] = 0.
N640 _START[1] = 0.
N650 _START[2] = 0.
N660 _DIST[0] = 0.
N670 _DIST[1] = 30.
N680 _DIST[2] = 0.
N690 TRANS X10
N700 AROT Z45
N710 _STATUS = CALCPOSI(_START,_DIST, _LIMIT, _MAXDIST)
; 从 N690 和 N700 中再次删除框架
N720 TRANS
N730 _START[0] = 0.
N740 _START[1] = 10.
N750 _START[2] = 0.
; 矢量 _DIST 和 _MAXDIST 定义运动平面
N760 _DIST[0] = 30.
N770 _DIST[1] = 30.
N780 _DIST[2] = 0.
N790 _MAXDIST[0] = 1.
N800 _MAXDIST[1] = 0.
N810 _MAXDIST[2] = 1.
N820 _STATUS = CALCPOSI(_START, _DIST, _LIMIT, _MAXDIST,,17)
N830 M30
```

CALCPOSI() 的结果

N...	<Status>	<MaxDist>[0] ≙ X	<MaxDist>[1] ≙ Y	注释
420	3123	8.040	4.594	进入保护区 N3。
430	1122	20.000	11.429	无保护区监控， 超限。
440	1121	30.000	17.143	仅软件限位开关监控仍有效。
510	4213	0.000	0.000	起点在 C4 保护区中
520	0000	0.000	-27.000	预激活的 C4 没有被监控。可以完成指定行程。
540	2222	0.000	-25.000	_LIMIT[1] = 2，行程因工作区域限制受限。
620	4223	-13.000	0.000	由于 C2 和 _LIMIT[3]，到 C4 的间距共为 4 毫米。0.1 毫米的 C2 → N3 间距不会导致行程受限。
710	1221	0.000	21.213	平移和旋转的框架有效。_DIST 中的允许行程在经过位移和旋转的坐标系 (WCS) 中适用。
820	102121	18.000	18.000	越过了 Y 轴软件限位开关。设置 <_TESTLIM> = 17 请求计算后续方向。该方向位于 <_DIST> (0.707, 0.0, 0.707) 中。它是有效的，因为 <_STATUS> 的十万位已置位。

更多信息

轴状态“已回参考点”

所有需要由 CALCPOSI() 检查的机床轴必须回参考点。

圆弧相关的行程数据

所有与圆弧相关的行程数据**总是**被视为半径数据。在直径编程（DIAMON / DIAM90）激活的车床 X 轴上尤其要注意这一点。

行程缩短

如果某个轴指定的行程受限，则在<MaxDist>的返回值中，其他轴的行程也相应缩短。终点因此可仍位于指定的轨迹上。

回转轴

该功能只能监控非模数回转轴。

允许不为一个或多个轴设置软件限位开关、工作区域限制或者保护区。

软件限位开关和工作区域限制状态

仅当执行 CALCPOSI()时软件限位开关和工作区域限制已激活，这两个限位才会受到监控。状态可能会受下列因素影响，例如：

- 机床数据：MD21020 $MC_WORKAREA_WITH_TOOL_RADIUS
- 设定数据：$AC_WORKAREA_CS_...
- NC/PLC 接口信号：DB31, ... DBX12.2 / 3
- 指令：WALIMON / WALIMOF

软件限位开关和坐标转换

CALCPOSI() 和坐标转换（如 TRANSMIT 综合使用时，行程内的某些位置可能具有多个含义，因此系统无法再由几何轴的位置 (WCS) 计算出唯一的机床轴位置 (MCS)。而在正常运行中，WCS 中几何轴的连续运行和 MCS 中的机床轴连续运行一一对应，因此可确保位置的唯一性。所以在执行 CALCPOSI() 对软件限位开关进行监控前，需要使用当前的机床位置，以避免该问题。

说明

预处理停止

CALCPOSI()和坐标转换同时应用时，由用户自行负责在执行 CALCPOSI() 之前编程预处理停止指令 (STOPRE)，以便实现机床轴位置的同步。

保护区间距和传统的保护区

传统的保护区**不能**确保轴在指定行程中和所有保护区都相距 <Limit>[3] 中设置的安全间距。它只能保证，<Dist> 中返回的终点向前移动了安全间距，但没有进入保护区。但可能轴已经非常靠近保护区。

保护区间距和防撞保护区

防撞保护区能确保轴在指定行程中和所有保护区都相距参数 <Limit>[3] 中设置的安全间距。

参数 <Limit>[3] 中设置的安全间距仅在下列条件下有效：

<Limit>[3] > (MD10619 $MN_COLLISION_TOLERANCE)

如果参数 <TestLim> 中的位 4 置位（计算后续运行方向），那么在 <DIST> 中包含的方向矢量仅在该函数返回值十万位 (<Status>) 置位时有才效。如果因进入保护区或坐标转换激活而不能确定该方向，则 <DIST> 中的输入值保留不变。也不会返回错误。

特殊的位移指令 5

5.1 逼近已经过编码处理的位置 (CAC, CIC, CDC, CACP, CACN)

通过下列指令可以将直线轴和回转轴通过位置号码返回到机床数据表中设定的固定轴位置。 这种编程类型称作“返回到编码位置”。

句法

```
CAC(<n>)
CIC(<n>)
CACP(<n>)
CACN(<n>)
```

含义

CAC(<n>):	返回到位置号码 n 的编码位置
CIC(<n>):	编码位置，起自于当前的位置编号，向前 (+n) 或向后 (–n) 返回到 n 位置。
CDC(<n>):	以最短的行程返回到位置编号 n 的编码位置 （仅用于回转轴）
CACP(<n>):	按正方向返回到位置编号 n 的编码位置 （仅用于回转轴）
CACN(<n>):	按负方向返回到位置编号 n 的编码位置 （仅用于回转轴）
<n>:	机床数据表中的位置编号 取值范围： 0, 1, ... (最大表位数 - 1)

示例： 返回到某个定位轴的编码位置

编程代码	注释
N10 FA[B]= 300	; 用于定位轴 B 的进给
N20 POS[B]=CAC(10)	; 返回到位置号码 10 的编码位置
N30 POS[B]=CIC(-4)	; 返回到“当前位置号码” －4 的编码位置

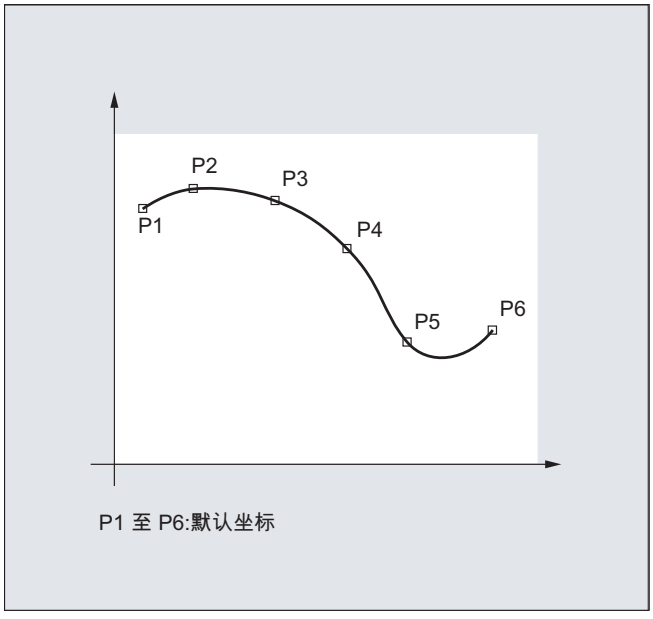
文档

- 功能手册 扩展功能部分；分度轴（T1）
- 功能手册 同步动作

5.2 样条插补 (ASPLINE, BSPLINE, CSPLINE, BAUTO, BNAT, BTAN, EAUTO, ENAT, ETAN, PW, SD, PL)

无法精确分析描述工件上任意曲线轮廓。 因此这种类型的轮廓通过一个限定的支点数近似描述，例如表面数字化。 为了建立工件上的数字化表面，支点必须连接到一个轮廓描述。 这可以是样条插补。

样条定义一个由 2 阶或 3 阶多项式合并的曲线。 可定义样条支点上的特性，**取决于使用的样条类型** 。



SINUMERIK solution line 提供下列样条类型：

- A 样条
- B 样条
- C 样条

句法

通用：

```
ASPLINE X... Y... Z... A... B... C...
BSPLINE X... Y... Z... A... B... C...
CSPLINE X... Y... Z... A... B... C...
```

对于 B 样条可另外编程：

```
PW=<n>
SD=2
PL=<值>
```

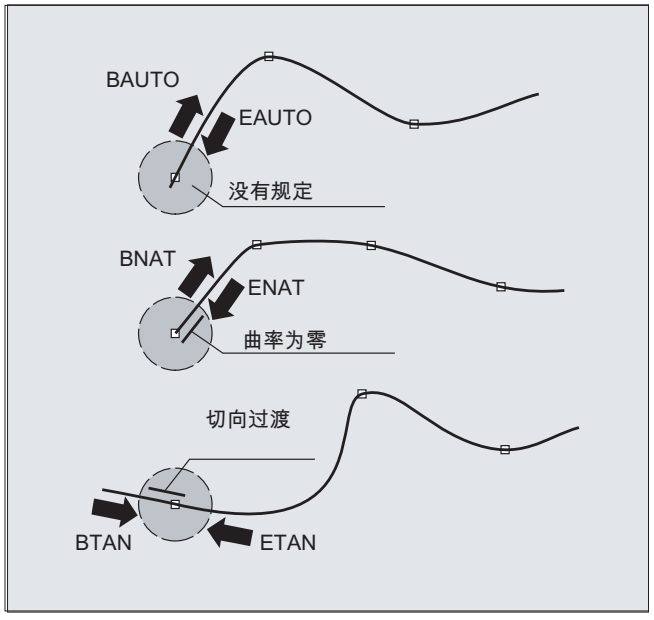
对于 A 和 C 样条可另外编程：

BAUTO / BNAT / BTAN

EAUTO / ENAT / ETAN

含义

样条插补类型：			
ASPLINE:	用于打开 A 样条插补的指令		
BSPLINE:	用于打开 B 样条插补的指令		
CSPLINE:	用于打开 C 样条插补的指令		
	指令 ASPLINE, BSPLINE 和 CSPLINE 是模态有效，并属于位移指令组。		
支点或检查点：			
X... Y... Z... A... B... C...	直角坐标的位置		
点权重（仅 B 样条）：			
PW:	通过指令 PW 可对每个支点进行所谓的“点权重”编程。		
<n>:	“点权重”		
	取值范围：	0 ≤ n ≤ 3	
	步进宽度：	0.0001	
	作用：	n > 1	曲线被检查点更为紧密地拉近。
		n < 1	曲线被检查点略微拉近。
样条阶（仅 B 样条）：			
SD:	正常情况下使用一个 3 级多项式。 通过编程 SD=2 也可以使用一个 2 阶多项式。		
节点间距（仅 B 样条）：			
PL：	节点间距适合于内部计算。 但是，控制系统也可以对规定的节点间距进行处理，在用指令 PL 规定作为所谓的参数间隔长度。		
<值>:	参数－间隔－长度		
	取值范围：	如位移尺寸	

样条曲线开始处的过渡特性（仅 A 或 C 样条）：	
BAUTO:	没有给定过渡特性。 由第一个点的位置开始。
BNAT:	曲率为零
BTAN:	切线过渡到上一段（清除位置）
样条曲线末尾处的过渡特性（仅 A 或 C 样条）：	
EAUTO:	没有给定过渡特性。 从最后一个点的位置结束。
ENAT:	曲率为零
ETAN:	切线过渡到上一段（清除位置）
	

说明

可编程过渡特性不影响 B 样条。 在起始点和终点处 B 样条始终与控制多边形轮廓相切。

边界条件

- 可以使用刀具半径补偿。
- 以投影到平面上的方式来进行碰撞监控。

示例

示例 1：B 样条

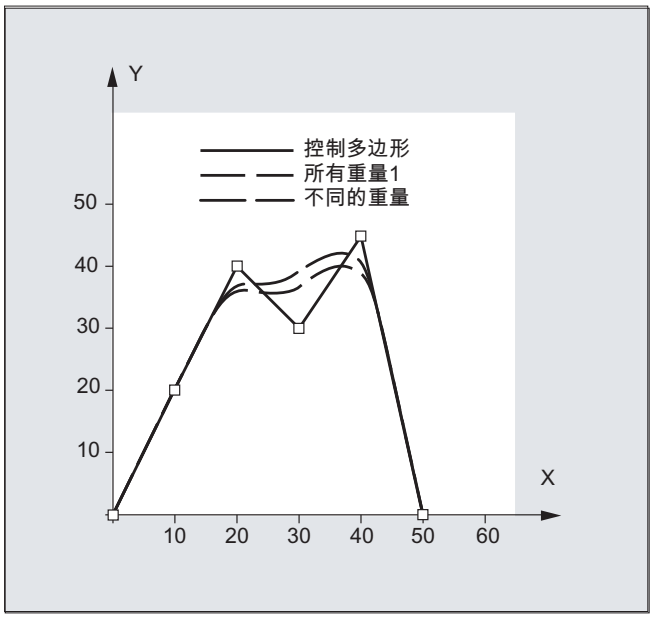
程序代码 1 (全部权重 1)

```
N10 G1 X0 Y0 F300 G64
N20 BSPLINE
N30 X10 Y20
N40 X20 Y40
N50 X30 Y30
N60 X40 Y45
N70 X50 Y0
```

程序代码 2 (不同的权重)

```
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
```

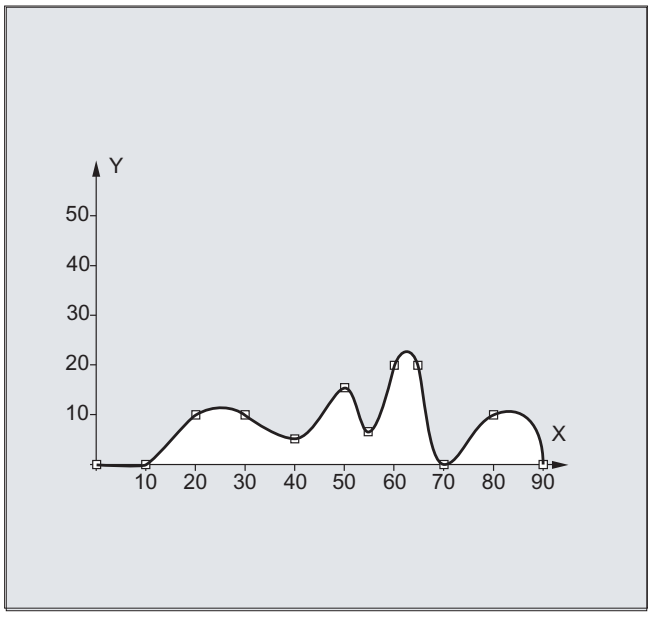
程序代码 3 (检查多项式)	注释
N10 G1 X0 Y0 F300 G64	
N20	；已取消
N30 X10 Y20	
N40 X20 Y40	
N50 X30 Y30	
N60 X40 Y45	
N70 X50 Y0	



示例 2： C 样条举例，在曲面开始和结束处为零

程序代码

```
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
```



示例 3： 样条插补 (A 样条) 和坐标转换 (ROT)

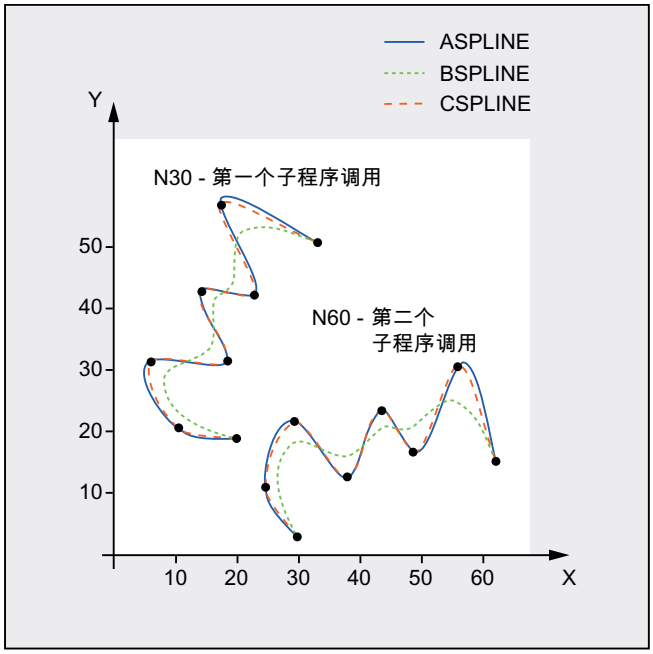
主程序：

程序代码	注释
N10 G00 X20 Y18 F300 G64	; 运行到起点。
N20 ASPLINE	; 激活 A 样条插补类型。
N30 KONTUR	; 子程序首次调用。
N40 ROT Z-45	; 坐标转换： WKS 旋转 -45° 到 Z 轴。
N50 G00 X20 Y18	; 返回轮廓起始点。
N60 KONTUR	; 子程序第二次调用。
N70 M30	; 程序结束

子程序 “轮廓” (包含支点坐标)：

程序代码
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

在下列图形中除了由程序示例得出的样条曲线（ASPLINE），也包含激活 B 或 C 样条插补时得出的样条曲线（BSPLINE, CSPLINE）：



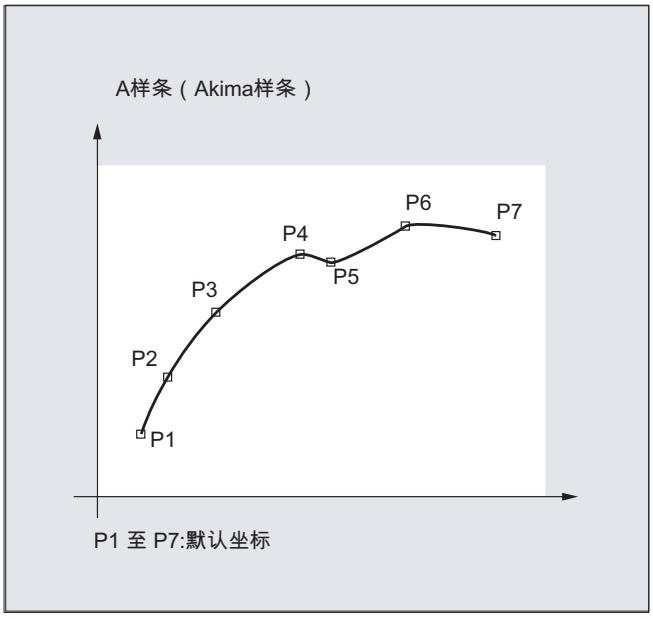
其它信息

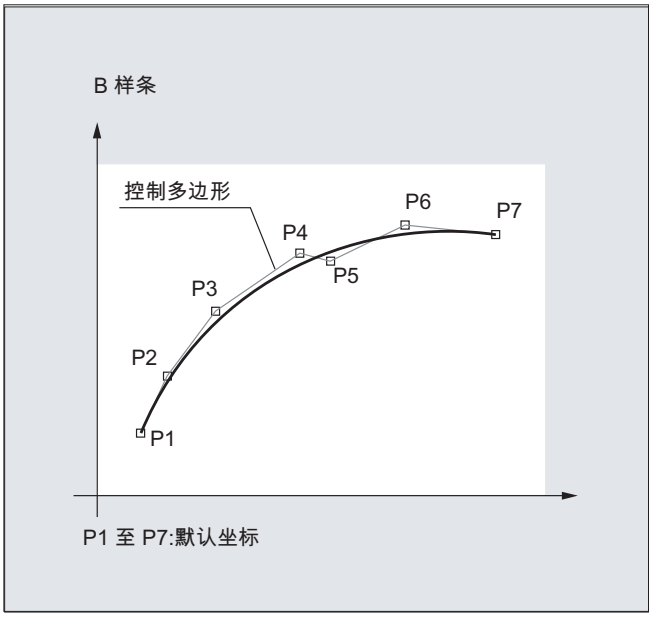
样条插补优点

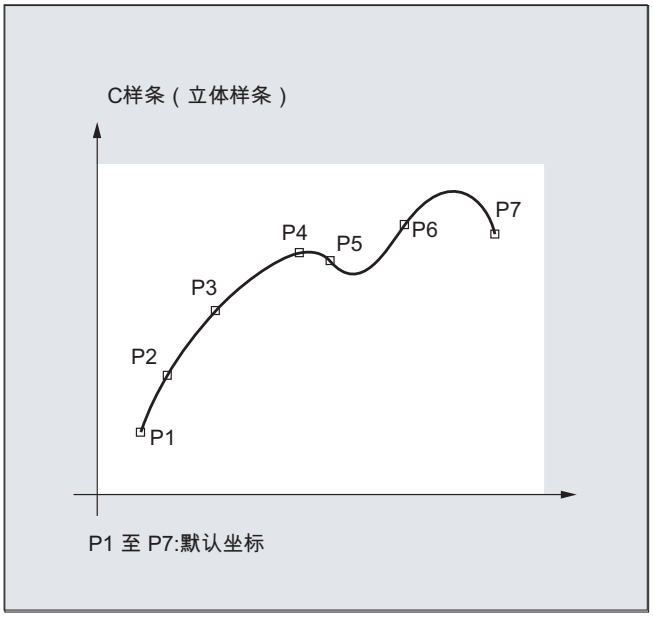
通过使用样条插补，与使用阶数组 G01 相比，有下列优点：

- 减少了用于描述轮廓所需零件程序段的数目
- 在零件程序段之间过渡时机械特性的曲线走向更为平滑

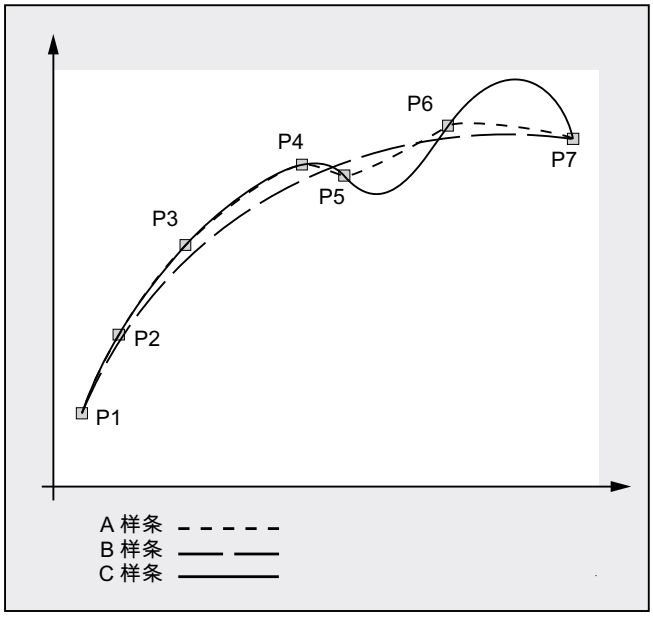
不同样条类型的特性和用途

样条类型	特性和用途
A 样条	 **特性：** ● 通过规定的支点精确走向。 ● 曲线走向为始终相切，但非曲面。 ● 几乎不产生不期望的摆动。 ● 支点改变影响范围是局部的，即支点的变化仅对 6 个以下相邻的支点起作用。 **应用：** A 样条首先适用于带有较大升幅改变的曲线插补（例如台阶式的曲线）。

样条类型	特性和用途
B 样条	 **特性：** ● 走向不通过规定的支点，而是仅在其附近。曲线通过支点拉近。通过支点权重及一个系数，可以另外影响曲线走向。 ● 曲线走向为始终相切和曲面。 ● 不产生不期望的摆动。 ● 支点改变影响范围是局部的，即支点的变化仅对 6 个以下相邻的支点起作用。 **应用：** B 样条主要是考虑与 CAD 系统的接口。

样条类型	特性和用途
C 样条	 **特性：** ● 通过规定的支点精确走向。 ● 曲线走向为始终相切和曲面。 ● 经常产生不期望的摆动，特别在带有较大升幅改变的位置。 ● 支点改变影响范围是全局的，即一个支点的改变影响整个曲线走向。 **应用：** 当支点位于一个已知的曲线上时（圆，抛物线，双曲线），可以很好地采用 C 样条。

在相同的支点处对比三种类型的样条



样条程序段的最低数量

使用 G 代码 ASPLINE, BSPLINE 和 CSPLINE，使样条与程序段终点相联系。对此必须同时计算一系列的程序段（终点）。用于计算的缓冲器的大小一般情况下为 10 个程序段。不是每条程序段信息都是一个样条终点。然而，控制系统每 10 个程序段中必须有一定数量的样条终点程序段：

样条类型	样条程序段的最低数量
A 样条：	每 10 个程序段中必须至少有 **4** 个样条程序段。 注释程序段和参数计算不在此列。
B 样条：	每 10 个程序段中必须至少有 **6** 个样条程序段。 注释程序段和参数计算不在此列。
C 样条：	所需的样条程序段最低数量由下列求和得出： MD20160 $MC_CUBIC_SPLINE_BLOCKS 的值 + 1 在 MD20160 中设定通过样条段计算出的终点数量。缺省设置为 8。在标准情况下，每 10 个程序段中必须有 **9** 个样条程序段。

说明

如果低于允许值的范围，则会产生一个报警，同样当样条插补的轴当作定位轴编程时也会发出一个报警。

合并较短样条程序段

在样条插补中可能会存在较短的样条程序段，它们会降低轨迹速度。通过功能“合并较短样条程序段”可以合并这些程序段，从而产生足够长的程序段并避免降低轨迹速度。

通过通道专用的机床数据：

MD20488 $MC_SPLINE_MODE （样条插补的设置）激活该功能。

文献:

功能手册 基本功能；轨迹控制运行，准停，预读(B1),
章节：合并较短样条程序段

5.3 样条组合(SPLINEPATH)

使用指令 SPLINEPATH 选出样条组合中需要进行插补的轴。 样条插补中最多可以有八个轨迹轴。

说明

如果 SPLINEPATH 没有明确地编程，则通道中开始的三个轴作为样条组合运行。

句法

这需要在一个独立的程序段中确定样条组合：

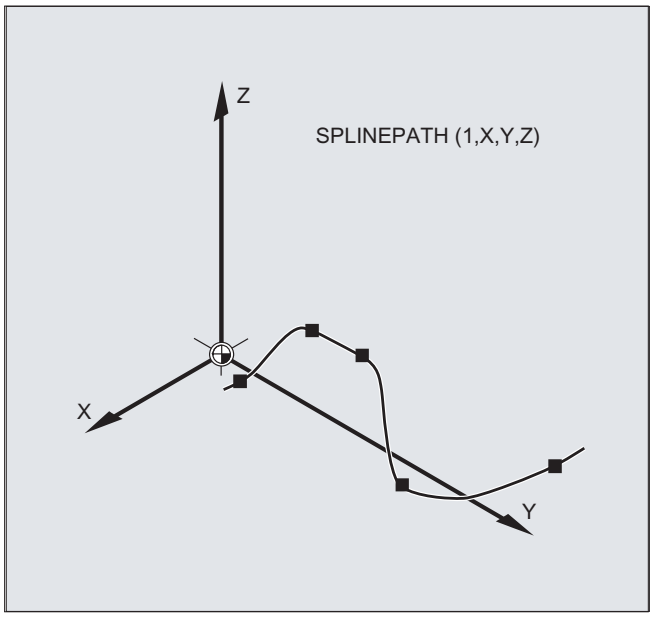
```
SPLINEPATH(n,X,Y,Z,…)
```

含义

SPLINEPATH:	用于确定样条组合的指令
n:	=1 (固定值)
X,Y,Z,… :	样条组合中要插补的轨迹轴名称

示例： 带有三个轨迹轴的样条组合

程序代码	注释
N10 G1 X10 Y20 Z30 A40 B50 F350	
N11 SPLINEPATH(1,X,Y,Z)	;样条组合
N13 CSPLINE BAUTO EAUTO X20 Y30 Z40 A50 B60	;C 样条
N14 X30 Y40 Z50 A60 B70	;节点
…	
N100 G1 X… Y…	;取消样条补


Z
SPLINEPATH (1,X,Y,Z)
X
Y

5.4 激活/关闭 NC 程序段压缩 (COMPON, COMPCURV, COMPCAD, COMPSURF, COMPOF)

直线程序段的压缩功能（取决于参数设置也包括圆弧和/或快速运行程序段）可使用 G 指令组 30 的 G 指令激活/关闭。指令模态有效。

句法

```
COMPON / COMPCURV / COMPCAD / COMPSURF
...
COMPOF
```

含义

COMPON:	激活压缩器功能 COMPON
COMPCURV:	激活压缩器功能 COMPCURV
COMPCAD:	激活压缩器功能 COMPCAD
COMPSURF:	激活压缩器功能 COMPSURF
COMPOF:	关闭当前激活的压缩器功能

说明

除此以外，改善表面质量还可以使用平滑功能 G642 和急动度限制 SOFT。这些指令应写在程序开始处。

示例：COMPCAD

程序代码	注释
N10 G00 X30 Y6 Z40	
N20 G1 F10000 G642	; 激活：平滑功能 G642
N30 SOFT	; 激活：急动度限制 SOFT
N40 **COMPCAD**	; 激活：压缩器功能 COMPCAD
N50 STOPFIFO	
N24050 Z32.499	; 第 1 运行程序段
N24051 X41.365 Z32.500	; 第 2 个运行程序段
...	
N99999 X...Z...	; 上一个运行程序段
COMPOF	; 关闭压缩器功能。
...	

5.5 多项式插补 (POLY, POLYPATH)

就本义来说，多项式插补（POLY）并不是一种样条插补。 首先它是用作编程外部生成的样条曲线的接口。 在此，样条区段可以直接编程。

使用此插补方式时，NC 不需计算多项式系数。 当系数直接来源于 CAD 系统或者后置处理程序时，此方式非常理想。

句法

3 阶多项式：

```
POLY PO[X]=(xe,a2,a3) PO[Y]=(ye,b2,b3) PO[Z]=(ze,c2,c3) PL=n
```

5 阶多项式和新多项式句法：

```
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")
```

说明

在一个 NC 程序段中编程的多项式系数和轴的总和不能超出每个程序段允许的最大轴数量。

含义

POLY：	程序段中写入 POLY，启用多项式插补。
POLYPATH：	可为两个轴组 AXIS 或 VECT 选择多项式插补
PO[轴名称/变量]：	终点和多项式系数
X, Y, Z：	轴名称
xe, ye, ze ：	各个轴的终点位置说明；取值范围如同位移尺寸

a2, a3, a4, a5：	系数 $_2$, a_3, a_4, 和 a_5 使用其值写入；取值范围如同位移尺寸。 如果最后的系数值为零，则可以略去该值。
PL：	定义多项式的参数间隔长度（功能 f(p) 的定义范围）。 间隔从 0 开始，p 可以为 0 到 PL 之间的数值。 PL 的理论值范围： 0.0001 …99 999,9999 **提示：** PL 值用于它所在的程序段。 如果没有编程 PL，则 PL=1。

激活/取消多项式插补

在零件程序中通过 G 指令 POLY 激活多项式插补。

G 指令 POLY 和 G0、G1、G2、G3、ASPLINE、BSPLINE 和 CSPLINE 同属于 G 功能组 1。

仅使用其名称和终点编程的轴（例如 X10）以直线运行。 当一个 NC 程序段的所有轴都这样编程时，控制系统特性与采用 G1 时相同。

编程 G 功能组 1 中另一个指令（例如 G0、G1）会自动取消多项式插补。

多项式系数

PO 值(PO[]=) 或者 ...=PO(...) 用于设定轴的所有多项式系数。 根据多项式的阶数，可设定多个值并以逗号隔开。 在一个程序段中，不同的轴可以有不同的多项式阶数。

子程序 POLYPATH

使用 POLYPATH(...) 可选择性地为特定轴组使能多项式插补：

仅路径轴和辅助轴：	POLYPATH("AXES")
仅方向轴： （在包含方向转换的运行中）	POLYPATH ("VECT")

未使能的轴则以直线运行。

默认情况下会将两个轴组的多项式插补都使能。

编程时不写入参数 POLYPATH( ) 即可取消激活所有轴的多项式插补。

示例

程序代码	注释
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	;启用多项式插补
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	;轴的混合说明
N27 PO[X]=(10,2.5) PO[Y]=(2.3)	;没有编程 PL ; 则 PL=1
N30 G1 X… Y… Z.	;关闭多项式插补
…	

示例： 新多项式句法

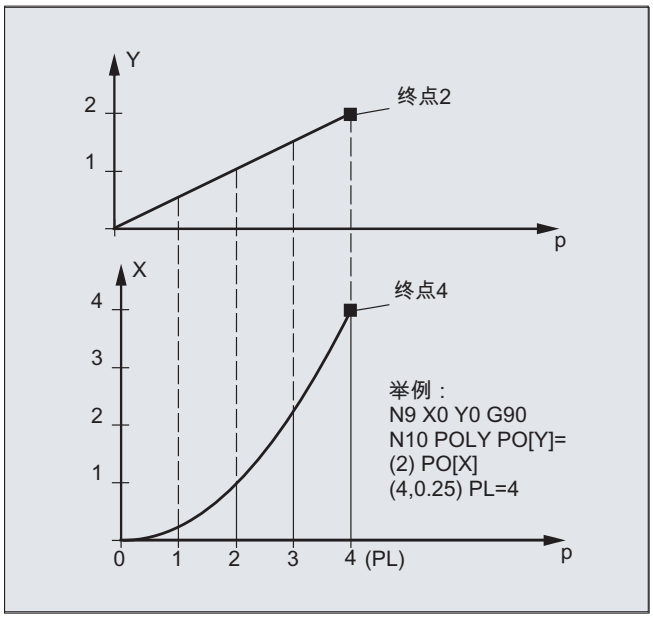
继续生效的多项式句法	新多项式句法
PO[轴名称]=(.. , ..)	轴名称=PO(.. , ..)
PO[PHI]=(.. , ..)	PHI=PO(.. , ..)
PO[PSI]=(.. , ..)	PSI=PO(.. , ..)
PO[THT]=(.. , ..)	THT=PO(.. , ..)
PO[]=(.. , ..)	PO(.. , ..)
PO[变量]=IC(.. , ..)	变量=PO IC(.. , ..)

示例： X/Y 平面中的曲线

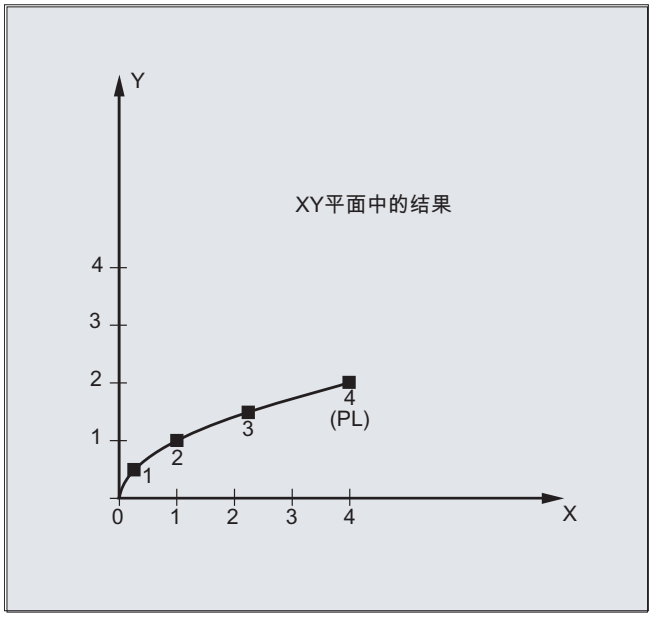
编程

程序代码
N9 X0 Y0 G90 F100
N10 POLY PO[Y]=(2) PO[X]=(4,0.25) PL=4

曲线 X(p) 和 Y(p)


Y
2
1
终点2
p
X
4
3
2
1
终点4
举例：
N9 X0 Y0 G90
N10 POLY PO[Y]=
(2) PO[X]
(4,0.25) PL=4
0
1
2
3
4 (PL)
p

XY 平面中的曲线


Y
XY平面中的结果
4
3
2
1
4
(PL)
3
2
1
0
1
2
3
4
X

说明

多项式函数的一般形式为：

$$f(p)= a_0 + a_1p + a_2p^2 + . . . + a_np^n$$

其中：
a_i:恒定系数 (i = 0, 1, ..., n)
p:参数

在控制系统中最多可编程 5 阶的多项式：

$$f(p)= a_0 + a_1p + a_2p^2 + a_3p^3 + a_4p^4 + a_5p^5$$

通过给系数设定具体的数值，可以产生各种曲线，如直线、抛物线和幂函数。

通过 $a_2 = a_3 = a_4 = a_5 = 0$ 生成直线：

$$f(p) = a_0 + a_1p$$

其中：

a_0:前一程序段结束时轴的位置

$p = PL$

$a_1 = (x_E - a_0 - a_2*p^2 - a_3*p^3) / p$

可以在**没有**使用 G 指令 POLY 激活多项式插补的情况下，对多项式进行编程。在这种情况下不会插补编程的多项式，而是以直线（G1）逼近编程的轴终点。只有在零件程序中以确定的方式激活多项式插补（POLY）后，才对编程的多项式进行插补。

特殊情况：除数多项式

对于几何轴，可使用 PO[]=(…) 不设定轴名称来编程一个共同的除数多项式，也就是说，将几何轴的运动作为两个多项式的商进行插补。

这样可精确描述例如圆锥面（圆，椭圆，抛物线，双曲线）等表面。

示例：

程序代码	注释
POLY G90 X10 Y0 F100	；几何轴直线运行到位置 X10, Y0。
PO[X]=(0,-10) PO[Y]=(10) PO[]=(2,1)	；几何轴沿四分圆运行到 X0 Y10。

除数多项式的固定系数（a_0）总是取 1。编程的终点与 G90 或者 G91 无关。

X(p) 和 Y(p) 通过编程的值计算得出：

$X(p) = (10 - 10 * p^2) / (1 + p^2)$

$Y(p) = 20 * p / (1 + p^2)$

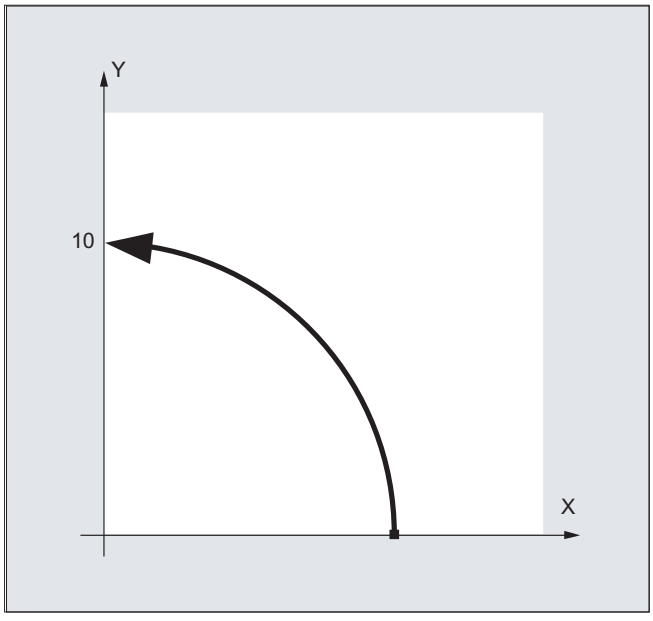
其中 $0 \le p \le 1$

根据已编程的起始点、终点、系数 a_2 和 PL=1，得出下列中间结果：

分子 (X) = $10 + 0 * p - 10 * p^2$

分子 (Y) = $0 + 20 * p + 0 * p^2$

分母＝ $1 + p^2$



多项式插补激活时，区间 [0,PL] 内以零编程的除数多项式会被拒绝，并输出报警。除数多项式不会对辅助轴的运动产生影响。

说明

在多项式插补中可以通过 G41, G42 启用刀具半径补偿，使用方法同直线插补或者圆弧插补。

5.6 可设置的轨迹基准 (SPATH, UPATH)

在多项式插值中（POLY, ASPLINE, BSPLINE, CSPLINE, COMPON, COMPCURV），轨迹轴 i 的位置可通过多项式 $p_i(U)$指定。 此时，曲线参数 U 在一个 NC 程序段之内从 0 到 1。

通过 FGROUP 选择与轨迹进刀 F 有关的轴（FGROUP 轴）。 FGROUP 轴路径 S 上恒定速度的插值表示在多项式插值中曲线参数 U 通常为非恒定变化。对于 FGROUP 中没有包含的轴可以在两个选项之间选择，如同 FGROUP 轴一样：

- 与行程 S (SPATH)同步
- 与曲线参数 U（UPATH）同步

句法

```
SPATH
UPATH
```

含义

SPATH:	FGROUP 中不包含的轴将根据行程 S 移动
UPATH:	FGROUP 中不包含的轴将根据曲线参数 U 移动

说明

UPATH 和 SPATH 也用来确定 F 字多项式 (FPOLY, FCUB, FLIN) 与轨迹运动之间的关系。

边界条件

SPATH 或 UPATH 在以下情况下没有意义：

- 线性插补（G1）
- 圆弧插补 (G2, G3)
- 线程块(G33, G34, G35, G33x, G63)
- 所有轨迹轴都包含在 FGROUP 内

示例

下列示例用来阐明两种运动控制之间的区别。

程序代码

```
N10 FGROUP(X,Y,Z)
```

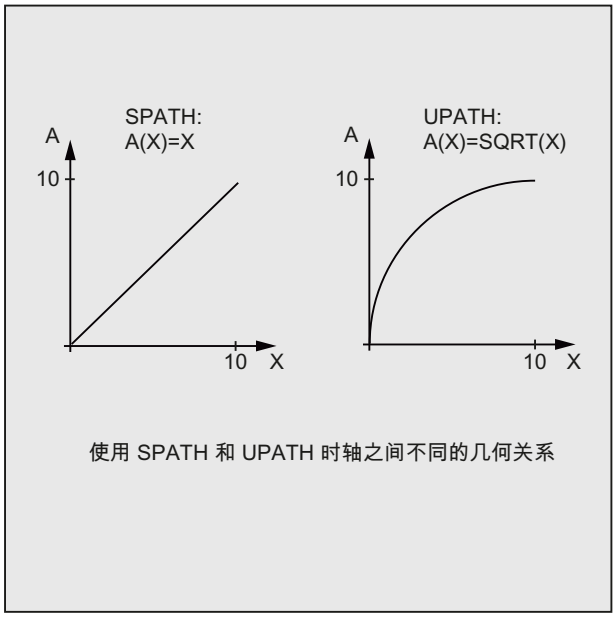
程序代码

```
N15 G1 X0 A0 F1000 SPATH                    ; SPATH
N20 POLY PO[X]=(10,10) A10
```

程序代码

```
N10 FGROUP(X,Y,Z)
N15 G1 X0 A0 F1000 UPATH                    ; UPATH
N20 POLY PO[X]=(10,10) A10
```

在两个程序段中， N20 中 FGROUP 轴的位移 S 与曲线参数 U 的平方有关。 因此，沿着 X 轴的位移得出同步轴的不同位置，视情况而定，是否 SPATH 或者 UPATH 已激活。



使用 SPATH 和 UPATH 时轴之间不同的几何关系

其它信息

复位时的控制特性和机床数据/可选数据

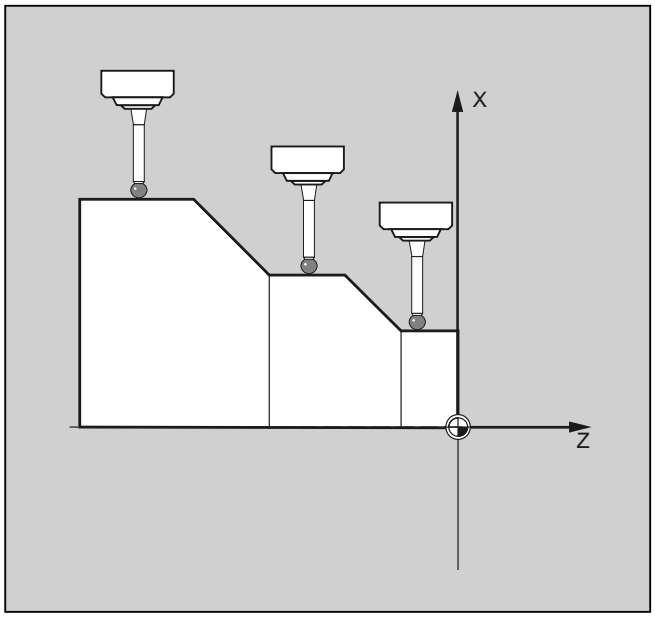
在复位后，通过 MD20150 $MC_GCODE_RESET_VALUES[44] 确定的 G 代码生效（第 45 个 G 代码组）。

平滑方式的缺省值通过 MD20150 $MC_GCODE_RESET_VALUES[9] 定义（第 10 个 G 代码组）。

轴特有的机床数据 MD33100 $MA_COMPRESS_POS_TOL[<n>]有一个引申义： 它含有压缩器功能的公差和用 G642 进行精磨削的公差。

5.7 用接触式探头测量 (MEAS, MEAW)

通过功能“采用触发式探头测量”可以使轴移动到工件上的实际位置；在测量探头发出脉冲沿时，它可以测定所有测量程序段中写入的轴的位置，并将每个轴的位置写入到存储单元中。



在编程该功能，可以使用下列两个固定地址：

- MEAS
  通过 MEAS 可以删除实际位置和给定位置之间的剩余行程。
- MEAW
  而 MEAW 则适用于那些在任何情况下都需要逼近到编程位置的测量任务。

MEAS 和 MEAW 为逐段生效且可以和运行指令一同编程。 进给率、插补类型 (G0, G1, ...) 以及轴的数量须根据相应的测量问题进行调整。

句法

```
MEAS=<TE> G... X... Y... Z...
MEAW=<TE> G... X... Y... Z...
```

含义

MEAS:	指令： 测量，**带**剩余行程删除	
	生效方式：	逐段式

MEAW:	指令： 测量，**不带**剩余行程删除	
	生效方式：	逐段式
<TE>：	触发测量的触发事件	
	类型：	INT
	取值范围：	-2, -1, 1, 2
	含义：	
	(+)1	测量探头 1 （测量输入 1 上）的上升沿
	-1	测量探头 1 （测量输入 1 上）的下降沿
	(+)2	测量探头 2 （测量输入 2 上）的上升沿
	-2	测量探头 2 （测量输入 2 上）的下降沿
	提示： 最多可使用 2 个测量探头，视扩建阶段而定。	
G...：	插补类型，例如 G0, G1, G2 或 G3	
X... Y... Z...:	直角坐标的终点	

示例

程序代码	注释
N10 MEAS=1 G1 F1000 X100 Y730 Z40 ...	；带有第一个测量输入端的测量探头和直线插补的测量程序段。 自动产生预处理停止。

其它信息

测量任务状态

如果需要在程序中分析测量探头是否已工作，可以查询状态变量 $AC_MEA[<n>]（<n> = 测量探头的编号)：

值	含义
0	测量任务未履行
1	测量任务已顺利结束（测量探头已工作）

说明

当探头在程序中偏转时，就将变量置为 1。当某个测量程序段开始执行时，自动将变量设定成探头的初始状态。

记录测量值

采集程序段所有运动过的轨迹轴和定位轴的位置（轴上的最大数量要视控制系统配置而定）。如果编程 MEAS，在测量探头触发之后按照定义使运动停止。

说明

如果在测量程序段中编程几何轴，所有几何轴的测量值将被存储。
如果在某个测量程序段中编程了某个参与转换的轴，就保存所有参与该转换的轴的测量值。

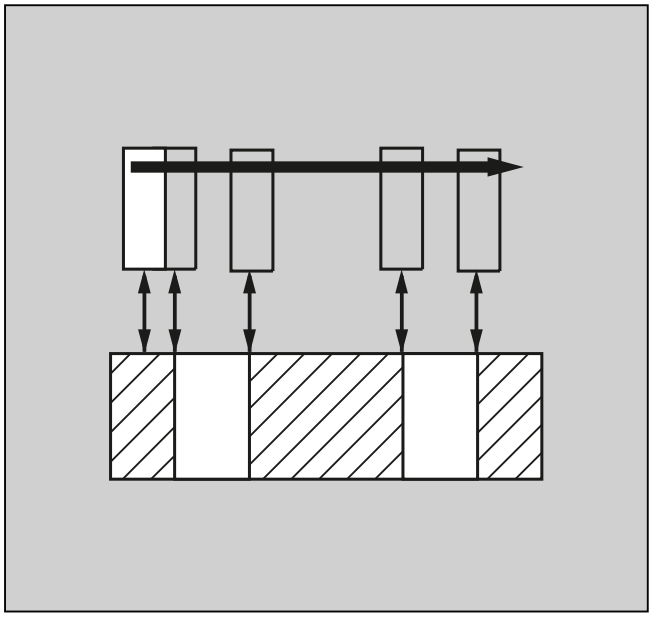
读取测量结果

通过测量探头采集的轴的测量结果可通过以下系统变量读取：

- $AA_MM[<轴>]
  机床坐标系中的测量结果
- $AA_MW[<轴>]
  工件坐标系中的测量结果

5.8 轴测量 (MEASA、MEAWA、MEAC)（选件）

如果是轴向测量，可以使用多个探头和多个测量系统。



编程此功能时所涉及的关键指令包括 MEASA、MEAWE 和 MEAC。

使用 MEASA 或 MEAWA 时，每次测量最多可为相应已编程的轴采集四个测量值，并且针对触发事件将其保存在系统变量中。

还可使用 MEAC 执行连续的测量任务。在这种情况下，将测量结果保存在 FIFO 变量中。

句法

```
MEASA[<轴>]=(<模式>,<TE1>,...,<TE4>)
MEAWA[<轴>]=(<模式>,<TE1>,...,<TE4>)
MEAC[<轴>]=(<模式>,<测量存储器>,<TE1>,...,<TE4>)
```

说明

MEASA 和 MEAWA 为逐段有效且可以在某个程序段中共同编程。但如果 MEASA/MEAWA 与 MEAS/MEAW 编程在一个程序段中，就会发出出错信息。

含义

MEASA：	关键字：轴测量，带剩余行程删除	
	生效方式：	逐段式

MEAWA：	关键字：轴测量，不带剩余行程删除	
	生效方式：	逐段式
MEAC：	关键字：轴连续测量，不带剩余行程删除	
	生效方式：	逐段式
<轴>:	名称，用于测量所使用的通道轴	
<模式>:	指定运行模式（测量模式和测量系统）的两位数	
	十进制个位（测量模式）：	
	0	中断测量任务。
	1	最多 4 个可同时激活的触发事件。
	2	最多 4 个可依次激活的触发事件。
	3	最多 4 个可依次激活的触发事件，不过在启动时没有对触发事件 1 的监控（报警 21700/21703 被抑制）。 **提示：** 该模式不适用于 MEAC。
	十进制十位（测量系统）：	
	0（或者没有说明）	已激活的测量系统
	1	测量系统 1
	2	测量系统 2
	3	两个测量系统
<TE>：	触发测量的触发事件	
	类型：	INT
	取值范围：	-2, -1, 1, 2
	含义：	
	(+)1	测量探头 1 的上升沿
	-1	测量探头 1 的下降沿
	(+)2	测量探头 2 的上升沿
	-2	测量探头 2 的下降沿
<测量存储器>：	FIFO 号（循环存储器）	

示例

示例 1：在模式 1 中进行轴的测量，带剩余行程删除（按时间顺序分析）

a) 有 1 个测量系统

程序代码	注释
...	
N100 MEASA[X]=(1,1,-1) G01 X100 F100	；使用激活的测量系统在模式 1 中测量。等待测量信号，测量探头 1 的上升沿/下降沿在 X = 100 后面的运动行程上。
N110 IF $AC_MEA[1]==FALSE GOTOF ENDE	;检查测量结果。
N120 R10=$AA_MM1[X]	；保存属于第一个已编程触发事件（上升沿）的测量值。
N130 R11=$AA_MM2[X]	；保存属于第二个已编程触发事件（下降沿）的测量值。
N140 ENDE:	

b) 有 2 个测量系统

程序代码	注释
...	
N200 MEASA[X]=(31,1,-1) G01 X100 F100	；使用两个测量系统在模式 1 中测量。等待测量信号，测量探头 1 的上升沿/下降沿在 X = 100 后面的运动行程上。
N210 IF $AC_MEA[1]==FALSE GOTOF ENDE	;检查测量结果。
N220 R10=$AA_MM1[X]	；在出现上升沿时保存测量系统 1 的测量值。
N230 R11=$AA_MM2[X]	；在出现上升沿时保存测量系统 2 的测量值。
N240 R12=$AA_MM3[X]	；在出现下降沿时保存测量系统 1 的测量值。
N250 R13=$AA_MM4[X]	；在出现下降沿时保存测量系统 2 的测量值。
N260 ENDE:	

示例 2：在模式 2 中进行轴的测量，带剩余行程删除（按编程的顺序分析）

程序代码	注释
...	
N100 MEASA[X]=(2,1,-1,2,-2) G01 X100 F100	；使用激活的测量系统在模式 2 中测量。在 X = 100 后面的运动行程上，等候测量信号，顺序为测量探头 1 的上升沿，测量探头 1 的下降沿；测量探头 2 的上升沿，测量探头 2 的下降沿。
N110 IF $AC_MEA[1]==FALSE GOTOF MESSTASTER2	；使用测量探头 1 检查测量结果。
N120 R10=$AA_MM1[X]	；保存属于第一个已编程触发事件（测量探头 1 上升沿）的测量值。
N130 R11=$AA_MM2[X]	；保存属于第二个已编程触发事件（测量探头 1 上升沿）的测量值。
N140 MESSTASTER2:	

程序代码	注释
N150 IF $AC_MEA[2]==FALSE GOTOF ENDE	；使用测量探头 2 检查测量结果。
N160 R12=$AA_MM3[X]	；保存属于第三个已编程触发事件（测量探头 2 上升沿）的测量值。
N170 R13=$AA_MM4[X]	；保存属于第四个已编程触发事件（测量探头 2 上升沿）的测量值。
N180 ENDE:	

示例 3：在模式 1 中进行轴的持续测量（按时间顺序分析）

a) 测量 100 个以下的测量值

程序代码	注释
...	
N110 DEF REAL MESSWERT[100]	
N120 DEF INT 循环=0	
N130 MEAC[X]=(1,1,-1) G01 X1000 F100	；使用激活的测量系统在模式 1 中测量，将测量值保存在 $AC_FIFO1 中，等待测量探头 1 在 X = 1000 之后的运动行程上有下降沿的测量信号。
N135 STOPRE	
N140 MEAC[X]=(0)	；在到达轴位置之后中断测量。
N150 R1=$AC_FIFO1[4]	；将累计测量值的数量保存在参数 R1 中。
N160 FOR 循环=0 TO R1-1	
N170 测量值[循环] = $AC_FIFO1[0]	；从 $AC_FIFO1 中读取并且保存测量值。
N180 ENDFOR	

b) 在 10 个测量值之后使用剩余行程删除来测量

程序代码	注释
...	
N10 WHEN $AC_FIFO1[4]>=10 DO MEAC[x]=(0) DELDTG(x)	；删除剩余行程。
N20 MEAC[x]=(1,1,1,-1) G01 X100 F500	
N30 MEAC[X]=(0)	
N40 R1=$AC_FIFO1[4]	；测量值的数量。
...	

c) 通过 2 个测量探头测量上升沿/下降沿

程序代码	注释
...	
N110 DEF REAL MESSWERT[16]	
N120 DEF INT 循环=0	

程序代码	注释
N130 MEAC[X]=(1,1,-1,2) G01 X100 F100	；使用激活的测量系统在模式 1 中测量，将测量值保存在 $AC_FIFO1 中，等待测量信号（依次为测量探头 1 的下降沿、测量探头 2 的上升沿），沿着向 X=100 的路径运行。
N140 STOPRE	；进给停止
N150 MEAC[X]=(0)	；在到达轴位置之后中断测量。
N160 R1=$AC_FIFO1[4]	；将累计测量值的数量保存在参数 R1 中。
N170 FOR Schleife=0 TO R1-1	
N180 MESSWERT[Schleife]=$AC_FIFO1[0]	；从 $AC_FIFO1 中读取并且保存测量值。
N190 ENDFOR	

其它信息

测量任务

可以在零件程序中或者从某个同步动作中（参见“同步动作 (页 655)”一章）编程一个测量任务。某一时刻和同一时刻每个轴只能有一个测量任务激活。

说明

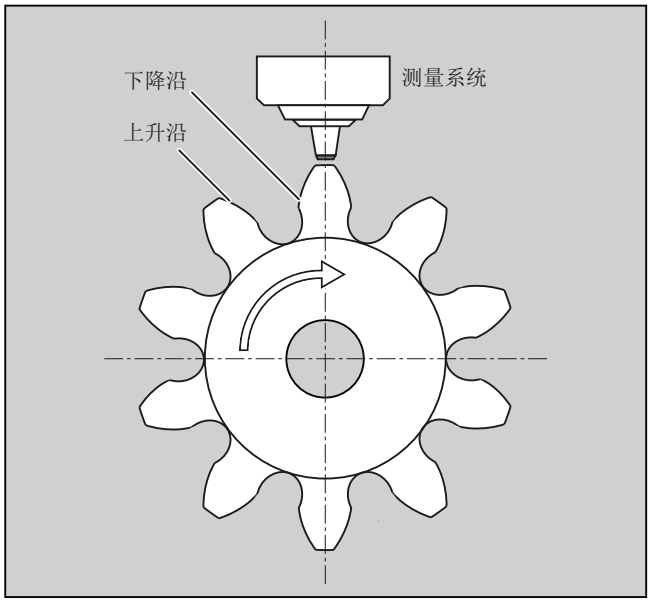
进给应和相应的测量问题适配。

如果是 MEASA 和 MEAWA ，那么只有当进给不再作为一个相同的触发事件、且不再作为每个位置控制器周期的 4 个不同的触发事件出现时，才能保证结果正确。

如果是带有 MEAC 的连续测量，那么插补周期和位置控制器周期之间的比例不得大于 8:1。

触发事件

一个触发事件由探头的编号和测量信号的触发条件（上升或者下降沿）组成。



每次测量时，可以分别处理 4 个以下的已响应探头的触发事件，即最多两个探头，每个探头分别有两个测量脉冲沿。处理的顺序和触发事件的最大数量与所选的模式有关。

说明

针对测量模式 1：在同一个测量任务中只能编程一个触发事件！

使用 MEAC 时，可选择 PROFIBUS 报文 395 将该值提高到每个测头、每个脉冲沿、每个位置控制周期 8 个测量值（上升沿和下降沿各 8 个）。

- 一个探头：上升沿和下降沿各 8 个
- 两个探头：每个探头的上升沿和下降沿各 4 个

相比使用 PROFIBUS 报文 395，此时可实现更高的进给率和转速。

文档：

功能手册之扩展功能；测量 (M5)，章节：轴测量

运行模式

使用运行模式的第一个数字（十位数）来选择所需的测量系统。如果只有一个测量系统存在，但是仍然编程了第二个测量系统，就自动使用存在的测量系统。

使用第二个数字（个位数）可以选择所需的测量模式。以此可以根据相应的控制方法调节测量过程：

- **模式 1:**
  按照触发事件出现的时间顺序对其进行分析。在这种模态中使用六轴模块时仅可编程一个触发事件，或者在参数说明多个触发事件时自动切换到第 2 个模态（没有提示信息）。
- **模式 2:**
  按照编程顺序对触发事件进行分析。
- **模式 3:**
  按照编程顺序对触发事件进行分析，但是不在 START 时监控触发事件 1。

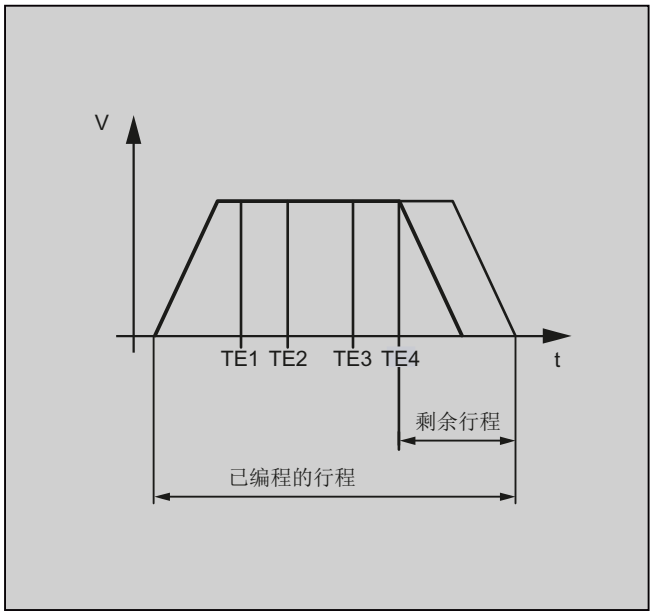
说明

如果使用 2 个测量系统，就只可编程两个触发事件。

有和没有剩余行程删除的测量

在编程 MEASA 时，只有在采集所有所要的测量值之后才会执行剩余行程删除。

对于在任何情况下均要逼近已编程位置的特殊测量任务而言，使用 MEAWA 。



说明

MEASA 不可以在同步动作中编程。取而代之的是可以将 MEAWA 加上剩余行程删除作为同步动作编程。

当使用 MEAWA 从同步动作中开始测量任务时，仅机床坐标系中的测量值可供使用。

MEASA, MEAWA 的测量结果

测量结果可在下列变量项下使用：

- 在机床坐标系中：

$AA_MM1[<轴>]	当出现触发事件 1 时已编程测量系统的测量值
...	...
$AA_MM4[<轴>]	当出现触发事件 4 时已编程测量系统的测量值

- 在工件坐标系中：

$AA_MW1[<轴>]	当出现触发事件 1 时已编程测量系统的测量值
...	...
$AA_MW4[<轴>]	当出现触发事件 4 时已编程测量系统的测量值

几何轴/坐标转换

如果要开始对某个几何轴进行轴向测量，就必须对所有剩余几何轴的相同测量任务进行显式编程。这同样适用于进行转换的轴。

示例：

```
N10 MEASA[Z]=(1,1) MEASA[Y]=(1,1) MEASA[X]=(1,1) G0 Z100
```

或者

```
N10 MEASA[Z]=(1,1) POS[Z]=100
```

有 2 个测量系统的测量任务

当使用两个测量系统执行某个测量任务时，相应轴的两个测量系统可能有两个触发事件，应采集其中可能有的每个事件。预定变量的配置规定为：

$AA_MM1[<轴>]	或者	$AA_MW1[<轴>]	当出现触发事件 1 时测量系统 1 的测量值
$AA_MM2[<轴>]	或者	$AA_MW2[<轴>]	当出现触发事件 1 时测量系统 2 的测量值

$AA_MM3[<轴>]	或者	$AA_MW3[<轴>]	当出现触发事件 2 时测量系统 1 的测量值
$AA_MM4[<轴>]	或者	$AA_MW4[<轴>]	当出现触发事件 2 时测量系统 2 的测量值

系统变量

测量探头状态包含在以下系统变量中：

$A_PROBE[<n>]

值	含义
1	测量探头偏转
0	测量探头未偏转

以下系统变量中提供测量探头限制功能：

$A_PROBE_LIMITED[<n>]

值	含义
1	测量探头限制生效
0	测量探头限制取消

<n> = 测量探头

文档：
参数手册之系统变量

当 MEASA, MEAWA 时的探头状态

如果有必要在程序中进行分析，可以通过 $AC_MEA[<n>]（n = 探头的编号）来查询测量任务状态。只要在某个程序段中已编程的探头的所有触发事件已经出现，该变量就会给出值 1。其它情况下值为 0。

说明

当从同步动作中开始测量时，就不再更新 $AC_MEA 。在这种情况下，应查询新的 PLC 接口信号 DB31, … DBX62.3 或者等效变量 $AA_MEAACT[<轴>]。

含义：

$AA_MEAACT==1:测量有效

$AA_MEAACT==0:测量未激活

连续测量(MEAC)

测量值在执行 MEAC 时存在于机床坐标系中并且被保存在指定的 FIFO[n]存储器中（循环存储器）。如果设计了两个探头用来进行测量，就会将第二个探头的测量值单独保存在额外为此而设计的 FIFO[n+1]-存储器中（可通过 MD 设置）。

FIFO 是一种循环存储器，按照循环原理将 $AC_FIFO 变量中的测量值记录在该存储器中，参见“同步动作 (页 655)”一章。

说明

FIFO 内容仅能从循环存储器中读出一次。如果要多次使用测量数据，就必须将其临时保存在用户数据中。

当测量值的数量超过机床数据中为 FIFO-存储器规定的最大数时，就会自动结束测量。

可通过循环读取测量值的方式来实现连续测量。此时必须至少以和新测量值的输入频率相同的频率来进行读取。

文档：

- 功能手册之同步动作；详细说明，
  章节：参数 ($AC_FIFO)
- 功能手册之扩展功能；测量 (M5)，章节：轴测量

防止错误编程

识别出下面的出错编程，并且显示一个出错：

- MEASA/MEAWA 与 MEAS/MEAW 被编写在一个程序段中
  示例：
  ```
  N01 MEAS=1 MEASA[X]=(1,1) G01 F100 POS[X]=100
  ```
- MEASA/MEAWA 参数个数 <2 或者 >5
  示例：
  ```
  N01 MEAWA[X]=(1) G01 F100 POS[X]=100
  ```
- MEASA/MEAWA 触发事件不等于 1/ -1/ 2/ -2
  示例：
  ```
  N01 MEASA[B]=(1,1,3) B100
  ```
- MEASA/MEAWA 模式错误
  示例：
  ```
  N01 MEAWA[B]=(4,1) B100
  ```
- MEASA/MEAWA 重复编写了触发事件
  示例：
  ```
  N01 MEASA[B]=(1,1,-1,2,-1) B100
  ```

- MEASA/MEAWA 缺少几何轴
  示例：
  ```
  N01 MEASA[X]=(1,1) MEASA[Y]=(1,1) G01 X50 Y50 Z50 F100 ;几何轴 X/Y/Z
  ```
- 几何轴中测量任务不一致
  示例：
  ```
  N01 MEASA[X]=(1,1) MEASA[Y]=(1,1) MEASA[Z]=(1,1,2) G01 X50 Y50 Z50 F100
  ```

5.9 OEM 专用函数(OMA1 ... OMA5, OEMIPO1, OEMIPO2, G810 ... G829)

OEM 地址

OEM 地址的含义由 OEM 定义，该功能通过编译循环实现。为此，预留了 5 个 OEM 地址 (OMA1 ... OMA5)。地址名称可以设定。每个程序段中都允许加入 OEM 地址。

预留的 G 代码调用

以下 G 代码调用预留给 OEM 用户：

- OEMIPO1, OEMIPO2 (来自 G 代码组 1)
- G810 ... G819 (G 代码组 31)
- G820 ... G829 (G 代码组 32)

该功能通过编译循环实现。

函数和子程序

此外，OEM 用户也可以创建预定义的函数和包含参数传递的子程序。

说明

工件仿真

在 SW 4.4 版本之前，工件模拟是不支持编译循环的，SW 4.4 版本之后也只支持个别的编译循环(CC)。

因此，在不被支持的编译循环中，如果不单独处理零件程序中的语言指令（OMA1 ... OMA5, OEMIPO1/2, G810 ... G829，自定义的程序和函数），会输出报警，并中断仿真。

解决方案：单独处理零件程序中不被支持的 CC 专用语言指令（$P_SIM 查询)。
示例：

```
N1 G01 X200 F500
IF (1==$P_SIM)
N5 X300 ，在仿真时 CC 未激活
ELSE
N5 X300 OMA1=10
ENDIF
```

5.10 带有角部减速的进给减速 (FENDNORM, G62, G621)

在自动拐角延迟时，在距离拐角很近处以钟形曲线降低进给速度。 除此之外，关系到加工的刀具性能的范围可以通过设定数据进行参数设定。 它们是：

- 开始和结束进给速度降低
- 用来减小进给速度的修调率
- 识别相关角

有些角部被视为重要的角部，即其内角小于通过调整数据所设定参数的角部。

使用 FENDNORM 缺省值关闭自动拐角倍率的功能。

文档：
/FBFA/ 功能说明 ISO 方言

句法

```
FENDNORM
G62 G41
G621
```

含义

FENDNORM:	自动拐角延迟关
G62：	激活刀具半径补偿时的内拐角减速
G621：	激活刀具半径补偿时在所有角部减速

G62 仅作用于内角，带有

- 有效的刀具半径补偿 G41, G42 和
- 有效的轨迹控制运行 G64, G641

以降低后的进给速度逼近相应的角部，该进给速度来自于：

F * (用于降低进给速度的倍率) * 进给速度倍率

当刀具（以中心点轨迹为基准）在相应角应该变换方向时，表明已经到达了最大可能的进给减速。

G621 与 G62 相似作用于通过 FGROUP 所确定的轴的每个角

5.11 可编程的运动结束条件 (FINEA, COARSEA, IPOENDA, IPOBRKA, ADISPOSA)

与轨迹插补 (G601, G602 和 G603) 的程序段转换条件相似，指令轴或 PLC 轴单轴插补的运动结束条件可以在一个零件程序或者同步动作中编程。

取决于编程的运动结束条件，在单轴运动中，零件程序段或工艺循环段以不同的方式迅速结束。 PLC 也是同样如此，通过 FC15/16/18。

句法

```
FINEA[<轴>]
COARSEA[<轴>]
IPOENDA[<轴>]
IPOBRKA(<轴>[,<时间>])
ADISPOSA(<轴>[,<模式>,<窗口大小>])
```

含义

FINEA:	运动结束条件：“精准停”	
	生效方式：	模态
COARSEA:	运动结束条件：“粗准停”	
	生效方式：	模态
IPOENDA:	运动结束条件：“插补器停止”	
	生效方式：	模态
IPOBRKA:	程序段转换条件： 制动斜坡	
	生效方式：	模态
ADISPOSA:	运动结束条件公差窗口	
	生效方式：	模态
<轴>:	通道轴名称 (X, Y, ...)	
<时间>:	程序段转换时间，是制动斜坡的 % 值： • 100% = 制动斜坡开始 • 0% = 制动斜坡结束，和 IPOENDA 含义相同	
	类型：	REAL

<模式>:	公差窗口的基准		
	取值范围：	0	公差窗口无效
		1	公差窗口与给定位置相关
		2	公差窗口与实际位置相关
	类型：	INT	
<窗口大小>:	公差窗口大小		
	类型：	REAL	

示例

示例 1： 运动结束条件：“插补器停止”

程序代码

```
; 定位轴 X 运行到 100，速度 200 m/min，加速度 90%，
; 运动结束条件： 插补器停止
N110 G01 POS[X]=100 FA[X]=200 ACC[X]=90 IPOENDA[X]

; 同步动作：
; 任何时候只要： 设置输入 1
; 然后定位轴 X 运行到 50，速度 200 m/min，加速度 140%，
; 运动结束条件： 插补器停止
N120 EVERY $A_IN[1] DO POS[X]=50 FA[X]=200 ACC[X]=140
IPOENDA[X]
```

示例 2： 程序段转换条件：“制动斜坡”

程序代码	注释
	; 缺省设定生效
N40 POS[X]=100	; X 轴定位到 100
	程序段转换条件： 精准停
N20 IPOBRKA(X,100)	; 程序段转换条件： “制动斜坡”，
	100% = 制动斜坡开始
N30 POS[X]=200	; 一旦 X 轴开始制动，就立即转换程序段
N40 POS[X]=250	; X 轴并未停止在 200 位置，而是继续运行到位置 250。
	一旦轴开始制动，就立即转换程序段。
N50 POS[X]=0	; X 轴停止，然后返回到位置 0。
	达到位置 0 和“精准停”后，转换程序段
N60 X10 F100	; X 轴作为轨迹轴运动到位置 10

其它信息

运动结束条件的系统变量

可以通过系统变量 $AA_MOTEND 读出当前生效的运动结束条件。

文献： /LIS2sl/ 参数手册，第 2 册

程序段转换条件："制动斜坡"（IPOBRKA）

如果激活了程序段转换条件"制动斜坡"，并为此指定了一个程序段转换时间（该时间是可选值），该值会在下一次定位运动中生效，并和主处理同步地写入设定数据中。 没有指定该时间时，设定数据中的当前值生效。

SD43600 $SA_IPOBRAKE_BLOCK_EXCHANGE

重新设定轴的运动结束条件 (FINEA, COARSEA , IPOENDA)时，对应轴的 IPOBRKA 被禁用。

附加程序段转换条件："公差窗口"（ADISPOSA）

通过 ADISPOSA 可以设定程序段终点的公差窗口，作为附加程序段转换条件，程序段终点可以是实际位置或目标位置。 现在，有两个程序段转换条件：

- 程序段转换条件："制动斜坡"
- 程序段转换条件："公差窗口"

文献

有关定位轴程序段转换条件的其它信息参见：

- 功能手册 扩展功能；定位轴（P2）
- 编程手册 基本原理，章节"进给控制"

坐标转换（框架） 6

6.1 通过框架变量转换坐标

除了在编程手册基本原理章节“坐标转换（框架）”中介绍的指令以外，例如 ROT, AROT, SCALE 等，工件坐标（WKS）也可通过框架变量$P_...FR（数据管理框架）和$P_...FRAME（生效框架）被转换。

下图为框架变量结构的概述：

- 数据管理框架
- 生效框架
- 激活的总框架： 所有激活框架的连接
- NCU 全局框架
- 通道专用框架

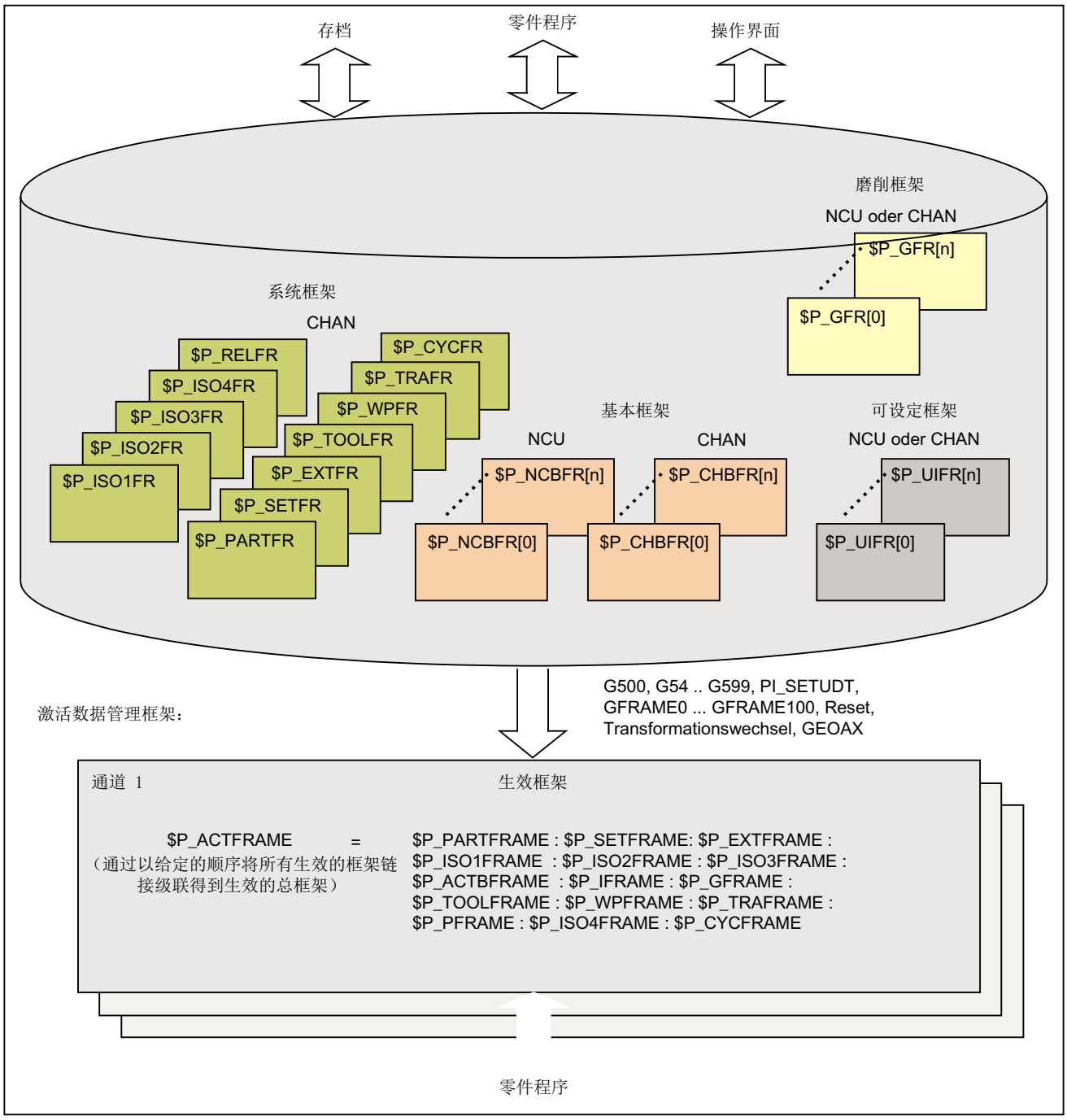

存档
零件程序
操作界面
磨削框架
NCU oder CHAN
$P_GFR[n]
$P_GFR[0]
系统框架
CHAN
$P_RELFR
$P_ISO4FR
$P_ISO3FR
$P_ISO2FR
$P_ISO1FR
$P_CYCFR
$P_TRAFR
$P_WPFR
$P_TOOLFR
$P_EXTFR
$P_SETFR
$P_PARTFR
基本框架
NCU
CHAN
$P_NCBFR[n]
$P_CHBFR[n]
$P_NCBFR[0]
$P_CHBFR[0]
可设定框架
NCU oder CHAN
$P_UIFR[n]
$P_UIFR[0]
激活数据管理框架：
G500, G54 .. G599, PI_SETUDT,
GFRAME0 ... GFRAME100, Reset,
Transformationswechsel, GEOAX
通道 1
生效框架
$P_ACTFRAME =
（通过以给定的顺序将所有生效的框架链接级联得到生效的总框架）
$P_PARTFRAME : $P_SETFRAME: $P_EXTFRAME :
$P_ISO1FRAME : $P_ISO2FRAME : $P_ISO3FRAME :
$P_ACTBFRAME : $P_IFRAME : $P_GFRAME :
$P_TOOLFRAME : $P_WPFRAME : $P_TRAFRAME :
$P_PFRAME : $P_ISO4FRAME : $P_CYCFRAME
零件程序

图 6-1　框架变量概述

6.1.1 预定义框架变量 ($P_CHBFRAME, $P_IFRAME, $P_PFRAME, $P_ACTFRAME)

激活： 通道专用的基本框架$P_CHBFRAME[<n>] ($P_BFRAME)

说明

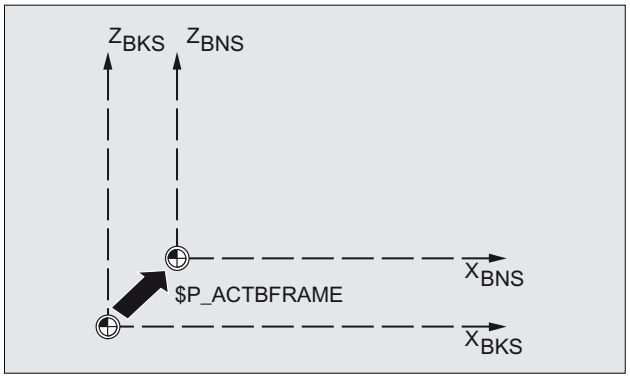
由于兼容性原因保留当前的基本框架$P_BFRAME 和数据管理基本框架$P_UBFR。

- $P_BFRAME □ $P_CHBFRAME[0]
- $P_UBFR □ $P_CHBFR[0].

框架变量$P_CHBFRAME[<n>]定义基准坐标系(BKS)和基准零点坐标系(BNS)之间的关系。

如果当前的通道专用基本框架$P_CHBFRAME[<n>] 在 NC 程序立即生效，则下列选项可用

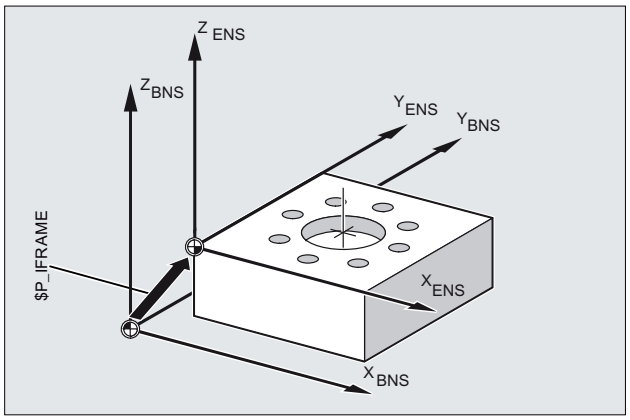
- 指令：
  - G500 （关闭所有可设置框架，基本框架仍然有效）
  - G54 ... G599 (可设置零点位移)
- 数据管理的通道专用基本框架分配到当前的通道专用基本框架：
  $P_CHBFRAME[<n>] = $P_CHBFR[<m>]



激活： 通道专用可设定框架 $P_IFRAME

框架变量$P_IFRAME 定义了基准零点坐标系(BNS)和可设定零点坐标系(ENS)之间的关系。

- $P_IFRAME 相当于$P_UIFR[$P_IFRNUM]
- 例如，在编程了 G54 之后，$P_IFRAME 就会含有通过 G54 所定义的转换、旋转、缩放和镜像。

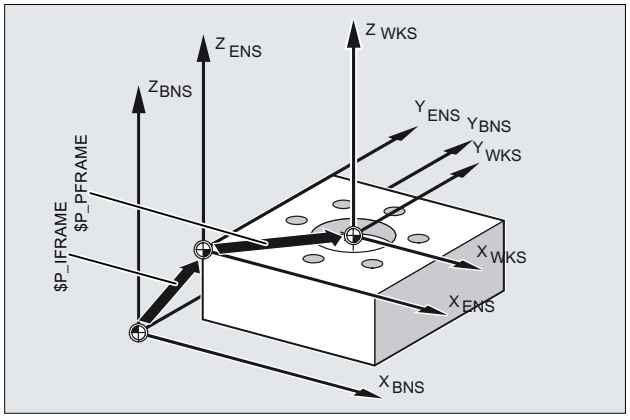


激活： 通道专用可编程框架 $P_PFRAME

框架变量$P_PFRAME 定义了可设定零点坐标系(ENS)和工件坐标系(WCS)之间的关系。

$P_PFRAME 含有

- **从编程** TRANS/ATRANS, ROT/AROT, SCALE/ASCALE, MIRROR/AMIRROR 或者
- **从赋值** CTRANS, CROT, CMIRROR, CSCALE 给可编程的 FRAME 得出的合成框架。



激活： 总框架$P_ACTFRAME

通道内有效的总框架是所有通道内有效框架的连接。

```
$P_ACTFRAME =   $P_PARTFRAME : $P_SETFRAME  : $P_EXTFRAME :
                $P_ISO1FRAME : $P_ISO2FRAME : $P_ISO3FRAME :
                $P_ACTBFRAME :$P_I 框架  : $P_GFRAME :
                $P_TOOLFRAME : $P_WPFRAME   : $P_TRAFRAME :
                $P_PFRAME    : $P_ISO4FRAME : $P_CYCFRAME
```

$P_ACTFRAME 所描述的是当前工件坐标系的零点。

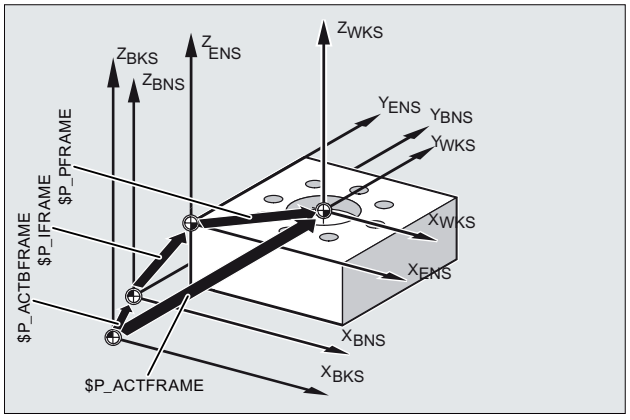
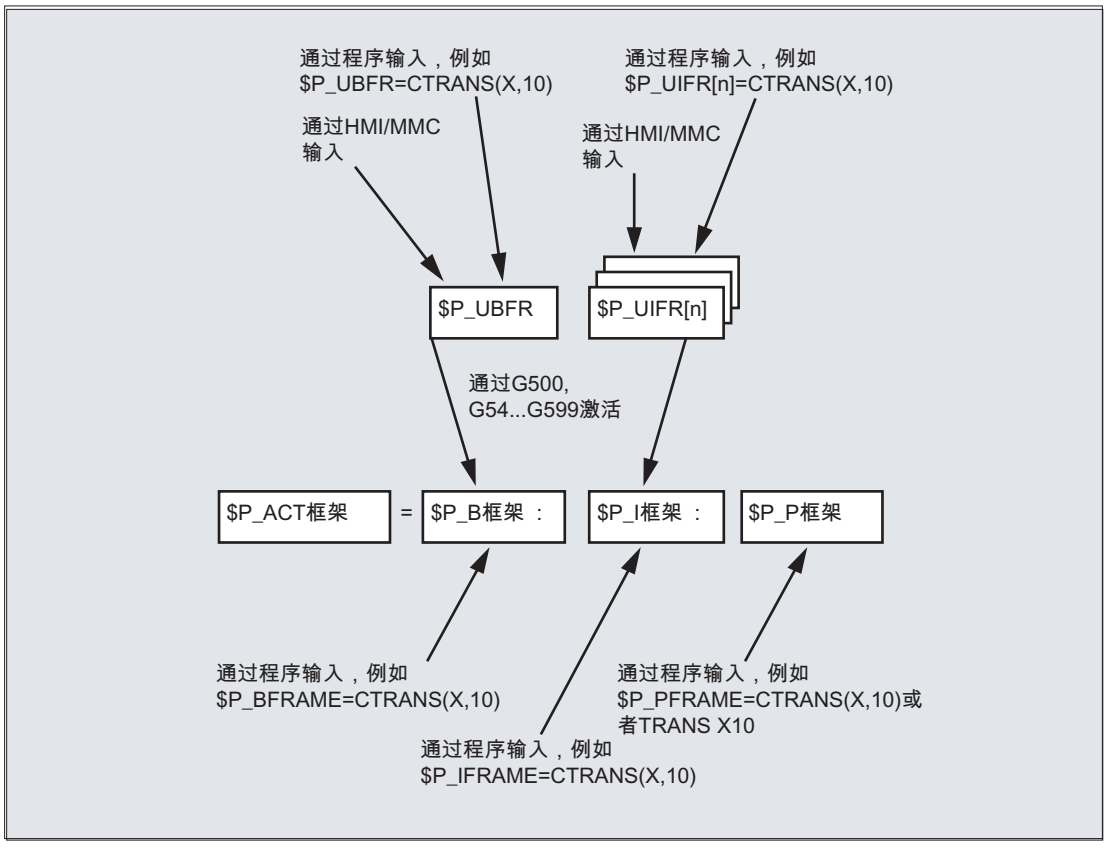


图 6-2　框架变量 $P_ACTFRAME

如果框架$P_BFRAME / $P_CHBFRAME[<n>], $P_IFRAME 或 $P_PFRAME 中的任何一个发生改变，则当前总框架$P_ACTFRAME 将重新计算。



如果 MD20110RESET_MODE_MASK 按照如下方式设定，则复位之后基准框架和可设定框架生效：

位 0=1, 位 14=1 --> `$P_UBFR` (基本框架)有效

位 0=1, 位 5=1 --> `$P_UIFR[$P_UIFRNUM]` (可设置的框架)有效

数据管理： 通道专用基本框架$P_CHBFR[<n>]

通过框架变量$P_CHBFR[<n>]在数据管理中读/写基本框架。 数据管理框架不会通过写入立即在通道中激活。 写入框架激活的步骤如下：

- 通道复位和 MD20110 $MC_RESET_MODE_MASK，位 0==1 位 14==1
- 指令 `G500`, `G54` ... `G57`, `G505` ... `G599`（开启/关闭基本框架，随后重新计算当前总框架）

数据管理： 通道专用可设定框架 $P_UIFR[<n>]

通过框架变量$P_UIFR[<n>]在数据管理中读/写可设定框架。 框架通过写入不会立即在通道激活。 通道内写入框架的计算如下：

- 指令 G500 （关闭所有可设定框架或零点位移）
- 指令 G54 ... G57, G505 ... G599（开启一个可设定框架或零点位移）

被激活的可设定框架	数据管理框架	（符合指令）
$P_IFRAME =	$P_UIFR[0]	G500
	$P_UIFR[1]	G54
	$P_UIFR[2]	G55
	$P_UIFR[3]	G56
	$P_UIFR[4]	G57
	$P_UIFR[5]	G505
	$P_UIFR[6]	G506
	...	...
	$P_UIFR[99]	G599

6.2 给框架赋值

6.2.1 直接赋值（轴值，角度，尺寸）

在 NC 程序中可以直接给框架或者框架变量赋值。

句法

句法

```
$P_PFRAME = CTRANS(X, <位移值>, Y, <位移值>, Z, <位移值>, ...)
$P_PFRAME = ROT(X, <角度>, Y, <角度>, Z, <角度>, ...)
$P_UIFR[..] = CROT(X, <角度>, Y, <角度>, Z, <角度>,  ...
$P_PFRAME = CSCALE(X, <比例>, Y, <比例>, Z, <比例>, ...)
$P_PFRAME = CMIRROR(X, Y, Z)
```

$P_CHBFRAME[<n>]的句法与$P_PFRAME 的句法相同。

含义

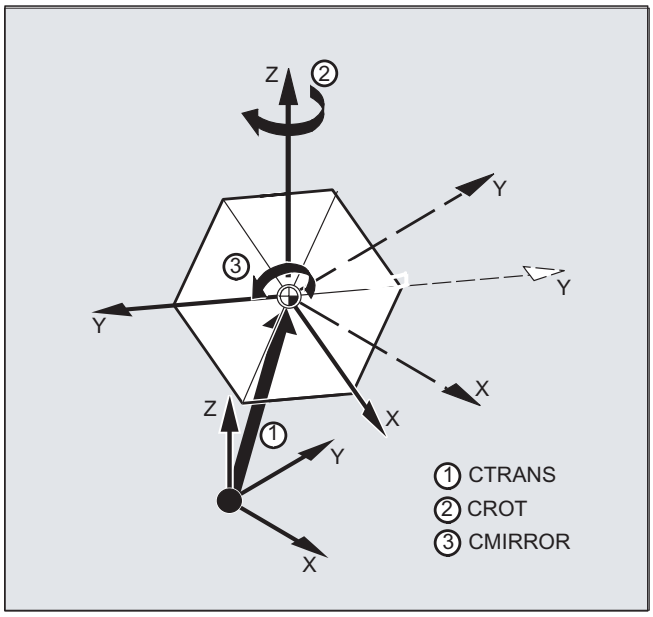
CTRANS：	在给定轴上的偏移
CROT：	围绕给定轴旋转
CSCALE:	在给定轴上的比例改变
CMIRROR:	在给定轴上的反向
X, Y, Z:	在所给定的几何轴方向的偏移值
<位移值>:	位移值
<角度>:	围绕旋转的角度
<比例>:	比例说明

示例

当前可编程框架的框架组件赋值

当前可编程框架的框架组件转换、旋转和镜像的赋值：

```
$P_PFRAME = CTRANS(X,10,Y,20,Z,5) : CROT(Z,45) :CMIRROR(Y)
```



写框架的旋转组件

可设定数据管理框架旋转组件上所有三根轴的赋值，$P_UIFR 带 CROT：

```
$P_UIFR[5]=CROT (X, 0, Y, 0, Z, 0)
```

可选择直接对数据管理框架旋转组件上的各轴进行单独赋值：

```
$P_UIFR[5, Y, RT]=0
$P_UIFR[5, X, RT]=0
$P_UIFR[5, Z, RT]=0
```

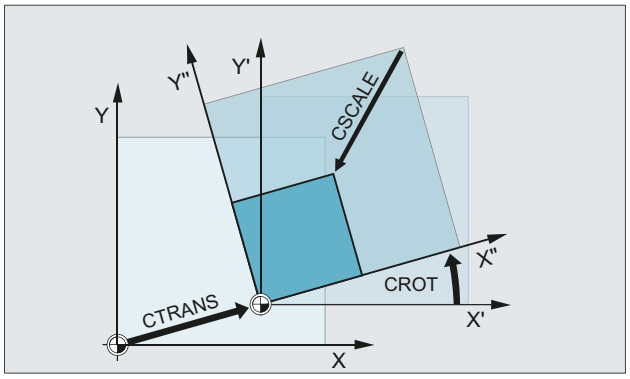
说明

在框架上多个操作可以通过连接符：相互联系起来。 操作按顺序地从左至右进行。

示例

$P_PFRAME 上的链式操作，带位移、旋转和缩放：

```
$P_PFRAME = CTRANS(...) : CROT(...) : CSCALE...
```



6.2.2 读取和修改框架组件 (TR, FI, RT, SC, MI)

可以对某个框架的**各个**数据进行访问，例如某个特定的位移值或者旋转角度。 这些值可以修改，或者赋值给另一个变量。

句法

`R10=$P_UIFR[$P_UIFNUM,X,RT]`	从当前的可设置零点位移 $P_UIFRNUM 得出的围绕 X 轴的旋转角度 RT 应当赋给变量 R10。
`R12=$P_UIFR[25,Z,TR]`	从已设置的编号为 25 的框架的数据集得出的 Z 轴中的位移值 TR 应当赋给变量 R12。
`R15=$P_PFRAME[Y,TR]`	给变量 R15 赋值 Y 轴的偏移值 TR，在当前可编程的框架中。
`$P_PFRAME[X,TR] = 25`	在当前可编程的框架中，改变 X 轴的偏移值 TR。 X25 立即适用。

含义

$P_UIFRNUM:	使用该变量可以自动建立与当前可设定零点偏移坐标系的联系。
P_UIFR[n,…,…]：	通过给出框架号 n，从而使用可设定框架 n。
	对需要读出或者修改的分量的说明：
TR:	TR 转换
FI:	FI 精细转换
RT:	RT 旋转
SC:	SC Scale 改变比例尺

MI:	MI 镜像
X, Y, Z:	此外（参见示例）还指定相应的轴 X，Y，Z。

RT 旋转的数值范围

围绕第 1 个几何轴旋转： -180° ～ +180°
围绕第 2 个几何轴旋转： -90° ～ +90°
围绕第 3 个几何轴旋转： -180° ～ +180°

说明

调用框架

通过指定系统变量 $P_UIFRNUM 可以直接访问使用 $P_UIFR 或者 G54, G55, … 最新设置的零点位移
($P_UIFRNUM 含有最新设置的框架的编号)。

所有其它所保存的可设置框架 $P_UIFR 可通过指定相应的编号 $P_UIFR[n] 来调用。

可以为预定义框架变量和自定义框架指定名称，例如 $P_IFRAME.

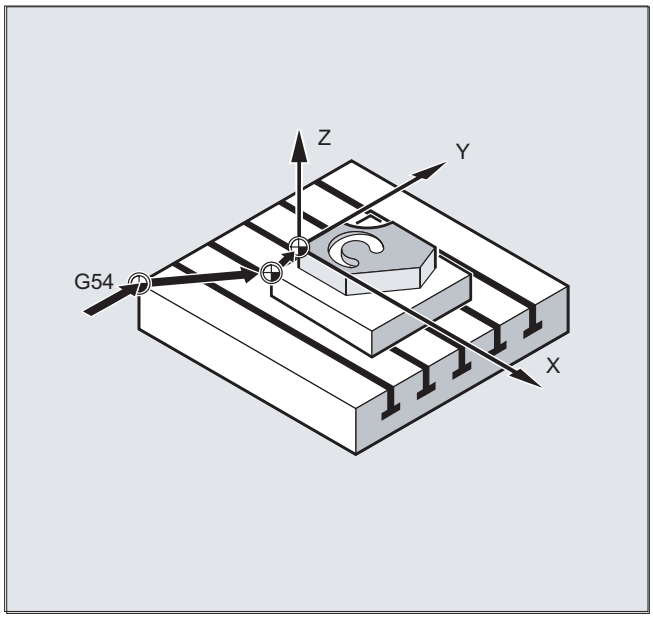
数据调用

在方括号中的是要访问或者修改的轴名称和值的框架组件，例如 [X, RT] 或者 [Z, MI].

6.2.3 通过框架计算

在 NC 程序中，可以将框架赋给另外一个框架或者使框架级联。

例如，框架级联适合用来描述排列在一个托盘上且应在一个加工流程中进行加工的多个工件。



描述托盘任务时，可以例如仅含有一些部分值，通过其级联来生成各种工件零点。

示例

赋值

程序代码	注释
DEF FRAME EINSTELLUNG_1	; 定义本地框架变量
EINSTELLUNG_1 = CTRANS(X,10)	; 功能结果分配至框架变量
$P_PFRAME = EINSTELLUNG_1	; 框架变量分配至当前框架
DEF FRAME EINSTELLUNG_4	; 定义本地框架变量
EINSTELLUNG_4 = $P_PFRAME	; 当前框架缓存至框架变量
...	
$P_PFRAME = EINSTELLUNG_4	; 从框架变量读回当前框架

连接

框架以编程顺序通过符号：相互连接。 框架组件，例如位移、旋转等先后相加。

程序代码	注释
$P_IFRAME = $P_UIFR[15] : $P_UIFR[16]	; 结果赋值 - 连接框架 ; 两个可设定数据管理框架，在激活的 ; 可设定总框架。 ; 应用示例： ; $P_UIFR[15]:偏移 ; $P_UIFR[16]:旋转

程序代码	注释
`$P_UIFR[3] = $P_UIFR[4] : $P_UIFR[5]`	; 结果赋值 - 连接框架 ; 两个可设定数据管理框架，在一个 ; 其他可设定数据管理框架

6.2.4 定义框架变量 (DEF FRAME)

除了预定的框架变量以外，也可定义自己的框架变量。 自定义的框架变量的用户变量类型为 FRAME。 框架的名称可以在用户变量规定的范围内自由指定。

通过功能 CTRANS, CROT, CSCALE, CMIRROR 可以为自定义框架变量赋值。

句法

```
DEF FRAME <名称>
```

含义

DEF FRAME:	定义用户变量类型 FRAME。
<名称>:	框架变量名称

示例

定义一个框架变量"PALETTE"并分配位移和旋转值：

程序代码	注释
`DEF FRAME PALETTE`	; 定义框架变量 PALETTE
`PALETTE = CTRANS(...) : CROT(...)`	; 连接的结果框架分配 ; 框架变量 PALETTE 的位移和旋转

6.3 粗位移和精位移 (CTRANS, CFINE)

精位移

精位移 CFINE(...)可用于以下框架：

- 可设定的框架： $P_UIFR bzw. $P_IFRAME
- 基本框架： $P_NCBFR[<n>], $P_CHBFR[<n>] 或 $P_CHBFRAMES[<n>] 或 $P_ACTBFRAME
- 可编程的框架：$P_PFRAME

使用 CFINE(...) 指令编程框架的精位移。

粗位移

粗位移 CTRANS(...)可用于所有框架。

总偏移

总位移由粗位移和精位移相加而得。

机床数据

解锁精位移

解锁精位移和机床数据：

MD18600 $MN_MM_FRAME_FINE_TRANS = 1

句法

精位移

- 总框架
  - <框架> = CFINE(<K_1>,<值>)
  - <框架> = CFINE(<K_1>,<值>, <K_2>, <值>)
  - <框架> = CFINE(<K_1>,<值>, <K_2>, <值>, <K_3>, <值>)
- 框架分量
  - <框架>[<n>, <K_1>, FI] = <值>

粗位移

- 总框架
  - <框架> = CTRANS(<K_1>,<值>)
  - <框架> = CTRANS(<K_1>,<值>, <K_2,<值>)
  - <框架> = CTRANS(<K_1>,<值>, <K_2,<值>, <K_3,<值>)
- 框架分量
  - <框架>[<n>,<K_1>,TR] = <值>

专用于可编程框架$P_PFRAME:

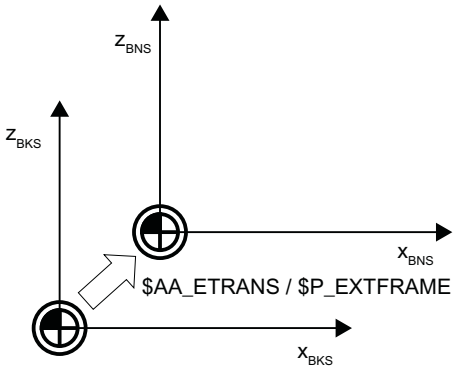
- TRANS <K_1> <值>
- TRANS <K_1> <值> <K_2> <值>
- TRANS <K_1> <值> <K_2> <值> <K_3> <值>

含义

<框架>:	框架，例如可设定数据管理框架$P_UIFR[<n>]
CFINE:	精位移，累加式位移。
CTRANS:	粗位移，绝对位移
TRANS:	只有可编程的框架： 粗位移，绝对位移
<K_n>:	坐标轴 X, Y, Z
<值>:	位移值

6.4 外部零点偏移 ($AA_ETRANS)

外部零点偏移是基本坐标系（BKS）和基础零点系统（BNS）之间的线性位移。



外部零点偏移通过$AA_ETRANS 生效，取决于机床数据的参数设置，有两种方式：

1. 系统变量$AA_ETRANS 在通过 NC/PLC 接口信号激活后直接作为偏移值生效
2. 系统变量$AA_ETRANS 的值在通过 NC/PLC 接口信号激活后被接收成为有效的系统框架$P:EXTFRAME 和数据管理框架$P_EXTFR 的值。随后激活的总框架$P_ACTFRAME 将被重新计算。

机床数据

与系统变量 $AA_ETRANS 相关，有两种方法可以区分，通过以下机床数据选择：

MD28082 $MC_MM_SYSTEM_FRAME_MASK,位 1= <值>

<值>	含义
0	功能：通过 PLC，HMI 或 NC 程序直接写$AA_ETRANS[<轴>]。 解锁，从$AA_ETRANS[<轴>]移出零点偏移至下一个可能的运行偏差：DB31, ... DBX3.0
1	功能：激活有效的系统框架$P:EXTFRAME 和数据管理框架$P_EXTFR 解锁，从$AA_ETRANS[<轴>]移出零点偏移，通过：DB31, ... DBX3.0 之后在通道内： ● 停止通道中的所有运行移动（除命令和 PLC 轴） ● 预处理站，随后重组（STOPRE） ● 粗位移有效框架$ P_EXTFRAME[<轴>]=$ AA_ETRANS[<轴>] ● 粗位移数据管理框架$ P_EXTFRAME[<轴>]=$ AA_ETRANS[<轴>] ● 重新计算有效的总框架$P_ACTFRAME ● 位移移出至编程轴。 ● 继续中断的运行移动或 NC 程序

编程

- 句法

  ```
  $AA_ETRANS[<轴>] == <值>
  ```

- 含义

$AA_ETRANS:	系统变量，用于缓存外部零点偏移
<轴>：	通道轴
<值>：	位移值

NC/PLC 接口信号

DB31, ... DBX3.0 = 0 → 1 ⇒ $P_EXTFRAME[<轴>] = $P_EXTFR[<轴>] = $AA_ETRANS[<轴>]

6.5 参考点状态的实际值设置和损失（PRESETON）

程序 PRESETON() 在机床坐标系（MKS）为一个或多个轴设置新的实际值。 该值等于轴的 MKS 零点偏移。 因此不运行此轴。

通过 PRESETON 可触发一个带同步的预处理停止。 实际位置在轴静止时分配。

如果轴在 PRESETON 指令下未分配到通道，随后的过程取决于轴专用的配置：

MD30552 $MA_AUTO_GET_TYPE

回参考点状态

通过在机床坐标系设置新的实际值使机床轴参考点状态复位：

DB31, ... DBX60.4 / .5 = 0（已回参考点/已同步，测量系统 1/2）

建议 PRESETON 只用于没有参考点义务的轴。

为了恢复原来的机床坐标系，机床轴的测量系统必须通过例如从零件程序接近参考点（G74）来重新返回到参考点。

⚠ 小心
参考点状态丢失 通过 PRESETON 在机床坐标系设置新的实际值，使机床轴的参考点状态重置到“未返回参考点/未同步”。

编程

句法

```
PRESETON(<轴_1>, <值_1> [, <轴_2>, <值_2>, ... <轴_8>, <值_8>])
```

含义

PRESETON：	参考点状态的实际值设置和损失	
	预处理停止：	是
	在单独程序段中编程：	是
<轴_x>:	机床进给轴名称	
	类型：	AXIS
	取值范围：	在通道定义的机床轴名称

<值_x>:	机床坐标系中机床轴的新实际值（MKS） 在当前有效的尺寸系统输入（英制/公制） 考虑到有效的直径编程（DIAMON）	
	类型：	REAL

文档

在 NC 程序的 PRESETONS

NC 程序中 PRESETON 的详细描述请参阅：

功能手册之基本功能；章节"K2: 轴，坐标系，框架"**>**"坐标系"**>**"机床坐标系（MCS）"**>**"参考点状态实际值设置和损失（PRESETON）"

同步动作的 PRESETONS

同步动作的 PRESETON 的详细描述请参阅：

功能手册 同步动作，章节："详细描述"**>**"同步动作操作"**>**"参考点状态实际值设置和损失（PRESETON）"

6.6 参考点状态的实际值设置，无损失（PRESETONS）

程序 PRESETONS() 在机床坐标系（MKS）为一个或多个轴设置新的实际值。 该值等于轴的 MKS 零点偏移。 因此不运行此轴。

通过 PRESETON 可触发一个带同步的预处理停止。 实际位置在轴静止时分配。

如果轴在 PRESETONS 指令下未分配到通道，随后的过程取决于轴专用的配置：

MD30552 $MA_AUTO_GET_TYPE

回参考点状态

通过 PRESETONS 在机床坐标系（MKS）设置新的实际值，机床轴的参考点状态**不会**改变。

前提条件

- **编码器类型**

  PRESETONS 仅适用有效测量系统的以下编码器类型：
  - MD30240 $MA_ENC_TYPE[<测量系统>] = 0 (仿真编码器)
  - MD30240 $MA_ENC_TYPE[<测量系统>] = 1 (原始信号传感器)
- **参考点模式**

  PRESETONS 仅适用有效测量系统的以下参考点模式：
  - MD34200 $MA_ENC_REFP_MODE[<测量系统>] = 0 (无需接近参考点)
  - MD34200 $MA_ENC_REFP_MODE[<测量系统>] = 1 (参考增量、旋转或线性测量系统： 零脉冲位于编码器轨迹上）

编程

句法

```
PRESETONS(<轴_1>, <值_1> [, <轴_2>, <值_2>, ... <轴_8>, <值_8>])
```

含义

PRESETONS:	参考点状态的实际值设置，无损失	
	预处理停止：	是
	在单独程序段中编程：	是
<轴_x>:	机床进给轴名称	
	类型：	AXIS
	取值范围：	在通道定义的机床轴名称

<值_x>:	机床坐标系中机床轴的新的当前实际值（MKS） 在有效的尺寸系统输入（英制/公制） 考虑到有效的直径编程（DIAMON）	
	类型：	REAL

文档

在 NC 程序的 PRESETONS

NC 程序中 PRESETONS 的详细描述请参阅：

功能手册之基本功能；章节"K2: 轴，坐标系，框架"**>**"坐标系"**>**"机床坐标系（MCS）"**>**"参考点状态实际值设置和损失（PRESETONS）"

同步动作的 PRESETONS

同步动作的 PRESETONS 的详细描述请参阅：

功能手册 同步动作，章节："详细描述"**>**"同步动作操作"**>**"参考点状态实际值设置，无损失（PRESETONS）"

6.7 从空间中的三个测量点计算框架 (MEAFRAME)

MEAFRAME 功能用于支持测量循环。 使用此功能可从三个理想的点及其相应的测量点计算出框架。

如果定位一个供加工的工件，则其位置相对于直角的机床坐标系及其理想位置可以偏移或者旋转。 用于精确加工或者测量时，要么需要进行成本高昂的物理调整，要么在零件程序中对运动进行修改。

通过在空间探测已知理想位置的三个点可以确定一个框架。 使用一个触碰标板上精确定位的专用孔或者测量球的接触式或者光电传感器进行探测。

句法

```
MEAFRAME(<理想点>,<测量点>,<质量>)
```

含义

MEAFRAME：	功能调用		
<理想点>：	二维 实数数组，它包含理想点的三个坐标		
<测量点>：	二维 实数数组，它包含测量点的三个坐标		
<质量>：	变量，通过其反馈框架计算质量信息		
	类型：	VAR REAL	
	值：	-1	这些理想点位于直线附近： 无法计算框架。 返回的框架变量含有一个中性框架。
		-2	测量点几乎在一条直线上： 无法计算框架。 返回的框架变量含有一个中性框架。
		-4	旋转矩阵的计算因另外一个原因而失败。
		≥ 0.0	将测定的三角形转换成一个与理想三角形一致的三角形所需的变形之和（点之间的距离）。

说明

测量的质量

为了能够使用旋转/平移组合将所测定的坐标分配给理想的坐标，由测量点所确定的三角形必须与理想三角形一致。 应设法用一种可将偏差的平方之和减小到最小程度的补偿算法，将所测定的三角形转换成理想三角形。

测量点的有效所需变形可作为测量质量的指标，因此被 MEAFRAME 作为辅助变量输出。

说明

使用 MEAFRAME 创建的框架可通过 ADDFRAME 功能转换为框架级联中的另一个框架（参见示例“使用 ADDFRAME 级联”）。

示例

示例 1：

零件程序 1：

程序代码
...
DEF FRAME CORR_FRAME

设定测量点：

程序代码	注释
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)	；用于测试。
DEF REAL FIT_QUALITY=0	
DEF REAL ROT_FRAME_LIMIT=5	；最大允许有 5 度的零件位置旋转。
DEF REAL FIT_QUALITY_LIMIT=3	；在理想和测量的三角之间 最大允许有 3 mm 的偏移。
DEF REAL SHOW_MCS_POS1[3]	
DEF REAL SHOW_MCS_POS2[3]	
DEF REAL SHOW_MCS_POS3[3]	

程序代码	注释
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	; 第 1 个 RPY 角的极限值。
SETAL(65020)	
GOTOF NO_FRAME	
ENDIF	
N260 IF CORR_FRAME[Y,RT] > ROT_FRAME_LIMIT	; 第 2 个 RPY 角的极限值。
SETAL(65021)	
GOTOF NO_FRAME	
ENDIF	
N270 IF CORR_FRAME[Z,RT] > ROT_FRAME_LIMIT	; 第 3 个 RPY 角的极限值。
SETAL(65022)	
GOTOF NO_FRAME	
ENDIF	
N300 $P_IFRAME=CORR_FRAME	; 激活带有一个可设置的框架的探测框架。 ; 通过将几何轴向理想点定位的方式来检查框架。
N400 X=IDEAL_POINT[0,0] Y=IDEAL_POINT[0,1] Z=IDEAL_POINT[0,2]	
N410 SHOW_MCS_POS1[0]=$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	; 取消可设定的框架，因为已使用零框架预置（没有事先输入值）
No_FRAME	; 取消可设定的框架，因为已使用零框架预置（没有输入值）。

程序代码	注释
M0	
M30	

示例 2： 框架级联

级联 MEAFRAME 用于补偿

MEAFRAME 功能提供一个补偿框架。 若将此补偿框架与调用功能时生效（例如 G54）的可设定框架 $P_UIFR[1] 级联，则可得到一个能够进一步换算用于运行或加工的可设定框架。

使用 ADDFRAME 级联

如果要让框架级联中的该补偿框架在另一个位置上发挥作用，或者在可设置框架之前尚有其它框架激活，则可将功能 ADDFRAME 用来在其中一个通道基本框架或者某个系统框架中进行级联。

在这些框架中，以下功能不可生效：

- 使用 MIRROR 镜像
- 使用 SCALE 缩放

用于给定值和实际值的输入参数为工件坐标。 在控制器的基本系统中，这些坐标始终须以公制或英制尺寸（G71/G70）以及半径相关（DIAMOF）尺寸给定。

文档：
ADDFRAME 的更多相关信息请见：
功能手册 基本功能；K2：轴、坐标系、框架、

6.8 NCU 全局框架

对于所有的通道，每个 NCU 仅有一个 NCU 全局框架。 NCU 全局框架可以由所有的通道读写。 分别在各个通道中激活 NCU 全局框架。

通过全局框架可以对带有位移的**通道轴和加工轴** 进行缩放和镜像。

几何关系与框架级联

在全局框架中各个轴之间没有几何关系。 因此不可以进行旋转和编程几何轴名称。

- 全局框架中不可以使用旋转。 编程旋转时会产生报警： "18310 通道 %1 程序段 %2 框架: 不可以旋转”
- 可以进行全局框架和通道专用框架的级联。 最后生成的框架包含所有的框架分量，包括用于所有轴的旋转。 如果带旋转分量的框架赋值于一个全局框架，则产生报警“框架： 不可以旋转”。

NCU 全局框架

NCU-全局基本框架 $P_NCBFR[n]

可以设计 8 个以下的 NCU 全局基本框架：

通道专用的基本框架可以同时存在。

全局框架可以由一个 NCU 的所有通道读写。 在写全局框架时，由用户考虑通道的协调。例如可以通过等候标记（WAITMC）来实现这一点。

机床制造商

全局基本框架的数量通过机床数据配置。

文档：
功能手册 基本功能；轴、坐标系、框架(K2)

NCU 全局可设置框架 $P_UIFR[n]

所有可设置框架 G500、G54...G599 均可配置为 NCU 全局型或者通道专用型。

机床制造商

所有可设置框架可通过机床数据 MD18601 $MN_MM_NUM_GLOBAL_USER_FRAMES 重新配置为全局框架。

使用框架的编程指令时，可以使用通道轴名和加工轴名作为轴名称。 编程几何轴名称时会出现报警，从而无法进行。

6.8.1 通道专用框架 ($P_CHBFR, $P_UBFR)

可使用零件程序和 BTSS 通过操作系统及 PLC 读写可设定框架和基本框架。

精位移也可以用于全局框架。 和通道专用框架一样，也通过 G53、G153、SUPA 和 G500 来抑制全局框架。

机床制造商

通过机床数据 MD28081 $MC_MM_NUM_BASE_FRAMES 可以设定通道中基准框架的数量。 默认配置被设计成每个通道至少有一个基本框架的形式。 每个通道最多可以有 8 个基准框架。 在通道中除了 8 个基准通道之外，还可以有另外 8 个 NCU 全局基准框架。

通道专用框架

$P_CHBFR[n]

通过系统变量 $P_CHBFR[n] 可以读取和写入基本框架。 当写入某个基本框架时，级联的全部基本框架不会激活，而是在执行 G500、G54...G599 中的一个指令时才会激活。 该变量主要在从 HMI 或者 PLC 写入到基本框架的过程中作为存储器使用。 这些框架变量通过数据存储进行保护。

通道中的第一个基准框架

向预定义变量 $P_UBFR 写入时，不会同时激活数组索引为 0 的基本框架，而是在执行 G500、G54...G599 中的一个指令时才会激活。 变量也可以在程序中读写。

$P_UBFR

$P_UBFR 和 $P_CHBFR[0] 一样。 默认情况下通道中始终有一个基本框架，使得这些系统变量可与较早的版本兼容。 如果没有通道专用基准框架，则在读写时会产生报警“框架：“指令不允许”。

6.8.2 在通道中有效的框架

在通道中有效的框架由零件程序通过这些框架的有关系统变量来输入。 这里也包括系统变量。 通过这些系统变量可以在零件程序中读写当前的系统框架。

当前在通道中有效的框架

一览

当前的系统框架	用于：
$P_PARTFRAME	TCARR 和 PAROT

$P_SETFRAME	实际值设定和刮削
$P_EXTFRAME	外部零点偏移
$P_NCBFRAME[n]	当前的 NCU 全局基准框架
$P_CHBFRAME[n]	当前的通道基本框架
$P_BFRAME	通道中当前的第一个基准框架
$P_ACTBFRAME	总体基准框架
$P_CHBFRMASK 和 $P_NCBFRMASK	总体基准框架
$P_IFRAME	当前可设定的框架
当前的系统框架	用于：
$P_TOOLFRAME	TOROT 和 TOFRAME
$P_WPFRAME	工件基准点
$P_TRAFRAME	转换
$P_PFRAME	当前可编程的框架
当前的系统框架	用于：
$P_CYCFRAME	循环
P_ACTFRAME	当前的总框架
框架级联	当前框架由全部基本框架组成

$P_NCBFRAME[n] 当前的 NCU 全局基本框架

通过系统变量 $P_NCBFRAME[n] 可以读取和写入当前的全局基本框架数组元素。 在通道中写过程中，最后生成的总基准框架一起计算在内。

修改的框架仅在编程的通道中生效。 如果要求修改一个 NCU 所有通道的框架，则必须同时说明 $P_NCBFR[n] 和 $P_NCBFRAME[n]。 然后其它通道必须通过例如 G54 来激活框架。 在写一个基准框架时，重新计算总的基准框架。

$P_CHBFRAME[n] 当前的通道基本框架

通过系统变量 $P_CHBFRAME[n] 可以读取和写入当前的通道基本框架数组元素。 在通道中写过程中，最后生成的总基准框架一起计算在内。 在写一个基准框架时，重新计算总的基准框架。

$P_BFRAME 通道中当前的第 1 个基本框架

通过预定义框架变量 $P_BFRAME 可以在零件程序中读取和写入带有在通道中有效的数组索引 0 的当前基本框架。 写入的基准框架立即计算在内。

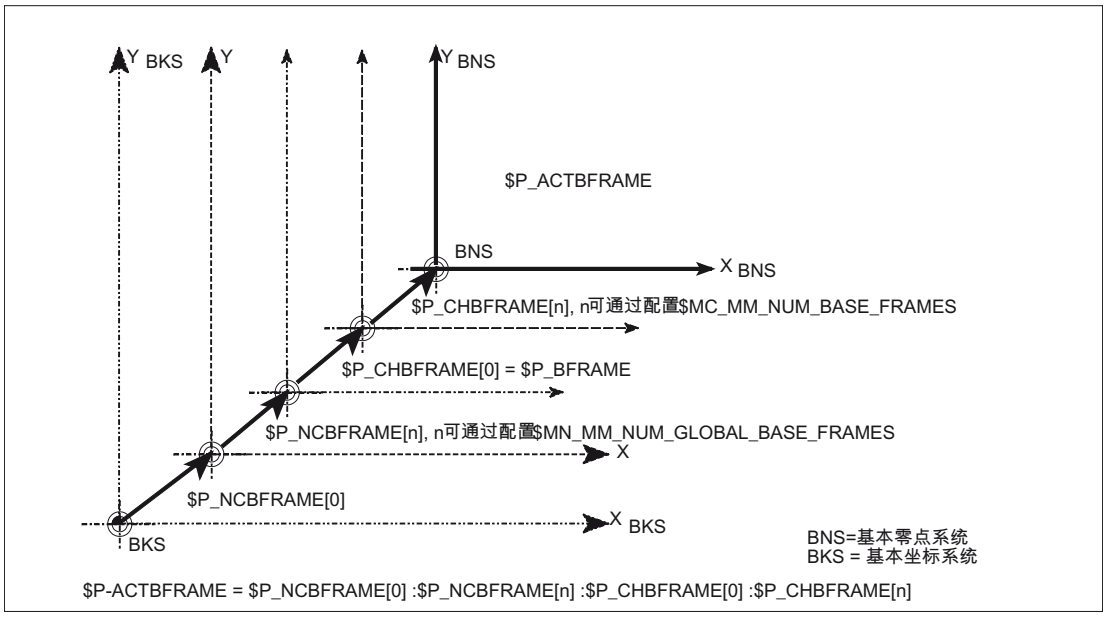
$P_BFRAME 和 $P_CHBFRAME[0] 一样。 在正常情况下，系统变量始终有一个有效值。 如果没有通道专用基准框架，则在读写时会产生报警“框架：“指令不允许”。

$P_ACTBFRAME 全部基本框架

变量 $P_ACTFRAME 用来检查级联的全部基本框架。 该变量仅可读。

$P_ACTFRAME 相当于：

$P_NCBFRAME[0] :... : $P_NCBFRAME[n] :$P_CHBFRAME[0] : ... : $P_CHBFRAME[n].



$P_CHBFRMASK 和 $P_NCBFRMASK 全部基本框架

用户可以通过系统变量 $P_CHBFRMASK 和 $P_NCBFRMASK 来选择要在计算“全部”基本框架时同时考虑哪些基本框架。 变量仅在程序中编程，通过机床控制面板读入。 将变量的值作为位掩码并且指定将 $P_ACTFRAME 的哪些基本框架数组元素考虑到计算中。

使用 $P_CHBFRMASK 可以设定将哪些通道专用基本框架考虑在内，且使用 $P_NCBFRMASK 来设定将哪些 NCU 全局基本框架考虑在内。

编程这些变量重新计算总的基准框架和总的框架。 复位后以及出厂设置中 $P_CHBFRMASK 和 $P_NCBFRMASK 的值如下：

$P_CHBFRMASK = $MC_CHBFRAME_RESET_MASK

$P_NCBFRMASK = $MC_CHBFRAME_RESET_MASK

示例：

```
$P_NCBFRMASK = 'H81' ;$P_NCBFRAME[0] : $P_NCBFRAME[7]
$P_CHBFRMASK = 'H11' ;$P_CHBFRAME[0] : $P_CHBFRAME[4]
```

$P_IFRAME 当前的可设置框架

通过预定义框架变量 $P_IFRAME 可以在零件程序中读取和写入在通道中有效的当前可设置框架。 写入的可设定框架立即计算在内。

在 NCU 全局的、可设定的框架中，修改的框架仅在编程的通道中生效。 如果要修改某个 NCU 所有通道的框架，就必须同时写入 $P_UIFR[n] 和 $P_IFRAME 。 然后其它通道必须通过例如 G54 激活相应框架。

$P_PFRAME 当前的可编程框架

$P_PFRAME 为可编程框架，其通过 TRANS/ATRANS、G58/G59、ROT/AROT、SCALE/ASCALE、MIRROR/AMIRROR 的编程以及分配至可编程 FRAME 的 CTRANS、CROT、CMIRROR、CSCALE 得出。

当前可编程的框架变量，建立可设定零点坐标系(ENS)和工件坐标系(WKS)之间的关系。

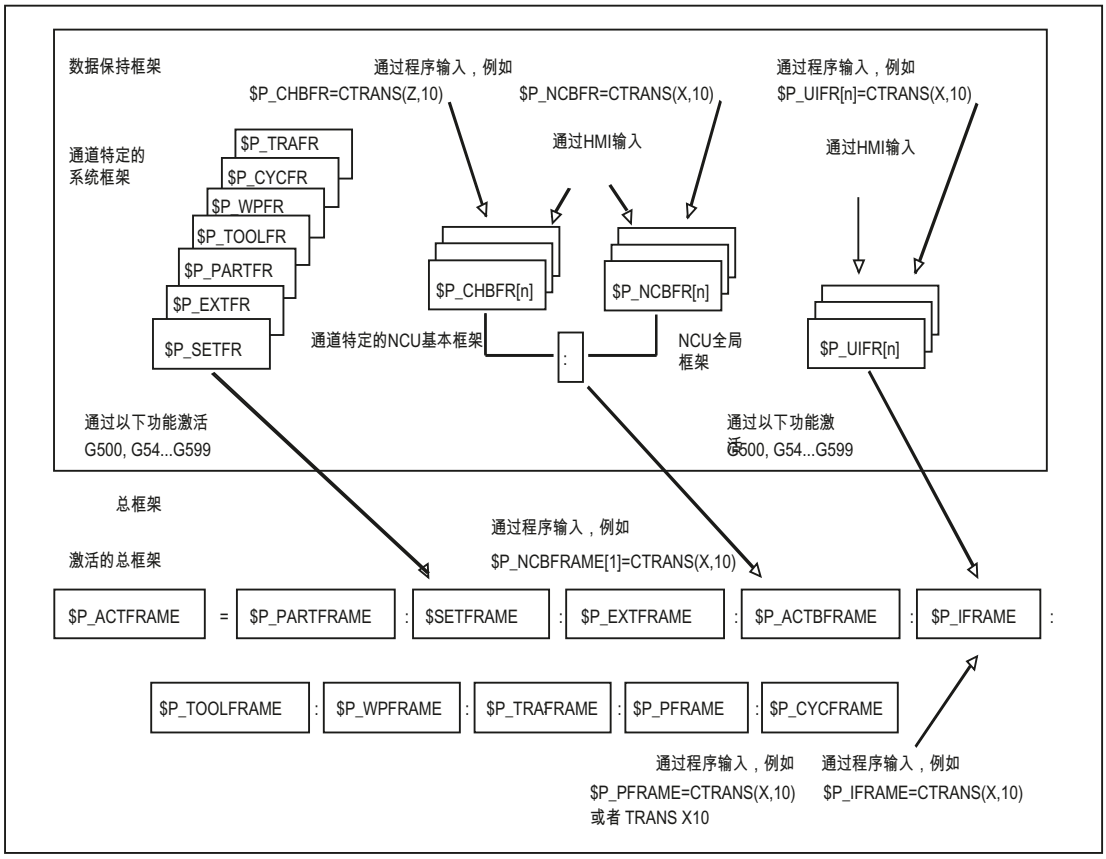
P_ACTFRAME 当前的总框架

当前的合成总框架 $P_ACTFRAME 现在作为级联受控于所有基本框架、当前的可设置框架和可编程框架。 如果框架分量改变，则当前框架会更新。

$P_ACTFRAME 相当于：

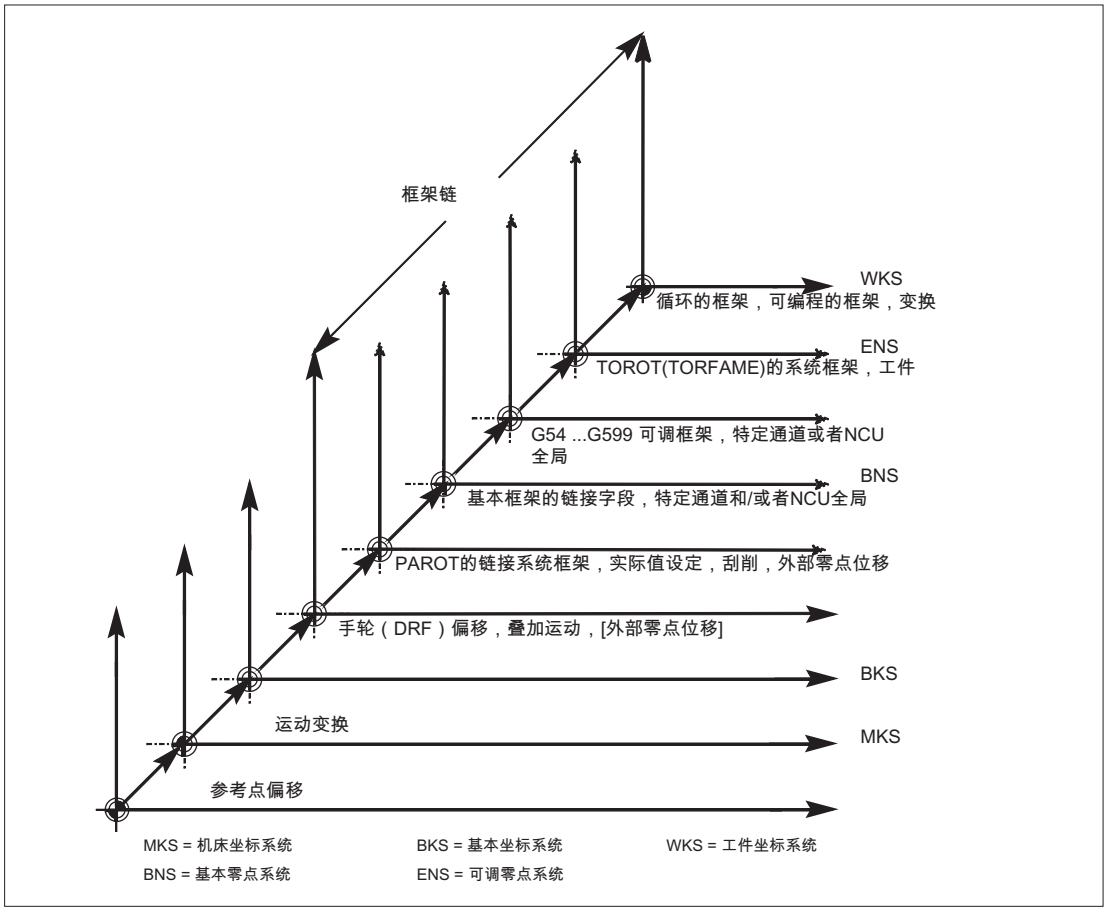
```
$P_PARTFRAME : $P_SETFRAME : $P_EXTFRAME : $P_ACTBFRAME :
$P_IFRAME :

$P_TOOLFRAME : $P_WPFRAME : $P_TRAFRAME : $P_PFRAME : $P_CYCFRAME
```


数据保持框架
通过程序输入，例如
$P_CHBFR=CTRANS(Z,10)
$P_NCBFR=CTRANS(X,10)
通过程序输入，例如
$P_UIFR[n]=CTRANS(X,10)
通道特定的
系统框架
$P_TRAFR
$P_CYCFR
$P_WPFR
$P_TOOLFR
$P_PARTFR
$P_EXTFR
$P_SETFR
通过HMI输入
通过HMI输入
$P_CHBFR[n]
$P_NCBFR[n]
$P_UIFR[n]
通道特定的NCU基本框架
:
NCU全局
框架
通过以下功能激活
G500, G54...G599
通过以下功能激
活
G500, G54...G599
总框架
通过程序输入，例如
$P_NCBFRAME[1]=CTRANS(X,10)
激活的总框架
$P_ACTFRAME
=
$P_PARTFRAME
:
$SETFRAME
:
$P_EXTFRAME
:
$P_ACTBFRAME
:
$P_IFRAME
:
$P_TOOLFRAME
:
$P_WPFRAME
:
$P_TRAFRAME
:
$P_PFRAME
:
$P_CYCFRAME
通过程序输入，例如
$P_PFRAME=CTRANS(X,10)
或者 TRANS X10
通过程序输入，例如
$P_IFRAME=CTRANS(X,10)

框架级联

根据以上所说的当前的总框架，当前的框架由总的基准框架、可设定的框架、系统框架和可编程的框架组成。



转换 7

7.1 转换方式的一般编程

一般功能

为了使控制系统能够匹配不同的机床运动，用合适的参数对所选的转换方式进行编程。 通过这些参数，使刀具的空间定向和回转轴定向运动在所选的转换中达成一致。

在三轴、四轴和五轴转换中，进行编程的位置标注总是以刀尖为参照，它正交于空间加工平面。 直角坐标系从基本坐标系转换到机床坐标系，并以几何轴为参照。 对工作点进行描述。 虚拟的回转轴描述了刀具的空间定向，用 TRAORI 对其进行编程。

运动转换时，在直角坐标系中对位置进行编程。 控制系统把用 TRANSMIT,TRACYL 和 TRAANG 编程的直角坐标系过程运动转换成真实加工轴的过程运动。

编程

三轴、四轴和五轴转换（TRAORI）

约定的方位转换通过指令 TRAORI 和转换编号、方位矢量和回转轴偏移这三个可能的参数来激活。

TRAORI(转换编号、方位矢量、回转轴偏移)

运动转换

约定的转换 TRANSMIT(转换编号)属于运动关系转换

TRACYL(加工直径、转换编号)

TRAANG(倾斜轴的角度、转换编号)

关闭有效转换

用 TRAFOOF 可以将正在激活中的转换关闭。

方位转换

三轴、四轴和五轴转换（TRAORI）

为了在机床加工空间中达到成形平面的最佳空间加工,机床除了 3 个线性轴 X、Y 和 Z 外还需要其他的轴。 这些辅助轴用来描述空间中的定向，因此被称作定向轴。 它们用作四种机床类型不同运动的旋转轴。

1. 两轴旋转头，例如：刀具台固定时，带有一个回转轴的万向刀具头平行于一个线性轴。
2. 两轴转台，例如固定式旋转头与可以围绕两个轴旋转的刀台
3. 单轴旋转头和单轴转台，例如一个可以旋转的旋转头，带有旋转式刀具，可围绕一个轴旋转的刀台。
4. 两轴旋转头和单轴转台，例如一个可围绕一个轴旋转的刀台和一个可绕自身旋转的带旋转刀具的旋转头。

3 轴和 4 轴转换是 5 轴转换的特殊形式，编程与 5 轴转换类似。

"生成的 3-/4-/5-/6-轴转换"可以用其用于垂直排列回转轴以及用于万向铣头转换的功能范围覆盖，可以象其它每个定向转换一样用 TRAORI 激活用于这四种机床类型。 在生成 5/6 轴转换时，刀具方位有一个附加的第三自由度，有它可在空间中任意至刀具方向，刀具绕自身的轴可以任意旋转。

文档： /FB3/ 功能手册特殊功能；3 至 5 轴转换（F2）

刀具定向的初始位置与运动无关

ORIRESET

如果用 TRAORI 激活了一个定向转换，则可以用 ORIRESET 规定初始位置为最多 3 个带可选参数 A、B、C 的定向轴。 根据通过转换确定的定向轴顺序分配已编程的参数到回转轴的顺序。 编程 ORIRESET（A, B, C），使得线性和同步定向轴自其当前位置向给定的初始位置运行。

运动转换

TRANSMIT 和 TRACYL

车床进行铣削加工时，对于约定的转换，

1. 可以用 TRANSMIT 在车削夹装中编程一个端面加工或者
2. 用 TRACYL 在圆柱体上编程一个任意走向槽的加工

。

TRAANG

如果进给轴例如为了工艺磨削也可以倾斜进给，则可以用 TRAANG 为约定的转换编程一个可参数化的角。

直角坐标 PTP 运动

属于运动转换的还有“直角坐标系中的 PTP-运动”， 此种运动可以编程有 8 个以下的不同关节位置 STAT= 。 这些位置在直角坐标系中进行编程，同时在机床坐标系中实现机床的运动。

文档：
/FB2/ 功能手册扩展功能；运动转换（M1）

级联的转换

每次可以前后进行两个转换。 当对由此形成的级联进行第二个转换时，就会从第一个转换接管轴的运动分量。

作为第一个转换的可以是：

- 定向转换 TRAORI
- 极转换 TRANSMIT
- 圆柱转换 TRACYL
- 斜向轴转换 TRAANG

第二个转换必须是斜向轴 TRAANG

7.1.1 转换时的定向运动

过程运行和定向运动

可编程定向的过程运行首先取决于机床类型。在用 TRAORI 进行三轴、四轴和五 轴转换时，旋转轴或者可摆动的线性轴体现了刀具的定向运动。

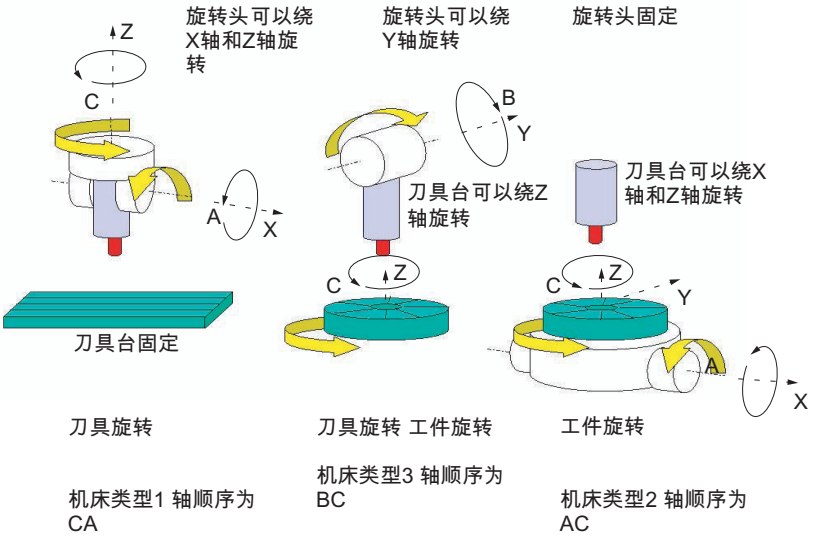
参与定向转换的回转轴的位置改变会导致其余加工轴的补偿运动。因此刀尖的位置保持不变。

刀具的定向运动可以通过虚拟轴的回转轴标识符 A...,B...,C...分别按照用途进行编程，即通过标注欧拉角、RPY 角或者方向矢量或平面垂线矢量、用于圆锥旋转轴的标准矢量编程，或者在圆锥外表面为了暂时定向而编程。

当用 TRANSMIT,TRACYL 和 TRAANG 进行运动转换时，控制系统把编程的直角坐标系过程运动转换成真实加工轴的过程运动。

三轴、四轴和五轴转换（TRAORI）时的机床运动

刀具可旋转或者最多两个回转轴的刀台可旋转。各单轴转动头和旋转台也可以组合在一起。



机床类型	对定向进行编程
机床类型 1 和 2 的三轴转换	仅在与旋转轴垂直的平面上对刀具定向进行编程。有 **两个** 移动的轴(线性轴)和 **一个** 旋转的轴(回转轴)。
机床类型 1 和 2 的四轴转换	仅在与旋转轴垂直的平面上对刀具定向进行编程。有 **三个** 移动的轴(线性轴)和 **一个** 旋转的轴(回转轴)。
机床类型 3 的 5 轴转换 单轴旋转头和单轴旋转台	对定向转换进行编程带有 **三个** 线性轴和**两个** 正交旋转轴的运动。 旋转轴平行于三个线性轴中的其中两个线性轴。第一个旋转轴被两个直角坐标线性轴所移动。该旋转轴使第三个线性轴与刀具一起旋转。第二个旋转轴使工件旋转。

生成 5/6 轴转换

机床类型	定向转换的编程
机床类型 4 的五/六轴转换带可旋转刀具的两轴旋转头和单轴旋转台	对定向转换进行编程带有 三个 线性轴和三个 正交旋转轴的运动。 旋转轴平行于三个线性轴中的其中两个线性轴。第一个旋转轴被两个直角坐标线性轴所移动。该旋转轴使第三个线性轴与刀具一起旋转。第二个旋转轴使工件旋转。可以通过一个以旋转角 THETA 围绕自身进行的额外旋转编程刀具的基本定向。

调用“生成的三轴、四轴、五轴和六轴转换”时，可以额外移交刀具基本定向。这不再适用于回转轴方向的限制。如果回转轴没有精确的彼此垂直或者现有的回转轴没有精确平行于线性轴，可能“生成的五/六轴转换”提供更好的刀具定向结果。

运动转换 TRANSMIT,TRACYL 和 TRAANG

对于车床上或者磨削时斜进给轴的铣削加工，取决于转换在标准情况下下列轴顺序有效：

TRANSMIT	激活极转换
卡装形式的端面加工	一个旋转轴 一个垂直于旋转轴的进给轴 一个平行于旋转轴的纵轴

TRACYL	激活圆柱面转换
在圆柱体上加工任意走向的槽口	一个旋转轴 一个垂直于旋转轴的进给轴 一个平行于旋转轴的纵轴

TRAANG	激活斜向轴转换
用斜置的进给轴加工	一个旋转轴 一个具有可设定参数的进给轴 一个平行于旋转轴的纵轴

直角坐标 PTP 运动

机床在机床坐标系中运动，并且编程时使用：

TRAORI	激活转换
PTP 点对点运行	在直角坐标系（MCS）中逼近位置
CP	直角轴的轨迹运动在（BKS）中
STAT	铰接的位置取决于转换。
TU	绕那个角度运行，轴的行程最短

生成 5/6 轴转换时的 PTP 运动

可以在机床坐标和刀具定向中通过回转轴位置和通过与运动无关的矢量欧拉以及 RPY 角或者方向矢量编程机床运动。

同时，可以沿一个圆锥表面进行回转轴插补、带大圆弧插补的矢量插补或者定向矢量插补。

举例：一个万向铣头的三到五轴转换

机床至少有 5 个轴，其中

- 用于直线运动的 3 个移动轴，在加工空间内把工作点向任意位置移动。
- 两个根据可设计的角度（通常为 45 度）排列的旋转运动轴，可以确定刀具在空间中的方位，当角度为 45 度时，这些方位局限到一个半球上。

7.1.2 定向转换 TRAORI 概述

与 TRAORI 相关的可能的编程方式

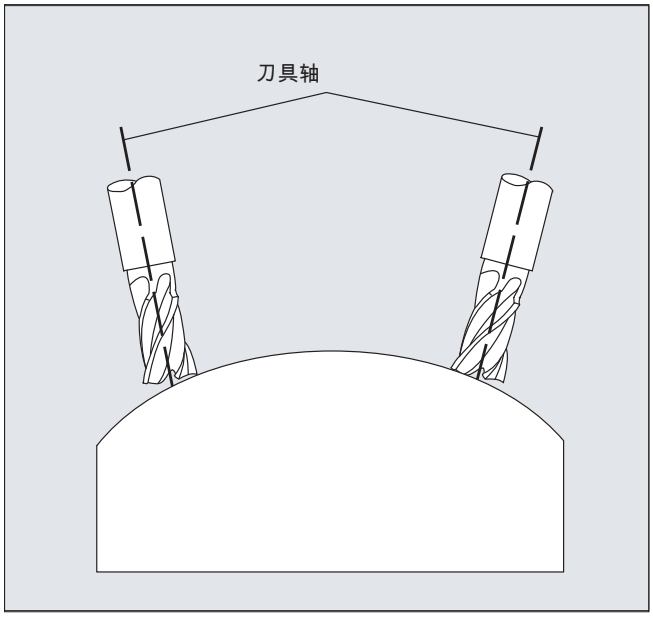
机床类型	当转换 TRAORI 有效时编程
机床类型 1、2 或者 3 两轴旋转头，两轴旋转台或者各单轴旋转头与旋转台的组合。	定向轴顺序和刀具定向方向 **以机床为参照** 时取决于机床运动结构， 可通过机床数据编程； **以工件为参照**时与机床运动结构无关， 可编程定向 编程定向轴的在参照系中的旋转方向时可以使用： - ORIMKS 参照系 = 机床坐标系 - ORIWKS 参照系 = 工件坐标系 缺省设置是 ORIWKS。 定向轴编程时使用： 直接的机床轴位置 A、B、C A2、B2、C2 虚拟轴角度编程 - ORIEULER 通过欧拉角（标准） - ORIRPY 通过 RPY 角 - ORIVIRT1 通过虚拟定向轴第 1 个定义 - ORIVIRT2 通过虚拟定向轴第 2 个定义 不同的插补方式： **直线插补** - 定向轴或机床轴的 ORIAXES **大圆插补**（定向矢量插补） - 定向轴的 ORIVECT 通过说明编程定向轴 矢量分量的 A3、B3、C3（方向/表面标准） 编程得出的刀具定向 程序段开始处表面标准矢量的 A4、B4、C4 程序段结束处表面标准矢量的 A5、B5、C5 用于刀具定向的提前角 LEAD 用于刀具定向的侧向角 TILT

机床类型	当转换 TRAORI 有效时编程
	定向矢量插补在一个圆锥表面上 定向变化在一个**在任意空间内**的 圆锥表面上，通过插补： - ORIPLANE 在平面中（大圆插补） - ORICONCW 在一个圆锥表面上，顺时针 - ORICONCCW 在一个圆锥表面上，逆时针 A6、B6、C6 方向矢量（圆锥的旋转轴） -OICONIO 在一个圆锥表面上插补： A7, B7, C7 中间矢量 (开始定向和结束定向)或者 - ORICONTO 在圆锥表面的切线过渡 改变定向参照 **一段轨迹** 用 - ORICURVE 两个触点运动的预设值通过 PO[XH]=(xe, x2, x3, x4, x5) 定向多项式至 5 次 PO[YH]=(ye, y2, y3, y4, y5) 定向多项式至 5 次 PO[ZH]=(ze, z2, z3, z4, z5) 定向多项式至 5 次 - ORIPATHS 用 A8, B8, C8 平滑 定向走向 刀具换向阶段适合: 退刀方向和行程长度
机床类型 1 和 3 其它带刀具绕自身额外旋转的机床类型要求第 3 个回转轴 定向转换，例如 6 轴转换。 方向矢量的旋转	编程刀具定向**旋转**用 LEAD 超前角与平面垂线矢量相对 PO[PHI] 编程一个至 5 次的多项式 TILT 侧向角绕轨迹切线(Z-方向)旋转 PO[PSI] 编程一个至 5 次的多项式 THETA 旋转角(绕刀具在 Z 的方向旋转) THETA= 在程序段结尾达到的值 THETA=AC(...) 以程序段方式切换到绝对尺寸 THETA=IC(...) 以程序段方式切换到累接尺寸 THETA=Θ_e 插补编程的角度 G90/G91 PO[THT]=(..) 编程一个至 5 次的多项式 编程旋转矢量 - ORIROTA 绝对旋转 - ORIROTR 相对旋转矢量 - ORIROTT 切向旋转矢量
轨迹相关的定向：定向改变与轨迹相关或者旋转矢量的旋转正切于轨迹	定向改变**相对于轨迹** 用 - ORIPATH 刀具定向参照该轨迹 - ORIPATHS 额外在定向走势的一个拐点处 旋转矢量编程 - ORIROTC 与轨迹切线的切向矢量、旋转

7.2 三轴、四轴和五轴转换 (TRAORI)

7.2.1 万向切削头的一般关系

当加工空间曲面时，为了获得最佳切削条件，刀具的定位角必须可以修改。

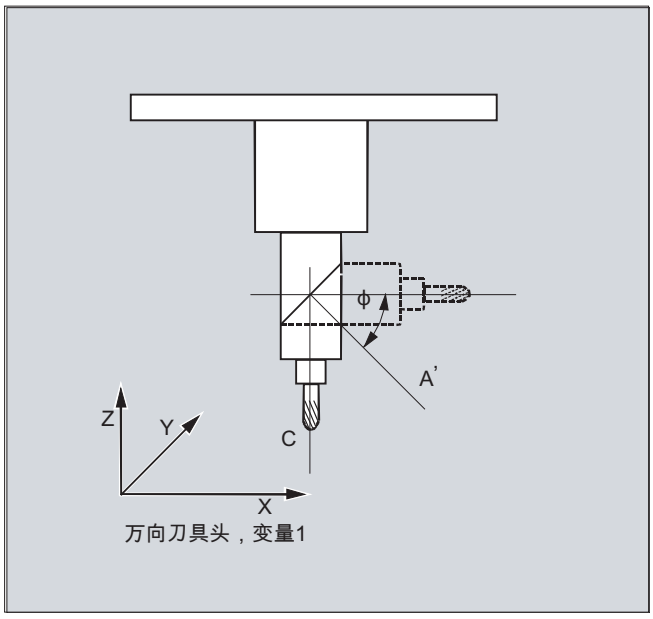


用哪一种机床结构达到这一点，这存储在轴数据中。

5 轴转换

万向切削头

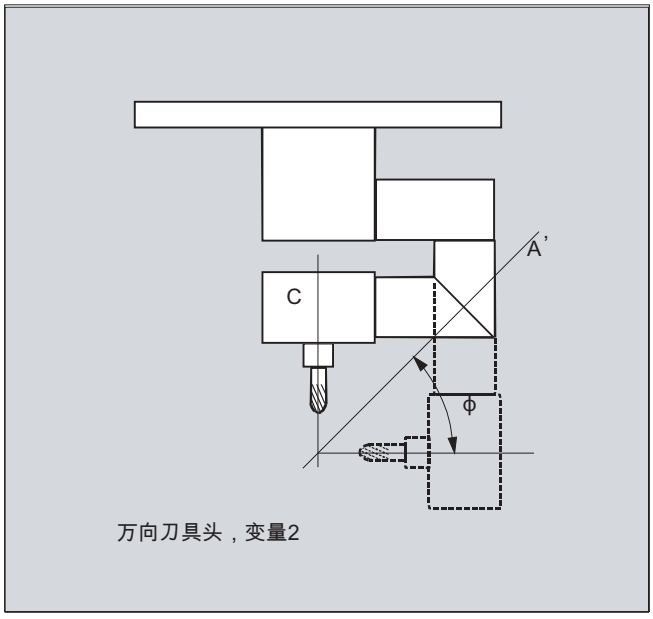
这里三个线性轴 (X, Y, Z) 和两个定向轴 (C, A) 用来确定刀具的定位角和工作点。 两个定向轴的其中之一是作为斜置轴设置的，在这里为 A'（在很多情况下是 45°}£©¡£



在这里所举的示例中，可以看到带有机床运动系统的 CA 的万向组合刀盘的布置！

机床制造商

定向轴的轴顺序和刀具的运动方向取决于通过机床数据的机床类型来设置。



在该例中，A’位于与 X 轴成 φ 的角度下

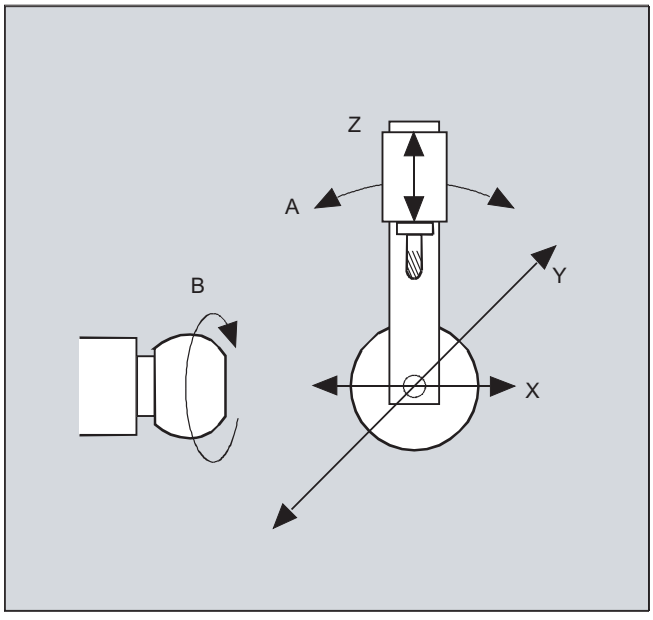
一般适用于下面的关系：

A'在角度 φ 下	X 轴
B' 在角度 φ 下	Y-轴
C' 在角度 φ 下	Z 轴

可以通过机床数据在 0° ～ +89° 的范围内设计角度 φ 。

带有可旋转的线性轴

这种情况与运动的工件和运动的刀具的布置有关。 运动由三个线性轴 (X, Y, Z) 和两个呈直角的旋转轴合成。 例如，通过一个有两个线性轴的滑板使第一个旋转轴运动，刀具平行于第三个线性轴。 第二个旋转轴使工件旋转。 第三个线性轴（转向轴）在十字滑板的平面上。



旋转轴的轴顺序和刀具的运动方向取决于通过机床数据的机床类型来设置。

适用于下面的关系：

进给轴	轴顺序：
1. 回转轴	A A B B C C
2. 回转轴	B C A C A B
转向的线性轴	Z Y Z X Y X

关于刀具定向方向可配置轴顺序的进一步说明参见

文档： /FB3/ 功能手册特殊功能；3 至 5 轴转换（F2），万向铣头一章中的“编程”。

7.2.2 三轴、四轴和五轴转换 (TRAORI)

用户可以设计两个或者三个直线移动轴和一个旋转轴。 坐标转换的条件：旋转轴正交于定向平面上。

刀具只有定位在和旋转轴垂直的平面上。 坐标转换支持带有运动刀具和运动工件的机床类型。

三轴和四轴转换的设计和编程与五轴转换类似。

文档：
功能手册 特殊功能；多轴转换 (F2)

句法

```
TRAORI(<n>)
TRAORI(<n>,<X>,<Y>,<Z>,<A>,<B>)
TRAFOOF
```

含义

TRAORI:	激活第一个设定的方向转换	
TRAORI(<n>):	激活用 n 个设定的方向转换	
<n>:	转换编号	
	值：	1 或 2
	示例： TRAORI(1) 激活方向转换 1	
<X>,<Y>,<Z>:	刀具所指向的方向矢量分量	
<A>,<B>:	旋转轴的可编程偏移	
TRAFOOF:	取消转换	

刀具定向

视所选的刀具定位方向而定，必须在 NC 程序中调整激活的工作平面 (G17, G18, G19)，使得刀具长度补偿在刀具定位的方向中有效。

说明

在启用坐标转换之后，位置数据（X，Y，Z）总是针对刀尖。 如果修改了参与坐标转换的旋转轴位置，其余加工轴也会开始补偿运动，使得刀尖的位置保持不变。

方向转换总是从刀尖指向刀夹。

方向轴的偏移

在激活方向转换时还可以直接编程一个方向轴的附加偏移。

如果在编程时确保了正确的顺序，参数可以不设。

示例：

```
TRAORI(, , , ,A,B) ；当只需输入一个唯一的偏移时
```

除了直接编程之外，方向轴的附加偏移还可以自动接收当前生效的零点偏移。 接收通过机床数据来设计。

示例

```
TRAORI(1,0,0,1)          ；刀具的初始方向指向 Z 轴方向
TRAORI(1,0,1,0)          ；刀具的初始方向指向 Z 轴方向
TRAORI(1,0,1,1)          ；刀具的初始方向指向 Y/Z 轴方向（相当于 -45°）
```

7.2.3 定向编程变量和初始位置（ORIRESET）

TRAORI 时刀具定向的定向编程

此外，对于线性轴 X, Y, Z，与可编程的定向转换 TRAORI 一起，也可以通过轴标识符 A.., B..., C...来编程轴位置，带角度或者矢量分量的虚拟轴。定向轴和加工轴可用不同的插补方式。与哪些定向多项式 PO[角度]和轴多项式 PO[轴]恰好激活无关，可以编程多个不同的多项式种类，例如 G1, G2, G3, CIP 或者 POLY

刀具定向的改变也可以通过定向矢量来编程。由此，每个程序段的结束定位可通过直接矢量编程或者通过回转轴位置实现。

三至五轴转换时定向编程的变量

以下定向编程变量互相排斥。

A、B、C	回转轴位置的直接指定
A2, B2, C2	通过欧拉角或 RPY 角进行虚拟轴的角度编程
A3, B3, C3	指定矢量分量
LEAD, TILT	指定参照于轨迹和表面的导角和侧向角
A4, B4, C4 A5, B5, C5	程序段段首和段尾的平面垂线矢量

A6, B6, C6 A7, B7, C7	在圆锥表面编程定向矢量插补
A8, B8, C8	退刀运动时的刀具换向、方向和长度

运行至刀具定向的初始位置（ORIRESET）

通过 ORIRESET(...)将各个机床运动中的定向轴线性并同步从当前位置运行至所编程的初始位置。如果未给轴编程初始位置，则使用所属机床数据 $MC_TRAFO5_ROT_AX_OFFSET_1/2 中定义的位置。

不考虑回转轴当前有效的框架。

机床运动 CA（通道轴名称 C，A）举例

指令	说明
ORIRESET(90, 45)	C 轴：90 ° A 轴：45 °
ORIRESET(, 30)	C 轴：$MC_TRAFO5_ROT_AX_OFFSET_1/2[0] A 轴：30 °
ORIRESET( )	C 轴：$MC_TRAFO5_ROT_AX_OFFSET_1/2[0] A 轴：$MC_TRAFO5_ROT_AX_OFFSET_1/2[1]

机床运动 CAC（通道轴名称 C，A，B）举例

指令	说明
ORIRESET(90, 45, 90)	C 轴：90 ° A 轴：45 ° B 轴：90 °
ORIRESET( )	C 轴：$MC_TRAFO5_ROT_AX_OFFSET_1/2[0] A 轴：$MC_TRAFO5_ROT_AX_OFFSET_1/2[1] B 轴：$MC_TRAFO5_ROT_AX_OFFSET_1/2[2]

说明

只允许在定向转换 TRAORI(...)生效的情况下通过 ORIRESET(...)运行刀具定向的初始位置。

编程旋转 LEAD、TILT 和 THETA

导角 LEAD 和侧向角 TILT

三轴到五轴的转换时，刀具方位的旋转通过超前角 LEAD 和侧向角 TILT 编程。

旋转角度 THETA

用一个**第三回转轴**既可以通过矢量分量定向编程也可以通过 LEAD，TILT 角编程以旋转角 THETA 使刀具绕自身旋转。

7.2.4 编程刀具定向 (A..., B..., C..., LEAD, TILT)

对于刀具的定向编程有下列可能：

1. 直接编程回转轴运动。定向改变总是在基准坐标系或者机床坐标系统中发生。定向轴作为同步轴运动。
2. 通过 A2，B2，C2 用欧拉角或者 RPY 角编程。
3. 通过 A3，B3，C3 编程方向矢量。方向矢量从刀尖指向刀夹。
4. 在程序段段首用 A4，B4，C4 编程表面法线矢量，在程序段段尾则用 A5，B5，C5 编程表面法线矢量（端面铣削）。
5. 通过导角 LEAD 和侧向角 TILT 编程
6. 通过 A6，B6，C6 编程圆锥旋转轴作为标准矢量或者编程圆锥表面的上的中间定向用 A7，B7，C7，
   ，参见章节“沿圆锥表面定向编程(ORIPLANE, ORICONxx)”。
7. 退刀运动时，通过 A8，B8，C8，
   编程刀具换向、方向和行程长度, 参见章节“平滑定向走势（ORIPATHS A8=, B8=, C8=）”

说明

在所有的情况下，只有当一个定向转换生效时，定向编程才是允许的。

优点：这些程序在每个机床运动类型都是可传送的。

通过 G 代码定义刀具定向

说明

机床制造商

通过机床数据可以在欧拉角或 RPY 角之间转换。对于相应的机床数据设置，可以根据也可以不根据组 50 有效的 G 代码进行切换。可以选择下列设置方式：

1. 当用于定义定向轴和定向角的两个机床数据通过 G 代码置为零时：
   用 A2，B2，C2 编程的角 **根据机床数据** 将定向编程的角度定义编译为欧拉角或 RPY 角。
2. 如果用于定义定向轴的机床数据通过 G 代码置为一，则
   根据组 50 的有效 G 代码进行切换：
   用 A2，B2，C2 编程的角根据有效 G 代码 ORIEULER、ORIRPY、ORIVIRT1、ORIVIRT2、ORIAXPOS 和 ORIPY2 中的一个进行编译。根据组 50 的有效 G 代码也可将用定向轴编程的值编译为定向角。
3. 如果用于定义定向角的机床数据通过 G 代码置为一，而用于定义定向轴的机床数据通过 G 代码置为零，则
   不根据组 50 有效的 G 代码进行切换：
   用 A2，B2，C2 编程的角根据有效 G 代码 ORIEULER、ORIRPY、ORIVIRT1、ORIVIRT2 ORIAXPOS 和 ORIPY2 中的一个进行编译。不根据组 50 的有效 G 代码总是将用定向轴编程的值编译为回转轴位置。

句法

回转轴位置

```
G1 X<值> Y<值> Z<值> A<值> B<值> C<值>
```

欧拉角

```
G1 X<值> Y<值> Z<值> A2=<值> B2=<值> C2=<值>
```

方向矢量

```
G1 X<值> Y<值> Z<值> A3=<值> B3=<值> C3=<值>
```

程序段段首的表面法线矢量

```
G1 X<值> Y<值> Z<值> A4=<值> B4=<值> C4=<值>
```

程序段段尾的表面法线矢量

```
G1 X<值> Y<值> Z<值> A5=<值> B5=<值> C5=<值>
```

导角

```
LEAD=<值>
```

侧向角

```
TILT=<值>
```

含义

G1：	线性插补
X， Y， Z：	线性轴位置
A， B， C：	回转轴位置
A2=， B2=， C2=：	角度编程（欧拉角或者 RPY 角）
A3=， B3=， C3=：	WCS 坐标系中 X， Y，Z 方向上的方向矢量
A4=， B4=， C4=：	WCS 坐标系中 X， Y，Z 方向上的程序段段首的表面法线矢量
A5=， B5=， C5=：	WCS 坐标系中 X， Y，Z 方向上的程序段段尾的表面法线矢量
LEAD= ：	导角 1)
TILT=：	侧向角 1)
1) 角度编译取决于 MD21094 $MC_ORIPATH_MODE 中的设置	

更多信息

通常情况下 5 轴程序由 CAD/CAM 系统生成并且没有输入到控制系统。因此，下列说明主要面向后处理器的编程人员。

提供以下用于定向编程的指令：

指令	含义
ORIEULER：	旋转顺序为 ZX'Z" 的欧拉角
ORIRPY：	旋转顺序为 XY'Z" 的 RPY 角
ORIRPY2：	旋转顺序为 ZY'X" 的 RPY 角
ORIVIRT1：	虚拟定向轴；可自由定义的旋转顺序，通过： MD21120 $MC_ORIAX_TURN_TAB_1
ORIVIRT2：	虚拟定向轴；可自由定义的旋转顺序，通过： MD21130 $MC_ORIAX_TURN_TAB_2
ORIAXPOS：	虚拟定向轴与回转轴位置

说明

通过机床数据可以由机床制造商定义各种变量。请注意机床制造商说明。

通过欧拉角 ORIEULER 编程，旋转顺序 Z X' Z"

在使用 A2、B2、C2 进行定向编程 ORIEULER 时所编程的值被编译为欧拉角（单位：度）。

新的定向矢量由原始定向矢量的以下三种旋转方式得出

1. 回转轴 A2 围绕坐标轴 Z 旋转
2. 回转轴 B2 围绕坐标轴 X' 旋转
3. 回转轴 C2 围绕坐标轴 Z" 旋转

在这种情况下， C2 的值(围绕新 Z 轴的旋转) 没有意义，且不必进行编程。

以 RPY 角 ORIRPY 编程，旋转顺序 X Y' Z"

在使用 A2、B2、C2 进行定向编程 ORIRPY 时所编程的值被编译为 RPY 角（单位：度）并带有旋转顺序 X Y' Z"。

说明

与 ORIEULER 编程相反，ORIRPY 所有三个值均对定向矢量有影响。

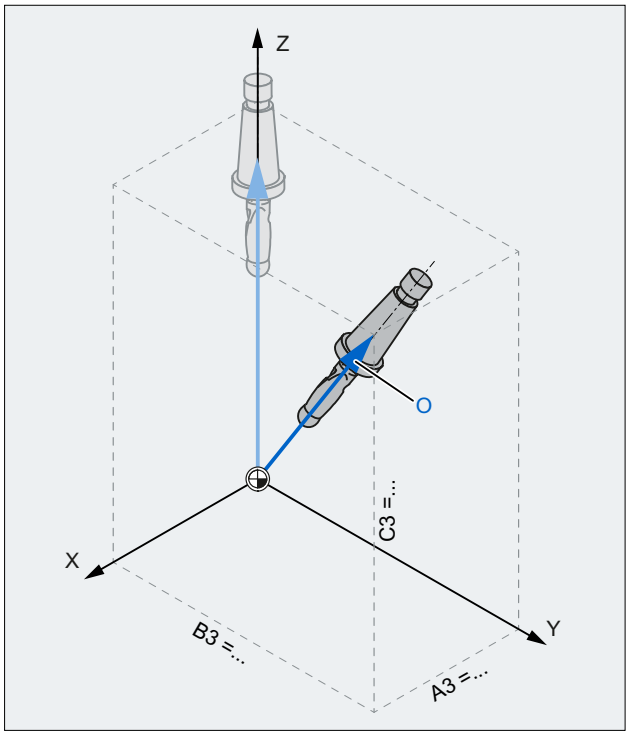
新的定向矢量由原始定向矢量的以下三种旋转方式得出

1. 回转轴 A2 围绕坐标轴 X 旋转
2. 回转轴 B2 围绕坐标轴 Y' 旋转
3. 回转轴 C2 围绕坐标轴 Z" 旋转

编程方向矢量

方向矢量的分量使用 A3, B3, C3 进行编程。矢量显示在刀具安装方向；矢量的长度在此没有意义。

没有编程的矢量组成部分设置为零。



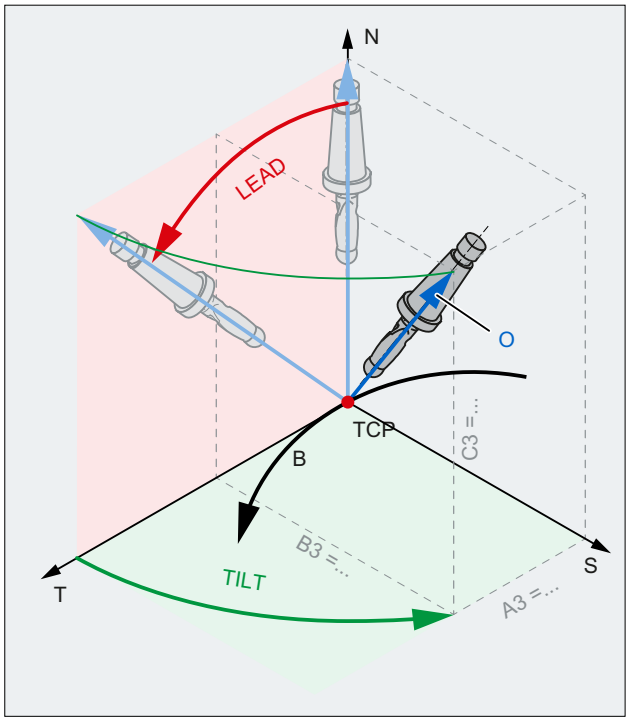
X, Y, Z	WCS 的坐标轴
A3, B3, C3	方向矢量的分量
O	方位矢量

图 7-1 编程方向矢量

编程刀具定向，使用 LEAD 和 TILT

生成的刀具定向通过以下几项得出：

- 轨迹切线
- 表面法线矢量
  位于程序段段首 A4, B4, C4 和程序段段尾 A5, B5, C5 中
- 导角 LEAD
  在由轨迹切线和表面法线矢量展开的平面上
- 程序段段尾处的侧向角 TILT
  垂直于轨迹切线的平面和表面法线矢量所成的角度



T 轨迹切线

S 垂直于轨迹切线

N 表面法线

B 轨迹

TCP Tool Center Point

O 方位矢量

图 7-2 LEAD TILT 的编程

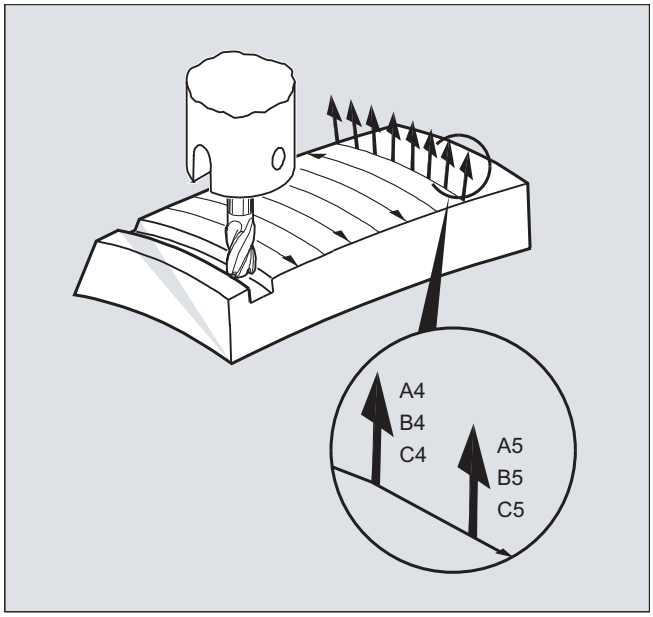
说明

内角的特性（在 3D-WZK 时）

当某个内角上的程序段被缩短时，同样可在程序段结束处获得最终刀具定向。

7.2.5 端面铣削 (A4, B4, C4, A5, B5, C5)

端面铣用来加工任意弯曲的表面。



对于这种 3D 铣削类型，需要在工件表面上按行描述 3D 轨迹。

计算需要考虑到刀具类型和刀具尺寸通常在 CAM 中执行。 计算完成的 NC 程序段然后通过后置处理器读取到控制器。

轨迹曲率编程

面描述

轨迹弯曲通过面正交矢量用下列组成部分来描述：

`A4, B4, C4` 程序段开始处的起始矢量

`A5, B5, C5` 程序段结束处的结束矢量

如果在一个程序段中只有起始矢量，那么整个程序段的面正交矢量是恒定不变的。 如果在某个程序段中仅有一个结束矢量，就会通过大圆弧插补从前一个程序段的终值插补到已编程的终值。

如果编程起始矢量和结束矢量，那么在这两个方向之间同样通过大圆弧插补来插补。 因此生成连续的平滑的轨迹行程。

在基本位置中平面法向矢量指向 Z-方向，和激活的平面 G17 ～ G19 无关。

矢量的长度没有意义。

没有编程的矢量组成部分设置为零。

当激活 ORIWKS 时 (参见“定向轴的关系 (ORIWKS, ORIMKS) (页 366)”)，平面法向矢量以激活的框架为参照并且在框架旋转时一起旋转。

机床制造商

面法向矢量必须在一个可通过机床数据设置的极限值范围内垂直于轨迹切线，否则就会发出报警。

7.2.6 定向轴的关系 (ORIWKS, ORIMKS)

在通过下列方式在工件坐标系中进行方位编程时

- Euler- 以及 RPY-角，或者
- 方位矢量

可以通过 ORIMKS/ORIWKS 来设置旋转运动的轨迹。

说明

机床制造商

定向的插补类型可以由以下机床数据确定：

MD21104 $MC_ORI_IPO_WITH_G_CODE

= FALSE:以 G 代码 ORIWKS 和 ORIMKS 为基准

= TRUE:以第 51 组 G 代码为基准 (ORIAXES, ORIVECT, ORIPLANE, ...)

句法

```
ORIMKS=...
ORIWKS=...
```

含义

ORIMKS：	在机床坐标系中旋转
ORIWKS：	在工件坐标系中旋转

说明

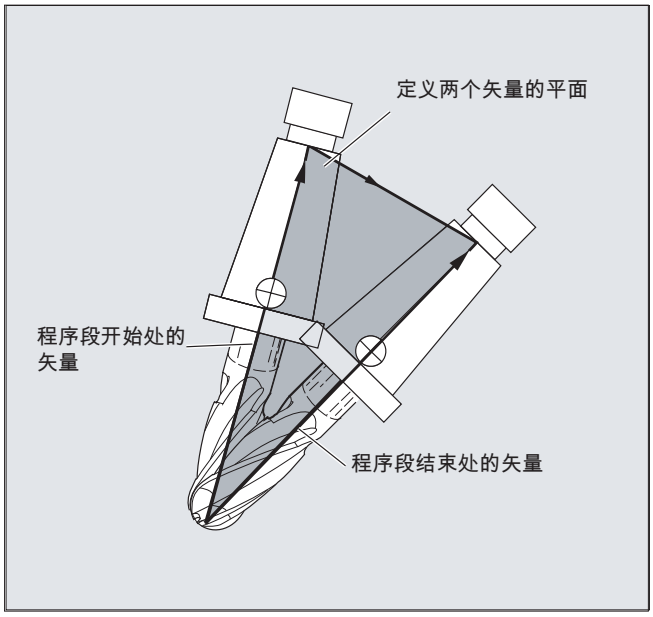
ORIWKS 是默认设置。如果某个五轴程序从开始起就不清楚应在哪个机床上执行，原则上应选择 ORIWKS 。机床实际上执行的运行取决于机床类型。

使用 ORIMKS 可以对实际的机床运动进行编程，以避免与装置等发生碰撞。

其它信息

对于 ORIMKS 而言，已执行的机床运动**取决于**机床运动机构。 用空间固定的刀尖改变方向时在回转轴位置之间进行线性插补。

对于 ORIWKS 而言，已执行的机床运动**不取决于**机床运动机构。 用空间固定的刀尖改变方向时刀具在由起始矢量和终点矢量展开的平面上运行。



单位置

说明

ORIWKS

在五轴机床单位置范围内的定向运行要求加工轴大幅度运行。 (例如，当旋转式回转头将 C 轴作为旋转轴且将 A 轴作为回转轴时，所有位置均为单值 A=0。)

机床制造商

为了不使加工轴过载，速度导向将单个位置附近的轨迹速度大幅减小。

用机床数据

$MC_TRAFO5_NON_POLE_LIMIT

$MC_TRAFO5_POLE_LIMIT

能适当设定转换参数，使得极点附近的定向运动通过极点进行设定并且可以进行流畅的加工。

单位置仅使用 MD $MC_TRAFO5_POLE_LIMIT 进行处理。

文档：
/FB3/ 功能手册特殊功能；3 至 5 轴转换（F2），
章节“单位置及其处理”。

7.2.7 定位轴编程(ORIAXES, ORIVECT, ORIEULER, ORIRPY, ORIRPY2, ORIVIRT1, ORIVIRT2)

“定向轴”功能描述的是刀具在空间内的定向，通过编程回转轴的偏移来实现。 可以通过刀具绕自身的额外旋转来达到较大的第三自由度。 这种刀具定向是在空间中任意通过一个第三回转轴实现，并需要六轴转换。刀具绕自身的旋转取决于旋转矢量的插补方式，通过旋转角 THETA 来确定（参见“刀具定向旋转(ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA) (页 379)”）。

通过轴标识符 A2, B2, C2 对定向轴编程。

句法

```
N... ORIAXES/ORIVECT                ; 线性插补或者大圆弧插补
N... G1 X Y Z A B C

N... ORIPLANE                       ; 平面的定向插补

N... ORIEULER/ORIRPY/ORIRPY2        ; 欧拉角/RPY 角定向角
N... G1 X Y Z A2= B2= C2=           ; 虚拟轴的角度编程

N... ORIVIRT1/ORIVIRT2              ; 虚拟定向轴 Def. 1/2
N... G1 X Y Z A3= B3= C3=           ; 方向矢量编程
```

说明

对于空间中沿着锥面的定向变化，还可编程其它定向轴回转轴偏移（参见“沿一个圆锥表面定向编程(ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO) (页 371)”）。

含义

ORIAXES:	加工轴或者定向轴的线性插补
ORIVECT:	大圆弧插补（和 ORIPLANE 一致）
ORIMKS: ORIWKS:	在机床坐标系中旋转 在工件坐标系中旋转 说明参见“定向轴的关系 (ORIWKS, ORIMKS) (页 366)”。
A= B= C=:	加工轴位置编程
ORIEULER:	通过欧拉角进行定向编程
ORIRPY:	通过 RPY 角进行定向编程 旋转顺序为 XYZ，此时： • A2 为围绕 X 的旋转角度 • B2 为围绕 Y 的旋转角度 • C2 为围绕 Z 的旋转角度
ORIRPY2:	通过 RPY 角进行定向编程 旋转顺序为 ZYX，此时： • A2 为围绕 Z 的旋转角度 • B2 为围绕 Y 的旋转角度 • C2 为围绕 X 的旋转角度
A2= B2= C2=:	虚拟轴的角度编程
ORIVIRT1/ORIVIRT2:	通过虚拟定向轴的定向编程 定义 1： 根据 MD21120 $MC_ORIAX_TURN_TAB_1 确定 定义 2： 根据 MD21130 $MC_ORIAX_TURN_TAB_2 确定
A3= B3= C3=:	方向轴的方向矢量编程

更多信息

机床制造商

使用 MD21102 $MC_ORI_DEF_WITH_G_CODE 确定如何对已编程的角 A2, B2, C2 进行定义：

根据 MD21100 $MC_ORIENTATION_IS_EULER（默认）进行定义，或者根据 G 代码组 50 进行定义 (ORIEULER, ORIRPY, ORIVIRT1, ORIVIRT2)。

使用 MD21104 $MC_ORI_IPO_WITH_G_CODE 来确定哪种插补方式有效：ORIWKS/ORIMKS 或 ORIAXES/ORIVECT。

运行方式 JOG

定向角在这样的运行方式下总是线性插补。在通过运行键操作的连续的，增量的运动时只能运行一个定向轴。通过手轮可以同时运行定向轴。

对于定向轴的手动运行，通道专用进给倍率开关或者快移倍率开关在快速叠加时有效。

用下列的机床数据可以有一个单独的速度说明：

MD21160 $MC_JOG_VELO_RAPID_GEO

MD21165 $MC_JOG_VELO_GEO

MD21150 $MC_JOG_VELO_RAPID_ORI

MD21155 $MC_JOG_VELO_ORI

说明

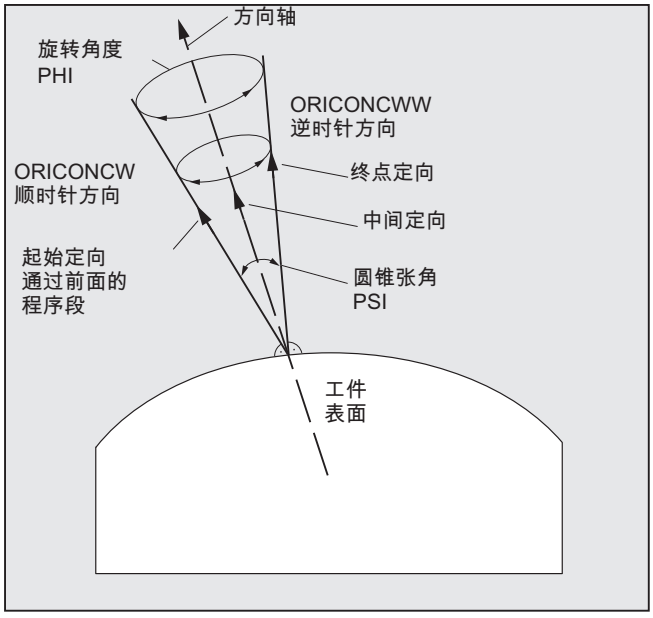
SINUMERIK 840D sl，带“转换包处理”功能

使用功能“笛卡儿手动方式”可以在 JOG 运行方式下分别在参考系统 MCS、WCS 和 TCS 中以相互独立的方式分别设置几何轴转换。

文档：
功能手册 扩展功能；运动转换（M1）

7.2.8 沿一个圆锥表面定向编程(ORIPLANE, ORICONCW, ORICONCCW, ORICONTO, ORICONIO)

用扩展的定向可以沿位于空间的圆锥表面执行方向改变。 在一个圆锥表面定向矢量插补可以用模态指令 ORICONxx 实现。 要在一个平面内插补，可以用 ORIPLANE 对结束定向编程。 通常起始定向通过前面的程序段确定。



编程

结束定向可以用 A2, B2, C2 以欧拉角或者 RPY 角编程角度数据，或者用 A,B,C 编程回转轴位置来确定。 对于沿圆锥表面的定向轴，必须有其它的编程数据：

- 锥体的旋转轴作为带有 A6, B6, C6 的矢量
- 张角 PSI 带有标识符 槽
- 用 A7, B7, C7 在圆锥面中间定向

说明

为圆锥的旋转轴编程方向矢量 A6, B6, C6

不一定需要对结束定向编程。 如果没有给出结束定向的数据，则整个圆锥面用 360 度插补。

用 NUT（槽）=角度编程圆锥张角

必须有结束定向的参数说明。

360 度的整圆锥面不能以这种方式插补。

在圆锥面中编程中间定向 A7,B7,C7

必须有结束定向的参数说明。 定向改变和旋转方向只能通过三个矢量来确定，即：起始定向、结束定向和中间定向。 因此三个矢量必须不同。 如果编程的中间定向平行于起始定向或结束定向，那么必须在起始矢量和结束矢量构成的平面上进行定向的直线大圆弧插补。

圆锥表面的扩展定向插补

N... ORICONCW 或 ORICONCCW	在圆锥面上**插补** 用
N... A6= B6= C6= A3= B3= C3=	圆锥的顺时针/逆时针方向矢量和结束定向
或者	或者
N... ORICONTO	切向过渡和
N... G1 X Y Z A6= B6= C6=	结束定向的参数说明
或者	或者
N... ORICONIO	结束定向的参数说明和
N... G1 X Y Z A7= B7= C7=	圆锥表面的中间定向用
N... PO[PHI]=(a2, a3, a4, a5)	旋转角的多项式和
N... PO[PSI]=(b2, b3, b4, b5)	张角多项式

参数

ORIPLANE:	平面上的插补（大圆弧插补）
ORICONCW:	顺时针方向圆锥表面上的插补
ORICONCCW:	逆时针方向圆锥表面上的插补
ORICONTO:	在切线过渡的圆锥表面上的插补
A6= B6= C6=:	圆锥的旋转轴的编程（标准化的矢量）
NUT（槽）=角度：	圆锥张角用度表示
NUT（槽）=+179:	运行角度小于或等于 180 度
NUT（槽）=-181:	运行角度大于或等于 180 度
ORICONIO:	在圆锥表面插补
A7= B7= C7=:	中间定向（作为标准矢量编程）
PHI:	绕圆锥方向轴定向的旋转角
PSI:	圆锥张角
可能的多项式 PO[PHI]=(a2, a3, a4, a5) PO[PSI]=(b2, b3, b4, b5)	除了每个角度外，多项式也最多只能编程为 5 次多项式。

示例： 不同的方向变化

程序代码	注释
…	
N10 G1 X0 Y0 F5000	
N20 TRAORI(1)	；定向转换开。
N30 ORIVECT	；刀具定向插补为矢量。
…	；在平面中定向刀具。
N40 ORIPLANE	；选择大圆弧插补。
N50 A3=0 B3=0 C3=1	
N60 A3=0 B3=1 C3=1	；在 Y/Z 平面上定向绕 45 度旋转，在程序段结尾达到定向(0,1/√2,1/√2)。
…	
N70 ORICONCW	；在圆锥表面定向编程：
N80 A6=0 B6=0 C6=1 A3=0 B3=0 C3=1	；在圆锥表面上以顺时针方向从 (0,0,1) 到定向 (1/√2,0,1/√2)插补定向矢量，旋转角度达到 270 度。
N90 A6=0 B6=0 C6=1	；刀具定向在同一个圆锥表面上连续转了一整圈。

其它信息

如果要在空间任意存在的一个圆锥表面上描述定向改变，则刀具定向所绕的矢量必须已知。此外，还必须预先确定起始定向和结束定向。起始定向由前一程序段得出，而结束定向必须被编程或通过其它条件确定。

在平面 ORIPLANE 中按照 ORIVECT 编程

编程大圆弧插补和角度多项式必须符合轮廓的线性插补和多项式插补。刀具定向在起始定向和结束定向构成的平面中进行插补。如果附加编程多项式，则定向矢量也可以从平面中翻转。

在平面 G2/G3，CIP 和 CT 中编程圆

扩展定向符合一个平面中圆的插补。关于相应的用圆心数据或半径数据如 G2/G3 编程圆，圆通过中间点 CIP 和切向圆 CT 参见

文档：编程手册之基本原理，“编程行程指令”。

定向编程

在圆锥表面 ORICONxx 编程定向矢量插补

为了在圆锥表面定向插补，可以从 G 代码组 51 中选择四种不同的插补方式：

1. 在圆锥表面顺时针插补 ORICONCW 运用结束插补、圆锥方向或者张角数据。方向矢量用标识符 A6, B6, C6，圆锥张角用标识符 NUT（槽）= 取值范围在区间 0 到 180 度来编程。
2. 在圆锥表面逆时针插补 ORICONCWW 运用结束插补、圆锥方向或者张角数据。方向矢量用标识符 A6, B6, C6，圆锥张角用标识符 NUT（槽）= 取值范围在区间 0 到 180 度来编程。
3. 在圆锥表面插补 ORICONIO 用结束定向数据和以标识符 A7, B7, C7 编程的中间定向数据。
4. 在圆锥表面插补 ORICONTO 用切向过渡和结束定向数据。方向矢量用标识符 A6, B6, C6 来编程。

7.2.9 两个接触点的定向预设值(ORICURVE, PO[XH]=, PO[YH]=, PO[ZH]=)

通过第二个空间曲线编程定向改变 ORICURVE

编程定向改变的另一个方法是，除了刀尖沿着一个空间曲线外，第二个刀具接触点的运动要用 ORICURVE 来编程。因此如同编程刀具矢量本身，可以清楚地确定刀具定向改变。

机床制造商

请注意机床制造商对通过机床数据设置的用于刀具第 2 定向轨迹编程的轴标识符的提示。

编程

在这种插补方式下，对于这两个空间曲线用 G1 对点或者 POLY 对多项式进行编程。圆和渐开线不允许。此外，可以用 BSPLINE 激活样条插补和功能“合并较短样条程序段”。

文档:
功能手册 基本功能；连续路径运行，准停，预读 (B1)，章节:合并较短样条程序段

不允许其他的样条方式 ASPLINE 和 CSPLINE 以及用 COMPON, COMPCURV 或者 COMPCAD 激活压缩程序。

为坐标系进行定向多项式编程时，刀具两个接触点的运动可以最多可以设为 5 次。

用附加的空间曲线和多项式为坐标系扩展定向插补

```
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)
```

刀具两个接触点的运动数据和各坐标系的附加多项式

参数

ORICURVE	带刀具两个接触点运行说明的定向插补
XH YH ZH	附加轮廓的刀具两个接触点坐标系的标识符作为空间曲线
可能的多项式 PO[XH]=(xe, x2, x3, x4, x5) PO[YH]=(ye, y2, y3, y4, y5) PO[ZH]=(ze, z2, z3, z4, z5)	除了各终点外，空间曲线还可以用多项式编程。
xe, ye, ze	空间曲线终点
xi, yi, zi	多项式系数最多 5 次

说明

标识符 XH YH ZH 用于第 2 定向轨迹的编程

选择的标识符必须与其它直线轴

X Y Z 轴

和回转轴如

A2 B2 C2 欧拉角或者 RPY 角

A3 B3 C3 方向矢量

A4 B4 C4 或者 A5 B5 C5 平面法向矢量

A6 B6 C6 旋转矢量或者 A7 B7 C7 中间点坐标

或者其它插补参数没有冲突。

7.3 定向多项式(PO[角度], PO[坐标])

进行三轴到五轴转换时，与 G 代码组 1 的哪个多项式恰好激活无关，两种不同类型的定向多项式最多只能编程到 5 次。

1. 多项式用于**角度:**超前角 LEAD、侧向角 TILT
   与起始定向与结束定向构成的平面有关。
2. 多项式用于**坐标:**两个空间曲线的 XH, YH, ZH 用于刀具上参考点的刀具定向。

六轴转换时为了刀具定向，另外还可以用最多 5 次的多项式为刀具旋转编程旋转矢量 THT 的旋转。

句法

定向多项式类型 1 用于**角度**

```
N… PO[PHI]=(a2, a3, a4, a5)
N… PO[PSI]=(b2, b3, b4, b5)
```

三轴至五轴转换

定向多项式类型 2 用于**坐标**

```
N… PO[XH]=(xe, x2, x3, x4, x5)
N… PO[YH]=(ye, y2, y3, y4, y5)
N… PO[ZH]=(ze, z2, z3, z4, z5)
```

两个用于刀具定向的定向轨迹的坐标符

另外，在两种情况下可以编程一个用于**旋转**的多项式在六轴转换时用

```
N… PO[THT]=(c2, c3, c4, c5)
```

轨迹相对的旋转插补

或者

```
N… PO[THT]=(d2, d3, d4, d5)
```

与定向矢量改变的绝对插补

相对插补和切向插补。如果该转换支持一个可用旋转角度 THETA 编程和插补的偏移，则上述编程可以实现。

含义

PO[PHI]	在起始定向和结束定向间的平面上的角度
PO[PSI]	描述起始定向和结束定向间的平面的定向倾斜角度
PO[THT]	用 THETA 编程的组 54 的 G 代码中的一个的旋转矢量旋转而成的旋转角
PHI	超前角 LEAD
PSI	侧面角 TILT
THETA	绕沿 Z 的刀具方向旋转
PO[XH]	刀具参考点的 X 坐标
PO[YH]	刀具参考点的 Y 坐标
PO[ZH]	刀具参考点的 Z 坐标

其它信息

不能编程定向多项式

- 如果样条插补 ASPLINE, BSPLINE, CSPLINE 有效。
  用于定向角的多项式类型 1 对于除样条外的每个插补方式都可能，即对于带快速行程G00 或者带进给 G01 的线性插补
  对于带 POLY 的多项式插补和
  带 G02, G03, CIP, CT, INVCW 和 INCCCW 的圆插补或者
  渐开线插补。
  与次相反，用于定向坐标的多项式类型 2，仅当
  带快速行程 G00 或带进给 G01 的线性插补或者
  当带 POLY 的多项式插补有效时才可能。
- 当定向插补通过轴插补 ORIAXES。 这种情况下，可以用 PO[A]和 PO[B]直接编程多项式用于轴定向 A 和 B。

带 ORIVECT、ORIPLANE 和 RICONxx 的定向多项式类型 1

对于带 ORIVECT、ORIPLANE 和 ORICONxx 的大圆弧插补和圆锥面插补，只有定向多项式类型 1 可以。

带 ORICURVE 的多项式类型 2

如果带额外空间曲线 ORICURVE 的插补有效，插补定向矢量的直角坐标分量且仅定向多项式类型 2 有效。

7.4 刀具定向旋转(ORIROTA, ORIROTR, ORIROTT, ORIROTC, THETA)

如果在带运动刀具的机床类型中，要求刀具的方向也可以改变，那么每个程序段均要编程结束方向。 取决于机床类型可以编程定向轴的运动方向或者编程方向矢量 THETA 的旋转方向。 对于这些旋转矢量可以编程不同的插补类型：

- ORIROTA: 规定的绝对旋转方向的旋转角度。
- ORIROTR: 相对于起始方向和结束方向平面的旋转角度。
- ORIROTT: 相对于方向矢量改变的旋转角度。
- ORIROTC: 轨迹切线的切向旋转角。

句法

仅当插补类型 ORIROTA 已激活时，才可以按照下列四种方式对旋转角或者旋转矢量进行编程：

1. 直接旋转轴位置 A, B, C
2. 欧拉角（单位：度），通过 A2, B2, C2
3. RPY-角（单位：度），通过 A2, B2, C2
4. 方向矢量，通过 A3, B3, C3 (旋转角通过 THETA=<值>)

如果 ORIOTR 或者 ORIOTT 已激活，则仅可直接使用 THETA 编程旋转角。

在没有定向变化的情况下，也可以单独在某个程序段对旋转进行编程。 此时 ORIROTR 和 ORIROTT 没有意义。 在这种情况下，始终参照绝对方向对旋转角进行插补（ORIROTA）。

```
N... ORIROTA                          确定旋转矢量的插补
N... ORIROTR
N... ORIROTT
N... ORIROTC
N... A3= B3= C3= THETA=<值>           确定定向矢量的旋转
N... PO[THT]=(d2, d3, d4, d5)         用 5 级多项式插补旋转角度
```

含义

ORIROTA:	规定的绝对旋转方向的旋转角度
ORIROTR:	相对于平面在起始方向和结束方向之间的旋转角度

ORIROTT:	旋转角度作为到定向改变的切向旋转矢量
ORIROTC:	旋转角度作为到轨迹切线的切向旋转矢量
THETA：	方向矢量的旋转
THETA=<值>：	以度表示的旋转角度，这个角度在程序段结束时达到
THETA=Θe：	旋转角，带有旋转矢量的终止角 Θ_e
THETA=AC(…)：	程序段方式转换到绝对尺寸说明
THETA=AC(…)：	程序段方式转换到相对尺寸说明
Θe：	旋转矢量的结束角度，绝对的用 G90，相对的用 G91（相对尺寸）均有效。
PO[THT]=(....)：	旋转角度的多项式

示例： 方向的旋转

程序代码	注释
N10 TRAORI	; 激活方位转换
N20 G1 X0 Y0 Z0 F5000	; 刀具定向
N30 A3=0 B3=0 C3=1 THETA=0	; 在 Z 方向旋转角度 0
N40 A3=1 B3=0 C3=0 THETA=90	; 在 X 方向旋转 90 度
N50 A3=0 B3=1 C3=0 PO[THT]=(180,90)	; 定向
N60 A3=0 B3=1 C3=0 THETA=IC(-90)	; 在 Y 方向旋转到 180 度
N70 ORIROTT	; 保持不变且旋转到 90 度
N80 A3=1 B3=0 C3=0 THETA=30	; 旋转角度与定向改变相对
	;旋转矢量与 X-Y 平面为 30 度

插补程序段 N40 时，旋转角度从起始值 0 度到结束值 90 度线性插补。 在程序段 N50 中，旋转角度从 90 度变为 180 度，按照抛物线 θ(u) = +90u2。 在 N60 中，无需改变定向也可以旋转。

当 N80 时，刀具定向从 Y 方向转到 X 方向。 从而定向改变在 X-Y 平面中，且旋转角度与该平面构成了 30 度。

其它信息

ORIROTA

旋转角 THETA 参照空间中的绝对设定方向进行插补。 基本旋转方向通过机床数据生成。

ORIROTR

旋转角 THETA 相对于由起始和终点方位所定义的平面进行解析。

ORIROTT

旋转角 THETA 相对于方位变化进行解析。 如果 THETA=0 ，旋转矢量以相对于方位变化的切向进行插补，并且只有当至少给方位编程了一个“PSI 倾斜角”多项式时，才会区别于 ORIROTR。 因此生成一个不在平面上运行的方向改变。 通过一个附加编程的旋转角 THETA ，就可以例如对旋转矢量进行适当插补，使其始终形成某个相对于方位变化的值。

ORIROTC

旋转矢量与轨迹切线相对用一个通过角度 THETA 编程的偏移来插补。 对于角度偏移也可以编程一个多项式 PO[THT]=(c2, c3, c4, c5) ，最多 5 次。

7.5 与轨迹相对的定向

7.5.1 定向方式相对于轨迹

使用扩展功能，不止在程序段结束也可以通过整个轨迹变化达到相对定向。在前面程序段达到的定向通过大圆弧插补转到编程的结束定向。原则上，有两种编程与轨迹相对的定向的方法：

1. 刀具定向和刀具旋转用 ORIPATH、ORPATHTS 相对于轨迹插补。
2. 按当前普遍的方式编程和插补定向矢量。用 ORIROTC 创建与轨迹切线相对的定向矢量旋转。

句法

对定向和刀具旋转的插补方式进行编程时用：

`N...ORIPATH`	与轨迹相对的定向
`N...ORIPATHS`	轨迹相对的定向带有定向变化平整
`N...ORIROTC`	与轨迹相对的旋转矢量插补

由轨迹变化中的角产生的定向拐点可以用 `ORIPATHS` 来平整。退刀运动的方向和行程长度可以通过带分量 `A8=X, B8=Y C8=Z` 的矢量来进行编程。

用 `ORIPATH/ORIPATHS` 能够使与轨迹切线相对的不同参照通过三个角度

- `LEAD=` 前置角参照于轨迹和表面的参数
- `TILT=` 侧向角参照于轨迹和表面的参数
- `THETA=` 旋转角度

为整个轨迹变换来编程。对于旋转角度 `THETA` 可以用 `PO[THT]=(...)` 额外编程最多 5 次的多项式。

说明

机床制造商

请注意机床制造商说明。通过设计的机床数据和设定数据，可以对与轨迹相对的定向方式进行其他设置。进一步说明参见

文档：
/FB3/功能手册 特殊功能；3 至 5 轴转换（F2）
章节“定向”

含义

角度 LEAD 和 TILT 的插补可以通过机床数据进行不同设置：

- 在整个程序段中将始终保持以 LEAD 和 TILT 编程的刀具定向的参考量。
- 超前角 LEAD ：绕与切线和平面垂线矢量 TILT 垂直的方向旋转:绕平面垂线矢量旋转定向。
- 超前角 LEAD ：绕与切线和平面垂线矢量侧向角 TILT 垂直的方向旋转:绕轨迹切线方向旋转定向。
- 旋转角度 THETA=：六轴转换时，刀具绕自身旋转且具有一个附加的第三回转轴作为定向轴。

说明

不允许轨迹相关的定向与 OSC、OSS、OSSE、OSD 和 OST 一起

与轨迹相对的定向插补 ORIPATH 或 ORIPATHS 与 ORIOTC 不能与用组 34 中某个 G 代码平整的定向变化一起编程。为此 OSOF 必须激活。

7.5.2 轨迹相关的刀具定向旋转（ORIPATH、ORIPATHS、旋转角）

六轴转换时为了在空间中任意定向刀具，也可以用一个第三回转轴使刀具绕自身旋转。 对于与轨迹相对的带 ORIPATH 或者 ORIPATHS 的刀具定向旋转，可以通过旋转角 THETA 编程额外的旋转。 也可以选择通过一个与刀具方向垂直的平面中的矢量来编程角度 LEAD 和 TILT。

机床制造商

请注意机床制造商说明。 通过机床数据可以设置角度 LEAD 和 TILT 的不同插补。

句法

刀具定向旋转和刀具旋转

用 ORIPATH 或者 ORIPATHS 激活与轨迹相对的刀具定向方式。

N... ORIPATH	激活参照于轨迹的定向方式
N... ORIPATHS	激活参照于轨迹的定向方式，带定向变化平滑

用旋转作用激活三种可能的角度：

N... LEAD=	编程的与平面法线矢量相对的定向角度

N... TILT=	用于编程定向的角度，该定向在与平面法线矢量相对的轨迹切线垂直的平面中
N... THETA=	旋转角度相对于定向改变，绕第三回转轴的刀具方向

用 LEAD=值、 TILT=值或 THETA=值编程程序末端的角度值。 除恒定的角度外，可以对全部三个角度编程最多 5 次多项式。

N... PO[PHI]=(a2, a3, a4, a5)	提前角多项式 LEAD
N... PO[PSI]=(b2, b3, b4, b5)	侧向角多项式 TILT
N... PO[THT]=(d2, d3, d4, d5)	旋转角多项式 THETA

编程时，可以去掉为零的较高的多项式系数。 举例 PO[PHI]=a2 为导角 LEAD 得出了一个抛物线。

含义

与轨迹相对的刀具定向

ORIPATH:	刀具定向参照于轨迹
ORIPATHS:	刀具定向和轨迹有关，会对定向运行中的折点进行平滑
LEAD:	相对于面正交矢量的角度，在由轨迹切线和面正交矢量展开的平面上
TILT:	绕 Z 方向定向旋转或绕轨迹切线旋转
THETA:	绕沿 Z 的刀具方向旋转
PO[PHI]:	用于导角 LEAD 的定向多项式
PO[PSI]:	用于侧向角 TILT 的定向多项式
PO[THT]:	用于旋转角 THETA 的定向多项式

说明

旋转角度 THETA

对于用第三回转轴作为定向轴的刀具绕自身的旋转，要求有一个六轴转换。

7.5.3 轨迹相关的刀具旋转插补（ORIROTC，THETA）

用旋转矢量插补

对于用 ORIROTC 编程的相对于轨迹切线的刀具旋转,旋转矢量也可用一个可通过旋转角 THETA 编程的偏移量来插补。 为此，可以用 PO[THT]为该偏移角编程一个最多 5 次的多项式。

句法

`N... ORIROTC`	创建一个相对于轨迹切线的刀具旋转
`N... A3= B3= C3= THETA=值`	确定定向矢量的旋转
`N... A3= B3= C3= PO[THT]=(c2, c3, c4, c5)`	用最多 5 次的多项式插补偏移角

在没有定向变化的情况下，也可以单独在某个程序段对旋转进行编程。

含义

六轴转换时，与轨迹相对的刀具旋转插补

`ORIROTC`:	创建到轨迹切线的切向旋转矢量
`THETA=值`:	以度表示的旋转角度，这个角度在程序段结束时达到
`THETA=θe`:	旋转角，带有旋转矢量的终止角 Θ_e
`THETA=AC(…)`:	程序段方式转换到绝对尺寸说明
`THETA=IC(…)`:	程序段方式转换到相对尺寸说明
`PO[THT]=(c2, c3, c4, c5)`:	用 5 次多项式插补偏移角

说明

旋转矢量 ORIROTC 的插补

如果与刀具定向方向相反，也要创建一个相对于轨迹切线的刀具旋转，那么仅当六轴转换时可能。

激活 ORIROTC 时

不能编程旋转矢量 ORIROTA 。 在编程的情况下，输出报警 14128“ORIROTC 激活时刀具旋转绝对编程”。

三轴至五轴转换时刀具定位方向

可以象三轴至五轴转换时，通常用欧拉角或 RPY 角或者方向矢量来编程刀具定向方向。可以用大圆弧插补 ORIVECT、定向轴 ORIAXES 直线插补、所有圆锥面上的插补 ORICONxx 以及带刀具两个接触点空间曲线的额外插补来编程空间中的刀具定向改变。

G....:	指定旋转轴的运动方式
X, Y, Z:	指定线性轴
ORIAXES:	加工轴或者定向轴的线性插补
ORIVECT:	大圆弧插补（和 ORIPLANE 一致）
ORIMKS: ORIWKS:	在机床坐标系中旋转 在工件坐标系中旋转 说明请参阅刀具定位的旋转章节
A= B= C=:	加工轴位置编程
ORIEULER:	通过欧拉角进行定向编程
ORIRPY:	通过 RPY 角进行定向编程
A2= B2= C2=:	虚拟轴的角度编程
ORIVIRT1: ORIVIRT2:	通过虚拟定向轴的定向编程 (定义 1)，根据 MD $MC_ORIAX_TURN_TAB_1 确定 (定义 2), 根据 MD $MC_ORIAX_TURN_TAB_2 确定
A3= B3= C3=:	方向轴的方向矢量编程
ORIPLANE:	平面上的插补（大圆弧插补）
ORICONCW:	顺时针方向圆锥表面上的插补
ORICONCCW:	逆时针方向圆锥表面上的插补
ORICONTO:	在切线过渡的圆锥表面上的插补

A6= B6= C6=:	圆锥的旋转轴的编程（标准化的矢量）
NUT（槽）=角度	圆锥张角用度表示
NUT（槽）=+179	运行角度小于或等于 180 度
NUT（槽）=-181	运行角度大于或等于 180 度
ORICONIO:	在圆锥表面插补
A7= B7= C7=:	中间定向（作为标准矢量编程）
ORICURVE XH YH ZH 例如带有多项式 PO[XH]=(xe, x2, x3, x4, x5)	带刀具两个接触点运行说明的定向插补 除了相应的终点之外，其它空间曲线多项式也可以编程。

说明

如果用激活的 ORIAXES 通过定向轴对刀具定向进行插补，则只能在程序结束创建与轨迹相对的旋转角。

7.5.4 平滑定向变化(ORIPATHS A8=, B8=, C8=)

当在轮廓上以恒定加速度改变定向时，不希望出现轨迹运动中断，尤其是在轮廓拐角处。可以通过插入一个自身中间程序段来平整在定向变化中由此而产生的拐点。 此后定向改变加速度恒定，如果在换向时 ORIPATHS 也激活了。 在该阶段执行一个刀具退刀。

机床制造商

请注意机床制造商关于激活该功能的预定义机床数据和设定数据的说明。

如何解释退刀矢量可以通过机床数据设置：

1. 在刀具坐标系中，通过刀具方向定义 Z 坐标。
2. 在工件坐标系中，通过有效平面定义 Z 坐标。

关于功能“轨迹相对的定向”的进一步说明参见

文档：

功能手册 特殊功能；多轴转换 (F2)

句法

对于以整个轨迹为参照的恒定的刀具定向，在轮廓拐角处要求有进一步的编程数据。 该运动的方向和行程长度可以通过带分量 A8=X、B8=Y、 C8=Z 的矢量来进行编程。

```
N... ORIPATHS A8=X B8=Y C8=Z
```

含义

ORIPATHS:	刀具定向参照这个轨迹，拐点在方向变化中被平滑。
A8= B8= C8=:	用于方向和行程长度的矢量分量
X, Y, Z:	刀具方向上的退刀运动

说明

编程方向矢量 A8, B8, C8

如果该矢量长度为零，则无法进行退刀。

ORIPATHS

用 ORIPATHS 激活以轨迹为参照的刀具定向。 否则，定向通过线性大圆弧插补从起始定向变为结束定向。

7.6 定向压缩 (COMPON, COMPCURV, COMPCAD, COMPSURF)

包含了生效的方向转换 (TRAORI) 和任意一种刀具定向的 NC 程序，在满足规定公差的前提下，可以使用这些指令加以压缩。

编程

刀具定向

如果一个定向转换 (TRAORI) 处于激活状态，对于 5 轴机床可以如下措施（与运动无关）对刀具定向编程：

- 通过以下参数对方向**矢量**编程：
  `A3=<...> B3=<...> C3=<...>`
- 通过以下参数编程欧拉**角**或者 RPY **角**：
  `A2=<...> B2=<...> C2=<...>`

刀具旋转

如果是 **6 轴** 机床，除了刀具定向之外，还可以对刀具的旋转进行编程。

通过以下参数实现旋转角编程：

`THETA=<...>`

参见“刀具定向旋转 (页 379)”。

说明

在其中已经附带编程有一个旋转的 NC 程序段只有在旋转角以**线性**变化时才可以被压缩。也就是说，不允许使用 PO[THT]=(...) 给旋转角编程多项式。

可压缩 NC 程序段的通常形式

因此可压缩 NC 程序段的通常形式为：

```
N...X=<...> Y=<...> Z=<...> A3=<...> B3=<...> C3=<...> THETA=<...> F=<...>
```

或者

```
N...X=<...> Y=<...> Z=<...> A2=<...> B2=<...> C2=<...> THETA=<...> F=<...>
```

说明

位置值可以直接标注（如：X90）或者通过参数赋值（例如 X=R1*(R2+R3)）间接标注。

通过回转轴位置进行刀具定向编程

刀具定向也可以通过回转轴位置来说明时，例如形式为：

```
N...X=<...> Y=<...> Z=<...> A=<...> B=<...> C=<...> THETA=<...> F=<...>
```

在这种情况下以两种不同的方式进行压缩，它由是否进行大圆弧插补而定。在没有大圆弧插补的情况下，一般通过回转轴的轴向多项式来描述已压缩的定向变化。

轮廓精度

根据设定的压缩模式（MD20482 $MC_COMPRESSOR_MODE），对于几何轴和定向轴，在压缩时要么设计的轴专用的公差（MD33100 $MA_COMPRESS_POS_TOL）有效，要么下列可通过设置数据设定的通道专用的公差有效：

SD42475 $SC_COMPRESS_CONTUR_TOL（最大轮廓偏差）

SD42476 $SC_COMPRESS_ORI_TOL（刀具定向最大角度偏差）

SD42477 $SC_COMPRESS_ORI_ROT_TOL（用于刀具旋转角的最大角度偏差）（仅对于 6 轴机床可用）

文档：
功能手册 基本功能；3 至 5 轴转换（F2），
章节：“定向压缩”

激活/取消激活

通过模态 G 指令 COMPON、COMPCURV、COMPCAD 或 COMPSURF 激活压缩器功能。

通过 COMPOF 退出压缩器功能。

参见“NC 程序段压缩（COMPON，COMPCURV，COMPCAD） (页 285)”。

说明

仅当大圆弧插补已激活时，才会对定向运动进行压缩（也就是说，在从起点和终点展开的平面中改变刀具定向）。

大圆弧插补在以下条件下进行：

- MD21104 $MC_ORI_IPO_WITH_G_CODE = 0，
  ORIWKS 已激活且
  通过矢量对定向编程（通过 A3、B3、C3 或者 A2、B2、C2）。
- MD21104 $MC_ORI_IPO_WITH_G_CODE = 1 且
  ORIVECT 或 ORIPLANE 激活。
  刀具方向可以作为方向矢量编程，或者使用回转轴位置编程。如果 G 代码 ORICONxx 或者 ORICURVE 中一个激活或者定向角多项式（PO[PHI] 和 PO[PSI]）已编程，则不进行大圆弧插补。

示例

在下面的程序示例中，压缩一条用一个多边形轮廓逼近的圆弧。这里刀具方向与一个圆锥面同步。尽管几个编程的方向改变并不恒定，但是压缩器功能却生成一个平滑的曲线。

编程	注释
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	; 轮廓的最大偏差= 0.05 mm
$SC_COMPRESS_ORI_TOL=5	; 最大定向偏差 = 5 度
TRAORI	
COMPCURV	; 跟踪一个从多边形形成的圆。此时在一个围绕 Z、张角为 45 度的圆锥上作定向运动。
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	

7.7 激活/取消定向曲线的平滑 (ORISON, ORISOF)

“定向曲线的平滑”通过 G 指令组 61 的指令在零件程序中进行激活/取消。指令模态生效。

前提条件

- 带 5/6 轴转换的系统
- 压缩器功能 COMPCAD 生效。

句法

```
ORISON
...
ORISOF
```

含义

ORISON:	激活定向曲线的平滑
ORISOF:	取消定向曲线的平滑

示例

程序代码	注释
...	
TRAORI()	; 启用定向转换。
COMPCAD	; 激活压缩器功能 COMPCAD。
ORISON	; 启用定向平滑。
$SC_ORISON_TOL=1.0	; 刀具定向的最大角度差 = 1.0 度。
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	

程序代码	注释
...	
ORISOF	；关闭定向平滑。
...	

轴方向在 XZ 平面内偏转 90 度，即从 -45 度到 +45 度。通过定向曲线的平滑，轴方向不再达到最大角度值 - 45 度或 +45 度。

7.8 运动变换

7.8.1 激活端面转换 (TRANSMIT)

端面转换（TRANSMIT）是在零件程序或同步动作中通过指令 TRANSMIT 激活的。

句法

```
TRANSMIT
TRANSMIT(<n>)
```

含义

TRANSMIT:	激活含第一个 TRANSMIT 数据组的 TRANSMIT 功能
TRANSMIT(n):	激活含第<n>个 TRANSMIT 数据组的 TRANSMIT 功能

说明

一个在通道中生效的转换 TRANSMIT 是通过以下指令取消的：

- 取消转换 TRAFOOF
- 激活其他转换：例如 TRACYL、TRAANG、TRAORI

7.8.2 激活柱面转换 (TRACYL)

柱面转换（TRACYL）是在零件程序或同步动作中通过指令 TRACYL 激活的。

句法

```
TRACYL(<d>)
TRACYL(<d>,<n>)
TRACYL(<d>,<n>,<k>)
```

含义

TRACYL(<d>):	激活含第一个 TRACYL 数据组和加工直径<d>的 TRACYL 功能	
TRACYL (<d>,<n>):	激活含第<n>个 TRACYL 数据组和加工直径<d>的 TRACYL 功能	
<d>:	基准直径或加工直径 该值必须大于 1。	
<n>:	TRACYL 数据组编号（可选）	
	取值范围：	1, 2
<k>:	参数<k>只与转换类型 514 相关	
	k = 0:	无槽壁补偿
	k = 1:	带槽壁补偿
	如未给定参数，则设定的基本设置生效： $MC_TRACYL_DEFAULT_MODE_<n> 其中<n> = TRACYL 数据组编号	

说明

一个在通道中生效的转换 TRACYL 是通过以下指令取消的：

- 取消转换 TRAFOOF
- 激活其他转换：例如 TRAANG、TRANSMIT、TRAORI

示例

程序代码	注释
...	
N40 TRACYL(40.)	；激活含第一个 TRACYL 数据组和加工直径 40 mm 的 TRACYL 功能。
...	

其他信息

程序结构

使用 TRACYL 转换 513（带槽壁补偿的 TRACYL）进行铣槽加工的零件程序通常由以下几个工步组成：

1. 选择刀具。
2. 选择 TRACYL。

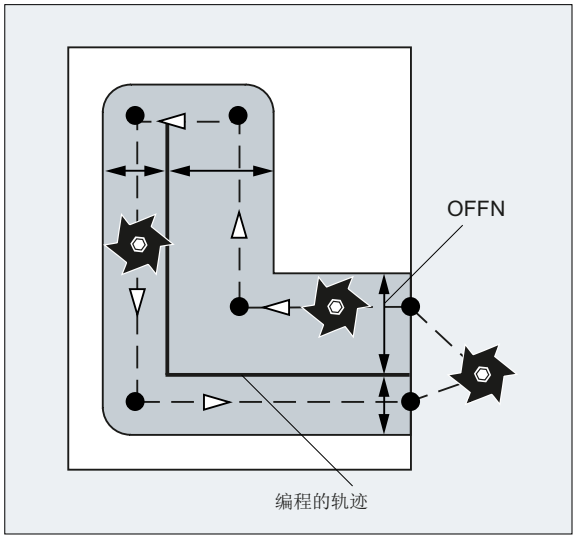
3. 选择合适的坐标偏移（框架）。
4. 定位。
5. 编程 OFFN。
6. 选择 TRC。
7. 返回程序段（运行到 TRC 并且返回槽壁）。
8. 槽中心线的轮廓。
9. 取消 TRC。
10. 退刀程序段（离开 TRC 并且离开槽壁）。
11. 定位。
12. TRAFOOF。
13. 再次选择原始的坐标偏移（框架）。

轮廓偏移(OFFN)

为了使用 TRACYL 转换 513 进行铣槽，在零件程序中对槽的中心线进行编程并通过地址 OFFN 对**槽宽的一半**进行编程。

为避免损伤槽壁，OFFN 只有在选中刀具半径补偿时才生效。

可以在零件程序内更改 OFFN 。因此实际的槽中心线可以偏离中心：



说明

OFFN 至少应与刀具半径相等，以避免损伤对面的槽壁。

说明

OFFN 与 TRACYL 配合使用的作用与没有 TRACYL 不同。因为当 TRC 激活时， OFFN 也可在没有 TRACYL 的情况下被考虑在内，所以 OFFN 应在 TRAFOOF 之后重新复位为零。

注意
OFFN 的作用取决于转换类型 使用 TRACYL 转换 513（带槽壁补偿的 TRACYL）时会为 OFFN 编程一半的槽宽。 使用 TRACYL 转换 512（无槽壁补偿的 TRACYL）时，相反 OFFN 的值会作为 TRC 的余量。

刀具半径补偿(TRC)

使用 TRACYL 转换 513 时，TRC 并不是相对于槽壁，而是相对于所编程的槽中心来计算的。为了使刀具在槽壁左侧行驶，应使用指令 G42 代替 G41 进行编程或使用负号为 OFFN 赋值。

刀具直径

如果使用 TRACYL 和直径小于槽宽的刀具，则不会生成与使用直径与槽宽相同的刀具时同样的槽壁几何数据。为提高精度，建议选择直径略小于槽宽的刀具。

轴使用

说明

下列轴不可以作为定位轴或者摆动轴使用：

- 几何轴沿圆柱表面（Y 轴）的圆周方向
- 附加的线性轴在槽壁补偿时（Z 轴）

7.8.3 激活可编程角度的斜角转换 (TRAANG)

可编程角度的斜角转换是在零件程序或同步动作中通过指令 TRAANG 激活的。

句法

```
TRAANG
TRAANG()
TRAANG(, <n>)
TRAANG(<α>)
TRAANG(<α>,<n>)
```

含义

TRAANG: TRAANG():	激活带有第一个 TRAANG 数据组和最后生效的角度<α>的 TRAANG 功能	
TRAANG(, <n>):	激活带有第<n>个 TRAANG 数据组和最后生效的角度<α>的 TRAANG 功能	
TRAANG(<α>):	激活带有第一个 TRAANG 数据组和角度<α>的 TRAANG 功能	
TRAANG(<α>,<n>):	激活带有第<n>个 TRAANG 数据组和角度<α>的 TRAANG 功能	
<α>:	斜置轴的角度(可选)	
	取值范围：	-90° < α < + 90°
	如未指定角度，则在机床数据中设置的基本设置生效： MD2xxxx $MC_TRAANG_ANGLE_<n>	
<n>:	TRAANG 数据组编号（可选）	
	取值范围：	1, 2

说明

一个在通道中生效的斜角转换 TRAANG 是通过以下指令取消的：

- 取消转换 TRAFOOF
- 激活其他转换：例如 TRACYL、TRANSMIT、TRAORI

示例

程序代码	注释
N20 TRAANG(45)	；激活带有第一个 TRAANG 数据组和 45°角的 TRAANG 功能。

7.8.4 在磨床上斜向切入 (G5, G7)

G 指令 G7 和 G5 可简化在使用“斜置轴 (TRAANG)”转换功能的磨床上进行斜向切入的编程，可在切入时只运行斜置轴。

此时只需对所需切入运动的最终位置的 X 和 Z 坐标进行编程。相应的起始位置可由 NC 在执行 G7 功能时根据 X 轴的当前位置，所编程的最终位置以及轴的倾斜角 α 加以计算并逼近该位置。

起始位置由两条直线的交点决定：

- 平行且距 Z 轴为当前 X 轴的位置的直线
- 平行于倾斜轴且过所编程最终位置的直线

接着使用 G5，倾斜轴将运行到所编程的最终位置。

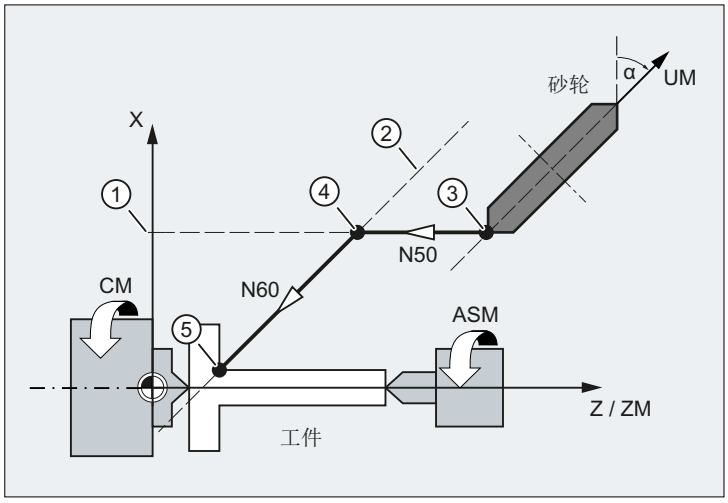
句法

```
G7 <最终位置_X> <最终位置_Z>
G5 <最终位置_X>
```

含义

G7：	计算并逼近斜向切入的起始位置
G5：	将倾斜轴运行至所编程的最终位置
<最终位置_X>:	X 轴的最终位置
<最终位置_Z>:	Z 轴的最终位置

示例



① 平行且距 Z 轴为当前 X 轴的位置的直线

② 平行于倾斜轴且过所编程最终位置的直线

③ 起始位置

④ 切入：起始位置

⑤ 切入：最终位置

X 几何轴

Z 几何轴

ZM 机床轴

UM 机床轴

程序代码	注释
N... G18	; 选择 XZ 平面。
N40 TRAANG(45.0)	; 激活 TRAANG 转换，角度 = 45°。
N50 G7 X40 Z70 F4000	; 计算并逼近起始位置。
N60 G5 X40 F100	; 将倾斜轴运行到最终位置。
N70 ...	

7.9 激活级联转换 (TRACON)

级联转换是在零件程序或同步动作中通过指令 TRACON 激活的。

句法

```
TRACON(<转换编号>,<参数_1>,<参数_2,...)
...
TRAFOOF
```

含义

TRACON:	激活级联转换 另一个之前有效的转换通过 TRACON() 隐含关闭。		
<转换编号>:	级联转换的编号		
	类型：	INT	
	取值范围：	0 ... 2	
	值：	0, 1	第一个/唯一的级联转换
		2	第二个级联转换
		无指示	与 0 或 1 的含义相同
		提示： 值不等于 0、1、2 会报错。	
<参数_1>,<参数_2,...:	相互间级联的转换的参数（例如斜置轴的角度） 如果是尚未设定的参数，则默认设置或者上一次所使用的参数有效。 通过置小数点必须考虑到：当要让前一个参数的基本设置有效时，对已知参数按照其等候的顺序进行分析。特别是当声明至少一个参数时，在该参数前就必须有一个逗号，即使当<转换编号>的声明没有必要时也是如此，例如 TRACON(, 3.7)。 执行带 Transmit 的 TRACON 或 TRAORI（第 5 轴加工）时，第二个参数 Par_2 并不作为斜置轴的倾角生效。 执行带 TRACYL（柱面加工）的 TRACON 时，Par 用作单位直径。		
TRAFOOF:	上一次激活的（级联）转换被关闭。		

示例

程序代码	注释
...	
N230 TRACON(1,45.)	；激活第一个级联转换。
	；之前生效的转换自动取消。
	；斜置轴的参数为 45°。
...	
N330 TRACON(2,40.)	；激活第二个级联转换。
	；斜置轴的参数为 40°。
...	
N380 TRAFOOF	；关闭第二个级联转换。
...	

7.10 直角坐标 PTP 运动

7.10.1 激活/取消坐标 PTP 运动 (PTP、PTPG0、PTPWOC、CP)

在 NC 程序中通过 G 指令组 49 中的指令激活/取消坐标点对点或 PTP 运动。

指令模态生效。默认通过坐标轨迹运动 (CP) 运行。

不同于 CP 的是，PTP 运动生效时，只执行坐标目标点的转换且机床轴同步运行。

为此，坐标目标点可换算为机床轴值，除位置和角度数据外，还需要轴位置信息。这些数据通过可设定地址 STAT (页 404) 和 TU (页 408) 指定。

前提条件

转换 TRAORI、TRANSMIT、RCTRA 或 ROBX 生效。

句法

```
PTP / PTPG0 / PTPWOC
...
CP
```

含义

PTP:	激活点对点运动 PTP 在 G0 和 G1 程序段中通过同步轴运动逼近编程的坐标位置。
PTPG0:	激活点对点运动 PTPG0 只在 G0 程序段中通过同步轴运动逼近编程的坐标位置。在 G1 程序段中切换为轨迹运动 CP。
PTPWOC:	激活点对点运动 PTPWOC（仅在定向转换生效时可行） 和 PTP 一样，但不带因回转轴和定向轴运动而引起的补偿运动。
CP:	取消点对点运动并激活轨迹运动 CP 通过 CP 执行坐标轨迹运动。

说明

PTPWOC

与 RCTRA 或 ROBX 组合使用时， PTPWOC 没有意义。

示例

参见：

- 示例 1：带 ROBX 转换的 6 轴机械手的 PTP 运动 (页 411)
- 示例 2：生成 5 轴转换时的 PTP 运动 (页 412)
- 示例 3：PTPG0 和 TRANSMIT (页 413)

7.10.2 给定铰接位置 (STAT)

通过笛卡尔坐标系给定位置及给定刀具定向不足以确定机床位置，因为同一个刀具定向可以有多个铰接位置。根据具体运动系统，可存在最多 8 个不同的关节位置。这些不同的关节位置针对特定坐标转换。

为避免不唯一性，铰接位置在地址 STAT 下给定。

说明

控制系统只在 PTP 运动中考虑编程的 STAT 值。在 CP 运动中会自动忽略，因为通常坐标转换生效时无法进行位置切换。通过生效的 CP 运行时，系统将起点用作目标点位置。

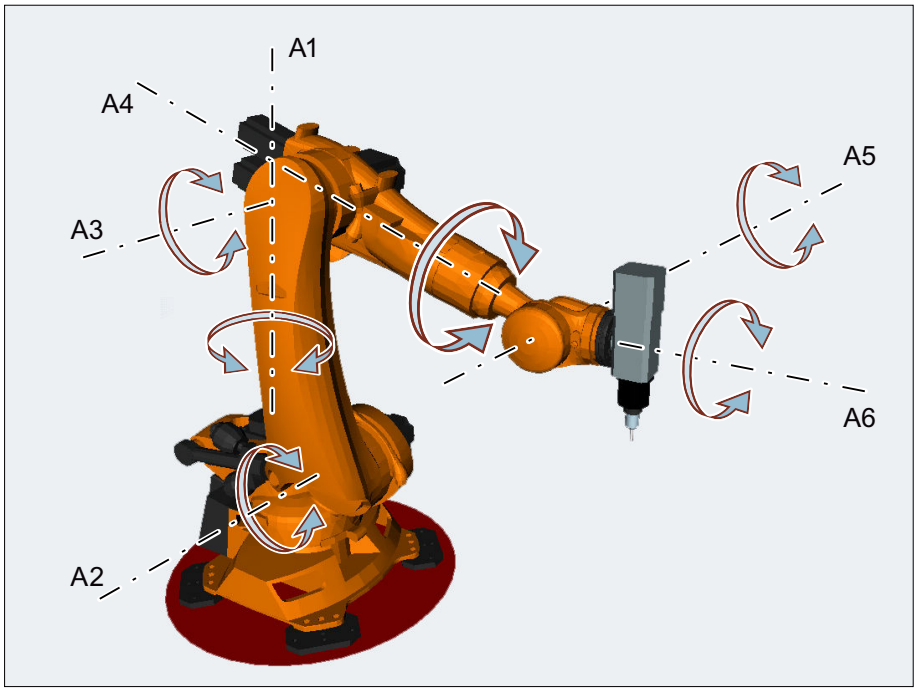
句法

```
STAT=<值>
```

含义

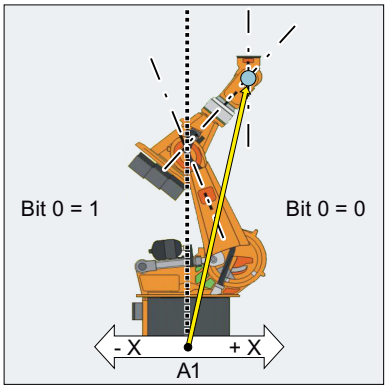
STAT：	用于给定铰接位置的可设定地址
<值>：	二进制或十进制值 每个可能有的位置均有一位。位的含义取决于相应的坐标转换。

STAT 的应用可通过带有铣削主轴的 6 轴关节型机械手为例来加以说明。运动转换应通过机械手转换 ROBX 来进行（前提：编译循环“RMCC/ROBX 扩展机械手转换”已加载并生效）。

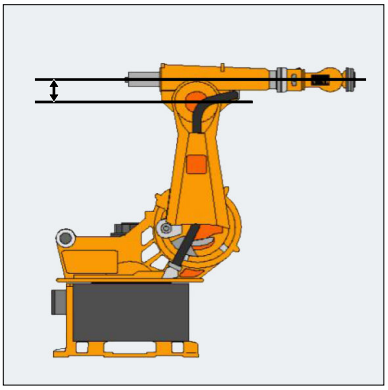


轴 A1、A2 和 A3 是关节型机械手的主要轴。使用主要轴可对称为头部轴或手臂轴的 A4、A5 和 A6 进行工作区域中的定位。通过手臂轴的附加运动方式，可根据加工任务对铣削主轴在工作区域中进行所需的定位。相同的刀具定位时，可以有不同的关节位置。

通过对 STAT 可设置地址的位 0 ... 2 的编程，选择加工时要使用的关节位置：

<table>
<tr><td rowspan="3">位 0</td><td colspan="2">手臂轴交点位置 (A4, A5, A6)</td><td rowspan="3"> 

示例：手臂轴交点位于初始区域中。</td></tr>
<tr><td>= 0</td><td>初始区域（Shoulder Right）
如果手臂轴交点的 X 值相对于 A1 坐标系为正，机械手则位于初始区域中。</td></tr>
<tr><td>= 1</td><td>过顶区域（Shoulder Left）
如果手臂轴交点的 X 值相对于 A1 坐标系为负，机械手则位于过顶区域中。</td></tr>
</table>

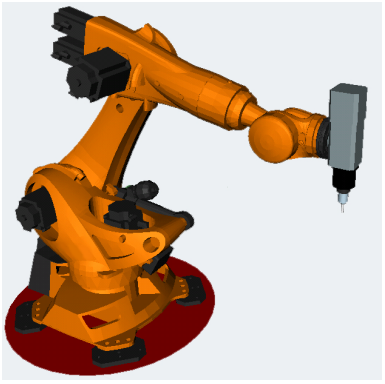
位 1	轴 3 的位置 位 1 值发生变化的角度取决于机械手类型。对于轴 3 和轴 4 相交的机械手：	
	= 0	A3 < 0° (**Elbow Down**)
	= 1	A3 ≥ 0° (**Elbow Up**)
	提示： 对于偏移在轴 3 和轴 4 之间的机械手，位 1 值发生变化的角度取决于该偏移的大小。	 A3 和 A4 之间的偏移
位 2	轴 5 的位置	
	= 0	A5 ≤ 0° (**No Handflip**)
	= 1	A5 > 0° (**Handflip**)

程序示例：

程序代码	**注释**
...	
N14 T="T8MILLD20" D1	; $TC_DP3[1,1]=132.95
N16 ORIMKS	
N17 G1 PTP X1665.67 Y0 Z1377.405 A=0 B=0 C=0 **STAT=**...F2000	; STAT 的值确定关节位置（见下）。
...	

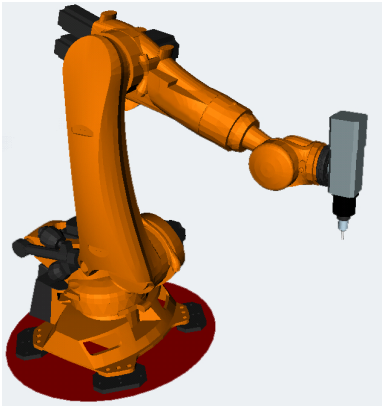
`STAT=1 ('B001')`

→ Shoulder Left
→ Elbow Down
→ No Handflip



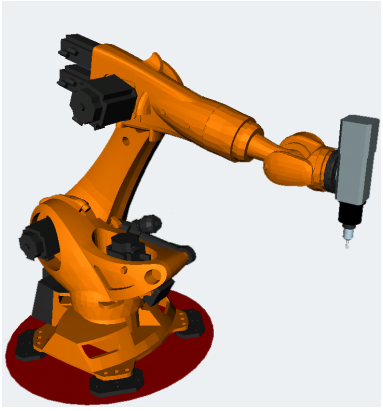
`STAT=2 ('B010')`

→ Shoulder Right
→ Elbow Up
→ No Handflip

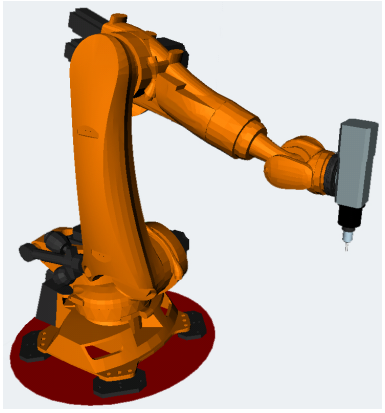


`STAT=5 ('B101')`

→ Shoulder Left
→ Elbow Down
→ Handflip



STAT=6 ('B110') → Shoulder Right
→ Elbow Up
→ Handflip



TRANSMIT

TRANSMIT 上使用地址 STAT，以避免极点不唯一。

如果回转轴必须旋转 180º 或 CP 时的轮廓由极引导，则：

位 0	仅在 $MC_TRANSMIT_POLE_SIDE_FIX_1/2 = 1 或 2 时相关：	
	= 0	回转轴旋转 +180º 或顺时针旋转。
	= 1	回转轴旋转 -180º 或逆时针旋转。
位 1	仅在 $MC_TRANSMIT_POLE_SIDE_FIX_1/2 = 0 时相关：	
	= 0	通过极运行。回转轴不转。
	= 1	围绕极旋转。此时，位 0 与 STAT 相关。

7.10.3 给定轴交角符号 (TU)

为了能够在旋转轴及大于 +180° 或小于 -180° 的轴交角上运行（不采用特殊运行方案，例如：中间点），必须在可设定地址 TU 下给定轴交角符号。

说明

控制系统只在 PTP 运动中考虑编程的 TU 值。在 CP 运动中会自动忽略。

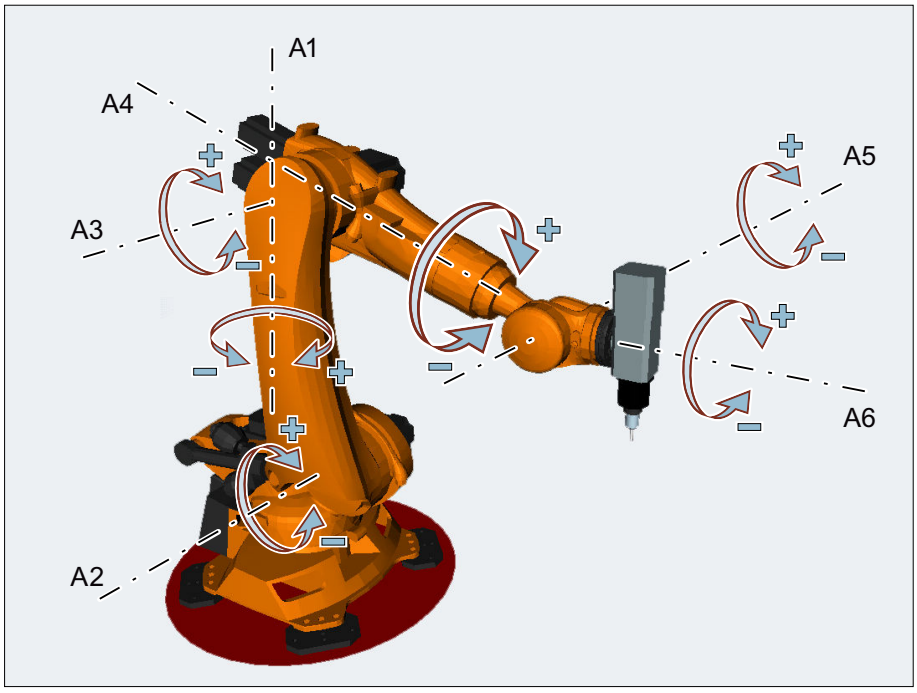
句法

TU=<值>

含义

TU：	用于给定轴交角符号的可设定地址			
<值>：	二进制或十进制值 每根参与转换的轴都有一个显示轴交角 (θ) 符号及运行方向的位。			
	位	= 0	轴交角符号：＋	角度范围：0° ≤ θ < 360°
		= 1	轴交角符号：-	角度范围：- 360° < θ < 0°

示例：6 轴关节型机械手



位	含义	值	轴交角符号	轴交角
位 0 1)	A1 轴交角的符号	= 0	+	≥ 0 °
		= 1	-	< 0 °
位 1 1)	A2 轴交角的符号	= 0	+	≥ 0 °
		= 1	-	< 0 °
位 2 1)	A3 轴交角的符号	= 0	+	≥ 0 °
		= 1	-	< 0 °
位 3 1)	A4 轴交角的符号	= 0	+	≥ 0 °
		= 1	-	< 0 °

位	含义	值	轴交角符号	轴交角
位 4 [1)]	A5 轴交角的符号	= 0	+	≥ 0 °
		= 1	-	< 0 °
位 5 [1)]	A6 轴交角的符号	= 0	+	≥ 0 °
		= 1	-	< 0 °

1) 实际的 TU 位号根据机械手轴的通道轴号得出！在示例中，机械手轴（A1 至 A6）为通道中的前六根轴，因此使用 TU 位 0 ... 5。如进行其他的机械手轴通道分配，机械手轴的 TU 位号也会相应变化（例如：机械手轴为第 3 至 8 通道轴，即机械手轴使用 TU 位 2 ... 7）。

TU=19（对应 TU='B010011）也表示：

位	值		轴交角
0	= 1	⇒	$\theta_{A1} < 0°$
1	= 1	⇒	$\theta_{A2} < 0°$
2	= 0	⇒	$\theta_{A3} \geq 0°$
3	= 0	⇒	$\theta_{A4} \geq 0°$
4	= 1	⇒	$\theta_{A5} < 0°$
5	= 0	⇒	$\theta_{A6} \geq 0°$

说明

运行范围 >±360° 的轴总是会沿最短轨迹运行，因为无法通过 TU 信息确定唯一的轴位置。

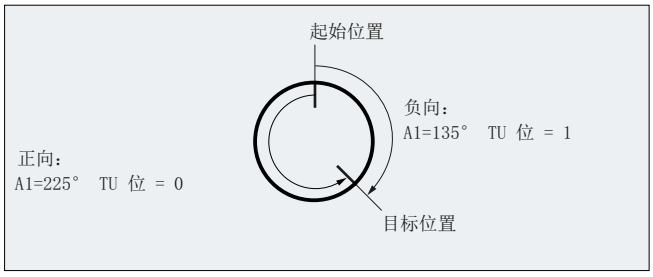
如果某个位置未进行 TU 编程，则根据 MD30455 $MA_MISC_FUNCTION_MASK 的设置运行较短或较长的行程（参见扩展功能手册中的章节“PTP 运动时的软限位”）。

TRANSMIT

TRANSMIT 生效时，PTP 运行中的地址 TU 无意义。

示例

下图中标出的回转轴位置可沿正方向或负方向达到。在地址 A1 下编程角位置。只有通过 TU 给定，运行方向才唯一。



7.10.4 示例 1：带 ROBX 转换的 6 轴机械手的 PTP 运动

在以下应用示例中，对坐标 PTP 运动以及与之相关的 NC 指令进行了典型的说明。

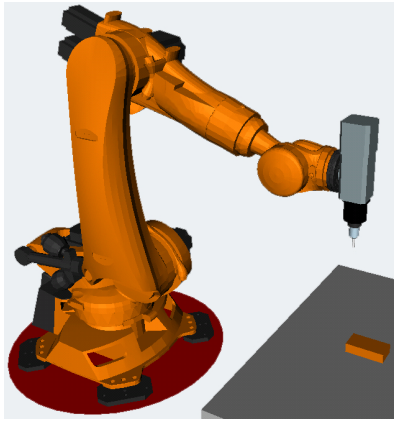


图 7-3 带铣削主轴的 6 轴关节型机械手

```
N1 G90
N2 T="T8MILLD20" D1 M6
N3 TRAORI
;$P_UIFR[1]=CTRANS(X,1500,Y,0,Z,400):CROT(X,0,Y,0,Z,-90)
N4 G54
N5 M3 S20000
N6 ORIWKS
N7 ORIVIRT1
N8 CYCLE832(0.01,_FINISH,1)
;HOME
N9 TRAFOOF
```

```
N10 G0 RA1=0.0000 RA2=-90.0000 RA3=90.0000 A=0.0000 B=90.0000 C=0.0000
N11 TRAORI
N12 G54
N13 G0 PTP X1369.2426 Y956.7528 Z502.5517 A=135.5761 B=-33.2223 C=161.1435
STAT='B010' TU='B001011'
N14 G0 X1355.1242 Y1014.9394 Z424.9695 A=135.8491 B=-33.1439 C=160.9941
STAT='B010' TU='B001011'
N15 G1 CP X1354.8361 Y1016.1269 Z423.3862 A=136.0635 B=-33.0819 C=160.8770
F1000
N16 G1 X1336.4283 Y1016.1269 Z426.6311 A=136.0484 B=-32.2151 C=160.9643
F2000
N17 G1 X1317.9831 Y1016.1269 Z429.6730 A=136.0175 B=-31.3394 C=161.0655
;HOME
N18 TRAFOOF
N19 G0 RA1=0.0000 RA2=-90.0000 RA3=90.0000 A=0.0000 B=90.0000 C=0.0000
N20 M30
```

7.10.5 示例 2：生成 5 轴转换时的 PTP 运动

接受：以一个直角 CA 运动为基础。

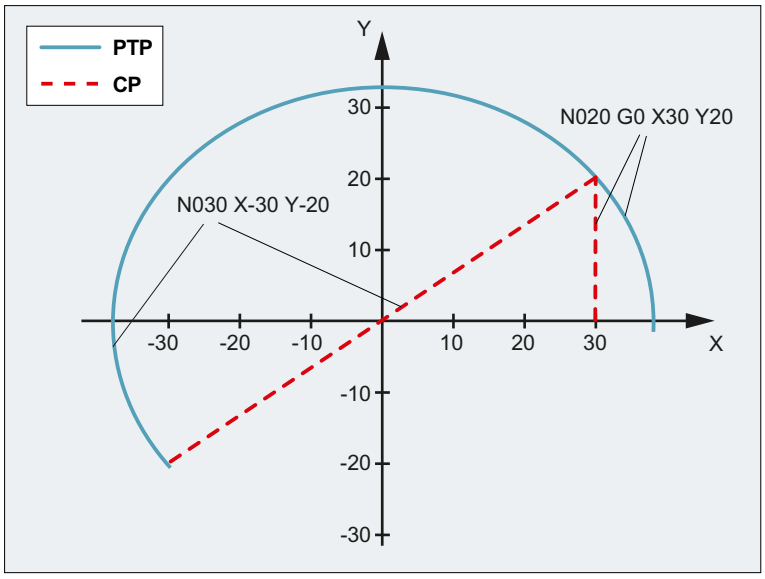
程序代码	注释
TRAORI	; CA 运动转换启动
PTP	; PTP 运动启动
N10 A3=0 B3=0 C3=1	; 回转轴位置 C=0 A=0
N20 A3=1 B3=0 C3=1	; 回转轴位置 C=90 A=45
N30 A3=1 B3=0 C3=0	; 回转轴位置 C=90 A=90
N40 A3=1 B3=0 C3=1 **STAT=1**	; 回转轴位置 C=270 A=-45

选择唯一的回转轴返回位置：

同时在程序段 N40 中，回转轴通过编程 **STAT = 1** 以最长行程自其起点 (C=90, A=90) 运行到终点 (C=270, A=–45)。**STAT=0** 时，回转轴则以最短的行程运行到终点 (C=90, A=45)。

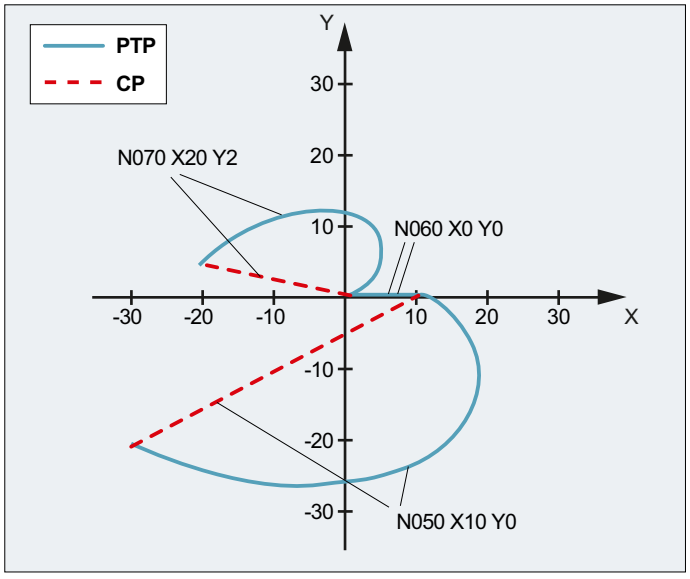
7.10.6 示例 3：PTPG0 和 TRANSMIT

通过 PTPG0 和 TRANSMIT 绕行极



程序代码	注释
N001 G0 X30 Z0 F10000 T1 D1 G90	；起始位置 绝对尺寸
N002 SPOS=0	
N003 **TRANSMIT**	; TRANSMIT 转换
N010 **PTPG0**	；自动到每个 G0 程序段 PTP 并重新 CP
N020 G0 X30 Y20	
N030 X-30 Y-20	
N120 G1 X30 Y20	
N110 X30 Y0	
M30	

通过 PTPG0 和 TRANSMIT 驶出极点



编程	注释
N001 G0 X90 Z0 F10000 T1 D1 G90	; 起始位置
N002 SPOS=0	
N003 **TRANSMIT**	; TRANSMIT 转换
N010 **PTPG0**	; 自动到每个 G0 程序段 PTP 并重新 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	

7.11 在选择一个转换时的边界条件

功能

选择转换可以通过零件程序或者 MDA。对此要注意：

- 不添加一个运行中间程序段（棱角/半径）。
- 一个样条程序段顺序必须已经结束；如果没有，就会显示一个信号提示。
- 刀具精密补偿必须已经取消 (FTOCOF); 如果没有，就会显示一个信号提示。
- 刀具半径补偿必须已经取消 (G40); 如果没有，就会显示一个信号提示。
- 一个激活的刀具长度补偿由控制器接收到转换。
- 在转换之前有效的当前框架由控制器取消。
- 一个当前有效的工作范围限制对于和转换有关的轴由控制器取消（和 WALIMOF 相适应）。
- 取消保护范围监控。
- 轨迹控制运动和精磨被中断。
- 所有在机床数据中说明的轴必须程序段同步化。
- 将更换的轴换回来；如果不这样，会出现一个信号提示。
- 在不独立的轴时输出一个信号。

换刀

换刀只有在取消刀具半径补偿时才可以。

刀具长度补偿的转换和刀具半径补偿的选择/取消不可以在同一个程序段内编程。

框架转换

所有仅以基本坐标系为参照的语句均允许（FRAME，刀具半径补偿）。但是 G91 时（增量尺寸）的框架转换不作特别处理 — 与未激活转换时不同。待执行的增量在新框架的工具坐标系中予以分析 — 与前一个程序段中哪一个框架在起作用没有关系。

排除在外

涉及转换的轴不可以：

- 用作预置轴（报警），
- 用于向固定点返回（报警），
- 回参考点（报警）。

7.12 取消转换 (TRAFOOF)

通过预定义程序 TRAFOOF 取消所有生效的坐标转换和框架。

说明

适用于取消转换的边界条件与选择的边界条件 (页 415)一样。

之后所需的框架必须通过更新的编程生效。

句法

```
...
TRAFOOF
```

含义

TRAFOOF:	取消所有生效的转换/框架

运动链

8

8.1 删除组件 (DELOBJ)

函数 DELOBJ() 通过将分配的系统变量恢复为缺省值来“删除”组件：

- 运动链元素
- 保护区、保护区元素和防撞对
- 转换数据

句法

```
[<RetVal>=] DELOBJ(<CompType>[,,,<NoAlarm>)])
[<RetVal>=] DELOBJ(<CompType>,<Index1>[,,<NoAlarm>])
[<RetVal>=] DELOBJ(<CompType>[,<Index1>][,<Index2>][,<NoAlarm>])
```

含义

DELOBJ：	删除运动链元素、保护区元素、碰撞对元素和转换数据元素

<CompType>：	需删除组件的类型	
	数据类型：	STRING
	值：“KIN_CHAIN_ELEM” **含义**：所有运动元素的系统变量：$NK_...	
	值：“KIN_CHAIN_SWITCH” **含义**：系统变量 $NK_SWITCH[<i>]	
	值：“KIN_CHAIN_ALL” **含义**：所有运动元素和开关。 等同于通过 “KIN_CHAIN_ELEM” 和 “KIN_CHAIN_SWITCH” 逐步调用 DELOBJ	
	值：“PROT_AREA” **含义**：保护区的系统变量： • $NP_PROT_NAME • $NP_CHAIN_NAME • $NP_CHAIN_ELEM • $NP_1ST_PROT	
	值：“PROT_AREA_ELEM” **含义**：机床保护区和/或自动刀具保护区的保护区元素的系统变量： • $NP_NAME • $NP_NEXT • $NP_NEXTP • $NP_COLOR • $NP_D_LEVEL • $NP_USAGE • $NP_TYPE • $NP_FILENAME • $NP_PARA • $NP_OFF • $NP_DIR • $NP_ANG	
	值：“PROT_AREA_COLL_PAIRS” **含义**：防撞对的系统变量： • $NP_COLL_PAIR • $NP_SAFETY_DIST	
	值：“PROT_AREA_ALL” **含义**：所有保护区、保护区元素和防撞对（系统变量 $NP_...）	

	等同于通过 “PROT_AREA”、“PROT_AREA_ELEM” 和 “PROT_AREA_COLL_PAIRS” 逐步调用 DELOBJ	
	值：“TRAFO_DATA” **含义**：所有转换的系统变量 $NT_...	
<Index1>：	第一个需删除组件的下标（**可选**）	
	数据类型：	INT
	缺省值：	-1
	值域：	-1 ≤ x ≤（所配置组件的最大数 -1）
	值	**含义**
	0, 1, 2, ...	需删除组件的下标。
	-1	所有指定类型的组件均已删除。不分析<下标 2>。
<Index2>：	最后需删除组件的下标（**可选**） 如果未编程 <Index2>，则只会删除 <Index1> 中指定的组件的系统变量。	
	数据类型：	INT
	缺省值：	仅 <Index1> 中指定组件的系统变量被删除。
	值域：	<Index1> < x ≤（所配置组件的最大数 -1）
<NoAlarm>：	报警抑制（**可选**）	
	数据类型：	BOOL
	缺省值：	FALSE
	值	**含义**
	FALSE	在故障时 (<RetVal> < 0) 停止程序执行并显示报警 .。
	TRUE	在故障时不停止程序执行且不显示报警。 使用场合：用户可根据返回值执行特定操作

<RetVal>:	函数的返回值	
	数据类型：	INT
	值域：	0, -1, -2, ... -7
	值	**含义**
	0	未出错。
	-1	调用的函数不含参数。至少要指定参数 **<CompType>**。
	-2	**<CompType>** 表示未知组件
	-3	**<**下标 1**>** 小于 -1
	-4	**<**下标 1**>** 大于所配置的组件数
	-5	**<**下标 1**>** 在删除组件**组**时值不等于 -1
	-6	**<**下标 2**>** 小于 **<**下标 1**>**
	-7	**<**下标 2**>** 大于所配置的组件数

8.2 通过名称确定下标 (NAMETOINT)

在 STRING 型的系统变量数组可以输入用户自定义的名称。借助系统变量名称，函数 NAMETOINT() 可以确定保存在系统变量数组中的名称所对应的下标值。

句法

```
<RetVal> = NAMETOINT(<SysVar>,<Name>[,<NoAlarm>])
```

含义

NAMETOINT:	确定系统变量下标	
<SysVar>:	STRING 型系统变量数组的名称	
	数据类型：	STRING
	值域：	NC 中所有 STRING 型系统变量数组的名称
<Name>:	需计算系统变量下标的字符串或名称。	
	数据类型：	STRING
<NoAlarm>:	报警抑制（**可选**）	
	数据类型：	BOOL
	缺省值：	FALSE
	值	**含义**
	TRUE	在故障时不停止程序执行且不显示报警。 使用场合：用户可根据返回值执行特定操作
	FALSE	在故障时 (<RetVal> < 0) 停止程序执行并显示报警。
<RetVal>:	系统变量下标或故障信息	
	数据类型：	INT
	值域：	-1 ≤ x ≤（所配置组件的最大数 -1）
	值	**含义**
	≥ 0	在指定的系统变量下标下找到了查找的名称。
	-1	名称未找到或出错。

示例

程序代码	注释
DEF INT INDEX	
$NP_PROT_NAME[27] = “隐藏”	
...	
INDEX = NAMETOINT（“$NP_PROT_NAME”，“隐藏”）	; INDEX == 27

含运动链的防撞

9

说明

保护区

后面章节中提到的保护区都与功能“机床几何模型”相关。

文档：

功能手册 特殊功能，章节“机床几何模型”

9.1 防撞对检查（COLLPAIR）

通过 COLLPAIR(...)计算两个保护区是否能形成一个防撞对。

句法

```
[<RetVal> =] COLLPAIR(<Name_1>,<Name_2>[,<NoAlarm>)])
```

含义

COLLPAIR:	检测保护区能否形成一个防撞对		
<RetVal>:	函数的返回值		
	数据类型：	INT	
	值：	≥ 0	两个保护区形成了防撞对。 返回值 == 防撞对下标 m（参见 $NP_COLL_PAIR）
		-1	没有指定两个字符串或者只要有一个字符串是零。
		-2	在第一个参数中指定的保护区未找到。
		-3	在第二个参数中指定的保护区未找到。
		-4	两个指定的保护区均未找到。
		-5	两个指定的保护区均找到了但无法形成一对。
<Name_1>:	第一个保护区的名称		
	数据类型：	STRING	
	取值范围：	编程的保护区名称	
<Name_2>:	第二个保护区的名称		
	数据类型：	STRING	
	取值范围：	编程的保护区名称	
<NoAlarm>:	报警抑制（**可选**）		
	数据类型：	BOOL	
	值：	FALSE （默认）	在故障时（**<RetVal> < 0**）停止程序执行并显示报警。
		TRUE	在故障时不停止程序执行且不显示报警。 使用场合：用户可根据返回值执行特定操作

9.2 要求对碰撞监测的机床模型进行重新计算（PROTA）

如果零件程序运动链 $NK_... 中、几何机床模型中或碰撞监测 $NP_... 中的系统变量已写入，必须调用程序 PROTA，这样 NC 内部碰撞监测的机床模型中的修改才会生效。

句法

```
PROTA[(<Par>)]
```

含义

PROTA:	要求对碰撞监测的机床模型进行重新计算		
	● 会触发预处理停止。 ● 必须位于单独的程序段中。		
<Par>:	参数（**可选**）		
	数据类型：	STRING	
	值：	---	不带参数。 不执行机床模型的重新计算。保持保护区状态。
		“R”	不执行机床模型的重新计算。根据$NP_INIT_STAT 将保护区设为初始状态。

边界条件

模拟

在零件程序中，PROTA 程序不可以与模拟程序（simNC）一起使用。

示例：模拟激活时，避免调用 PROTA。

程序代码	注释
...	
IF $P_SIM == FALSE	; IF 模拟未激活
PROTA	; THEN 碰撞模型重新计算
ENDIF	; ENDIF
...	

参见

设置保护区状态（PROTS） (页 428)

9.3 设置保护区状态（PROTS）

PROTS(...)程序将保护区状态设置为指定值。

句法

```
PROTS(<State>[,<Name_1>,...,<Name_n>])
```

含义

PROTS：	设置保护区状态		
	• 必须位于单独的程序段中。		
<State>：	指定保护区的目标状态		
	数据类型：	CHAR	
	值：	“A” 或 “a”	状态：有效
		“ I ” 或 “i”	状态：无效
		“P” 或 “p”	状态：预激活的或 PLC 控制的 1)
		“R” 或 “r”	状态：初始状态的 NC 内部值 2)
<Name_1> ... <Name_n>：	需要设为目标状态的一个或多个保护区的名称（**可选**） 如果没有指定名称，则目标状态适用于所有定义的保护区。		
	数据类型：	STRING	
	取值范围：	编程的保护区名称	
	提示： 可作为参数指定的保护区最大数只与每个程序行允许的最大字符数有关。		
1) 通过以下数据激活/取消激活：DB10.DBX234.0 - DBX241.7 2) 状态设置为初始状态的 NC 内部值，即：系统变量$NP_INIT_STAT 在最后一次调用 PROTA(...) (页 427)时的值。			

9.4 确定保护区间距（PROTD）

PROTD(...)计算两个保护区之间的间距。

功能特性：

- 间距计算与保护区的状态无关（激活、未激活、预激活）。
- 进行第二保护区的间距计算时，只涉及标识为 $NP_USAGE = "C" 或 "A" 的保护区元素。标识为 $NP_USAGE = "V" 的保护区元素不考虑。
- 所有元素都标识为 $NP_USAGE = "V" 的保护区无法参与间距计算。
- 间距计算采用前一个程序段结束时的有效位置。
- 在主处理中计算的叠加量（比如：DRF 偏移量或外部零点偏移）和**编译该函数**时有效的数值一起计入间距计算。

说明

同步

使用 PROTD(...)函数时，用户自行负责编写预处理停止指令 STOPRE，使主处理与预处理同步。

碰撞

如果指定保护区之间发生碰撞，该函数便会返回 0.0 的间距。如果两个保护区发生碰触或相互渗透，则发生碰撞。

间距计算中不考虑碰撞检测安全间距(MD10622 $MN_COLLISION_SAFETY_DIST) 。

句法

```
[<RetVal> =] PROTD([<Name_1>],[<Name_2>],VAR <Vector>[,<System>])
```

含义

PROTD:	计算两个指定保护区的间距	
	● 必须位于单独的程序段中。	
<RetVal>:	函数的返回值：这两个保护区间距的绝对值或碰撞时为 0.0（见上：碰撞段落）	
	数据类型：	REAL
	取值范围：	0.0 ≤ x ≤ + 最大实数值

<Name_1>, <Name_2>:	要计算间距的两个保护区的名称（**可选**）		
	数据类型：	STRING	
	取值范围：	编程的保护区名称	
	默认值：	""（空字符串） 如果没有指定保护区，该函数便会计算出碰撞模型中包含的所有激活的和预激活的保护区之间的最小间距。	
<Vector>:	返回值：保护区**<Name_2>**和保护区**<Name_1>**之间的 3 维间距矢量，其中： • **<Vector>**[0]：WCS 中的 X 轴坐标 • **<Vector>**[1]：WCS 中的 Y 轴坐标 • **<Vector>**[2]：WCS 中的 Z 轴坐标 碰撞时：**<Vector>** == 零矢量		
	数据类型：	VAR REAL [3]	
	取值范围：	**<Vector>** [n]：0,0 ≤ x ≤ ±最大 REAL 值	
<System>:	间距和间距矢量的单位（英制/公制）（**可选**）		
	数据类型：	BOOL	
	值：	FALSE （默认）	单位制取决于 G 代码组 13 中当前激活的 G 代码 (G70, G71, G700, G710)。
		TRUE	单位依据设置的基本单位制： MD52806 $MN_ISO_SCALING_SYSTEM

含运动链的转换 10

10.1 激活转换 (TRAFOON)

通过预定义程序 TRAFOON 激活借助运动链定义的转换。调用必须单独位于一个程序段中。

说明

作为替代方案，也可以通过传统的语言指令，如 TRAORI 或 TRANSMIT 来激活借助运动链定义的转换。为此必须在系统变量 $NT_TRAFO_INDEX 中输入对应的不等于零的值。$NT_TRAFO_INDEX 的更多信息参见“参数手册之系统变量分册”。

句法

```
TRAFOON(<Trafoname>,<Diameter>,<k>)
```

含义

TRAFOON:	程序，用于激活借助运动链定义的转换		
<Trafoname>:	转换数据组的名称		
	数据类型：	STRING	
	取值范围：	所有通过$NT_NAME 定义的转换数据组名称	
	提示： 转换数据组的名称必须具有唯一性。其只允许在 $NT_NAME 中出现一次。		
<Diameter>:	基准直径或加工直径（仅 TRACYL）		
	数据类型：	REAL	
	值必须 > 1。		
<k>:	定义槽壁补偿的使用（仅 TRACYL）。		
	数据类型：	BOOL	
	值：	FALSE	无槽壁补偿
		TRUE	有槽壁补偿
	对应 TRACYL 转换类型 514（槽壁补偿可编程）。若未给定 <k>，则 $NT_CNTRL[<n>] 中位 10 的设置生效。		

示例

程序代码	注释
TRAFOON["Trans_1"]	激活名称为 Trans_1 的转换。

10.2 根据机床测量修改定向转换（CORRTRAFO）

就采用借助运动链定义的定向转换的机床而言，用户可使用预定义功能 CORRTRAFO，以在机床测量后在机床的运动模型中修改定向轴的偏移矢量或方向矢量。

句法

```
<Corr_Status> = CORRTRAFO(<Corr_Vect>, <Corr_Index>, <Corr_Mode>
[, <No_Alarm>])
```

含义

CORRTRAFO:	功能调用

<Corr_Status>:	功能的返回值		
	数据类型：	INT	
	数值：	0	已无故障地执行功能。
		1	无转换生效。
		2	当前生效的转换不是定向转换。
		3	当前生效的定向转换并非借助运动链定义。
		10	调用参数 <Corr_Index> 为负。
		11	调用参数 <Corr_Mode> 为负。
		12	对子链区段的参考无效（<Corr_Index> 的个位）。值不允许大于子链中的定向轴的数量。
		13	对子链的定向轴的参考无效（<Corr_Index> 的个位）。值必须小于子链中的定向轴的数量。
		14	对子链的参考无效（<Corr_Index> 的十位）。仅允许值 0 和 1（对 Part 链或 Tool 链的参考）。在 <Corr_Index> 所参考的子链不存在的情况下，也会出现此故障编号。
		15	在通过参数 <Corr_Index> 参考的区段中，未定义补偿元素（$NT_CORR_ELEM_P 或 $NT_CORR_ELEM_T）。
		20	无效的补偿模式（<Corr_Mode> 的个位）。仅允许值 0 和 1。
		21	无效的补偿模式（<Corr_Mode> 的十位和/或百位）。在写入轴方向时，仅允许个位不等于零。
		30	<Corr_Mode> 的百位无效。仅允许值 0 和 1。
		31	<Corr_Mode> 的千位无效。仅允许值 0 和 1。
		40	需要作为轴方向接收的方向矢量为零矢量。仅当 <Corr_Mode> 的千位等于 0 时，才会出现此错误。若此参数的千位等于 1（最大补偿监控取消），则也可写入零矢量。
		41	在偏移矢量的补偿中，相对至少一个坐标中的当前值的偏差大于通过设定数据 SD41610 $SN_CORR_TRAFO_LIN_MAX 设定的最大值。参数 <Corr_Vect> 被误差矢量覆盖。这也适用于执行因报警而终止的情形（参见参数 <No_Alarm>）。

			误差矢量在其补偿值超出允许限值的分量中包含：测定的补偿值与限值之间的符号正确的差值。 未超出限值的分量的内容为零。
		42	在方向矢量的补偿中，相对当前方向的角度偏差大于通过设定数据 SD41611 $SN_CORR_TRAFO_DIR_MAX 设定的最大值。
		43	写入系统变量的尝试因缺少写入权限而被拒绝。
<Corr_Vect>:	补偿矢量 补偿矢量的内容通过以下参数 **<Corr_Index>** 和 <Corr_Mode> 定义。 若 <Corr_Status> = 41，则矢量的内容被覆盖（见上文）。		
	数据类型：	REAL	
<Corr_Index**>**:	需要修改补偿元素的区段/ 需要修改方向矢量的定向轴的序号		
	数据类型：	INT	
	参数 <Corr_Index> 为十进制编码（个位至十位）：		
	个位：	包含子链中的区段或定向轴的序号。	
	十位：	对子链的参考。	
		0x	工件链
		1x	刀具链

<Corr_Mode>:	补偿模式		
	数据类型：	INT	
	参数 <Corr_Mode> 为十进制编码（个位至千位）：		
	个位：	确定需要修正的元素。	
		xxx0	对线性偏移矢量的补偿
		xxx1	对定向轴的方向矢量的补偿
	十位：	确定如何修改 <Corr_Index> 之内容所参考的补偿元素。	
		xx0x	将补偿矢量直接写入补偿元素。 此方案用于直接写入补偿元素，而不需要获知相关系统数据（$NK_OFF_DIR[<n>, ...]）的序号 <n>。
		xx1x	类似于 0，但区别在于，传递的补偿值以世界坐标解释。 当运动链在初始设置（所有定向轴的位置等于 0）中包含其他旋转时，方案 0 和方案 1 便有区别。
		xx2x	类似于 1，但区别在于，补偿值基于整个区段，亦即，在补偿元素中输入一个值，使得整个区段达到通过补偿值定义的长度。
		提示： 在写入定向轴的方向矢量时不允许值 1 和 2。	
	百位：	确定参数 <Corr_Vect> 的内容的解释方式。	
		x0xx	传递的补偿矢量 <Corr_Vect> 包含： <Corr_Index> 结合 <Corr_Mode> 的十位所参考的补偿元素或区段的所有新长度（绝对补偿）。
		x1xx	传递的补偿矢量 <Corr_Vect> 仅包含：相对 <Corr_Index> 结合 <Corr_Mode> 的十位所参考的补偿元素或区段的当前长度的差（增量补偿）。
		提示： 在对定向轴的方向矢量的补偿中，百位的内容必须为 0。	
	千位：	确定是否需要通过下列最大值来限制补偿： • SD41610 $SN_CORR_TRAFO_LIN_MAX 或 • SD41611 $SN_CORR_TRAFO_DIR_MAX	
		0xxx	最大补偿监控生效。
		1xxx	最大补偿监控不生效。

<No_Alarm>：	故障情形（返回值 > 0）下的特性（**可选**）		
	数据类型：	BOOL	
	值：	FALSE（缺省）	在故障时停止程序执行，并显示报警 14103。
		TRUE	在故障时不停止程序执行且不显示报警。 使用场合：用户可根据返回值执行特定响应

说明

若在调用功能时出错，则或是输出报警，或是反馈故障编号（参见参数 <No_Alarm>），故用户自身可适当地对故障状态作出响应。故障原因通过报警参数详细表示。在报警的位置反馈的故障编号与报警参数相同。

有关 CORRTRAF 更多信息

机床的采用定向转换的运动结构通过一个或两个从世界坐标系零点出发的运动链（子链）描述。这两个链中的一个，即**刀具链**，在刀具的参考点处终止；另一个，即**工件链**，在基准坐标系的零点中终止。

CORRTRAFO 可以测量机床坐标转换过程中的杆臂长度和轴方向，并将这些数值写入到指定的轮廓元素中。运动链由通过 $NK_TYPE 定义的 OFFSET 型元素来描述。

区段中的 CORRTRAFO

工件链和刀具各自可以最多分为四个区段：

- 第 1 区段从链起点到定向轴 1。
- 第 2 区段从定向轴 1 到定向轴 2。
- 第 3 区段从定向轴 2 到定向轴 3。
- 第 4 区段从定向轴 3 到链终点。

各个区段可以包含 OFFSET 型或 ROT_CONST 型的恒定运动链元素。

下图展示了包含 2 个定向轴的坐标转换。

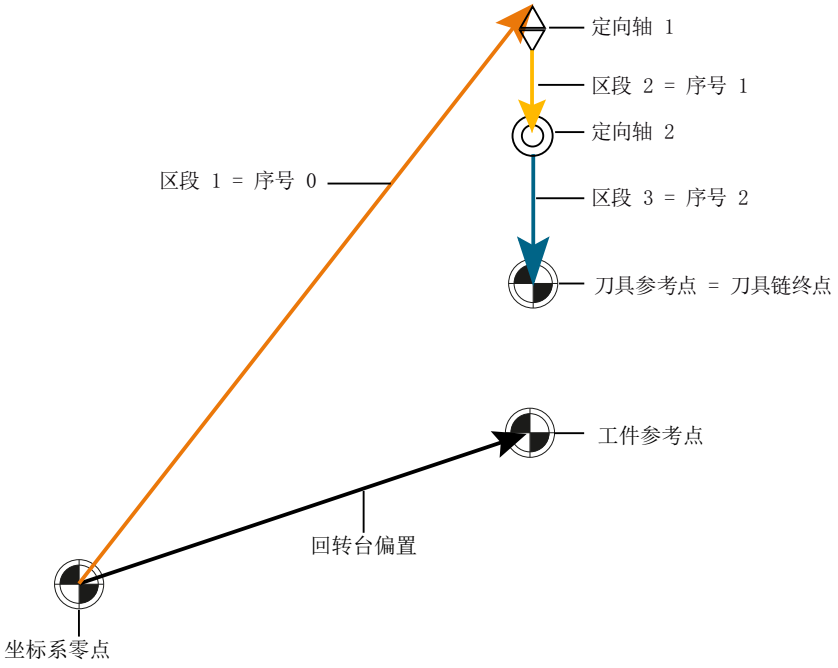


图 10-1　CORRTRAFO 示例

这些区段采用唯一性定义：在从起点到终点的整条运动链中，第一区段的序号为 0；第二区段序号为 1，以此类推。因此，最后一个区段的序号总是等于定向轴的数量。

补偿元素

可通过系统变量 $NT_CORR_ELEM_T[<n>, 0 ... 3] 或 $NT_CORR_ELEM_P[<n>, 0 ... 3] 对这些区段中的任一个中的恒定运动链元素（类型 $NK_TYPE[<n>] = "OFFSET" 的链元素）进行参考。借助 CORRTRAFO 功能可以将在机床测量中测定的补偿值写入以上述方法表示的元素中。

以坐标转换序号 1 为例：

- $NT_CORR_ELEM_T[1,0] = "C_AXIS_OFFSET"; 区段 1 上 C 轴（定向轴 1）的偏置，定义为轮廓元素。
- $NT_CORR_ELEM_T[1,1] = "B_AXIS_OFFSET"; 区段 2 上 B 轴（定向轴 2）的偏置，定义为轮廓元素。
- $NT_CORR_ELEM_T[1,2] = "BASE_TOOL_OFFSET"; 区段 3 上 B 轴相对于刀具参考点的偏置，定义为轮廓元素。

$NT_CORR_ELEM_T/P[<n>, 0 ... 3] 中的参考的顺序必须与上述区段对应，亦即，在 $NT_CORR_ELEM_T/P [<n>, 0] 中仅存在一个位于第一定向轴前的链元素，以此类推。

CORRTRAFO 可以将通过测量得出的补偿写入到以上述方式定义的轮廓元素中。补偿值的修改在 CORRTRAFO 中通过参数<Corr_Mode>来定义。

闭合转换链

若系统变量 $NT_CNTRL[<n>] 中的位 7 或位 8 置位，则系统内部自动在工件链的末尾（位 7）或在刀具链的起点（位 8）前自动插入额外的恒定链元素，该元素可以建立链终点和机床零点之间的连接（“闭合转换链”）。

这些自动插入的元素无法自外部写入，而是仅能读取（为此参见系统变量 $AC_TRAFO_CORR_ELEM_P/T）。

用于闭合刀具链的点

若系统变量 $NT_CLOSE_CHAIN_T 不为空，则并非在链的终点处，而是在指定的链元素的终点处将刀具链闭合。在激活坐标转换时，位于这个点后的其他链元素会导致对应的零点偏移。

定向轴的序号

除定向轴之间的恒定偏移以外，也可以通过 CORRTRAFO 功能描述定向轴的方向矢量。在此情形下，定向轴序号从零开始，贯穿起点到终点的整条运动链。故定向轴的序号总是等于位于该定向轴前的区段的序号。

也可通过系统变量 $AC_TRAFO_ORIAX_LOC 确定定向轴的序号。

允许的链元素的最大改变

可通过两个设定数据 SD41610 $SN_CORR_TRAFO_LIN_MAX（针对偏移矢量）和 SD41611 $SN_CORR_TRAFO_DIR_MAX（针对定向轴的方向矢量）限制所允许的链元素的最大改变。SD41610 $SN_CORR_TRAFO_LIN_MAX 给定各矢量分量相对其参考值所允许改变的最大量。SD41611 $SN_CORR_TRAFO_DIR_MAX 给定轴矢量之方向相对其参考值所允许改变的最大角度。其中参考值始终为对应的在调用 CORRTRAFO 时激活之坐标转换中生效的值。亦即，数据管理中的运动数据的可能在坐标转换激活后改变的内容对 CORRTRAFO 功能的生效方式无影响。

刀具补偿 11

11.1 补偿存储器

建立补偿存储器

每个数据字段均可用 T 和 D 编号调用（除了“平面 D 编号”之外）并且除了刀具的几何数据之外，还包含有其它记录，例如刀具类型。

平面 D 号结构

如果刀具管理不是由 NC 控制，那么使用“平面 D 号结构”。在这种情况下带从属刀具补偿程序段的 D 编号不对刀具进行赋值。

在零件程序中可以进一步编程 T。但是这个 T 和编程设计的 D 编号没有关系。

用户刀沿数据

通过机床数据文件可以配置用户切削数据。请注意机床制造商说明。

刀具参数

说明

补偿存储器中的各个参数值

补偿存储器 P1 ～ P25 的各个参数值可通过程序的系统变量读写。所有其他的参数被保留。

刀具参数 $TC_DP6 至 $TC_DP8，$TC_DP10 和 $TC_DP11 以及 $TC_DP15 至 $TC_DP17，$TC_DP19 和 $TC_DP20 视刀具类型不同有另一个含义。

刀具参数编号(DP)	系统变量的意义	附注
$TC_DP1	刀具类型	概要参见清单
$TC_DP2	刀沿位置	仅用于车刀
几何尺寸	**长度补偿**	
$TC_DP3	长度 1	计算
$TC_DP4	长度 2	类型和平面
$TC_DP5	长度 3	
几何尺寸	**半径**	

刀具参数编号(DP)	系统变量的意义	附注
$TC_DP6 [1)] $TC_DP6 [2)]	半径 1 / 长度 1 直径 d	铣刀/车刀/磨削刀具 切槽锯片
$TC_DP7 [1)] $TC_DP7 [2)]	长度 2 / 圆锥形铣刀的拐角半径 切槽锯片拐角半径	铣刀 切槽锯片
$TC_DP8 [1)] $TC_DP8 [2)]	铣刀的倒圆半径 1 突出长度 k	铣刀 切槽锯片
$TC_DP9 [1) 3)]	倒圆半径 2	保留
$TC_DP10 [1)]	刀具端面角度 1	圆锥形铣刀
$TC_DP11 [1)]	刀具纵轴角度 2	圆锥形铣刀
磨损	**长度和半径补偿**	
$TC_DP12	长度 1	
$TC_DP13	长度 2	
$TC_DP14	长度 3	
$TC_DP15 [1)] $TC_DP15 [2)]	半径 1 / 长度 1 直径 d	铣刀/车刀/磨削刀具 切槽锯片
$TC_DP16 [1)] $TC_DP16 [3)]	长度 2 / 圆锥形铣刀的拐角半径，拐角半径的槽 宽 b	铣刀 切槽锯片
$TC_DP17 [1)] $TC_DP17 [2)]	铣刀的倒圆半径 1 超出长度 k	铣削 / 3D 端面铣 切槽锯片
$TC_DP18 [1) 3)]	倒圆半径 2	保留
$TC_DP19[1)]	刀具端面角度 1	圆锥形铣刀
$TC_DP20[1)]	刀具纵轴角度 2	圆锥形铣刀
基本尺寸/适配器	**长度补偿**	
$TC_DP21	长度 1	
$TC_DP22	长度 2	
$TC_DP23	长度 3	
工艺		
$TC_DP24	后角	仅用于车刀
$TC_DP25		保留

[1)] 对于铣刀也适用于 3D 端面铣削

[2)] 对于切槽锯片刀具类型

[3)] 保留（SINUMERIK 840D sl 不使用）

备注

几何尺寸（例如长度 1 或者半径）存在多个记录组成部分。这些部分经相加得出结果尺寸 (例如长度总和 1，半径总和), 然后将成为有效尺寸。

不需要的补偿可以用值零来覆盖。

刀具参数 $TC-DP1 至 $TC-DP23，带轮廓刀具

说明

不评估表中未列出的刀具参数，例如 $TC_DP7，即其内容无意义。

刀具参数编号(DP)	含义	刀沿 Dn		附注
$TC_DP1	刀具类型			400 至 599
$TC_DP2	刀沿位置			
几何尺寸	**长度补偿**			
$TC_DP3	长度 1			
$TC_DP4	长度 2			
$TC_DP5	长度 3			
几何尺寸	**半径**			
$TC_DP6	半径			
几何尺寸	**极限角度**			
$TC_DP10	最小极限角度			
$TC_DP11	最大极限角度			
磨损	**长度和半径补偿**			
$TC_DP12	长度 1 磨损量			
$TC_DP13	长度 2 磨损量			
$TC_DP14	长度 3 磨损量			
$TC_DP15	磨损半径			
磨损	**极限角度**			
$TC_DP19	最小极限角度的磨损量			
$TC_DP20	最大极限角度的磨损量			
基本尺寸/适配器	**长度补偿**			

刀具参数编号(DP)	含义	刀沿 Dn		附注
$TC_DP21	长度 1			
$TC_DP22	长度 2			
$TC_DP23	长度 3			

基本值和磨损值

得出的尺寸分别作为总和由基本值和磨损值计算得出（例如用于半径的 $TC_DP6 + $TC_DP15）。此外，将基本尺寸（$TC_DP21 – $TC_DP23）加到第一个刀沿的刀具长度。此外，所有其他尺寸都影响该刀具长度，对于传统刀具，这些尺寸还可能影响有效刀具长度（适配器、可定向的刀架、设定数据）。

极限角度 1 和 2

极限角度 1 和 2 分别以刀沿终点到刀沿参考点的矢量为参照并以逆时针方向计数。

11.2 附加补偿

11.2.1 选择附加补偿（DL）

附加补偿实际上就是一种可以在加工过程中编程的过程补偿。 它们与一个切削刃的几何数据相关，因此是刀具切削刃数据的组成部分。

附加补偿数据通过一个 DL 号响应（DL: 与位置相关；根据使用位置补偿），通过操作界面输入。

应用

通过附加补偿，可以补偿使用地点条件下的尺寸误差。

句法

```
DL=<编号>
```

含义

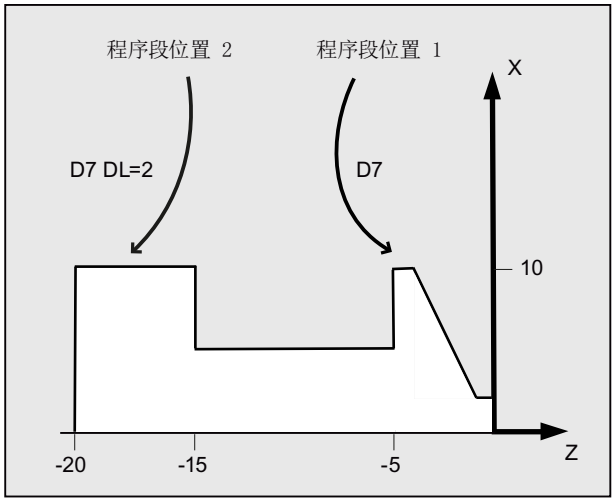
DL：	用于激活附加补偿的指令
<编号>：	通过参数 <编号> 可以指定待激活的附加刀具补偿数据段

说明

如何确定附加补偿的数量和激活附加补偿，可以通过机床数据进行（→ 请注意机床制造商的说明！）。

示例

同一个切削刃可以用于 2 个轴颈。



程序代码	注释
N110 T7 D7	; 转塔位于刀位 7。 D7 和 DL=1 被激活，并在下一个程序段中退回。
N120 G0 X10 Z1	
N130 G1 Z-6	
N140 G0 DL=2 Z-14	; D7 和 DL=2 被激活，在下一个程序段中退回。
N150 G1 Z-21	
N160 G0 X200 Z200	; 回到刀具更换点。
...	

11.2.2 确定磨损值和设置值（$TC_SCPxy[t,d], $TC_ECPxy[t,d]）

磨损量和设置值可以通过系统变量读取和写入。 这里的逻辑关系以刀具和刀沿相应的系统变量的逻辑为基准。

系统变量

$TC_SCPxy[<t>,<d>]:	磨损量通过 xy 分配给各自的几何参数，x 代表磨损量的数字，y 则是建立与几何参数的关系。
$TC_ECPxy[<t>,<d>]:	设置值通过 xy 分配给各自的几何参数，x 代表设置值的数字，y 则是建立与几何参数的关系。
<t>: 刀具 T 编号 <d>： 刀具切削刃的 D 号码	

说明

确定的磨损量和设置值附加到几何参数和其它的补偿参数（D 号码）。

示例

对于刀具（t）的切削刃（D 号码)，其长度 1 的磨损量确定为值 1.0。

参数： $TC_DP3 （长度 1， 用于车刀）

磨损量： $TC_SCP13 bis $TC_SCP63

设置值： $TC_ECP13 bis $TC_ECP63

$TC_SCP43 [<t>,<d>] = 1.0

11.2.3 清除附加补偿（DELDL）

用指令 DELDL 删除一个刀具切削刃的附加补偿（存储器使能）。 这时，不管是确定的磨损量还是设置值均清除。

句法

```
DELDL[<t>,<d>]
DELDL[<t>]
DELDL
<状态>=DELDL[<t>,<d>]
```

含义

DELDL:	用于删除附加补偿的指令
<t>:	刀具 T 编号
<d>:	刀具切削刃的 D 号码
DELDL[<t>,<d>]:	删除所有刀具 <t> 的切削刃 <d>附加补偿。
DELDL[<t>]:	刀具<t> 的所有切削刃的附加补偿均被删除。
DELDL:	TO 单元的所有刀具的所有附加切削刃补偿均被删除（对于编程了该指令的通道）

<状态>：	删除状态	
	值：	含义：
	0	已经成功地进行删除。
	-	没有进行删除（如果参数设定规定的是一个刀沿），或者删除没有完全进行（如果参数设定多个刀沿）。

说明

有效刀具的磨损量和设置值不可以被删除（类似于 D 补偿或刀具数据的删除特性）。

11.3 刀具补偿 - 特殊操作

设定数据 SD42900 - SD42960 可以用于控制刀具长度和磨损量符号的赋值。

这同样适用于几何轴镜像时的磨损量分量，或者在更换加工平面时的磨损量分量特性、以及在刀具方向上的进行温度补偿时。

磨损量

如果在磨损量之后给出一个参考基准，则表明是实际磨损量的和（$TC_DP12 到 $TC_DP20），以及磨损量（$SCPX3 到 $SCPX11）和设置值（$ECPX3 到 $ECPX11）的补偿值之和。

有关补偿值之和的完整信息，请见：

文献：

刀具管理功能手册

设定数据

SD42900 $SC_MIRROR_TOOL_LENGTH	镜像刀具长度分量和基准尺寸分量。
SD42910 $SC_MIRROR_TOOL_WEAR	镜像刀具长度分量的磨损量。
SD42920 $SC_WEAR_SIGN_CUTPOS	磨损量分量的符号赋值，与切削刃位置相关。
SD42930 $SC_WEAR_SIGN	反相磨损量尺寸的符号。
SD42935 $SC_WEAR_TRANSFORM	转换磨损量。
SD42940 $SC_TOOL_LENGTH_CONST	将刀具长度分量分配到几何轴。
SD42950 $SC_TOOL_LENGTH_TYPE	刀具长度分量的分配与刀具类型无关。
SD42960 $SC_TOOL_TEMP_COMP	刀具方向的温度补偿值在前面已编程的刀具方向上也有效。

文档

功能手册 基本功能；刀具补偿（W1）

其它信息

使修改的设定数据生效

只有在下次选择了一个刀沿时，设定数据修改后的刀具分量其新的赋值才生效。 如果一个刀具已经生效，则只有重新选择该刀具后，修改后的刀具的赋值数据才生效。

如果发生这种情况：即因为一个轴的镜像状态改变，使所产生的刀具长度改变，则这种情况与上述相同。 也就是说，在镜像指令后必须重新选择刀具，这样修改后的刀具长度分量才会生效。

可定向的刀架和新的设定数据

设置数据 SD42900 － SD42940 对一个可能激活的可定向刀架不起作用。 但是，一个刀具总是把所有的长度（刀具长度＋磨损量＋基准尺寸）加入到可定向刀架的计算中。 在计算所生成的总长时，要考虑所有由设定数据引起的改变；也就是说可定向刀架的矢量与加工平面无关。

说明

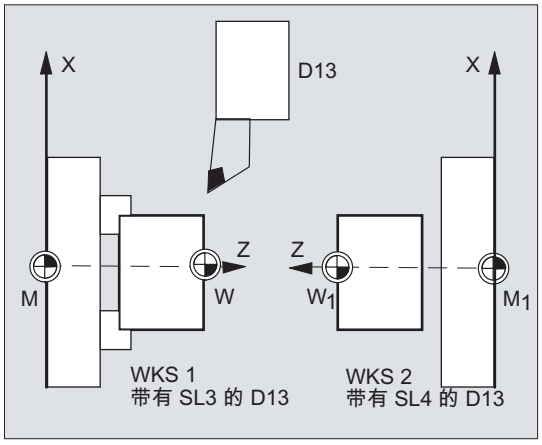
在使用可定向刀架时经常要求定义所有的刀具（在没有镜像的基准系统中），包括那些仅在镜像加工中使用的刀具。 这样在加工镜像轴时给刀架旋转，使刀具的实际位置正确表述。 刀具长度分量自动在正确的方向生效，从而就没有必要由控制系统通过设定数据给每个分量赋值（取决于各个轴的镜像状态）。

其它的应用可能

可定向刀架的这种功能非常有用，特别是在机床中，如果无法给刀具旋转，或者刀具在不同的方向已经固定安装。 这样刀具可以统一地在一个基准方向标注尺寸，然后通过一个虚拟地刀架的旋转产生加工时所需要的尺寸。

11.3.1 刀具长度镜像

采用设置好不为零的设置数据 SD42900 $SC_MIRROR_TOOL_LENGTH 和 SD42910 $SC_MIRROR_TOOL_WEAR， 可以用相应轴的磨损量对刀具长度分量和基本尺寸分量进行镜像。



SD42900 $SC_MIRROR_TOOL_LENGTH

设置数据 **不等于** 零：

刀具长度分量（$TC_DP3, $TC_DP4 和 $TC_DP5）和基本尺寸分量（$TC_DP21, $TC_DP22 和 $TC_DP23）（其关联轴镜像）通过符号反向而镜像。

磨损量 **没有**一起镜像。 如果磨损量也必须镜像，则必须设定设置数据 SD42910 $SC_MIRROR_TOOL_WEAR。

SD42910 $SC_MIRROR_TOOL_WEAR

设置数据 **不等于** 零：

关联轴镜像的刀具长度分量，其磨损量通过符号反相也同样进行镜像。

11.3.2 磨损量的符号赋值

用设定好不等于零的设置数据 SD42920 $SC_WEAR_SIGN_CUTPOS 和 SD42930 $SC_WEAR_SIGN，可以对磨损量的符号赋值进行反向。

SD42920 $SC_WEAR_SIGN_CUTPOS

设置数据 **不等于** 零：

设置数据不等于零：如果刀具带相应的切削刃方向（车刀和铣刀，刀具类型 400），则在加工平面中的磨损量分量的符号赋值取决于刀沿位置。 如果刀具类型不带相应刀沿方向，则该设定数据没有意义。

在下表中，尺寸通过 X 标记，其符号通过 SD42920（不等于 0）反向：

刀沿位置	长度 1	长度 2
1		
2		X
3	X	X
4	X	
5		
6		
7		X
8	X	
9		

说明

通过 SD42920 和 SD42910 进行后的符号赋值相互之间无关。 比如，一个尺寸参数的符号通过两个设定数据修改，则所产生的符号保持不变。

SD42930 $SC_WEAR_SIGN

设置数据 **不等于** 零：

所有磨损量尺寸的符号都反相。 这既作用于刀具长度上，也用于其他尺寸：比如刀具半径、倒圆半径等等。

如果输入一个正的磨损尺寸值，则借此使得刀具“变短”和“变薄”，请参见章节“刀具补偿，特殊操作”，更改的设置数据将生效“。

11.3.3 激活的加工的坐标系 (TOWSTD/TOWMCS/TOWWCS/TOWBCS/TOWTCS/TOWKCS)

取决于机床的运动性能，或者是可定向刀架的当前状态，在一个这样的坐标系中所测得的磨损量被换算到或者变换到一个合适的坐标系中。

有效加工的坐标系

由下面的坐标系可以计算出刀具长度补偿，可以用此长度补偿，通过组 56 的相应 G 代码，把刀具长度分量“磨损量”计算到有效的刀具中：

- 机床坐标系（MCS）
- 基准坐标系（BCS）

- 工件坐标系（WCS）
- 刀具坐标系（TCS）
- 运动转换的刀具坐标系（KCS）

句法

```
TOWSTD
TOWMCS
TOWWCS
TOWBCS
TOWTCS
TOWKCS
```

含义

TOWSTD:	初始设定值，用于刀具长度磨损量补偿
TOWMCS:	在 MKS 中刀具长度上的补偿
TOWWCS:	在 WKS 中刀具长度上的补偿
TOWBCS:	在 BKS 中刀具长度上的补偿
TOWTCS:	在刀架参考点上的刀具长度补偿（可定向刀架）
TOWKCS:	刀头上的刀具长度补偿（运动转换）

更多信息

区别标志

在下表中列出了最重要的区别特征：

G 代码	磨损量	有效的可定向刀架
TOWSTD:	初始设定值，刀具长度	磨损量受旋转控制。
TOWMCS:	在 MCS 中的磨损量如果没有可定向的刀架被激活，则 TOWMCS 与 TOWSTD 一致。	仅旋转所生成的刀具长度的矢量，不考虑磨损量。
TOWWCS:	强 WCS 中的磨损量换算到 MCS 中。	刀具矢量计算如同在 TOWMCS 中一样，不考虑磨损量。

TOWBCS:	将 BCS 中的磨损量换算到 MCS 中。	刀具矢量计算如同在 TOWMCS 中一样，不考虑磨损量。
TOWTCS:	将刀具坐标系中的磨损量换算到 MCS 中。	刀具矢量计算如同在 TOWMCS 中一样，不考虑磨损量。

TOWWCS, TOWBCS, TOWTCS:磨损量矢量加到刀具矢量中。

线性转换

如果 MCS 是从 BCS 中通过一个线性平移而产生的，则在 MKS 中的刀具长度定义才有意义。

非线性转换

比如，如果用 TRANSMIT 激活一个非线性转换，则在 MCS 作为所要求的坐标系说明时，自动使用 BCS。

没有运动转换并且没有定向刀架

如果既没有运动变换生效，也没有可定向刀架生效，则所有四个坐标系（除 WCS 之外）均同时生效。这样只有 WKS 与其它的坐标系相区别。因为只有刀具长度需要求值，则坐标系之间的平移就没有意义。

文献：
刀具补偿的其它信息，参见：
功能手册基本功能；刀具补偿（W1）

把磨损量计算在内

设置数据 **SD42935 $SC_WEAR_TRANSFORM** 确定下面三个磨损量分量中：

- 磨损
- 精补偿总和
- 粗补偿总和

哪一个受控于适配变换的旋转，或者受控于一个可定向的刀架，如果下面 G 代码中的一个被激活：

- TOWSTD
  初始位置。刀具长度中的补偿。
- TOWMCS
  机床坐标系中的磨损值（MCS）。
- TOWWCS
  工件坐标系中的磨损值（WCS）。

- TOWBCS
  基本坐标系中的磨损值（BCS）。
- TOWTCS
  刀架装置中（T 刀架参考系）刀具坐标系中磨损量。
- TOWKCS
  在运动变换时，刀头坐标系中的磨损量。

说明

各个磨损量分量（分配到几何轴，符号求值）的求值受以下因素影响：

- 有效的工作平面
- 适配器转换
- 设定数据：
  - SD42910 $SC_MIRROR_TOOL_WEAR
  - SD42920 $SC_WEAR_SIGN_CUTPOS
  - SD42930 $SC_WEAR_SIGN
  - SD42940 $SC_TOOL_LENGTH_CONST
  - SD42950 $SC_TOOL_LENGTH_TYPE

11.3.4 刀具长度和平面更换

采用设定好不为零的设置数据 SD42940 $SC_TOOL_LENGTH_CONST，可以在平面更换时将刀具长度分量例如长度、磨损和基本尺寸等分配到车刀和磨削工具的几何轴上。

SD42940 $SC_TOOL_LENGTH_CONST

设置数据 **不等于** 零：

在工作平面更换时（G17 - G19），刀具长度分量（长度、磨损量和基准尺寸）到几何轴的分配没有改变。

下表中说明在车刀和磨削工具（刀具类型 400 到 599）时刀具长度分量到几何轴的分配：

内容	长度 1	长度 2	长度 3
17	Y	X	Z
*)	X	Z	Y
19	Z	Y	X
-17	X	Y	Z

-18	Z	X	Y
-19	Y	Z	X

*) 每个不等于 0 的值，又不等同于六个值中的一个，则作为 18 求值。

下表中说明在其它的工具（刀具类型<400 或者>599）时刀具长度分量到几何轴的分配：

加工平面	长度 1	长度 2	长度 3
*)	Z	Y	X
18	Y	X	Z
19	X	Z	Y
-17	Z	X	Y
-18	Y	Z	X
-19	X	Y	Z

*) 每个不等于 0 的值，又不等同于六个值中的一个，则作为 17 求值。

说明

表中的说明以几何轴用 X、Y、Z 命名为前提。 补偿值到轴的分配不是由轴名称决定，而是由轴顺序来决定。

11.4 在线刀具补偿

11.4.1 定义多项式函数(FCTDEF)

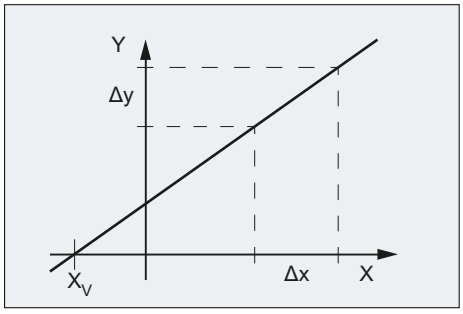
在一些修整方案中，比如：使用修整轮时，砂轮半径会随着修整轮的进给连续线性减小。为此需要使用由修整轮进给量和对应长度方向上的磨损量构成的线性函数。线性函数的定义通过最多定义三条直线的预定义程序 FCTDEF(...)FCTDEF(...) 进行。

直线

$y = f(x) = a_0 + a_1 * x_1$

a_1：直线斜率，使用 $a_1 = \Delta x / \Delta y$

a_0：直线沿 X 轴的偏移，使用 $a_0 = -a1 * X_V$



句法

```
FCTDEF(<Func>,<LLimit>,<ULimit>,<a0>,<a1>,<a2>,<a3>)
```

含义

FCTDEF(...):	PUTFTOCF(...) 的多项式函数定义： $y = f(x) = a_0 + a_1*x + a_2*x^2 + a_3*x^3$	
<Func>:	函数编号	
	数据类型：	INT
	值范围：	1, 2, 3
<LLimit>:	下限值	
	数据类型：	REAL
<ULimit>:	上限值	
	数据类型：	REAL

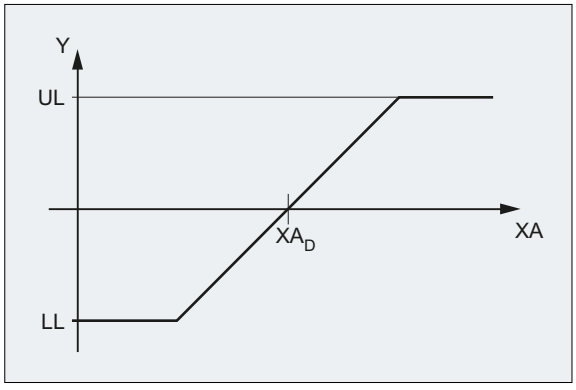
<a0>,<a1>,<a2>,<a3>:	多项式函数的系数	
	数据类型：	REAL

示例

定义

- 函数编号：1
- 下限和上限值：-100、100
- 特性曲线斜率：$a_1 = 1$
- 工作点应位于特性曲线中心点。特性曲线应依据 WCS 中 XA 轴的设定位置于 NC 程序中定义函数的时间点向 Y 的负方向偏移：$a_0 = -a_1 * XA_D = -1 * \$AA_IW$
- $a_2 = a_3 = 0$

特性曲线



UL 上限值

LL 下限值

XA_D NC 程序中定义函数的时间点上 XA 轴的设定值

编程

程序代码	注释
FCTDEF(1,-100,100,-$AA_IW[XA],1)	；函数定义

11.4.2 连续写入在线刀具补偿(PUTFTOCF)

通过预定义程序 PUTFTOCF(...) 可为之前使用 FCTDEF(...) (页 457) 定义的多项式函数进行在线刀具补偿。

说明

在线刀具补偿也可通过同步动作实现。
更多信息参见功能手册之同步动作。

句法

```
PUTFTOCF(<Func>,<RefVal>,<ToolPar>,<Chan>,<Sp>)
```

含义

PUTFTOCF(...):	连续写入在线刀具补偿，逐段生效，依据使用 FCTDEF(...) 定义的多项式函数	
<Func>:	函数编号，使用 FCTDEF(...) 定义函数时确定	
	数据类型：	INT
	值范围：	1, 2, 3
<RefVal>:	参考值，从该值可求出补偿（如轴的设定值）	
	数据类型：	VAR REAL
<ToolPar>:	磨削参数的编号（长度 1、2 或 3），补偿值应计入该参数。	
	数据类型：	INT
<Chan>:	通道号，在线刀具补偿在该通道中生效 **提示：** 只有当补偿无需在激活的通道中生效时，才需要给定该数据。	
	数据类型：	INT
<Sp>:	主轴号，在该主轴上在线补偿将生效 **提示：** 只有当需要补偿针对未激活的磨削砂轮，而不是针对激活的、使用中的刀具生效时，才需要给定该数据。	
	数据类型：	INT

11.4.3 不连续写入在线刀具补偿(PUTFTOC)

功能

通过预定义程序 PUTFTOC(...) 可实现使用固定补偿值的在线刀具补偿。

句法

```
PUTFTOC(<CorrVal>,<ToolPar>,<Chan>,<Sp>)
```

含义

PUTFTOC(...):	写入在线刀具长度补偿	
<CorrVal>:	在磨损参数中添加的补偿值	
	数据类型：	REAL
<ToolPar>:	磨削参数的编号（长度 1、2 或 3），补偿值应计入该参数。	
	数据类型：	INT
<Chan>:	通道号，在线刀具补偿在该通道中生效 **提示：** 只有当补偿无需在激活的通道中生效时，才需要给定该数据。	
	数据类型：	INT
<Sp>:	主轴号，在该主轴上在线补偿将生效 **提示：** 只有当需要补偿针对未激活的磨削砂轮，而不是针对激活的、使用中的刀具生效时，才需要给定该数据。	
	数据类型：	INT

11.4.4 激活/撤销在线刀具补偿(FTOCON/FTOCOF)

通过 G 指令 FTOCON 和 FTOCOF 激活或取消在线刀具补偿。

句法

```
FTOCON
...
```

```
FTOCOF
```

含义

FTOCON:	激活在线刀具补偿 该指令必须在在线刀具补偿应激活的通道中编程。
FTOCOF:	撤销在线刀具补偿 该指令必须在在线刀具补偿应撤销的通道中编程。 **提示：** 使用指令 FTOCOF，刀具补偿不会被继续走完。但在切削专用补偿数据中使用 PUTFTOC/PUTFTOCF 计算的值会保持不变。 为彻底取消在线刀具补偿，还必须在 FTOCOF 之后选择并取消该刀具 (T...)。

11.5 3D 刀具半径补偿

11.5.1 选择用于 3D 圆周铣削的 3D 刀具半径补偿（CUT3DC、CUT3DCD、ISD）

通过模态生效的 G 指令 CUT3DC 或 CUT3DCD 选择用于 3D 圆周铣削的、不考虑分界面的 3D 刀具半径补偿（3D-TRC）。

原本的激活通过 G41 或 G42 实现。通过 G40 取消刀具半径补偿。

句法

```
G41/G42 ORIC/ORID ISD=...CUT3DC/CUT3DCD CDOF2 X...Y...Z...
...
G40 X...Y...Z...
```

含义

CUT3DC:	用于圆周铣削的 3D 刀具半径补偿（仅在 5 轴转换生效时）	
CUT3DCD:	基于差分刀具的用于圆周铣削的 3D 刀具半径补偿（仅在 5 轴转换生效时） 通过刀具参数 $TC_DP15 定义半径差。	
G41/G42 X...Y...Z...:	激活刀具半径补偿	
	G41:	刀具半径补偿，轮廓左边
	G42:	刀具半径补偿，轮廓右边
	提示： 必须在一个线性程序段（G0/G1）中进行激活。	
CDOF2:	关闭 3D 圆周铣削的碰撞监控	
ORIC/ORID:	通过 G 指令 ORIC 和 ORID 定义外角处有定向变化时的特性。	
	ORIC:	将外角处的定向变化叠加至待插入的圆弧程序段。
	ORID:	外拐角上的定向变化在将要插入的圆弧程序段之前执行。

ISD=<值>：	通过地址 ISD 能够在圆周铣削和**生效**的 3D 刀具半径补偿中改变刀具的插入深度	
	<值>：	插入深度的长度
G40 X...Y...Z...：	取消刀具半径补偿 **提示：** 必须在一个包含几何轴运动的线性程序段（G0/G1）中进行取消。	

说明

系统在逼近程序段中，即通常在包含 G41 或 G42 的程序段中对用于选择 3D 刀具半径补偿的 G 指令进行分析。

G41 或 G42 也可以编写在不含补偿相关几何轴中之运行的程序段中。在此情形下，这种程序段后的第一个程序段便是逼近程序段。

在刀具半径补偿生效时的 3D 刀具半径补偿方案切换会被忽略，且不触发报警。

示例

程序代码	注释
	；定义刀具 D1：
$TC_DP1[1,1]=120	；类型（立铣刀）
$TC_DP3[1,1]=20	；长度补偿矢量
$TC_DP6[1,1]=8	；半径
N10 X0 Y0 Z0 T1 D1 F12000	；选择刀具。
N20 TRAORI(1)	；激活坐标转换
N30 G42 ORIC ISD=10 **CUT3DC** G64 X30	；激活 3D 圆周铣削，
	；外角处的定向变化为连续，
	；插入深度：10 毫米
N40 ORIWKS A30 B15	；通过给定轴位置实现拐角处的定向变化。
N50 Y20 A3=1 C3=1	；包含定向变化的运行程序段，
	；通过方向矢量给定定向。
N60 X50 Y30	；包含恒定定向的运行程序段。
N70 Y50 A3=0.5 B3=1 C3=5	；包含定向变化的运行程序段。
N80 M63	；无运行信息的程序段。
N90 X0 ISD=20	；包含插入深度更改的运行程序段。
N100 G40 Y0	；取消刀具半径补偿。
N110 M30	

更多信息

轨迹和定向

这里使用的圆周铣削方案通过设定一条轨迹（引导线）和对应的定向来实现。就这类加工而言，轨迹上的刀具形状无意义。起决定性作用的仅是刀具作用点上的半径。

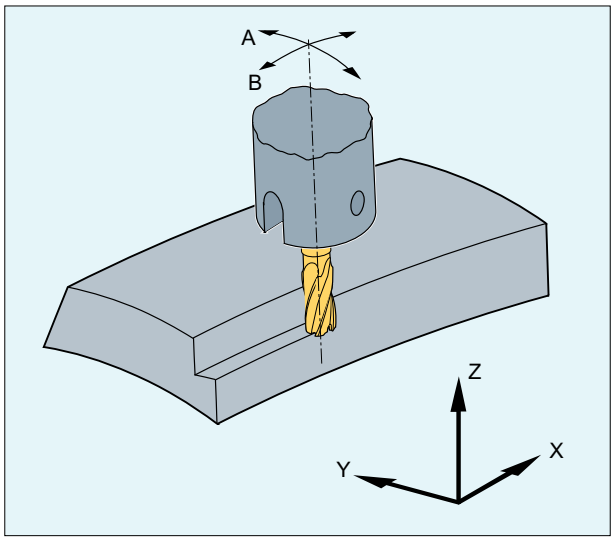


图 11-1　圆周铣削

逼近特性

在刀具半径补偿的 3D 方案中，逼近特性始终为 NORM。

外角处的特性

在采用 3D 刀具半径补偿的圆周铣削中，与采用 2½D 刀具半径补偿时的情况类似，在外角处对 G 指令组 18 的 G 指令（刀具补偿拐角特性）进行分析：

- G450：过渡圆（刀具在圆弧轨迹上绕工件拐角运行）
  与 2½D 刀具半径补偿中的解决方案的不同之处在于，在外角处插入的轮廓元素始终是半径为 0 的圆弧，刀具半径以同对编写的任意其他轨迹相同的方式对这个圆弧生效。无法在圆弧的位置上插入锥形截面。故地址 DISC 在此情形下无意义，且不会被分析。
- G451：等距线的交点（刀具在工件拐角中切削）
  通过延长两个参与程序段的偏移曲线，并且在垂直于刀具定向的平面中确定曲线在拐角处的交点，以此来确定交点。

在参与的运行程序段之间插入有至少一个包含刀具定向变化的程序段的情况下，不使用交点法（G451）。在此情形下，在拐角处总是插入圆弧。

外角处的定向变化的特性

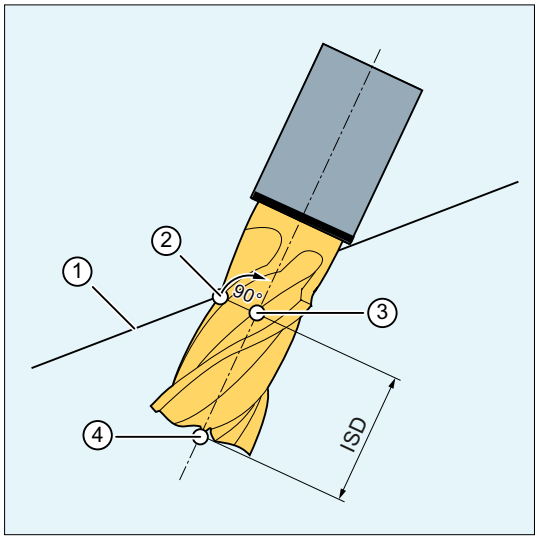
通过 G 指令 ORIC 和 ORID 可定义：在两个构成拐角的运行程序段之间编写的定向变化是在插入的圆弧程序段开始前执行（ORID），还是与这个圆弧程序段同时执行（ORIC）。

插入深度

铣刀的插入深度为刀尖与铣刀辅助点之间的距离。

铣刀辅助点为所编程轨迹上铣刀加工点在刀具纵向轴上的垂直投影。

因此通过插入深度，可以设置刀具柱面上的加工点位置。



① 编程的轨迹
② 铣刀加工点
③ 铣刀辅助点
④ 铣刀刀尖
ISD 插入深度 (InSertion Depth)

图 11-2 插入深度

以一把非标刀具为基准的刀具半径补偿

通过指令 CUT3DCD 激活以非标刀具为基准的用于圆周铣削的 3D 刀具半径补偿。如果轮廓编程以一把标准刀具的中心点轨迹为基准，但加工却采用一把非标刀具，则必须使用该刀具半径补偿。在计算 3D 刀具半径补偿时，只算入生效刀具半径的磨损值 ($TC_DP15) 及编程的刀具半径偏移 OFFN 和 TOFFR。不计算生效刀具的基圆半径 ($TC_DP6)。

带有斜侧壁的凹槽铣削，用于使用 CUT3DC 进行圆周铣削

在此 3D 刀具半径补偿中，通过朝向待加工面的表面法线进给，对铣刀半径的偏差进行补偿。当插入深度 ISD 保持相同时，铣刀端面所在的平面保持不变。在此情形下，半径例如小于标准刀具的铣刀将不到达构成分界面的凹槽底部。对于刀具的自动进给而言，控制系

统必须已知该分界面，参见章节“考虑分界面的 3D 圆周铣削（CUT3DCC、CUT3DCCD） (页 471)”。

精优曲面（Advanced Surface）/臻优曲面（Top Surface）

说明

在组合使用刀具半径补偿 CUT3DCD 和需获得授权的选件“精优曲面”或“臻优曲面”时，请注意与“精优曲面”/“臻优曲面”相关的推荐设置！

在 SIOS 网站上提供了专门的测试程序进行设定数据的测试：

- 精优曲面 (https://support.industry.siemens.com/cs/cn/zh/view/78956392)测试程序
- 臻优曲面 (https://support.industry.siemens.com/cs/cn/zh/view/109738423/en)测试程序

11.5.2 选择用于 3D 端面铣削的 3D 刀具半径补偿（CUT3DF、CUT3DFS、CUT3DFF、CUT3DFD）

通过模态生效的 G 指令 CUT3DF、CUT3DFS、CUT3DFF 或 CUT3DFD 选择用于 3D 端面铣削的 3D 刀具半径补偿（3D-TRC）。

原本的激活通过 G41 或 G42 实现。

为了计算刀具半径补偿，在 3D 端面铣削中需要定义待加工平面的表面法线。这必须在包含 G41 或 G42 的程序段中通过地址 A4、B4、C4 和 A5、B5、C5 实现。

通过 G40 取消刀具半径补偿。

句法

```
G41/G42 ORIC/ORID CUT3DF/CUT3DFS/CUT3DFF/CUT3DFD
X...Y...Z...A4=...B4=...C4=...A5=...B5=...C5=...
...
G40 X...Y...Z...
```

含义

CUT3DFS:	用于采用恒定定向的端面铣削的 3D 刀具半径补偿。刀具定向通过 G17 - G19 定义且不受框架影响。
CUT3DFF:	用于采用恒定定向的端面铣削的 3D 刀具半径补偿。刀具定向为通过 G17 - G19 定义、且视情况而定通过框架旋转的方向。

CUT3DF:	用于带有定向变化的端面铣削的 3D 刀具半径补偿（仅在 5 轴转换生效时）	
CUT3DFD:	基于差分刀具的用于带有定向变化的端面铣削的 3D 刀具半径补偿（仅在 5 轴转换生效时） 通过刀具参数 $TC_DP15 定义半径差。 **提示：** CUT3DFD 只允许和“3D 端面铣削中的表面法线平滑”组合使用。它可以通过 CYCLE832(...) (页 861) 调用授权功能“臻优曲面”时激活。	
G41/G42 X...Y...Z...:	激活刀具补偿 在 3D 端面铣削中， G41 和 G42 时的特性相同。 **提示：** 必须在一个线性程序段（G0/G1）中进行激活。	
A4/5=...B4/5=...C4/5=...:	定义待加工平面的表面法线	
	A4=...B4=...C4=...:	在程序段开始处定义
	A5=...B5=...C5=...:	在程序段结束处定义
ORIC/ORID:	通过 G 指令 ORIC 和 ORID 定义外角处有定向变化时的特性。	
	ORIC:	将外角处的定向变化叠加至待插入的圆弧程序段。
	ORID:	在圆弧程序段之前执行定向变化。
G40 X...Y...Z...:	取消刀具半径补偿 **提示：** 必须在一个包含几何轴运动的线性程序段（G0/G1）中进行取消。	

说明

G41 或 G42 也可以编写在不含补偿相关几何轴中之运行的程序段中。在此情形下，这种程序段后的第一个程序段便是逼近程序段。

在刀具半径补偿生效时的 3D 刀具半径补偿方案切换会被忽略，且不触发报警。

示例

示例 1：3D 端面铣削程序中编程 CUT3DF

程序代码	注释
N10	；定义刀具 D1：
N20 $TC_DP1[1,1]=121	；刀具类型（环形铣刀）
N30 $TC_DP3[1,1]=20	；长度补偿
N40 $TC_DP6[1,1]=5	；半径
N50 $TC_DP7[1,1]=3	；倒圆半径
N60	
N70	
N80 X0 Y0 Z0 A0 B0 C0 G17 T1 D1 F12000	；选择刀具。
N90 TRAORI(1)	；选择坐标转换。
N100 B4=-1 C4=1	；平面定义。
N110 **G41** ORID **CUT3DF** G64 X10 Y0 Z0	；激活刀具补偿。
N120 X30	
N130 Y20 A4=1 C4=1	；外角，重新定义平面。
N140 B3=1 C3=5	；通过 ORID 进行定向变化。
N150 B3=1 C3=1	；通过 ORID 进行定向变化。
N160 X-10 A5=1 C5=2 ORIC	
N170 A3=-2 C3=1	；通过 ORIC 进行定向变化
N180 A3=-1 C3=1	；通过 ORIC 进行定向变化
N190 Y-10 A4=-1 C4=3	；重新定义平面。
N200 X-20 Y-20 Z10	；包含前一程序段的内角。
N210 X-30 Y10 A4=1 C4=1	；内角，重新定义平面。
N220 A3=1 B3=0.5 C3=1.7	；通过 ORIC 进行定向变化
N230 X-20 Y30 A4=1 B4=-2 C4=3 ORID	
N240 A3 = 0.5 B3=-0.5 C3=1	；定向变化。
N250 X0 Y30 C4=1	；轨迹运行，新平面， ；通过相对编程进行定向。
N260 BSPLINE X20 Z15	；样条开端，定向的相对编程
N270 X30 Y25 Z18	；在样条期间保持生效
N280 X40 Y20 Z13	
N290 X45 Y0 PW=2 Z8	
N300 Y-20	
N310 G2 ORIMKS A30 B45 I-20 X25 Y-40 Z0	；螺旋线，通过轴编程进行定向。
N320 G1 X0 A3=-0.123 B3=0.456 C3=2.789 B4=-1 C4=5 B5=-1 C5=2	；轨迹运行，定向，非恒定平面。
N330 X-20 G40	；取消刀具半径补偿。
N340 M30	

示例 2：在一个从 CAD 系统中生成的 NC 程序段落中编程 CUT3DFD

程序代码	注释
N01 G710	

程序代码	注释
N03 T="12"	
N06 S5305 M03	
N07 G642	
; 移动到 MCS 中的起始位置，同时考虑刀长。	
G00 G90 X-250.62787 Y-38.37944 A=DC(253.12719) B-12.49543	
G00 G90 Z251.80052	
; MCS 中的定位结束。	
;	
TRAORI(1)	; 选择坐标转换。
G500	
D1	
CYCLE832(0.01, _TOP_SURFACE_SMOOTH_ON + _ORI_FINISH, 1)	; 调用 CYCLE832，其中： ; 轮廓误差 = 0.01 mm， ; 加工方式：臻优曲面，带平滑 ; 输入了定向公差的精加工， ; 定向公差 = 1 度
CUT3DFD	
N08 G90 G94	
N09 G00 X-269.21195 Y128.32027 Z1.18577 A3=-.216361688 B3=.934284397 C3=-.283373051	; 从 N09 到 N10，刀具以恒定定向量快速移动到工件。
N10 G00 X-251.90301 Y53.57752 Z23.85561	
N11 G01 X-247.57578 Y34.89183 Z29.52308 F50000.00000	; 从 N11 到 N21，刀具缓慢移动。刀具已经十分接近工件，另外，由于 G01 生效，现在 CYCLE832 中的模具制造设置（比如 COMPSURF）激活。模具制造设置“整定阶段”中的行程大概应是轮廓误差的 1000 倍，本例中为 10 mm。
N12 X-247.69126 Y33.82182 Z24.78219 F1061.00000	
N13 X-247.76560 Y33.13299 Z21.73022	
N14 X-247.82755 Y32.55897 Z19.18691	
N15 X-247.87918 Y32.08062 Z17.06748	
N16 X-247.92220 Y31.68200 Z15.30129	
N17 X-247.95805 Y31.34981 Z13.82947 A3=-.216361686 B3=.934284391 C3=-.283373071	
N18 X-247.98792 Y31.07299 Z12.60295 A3=-.216360662 B3=.934280801 C3=-.283385691	
N19 X-248.01282 Y30.84230 Z11.58085 A3=-.216336015 B3=.934194446 C3=-.283689030	
N20 X-248.03357 Y30.65006 Z10.72910 A3=-.216233089 B3=.933833626 C3=-.284952647	
N21 X-248.05086 Y30.48986 Z10.01931 A5=-.060687572 B5=.974940255 C5=-.214029243 A3=-.215712821 B3=.932005189 C3=-.291263295	
N22 **G41** X-248.06237 Y30.32400 Z9.36695 A5=-.060431854 B5=.973045457 C5=-.222554556 A3=-.214974689 B3=.929398552 C3=-.300007025 F1061.03295	; 从 N22 开始，才首次在整个程序段中完整定义表面法线，“完整定义”指：在 N21 段尾有表面法线，在 N22 段首和 N22 段尾也都有表面法线。因此，现在符合通过 G41/G42 来激活刀具补偿的条件。

程序代码	注释

```
N23 X-248.07130 Y30.15119 Z8.71082
    A5=-.060165696 B5=.971048883 C5=-.231179920
    A3=-.214177198 B3=.926684940 C3=-.308841625
N24 X-248.07829 Y29.97126 Z8.05094
    A5=-.059884286 B5=.968941717 C5=-.239928784
    A3=-.213318480 B3=.923853466 C3=-.317789237
N25 X-248.08317 Y29.78487 Z7.38844
    A5=-.059584206 B5=.966718449 C5=-.248807482
    A3=-.212397895 B3=.920898045 C3=-.326854594
N26 X-248.08578 Y29.59254 Z6.72679
    A5=-.059263963 B5=.964380907 C5=-.257793037
    A3=-.211418355 B3=.917822366 C3=-.336012474
...
```

说明

在 3D 端面铣削中使用 CUT3DFD 时，定义了表面法线后，才能通过 G41/G42 来激活刀具补偿。没有定义表面法线而直接编程 G41/G42 时，系统会报警。

更多信息

3D 端面铣削

对于这种 3D 铣削类型，需要逐行描述工件表面上的 3D 轨迹。通常在 CAM 中以将刀具形状和刀具尺寸考虑在内的方式执行计算。除了 NC 程序段外，后处理器将刀具定向（当 5 轴转换生效时）和所需的 3D 刀具补偿的 G 指令写入零件程序。这样机床操作人员就可以使用（与计算 NC 轨迹所使用的刀具不同的）略微小一些的刀具。

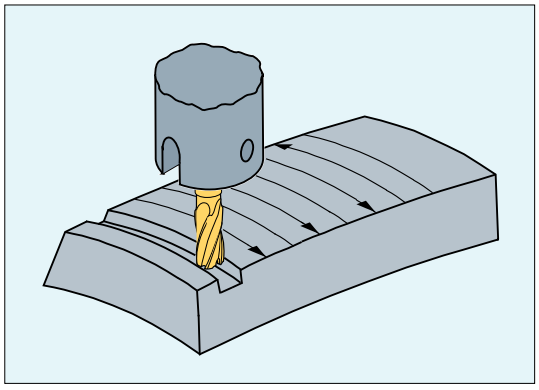


图 11-3　端面铣削

逼近特性

在刀具半径补偿的 3D 方案中，逼近特性始终为 NORM。

外角处的特性

在端面铣削中将外角作为半径为 0 的圆弧处理，其中圆弧平面由第一程序段的末尾切线与第二程序段的起始切线夹紧。故在程序段过渡中可采用定向变化。因此，在外角处总是将一个圆弧作为轮廓元素插入。交点法不可用于端面铣削。

外角处的定向变化的特性

通过 G 指令 ORIC 和 ORID 可定义：在两个构成拐角的运行程序段之间编写的定向变化是在插入的圆弧程序段开始前执行（ORID），还是与这个圆弧程序段同时执行（ORIC）。

以一把非标刀具为基准的刀具半径补偿

通过指令 CUT3DFD 可以激活以非标刀具为基准的 3D 刀具半径补偿。如果轮廓编程以一把标准刀具的中心点轨迹为基准，但加工却采用一把非标刀具，则必须使用该刀具半径补偿。在计算 3D 刀具半径补偿时，只算入生效刀具半径的磨损值 ($TC_DP15) 及编程的刀具半径偏移 OFFN 和 TOFFR。不计算生效刀具的基圆半径 ($TC_DP6)。

CUT3DFD 只允许和“3D 端面铣削中的表面法线平滑”组合使用。它可以通过 CYCLE832(...) (页 861) 调用授权功能“臻优曲面”时激活。该指令必须在 G41/G42 刀具补偿**之前**激活，也就是说：它不能等到刀具马上就要开始加工了才激活，而是必须在此前的一段行程中就激活，该行程大概是轮廓误差的 1000 倍，比如：1000 x 0.01 mm = 10 mm）。按相反顺序操作即可取消：首先使用 G40 关闭刀具补偿，然后在一段行程后，大约为轮廓公差的 1000 倍，通过 CUT2D 等指令进行取消。

“3D 端面铣削中的表面法线平滑”的使用还需要激活功能“通过多项式来插补表面法线”：

MD28291 $MC_MM_SMOOTH_SURFACE_NORMALS = TRUE

说明

在组合使用“3D 端面铣削+ CUT3DFD”和“臻优曲面”时，请注意与“臻优曲面”相关的推荐设置！

在 SIOS 网站上提供 (https://support.industry.siemens.com/cs/cn/zh/view/109738423/en) 了专门的测试程序进行设定数据的测试。

11.5.3 考虑分界面的 3D 圆周铣削（CUT3DCC、CUT3DCCD）

在带有连续或者恒定的刀具定向改变的 3D 圆周铣削时，经常编程一个定义的标准刀具的刀具中心点轨迹。因为实际操作上合适的标准刀具常常不可以使用，可以使用一个和标准刀具偏差不是太多 (≤ 5 %) 的刀具。

CUT3DCCD 指令可以为实际使用非标刀具的情况考虑一分界面，该分届面描述了使用标准刀具时的运行情况。NC 程序所描述的是标准刀具的中心点轨迹。

CUT3DCC 指令可以为圆柱形刀具考虑一分界面，该分届面描述了使用标准刀具时的运行情况。该 NC 程序所描述的是加工面上的轮廓。

如同 3D 端面铣削中那样，通过 A4、B4、C4 和 A5、B5、C5 来给定分界面的表面法线矢量。

句法

```
G41/G42 CUT3DCCD/CUT3DCC CDOF2 X...Y...Z...A4=...B4=...C4=...A5=...B5=...C5=...
...
G40 X...Y...Z...
```

含义

CUT3DCCD：	一种 3D 刀具半径补偿方式，适用于圆周铣削，考虑了一个分界面，位于非标刀具的中心点轨迹上。向分界面进给	
CUT3DCC：	一种 3D 刀具半径补偿方式，适用于圆周铣削，以 3D 形式进行半径补偿，并考虑了一个分界面：加工面上的轮廓	
G41/G42 X...Y...Z...：	激活刀具半径补偿	
	G41：	刀具半径补偿，轮廓左边
	G42：	刀具半径补偿，轮廓右边
	提示： 必须在一个线性程序段（G0/G1）中进行激活。	
CDOF2：	关闭 3D 圆周铣削的碰撞监控	
A4/5=...B4/5=...C4/5=...：	定义分界面的表面法线	
	A4=...B4=...C4=...：	在程序段开始处定义
	A5=...B5=...C5=...：	在程序段结束处定义
G40 X...Y...Z...：	取消刀具半径补偿 **提示：** 必须在一个包含几何轴运动的线性程序段（G0/G1）中进行取消。	

说明

系统在逼近程序段中，即通常在包含 G41 或 G42 的程序段中对用于选择 3D 刀具半径补偿的 G 指令进行分析。

G41 或 G42 也可以编写在不含补偿相关几何轴中之运行的程序段中。在此情形下，这种程序段后的第一个程序段便是逼近程序段。

在刀具半径补偿生效时的 3D 刀具半径补偿方案切换会被忽略，且不触发报警。

示例

程序代码	注释
N10 $TC_DP1[1,1]=120	; 圆柱铣刀
N20 $TC_DP6[1,1]=10	
N30 $TC_DP15[1,1]=-3	
...	
; 以圆柱铣刀和 CUT3DCCD 加工	
N110 TRAORI	; 激活坐标转换
N120 A4=0 B4=0 C4=1	; 在段首处定义分界面的表面法线
N130 X0 Y0 Z0 A0 C0 T1 D1 F20000	
N140 X10 Y0 Z0 G41 **CUT3DCCD** CDOF2 G64	; 激活 3D 圆周铣削，考虑分界面并关闭碰撞监控
N150 X20	
N160 X30 A45	; 钝角==> 无进给
N170 X40 A-45	; 锐角==> 进给
N180 X55	
N190 Y10 Z10	; 沿刀具方向运动
N200 Y20	
N210 C45	; 单纯的定向变化
N220 Y30 C90	
N230 A5=-1 B5=0 C5=2 Y40	; 表面变化
N240 Y50 G40	; 取消刀具半径补偿。
...	

更多信息

刀具类型

系统会对刀具类型（刀具参数 $TC_DP1）进行分析。仅允许带有圆柱形刀柄的铣刀（圆柱形铣刀或立铣刀、环形铣刀以及在极限情况下允许圆柱形模具铣刀）。这对应刀具类型 1 - 399，特例为编号 111 以及 155 至 157。

带圆角功能的标准刀具

标准刀具的圆角功能通过刀具参数 $TC_DP7 描述。从刀具参数 $TC_DP16 可得出真实刀具相对于标准刀具的圆角偏差。

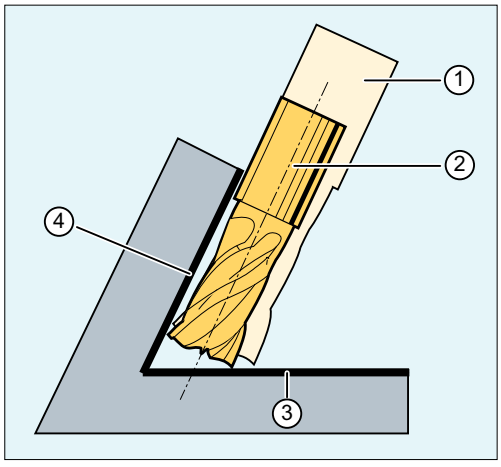
示例：半径较标准刀具有所减小的环形铣刀

刀具类型	刀柄半径（R）	拐角半径（r）
带圆角功能的标准刀具	R = $TC_DP6	r = $TC_DP7
带圆角功能的真实刀具 刀具类型 121 和 131 环形铣刀（带圆角功能的立铣刀）	R' = $TC_DP6 + $TC_DP15 + OFFN	r' = $TC_DP7 + $TC_DP16

在这个例子中，$TC_DP15 + OFFN 和 $TC_DP16 都为负。

采用 CUT3DCCD 的 3D 刀具半径补偿：进给至分界面的刀具中心点轨迹

如果使用一个和合适的标准刀具相比半径较小的刀具，那么一个纵向进给的铣刀会继续运行，直到再次碰到槽底。藉此将由加工面和分界面所构成的拐角去除一定程度，视刀具所允许的程度而定。在此，关系到圆周铣削和端面铣的混合加工方式。类似于半径减小的刀具，在采用半径增大的刀具时，沿相反方向进行相应的进给。



① 标准刀具
② 半径较小的刀具进给至分界面
③ 分界面
④ 加工面

相对于 G 指令组 22 的所有其他刀具补偿，为 CUT3DCCD 给定的刀具参数 $TC_DP6 对于刀具半径没有意义，且不影响产生的补偿。补偿偏移为刀具半径的磨损值（刀具参数 $TC_DP15）与为计算相对分界面的垂直偏移而编写的刀具偏移 OFFN 的和。

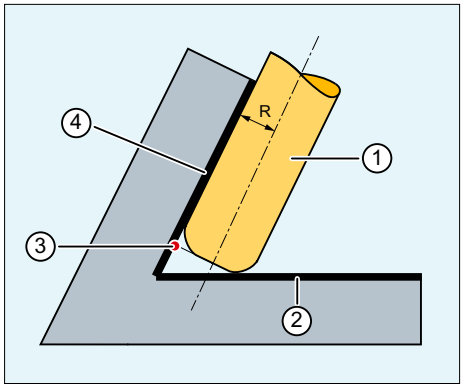
需要加工的表面在轨迹的左边或者右边，不可以从生成的零件程序中得知。因此从原始刀具的一个正半径和一个负磨损值得出。一个负向的磨损值总是描述一个缩小直径的刀具。

圆柱形刀具的使用

在使用圆柱形刀具时，只有当加工面和分界面构成一个锐角（小于 90 度）时，才需要横向进给。如果使用环形铣刀（带圆角功能的立铣刀），那么铣刀不仅在锐角时也在钝角时需要一个沿刀具纵向的进给。

采用 CUT3DCC 的 3D 刀具半径补偿：加工面上的轮廓

如果 CUT3DCC 与一个环形铣刀已激活，则编写的轨迹以具有相同直径的虚拟圆柱形铣刀为基准。当使用一个环形铣刀时，由此得出的轨迹参考点显示于下图中。



① 环形铣刀
② 分界面
③ 轨迹参考点
④ 加工面
R 刀柄半径（刀具半径）

加工面和分解面之间的夹角也允许在一个程序段中从锐角过渡到钝角，反之亦然。

相对于标准刀具，使用的实际刀具既可以大些，也可以小些。得出的角半径不可为负，且得出的刀具半径前置符号必须保留。

就 CUT3DCC 而言，NC 零件程序基于加工面上的轮廓。在此，同传统刀具半径补偿一样使用整个半径，其由以下分量组成：

- 刀具半径（刀具参数 $TC_DP6）
- 磨损值（刀具参数 $TC_DP15）
- 为计算相对分界面的垂直偏移而编写的刀具偏移 OFFN

分界面的位置由以下差值决定：

标准刀具的尺寸 - 刀具半径（刀具参数 $TC_DP6）

精优曲面（Advanced Surface）/臻优曲面（Top Surface）

说明

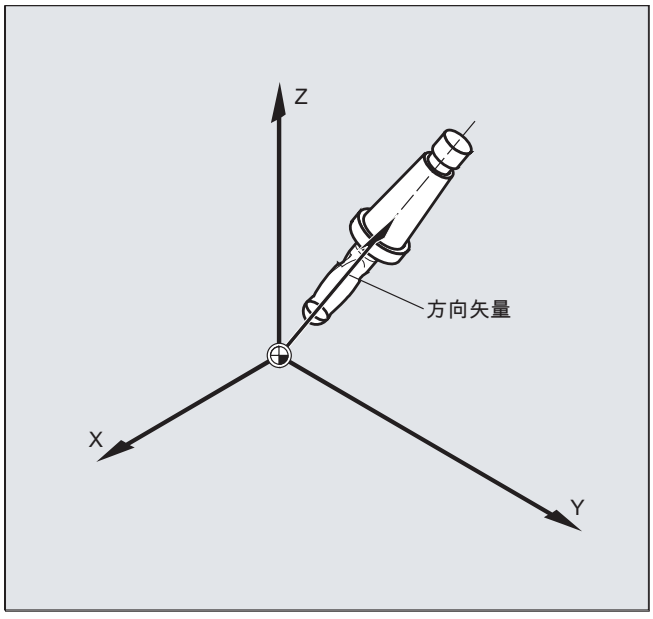
在组合使用刀具半径补偿 CUT3DCC / CUT3DCCD 和需获得授权的功能“精优曲面”或“臻优曲面”时，请注意与“精优曲面”/“臻优曲面”相关的推荐设置！

在 SIOS 网站上提供了专门的测试程序进行设定数据的测试：

- 精优曲面 (https://support.industry.siemens.com/cs/cn/zh/view/78956392)测试程序
- 臻优曲面 (https://support.industry.siemens.com/cs/cn/zh/view/109738423/en)测试程序

11.6 刀具定向(ORIC, ORID, OSOF, OSC, OSS, OSSE, ORIS, OSD, OST)

刀具定向可以理解为在空间中刀具的几何取向。使用一个 5 轴加工机床时刀具定向可以通过程序指令来设置。



根据不同刀具定向的插补类型会构成用 OSD 和 OST 激活的定向平滑。

矢量插补生效时，也可通过矢量插补来插补经过平滑的定向曲线。与此相反，回转轴插补生效时，则直接通过回转轴运动平滑定向。

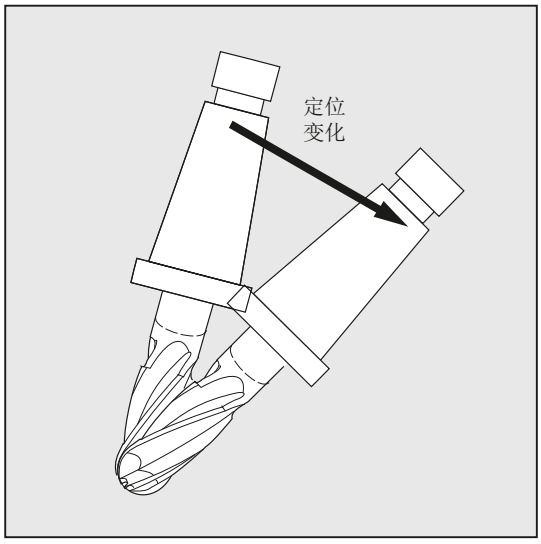
编程

定向变化的编程：

刀具定向的变化可以通过以下方式来编程：

- 直接编程回转轴 A, B, C（回转轴插补）
- 欧拉角或者 RPY 角
- 方向矢量（通过数据 A3 或 B3 或 C3 进行矢量插补）
- LEAD/TILT(端面铣削)

参考坐标系既可以是机床坐标系 (ORIMKS) 也可以是当前的工件坐标系 (ORIWKS)。



刀具定向编程：

ORIC:	定向和轨迹运动并行
ORID:	定向和轨迹运动先后进行
OSOF:	没有定向平滑
OSC:	定向恒定
OSS:	只在程序段开始处进行定向平滑
OSSE:	在程序段开始和结束处进行定向平滑
ORIS:	启用定向平滑后方向的改变速度，单位为度或毫米（适用于 OSS 和 OSSE）
OSD:	平滑定向，通过以下设定数据来规定平滑长度： SD42674 $SC_ORI_SMOOTH_DIST
OST:	平滑定向，矢量插补中在以下设定数据中规定角度公差（度）： SD42676 $SC_ORI_SMOOTH_TOL 在回转轴插补中，此处规定的公差被视为定向轴的最大公差。

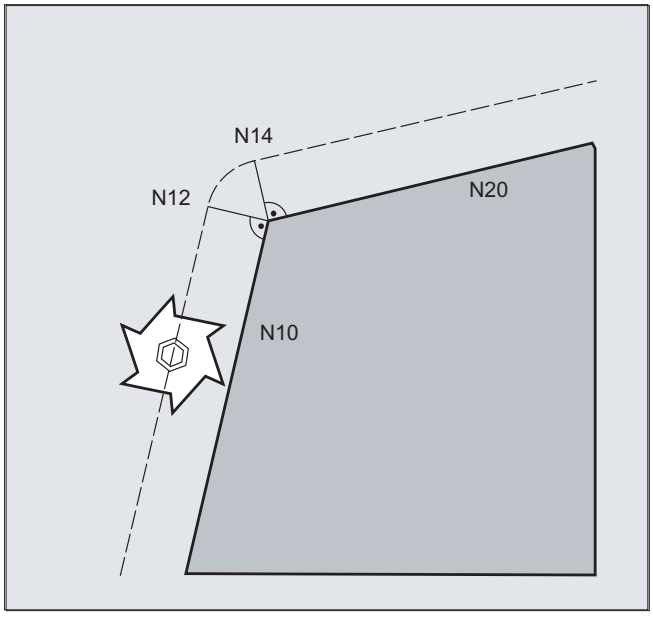
说明

所有用于平滑刀具定向的指令 (OSOF, OSC, OSS, OSSE, OSD 和 OST)都包含在 G 代码组 34 中。这些指令都是模态生效指令，即只有其中一个指令生效。

示例

示例 1：ORIC

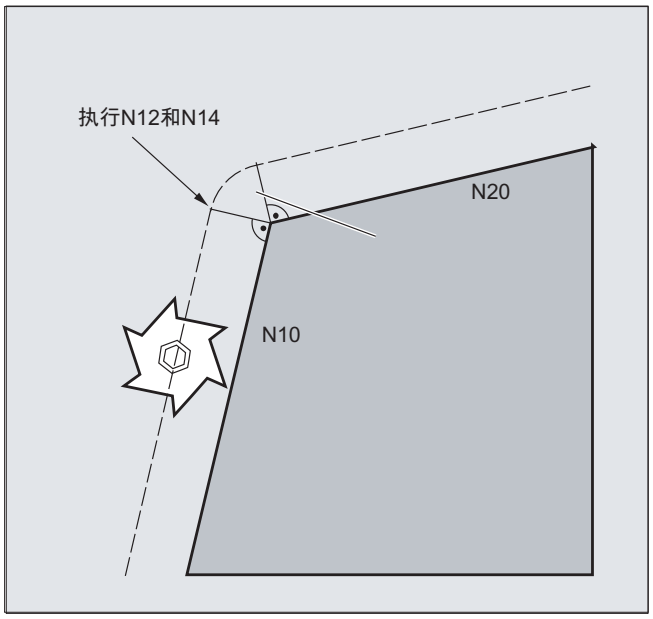
如果在运动程序段 N10 和 N20 之间已经编程了两个或者多个带有定向变化的程序段（例如 A2=...B2=...C2=...）并且 ORIC 已激活，则插入的圆程序段将根据角度变化值划分到这些中间程序段上。



程序代码	注释
ORIC	
N8 A2=… B2=… C2=…	
N10 X… Y… Z…	
N12 C2=… B2=…	；外角上插入的圆弧程序段会根据定向改变量分配到 N12 和 N14 上。圆周运动和定向变化此时会同时执行。
N14 C2=… B2=…	
N20 X =…Y=… Z=… G1 F200	

示例 2：ORID

如果 ORID 已激活，则两个运动程序段之间的所有程序段将在第一个运动程序段结束时执行。带有恒定定向的圆弧程序段将直接在第二个运动程序段之前执行。



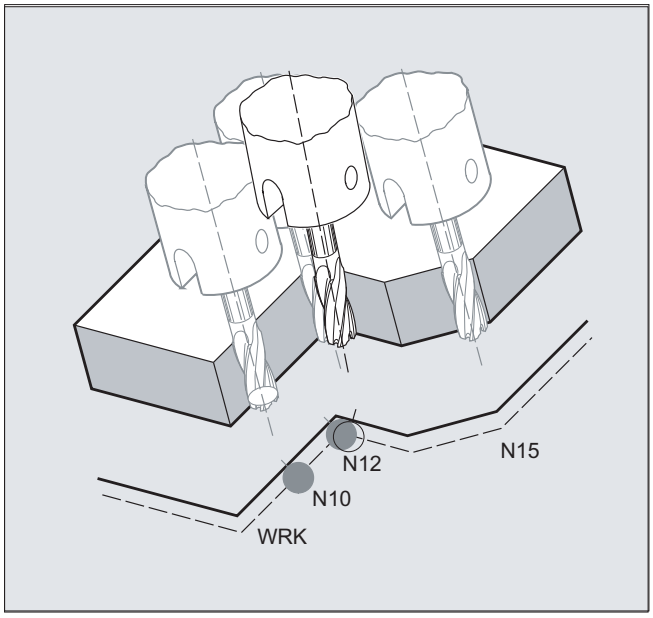
程序代码	注释
ORID	
N8 A2=… B2=… C2=…	
N10 X… Y… Z…	
N12 A2=… B2=… C2=…	；程序段 N12 和 N14 在 N10 结束处执行。然后圆弧程序段将以当前的方向运行。
N14 M20	;辅助功能等等
N20 X… Y… Z…	

说明

对于某个外角上的定向变化类型而言，起决定性作用的是在外角的第一个运动程序段中激活的程序指令。

没有定向变化：如果程序段极限上的定向没有变化，则刀具截面是一个接触两个轮廓的圆。

示例 3：内角上的定向变化



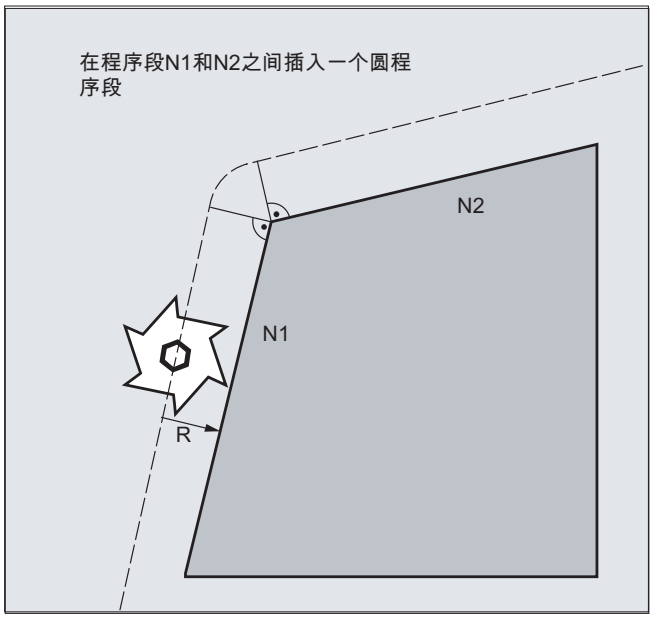
程序代码
ORIC
N10 X …Y… Z… G1 F500
N12 X …Y… Z… A2=… B2=… C2=…
N15 X …Y… Z… A2=… B2=… C2=…

其它信息

外角处的特性

在外角处总是插入一个带铣刀半径的圆弧程序段。

用程序指令 ORIC 或者 ORID 可以确定，在程序段 N1 和 N2 之间已编程的定向变化是否在插入的圆程序段之前还是与该程序段同时执行。



如果在外角处需要一个定向改变，那么定向改变可以选择和插补并行或者和轨迹运动分离来进行。

如果是 ORID ，首先执行已插入的、没有轨迹运动的程序段。 直接在两个运行程序段的第二个之前插入圆弧程序段，通过这两个程序段构成拐角。

如果在外角上插入了多个定向程序段并且 ORIC 已被选中，则圆周运动将根据各个插入程序段的定向变化分配给这些程序段。

用 OSD 或 OST 平滑定向

在用 G642 精磨时，轮廓轴和定向轴的最大偏差区别不会很大。 这两者之间较小的公差确定了平滑运动的形式或者角度公差，相对较强地平滑定向曲线，而避免出现较大的轮廓偏差。

通过激活 OSD 或 OST 可以用一个预设的平滑长度和角度公差，来平滑不明显的轮廓偏差、极小的定向曲线偏差。

说明

与用 G642 精磨轮廓（和定向曲线）不同，在用 OSD 或者 OST 平滑定向时，不再构成自身的程序段，而是直接在编程的原始程序段中插入平滑运动。

用 OSD 或者 OST 不能够平滑程序段过渡，因为在程序段过渡中用于刀具定向（矢量 **–>** 回转轴，回转轴 **–>** 矢量）的插补方式会发生变化。 必要时，这些程序段过渡可以用传统的平滑功能 G641、G642 或者 G643 进行平滑。

11.7 任意 D 编号赋值，切削刃编号

11.7.1 任意 D 编号赋值，切削刃编号 (地址 CE)

D 号码

D 编号可以作为补偿编号使用。 除此之外可以通过地址 CE 给切削刃的编号定址。 通过系统变量$TC_DPCE 可以描述切削刃编号。

预设置： 补偿编号 == 切削刃编号

通过机床数据确定 D 号码的最大数量（切削刃编号）和每个刀具的最大切削刃数量（→ 机床制造商）。 只有在确定最大切削刃编号（MD18105）大于每个刀具（MD18106）的切削刃数量时，下面的指令才有意义。 请注意机床制造商的说明。

说明

除了相对 D 号码分配之外，D 号码也可作为“平面”或者“绝对”D 号码（1-32000）在没有参照的情况下分配一个 T 编号（在“平面 D 号码结构”功能范围内）。

文献

功能手册 基本功能;刀具补偿(W1)

11.7.2 任意 D 编号赋值： 检查 D 号码(CHKDNO)

用指令 CKKDNO 可以检查现有的 D 号码是否是单一分配。 所有在一个 TO 单元内定义的刀具的 D 编号只能出现一次。 替代刀具在此不考虑。

句法

状态=CHKDNO(Tno1,Tno2,Dno)

含义

state:	=真：	对于检查范围单一的分配 D 编号。
	=假：	产生一个 D 编号的重合或者给定参数无效。 通过 Tno1，Tno2 和 D 编号来过渡导致重合的参数。 这些数据可以在零件程序中计算。

CHKDNO(Tno1,Tno2):	检查命名刀具的所有 D 编号。
CHKDNO(Tno1):	检查 Tno1 的所有 D 编号和其他所有的刀具比较。
CHKDNO:	检查所有刀具的所有 D 编号和其他所有刀具比较。

11.7.3 任意 D 编号赋值： 重命名 D 编号(GETDNO, SETDNO)

D 编号必须单一分配。 一个刀具的两个不同的切削刃不可以有同一个 D 编号。

GETDNO

该指令用来给某个带有 T 编号刀具的特定切削刃提供 D 编号。如果不存在 D 编号给已经输入的参数，则设定 d=0。 如果该 D 编号无效，将返回一个大于 32000 的值。

SETDNO

用这个指令可以给刀具 t 的切削刃 ce 的 D 编号值 d 赋值。 通过 状态 可返回该指令的结果 (正确 或者 错误). 如果没有输入参数的数据程序段，那么给回错误。 句法错误会导致报警。 不可以明显将 D 编号设置为 0。

句法

```
d = GETDNO (t,ce)
状态 = SETDNO (t,ce,d)
```

含义

d:	刀具刀刃的 D 编号
t:	刀具的 T 编号
ce:	刀具的刀刃编号（CE 编号）
state:	说明指令是否可以无误地执行（正确或者错误）。

重命名一个 D 编号举例

编程	注释
$TC_DP2[1.2]=120	
$TC_DP3[1,2] = 5.5	
$TC_DPCE[1,2] = 3	; 刀沿编号 CE
...	
N10 def int D旧编号, D新编号 = 17	

编程	注释
N20 D旧编号 = GETDNO(1,3)	
N30 SETDNO(1,3,D新编号)	

在此给切削刃 CE=3 赋值新的 D 值 17。 现在通过 D 编号 17 来响应该切削刃的数据; 既可以通过系统变量也可以在编程中使用 NC 地址。

11.7.4 任意 D 编号赋值： 求得预先给出 D 编号刀具的 T 编号（GETACTTD）

用指令 GETACTTD 可以相对于一个绝对 D 编号，得出所属的 T 编号。 不检查单一性。 如果在一个 TO 单元内有多个相同的 D 编号，就会返回第一个被发现刀具的 T 编号。 如果使用“平面”D 编号，则使用该指令就没有意义，因为此时始终返回数值 1（没有 T 编号在数据管理器中）。

句法

```
status=GETACTTD(Tnr,Dnr)
```

含义

Dnr:	D 编号，针对 D 编号应寻找对应的 T 编号。	
Tnr:	已发现的 T 编号	
状态：	值：	含义：
	0	找到 T 编号。 T 号包含 T 编号的值。
	-1	对于说明的 D 编号不存在 T 编号； T 号=0。
	-2	D 编号不是绝对的。 Tnr 包含第一个找到的刀具的值，这个刀具包含带有值 Dnr 的 D 编号。
	-5	该功能可以由于一个其他的原因不执行。

11.7.5 任意 D 编号赋值：设定无效的 D 编号 (DZERO)

指令 DZERO 用于在重新调整期间提供支持。 这样标记的补偿数据程序段不再由指令 CHKDNO 来检查。 为了重新对其进行访问，必须重新使用 SETDNO 设定 D 编号。

句法

```
DZERO
```

含义

DZERO:	将 TO 单元的所有 D 编号标识为无效。

11.8 刀架的运动关系

前提条件

刀架可以给一个刀具在所有可能的空间方向上定向，只有当

- 两个旋转轴 V_1 和 V_2 均存在。
- 旋转轴相互垂直。
- 刀具纵轴垂直于第二个旋转轴 V_2。

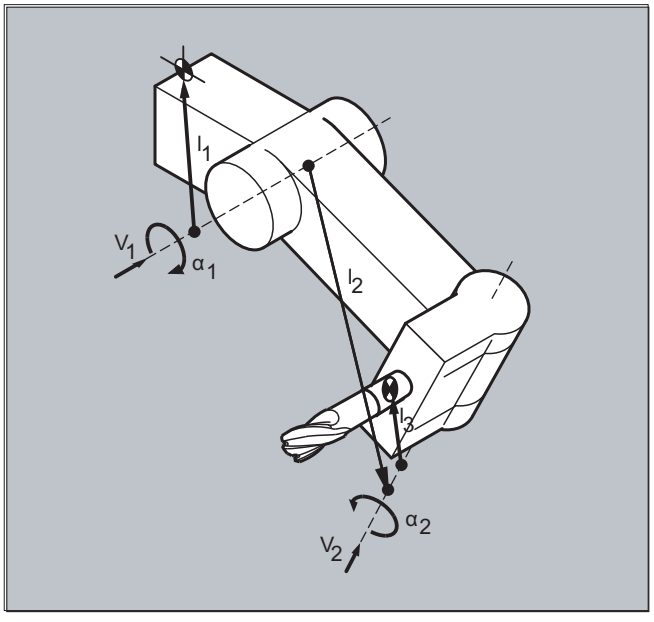
除此之外在使用机床时，在使用这些机床时所有可能的定向必须是可调节的，有下列要求：

- 刀具定向必须垂直于第一个旋转轴 V_1。

功能

带有最多两个旋转轴的刀架运动 v_1 或者 v_2 可通过 17 个系统变量 $TC_CARR1[m] 至 $TC_CARR17[m] 进行描述。 刀架的描述由以下几部分组成：

- 从第一个旋转轴到刀架参照点的矢量距离 I_1，从第一个旋转轴到第二个旋转轴的矢量距离 I_2，从第二个旋转轴到刀具参照点的矢量距离 I_3.
- 两个旋转轴的方向矢量 V_1，V_2.
- 围绕两个轴的旋转角 α1，α2。 旋转角以视角方向沿旋转轴矢量的方向顺时针正向来计算。



对于具有**分解运动** 的机床(刀具以及工件均可以转动)，系统变量已扩展了记录 $TC_CARR18[m] 至 $TC_CARR23[m]。

参数

可定向刀架系统变量的功能			
名称	**x-分量**	**y-分量**	**z-分量**
l_1 偏移矢量	$TC_CARR1[m]	$TC_CARR2[m]	$TC_CARR3[m]
l_2 偏移矢量	$TC_CARR4[m]	$TC_CARR5[m]	$TC_CARR6[m]
v_1 旋转轴	$TC_CARR7[m]	$TC_CARR8[m]	$TC_CARR9[m]
v_2 旋转轴	$TC_CARR10[m]	$TC_CARR11[m]	$TC_CARR12[m]
α_1 旋转角 α_2 旋转角	$TC_CARR13[m] $TC_CARR14[m]		
l_3 偏移矢量	$TC_CARR15[m]	$TC_CARR16[m]	$TC_CARR17[m]

可定向刀架系统变量的扩展			
名称	**x-分量**	**y-分量**	**z-分量**
l_4 偏移矢量	$TC_CARR18[m]	$TC_CARR19[m]	$TC_CARR20[m]
轴标识符 旋转轴 v_1 旋转轴 v_2	旋转轴 v_1 和 v_2 的轴标识符 (默认设置为零) $TC_CARR21[m] $TC_CARR22[m]		
运动关系类型	$TC_CARR23[m]		
Tool	运动类型-T ->	运动类型-P ->	运动关系类型-M
Part **M**ixed mode	仅刀具可以旋转（默认设置）	仅工件可以旋转	工件和刀具均可以旋转
偏移， 旋转轴 v_1 旋转轴 v_2	旋转轴 v_1 和 v_2 的角度（单位：度），当收到基本位置时 $TC_CARR24[m] $TC_CARR25[m]		
角偏移，旋转轴 v_1 旋转轴 v_2	旋转轴 v_1 和 v_2 的圆锥齿圈的偏移量（单位：度） $TC_CARR26[m] $TC_CARR27[m]		
角度增量 v_1 旋转轴 v_2 旋转轴	旋转轴 v_1 和 v_2 圆锥齿圈的增量 $TC_CARR28[m] $TC_CARR29[m]		

可定向刀架系统变量的扩展			
最小位置 旋转轴 v_1 旋转轴 v_2	旋转轴 v_1 和 v_2 的最小位置软件极限值 $TC_CARR30[m] $TC_CARR31[m]		
最大位置 旋转轴 v_1 旋转轴 v_2	旋转轴 v_1 和 v_2 最大位置的软件极限值 $TC_CARR32[m] $TC_CARR33[m]		
刀架名称	代替一个数字，刀架可以获得一个名称。 $TC_CARR34[m]		
用户： 轴名称 1 轴名称 2 特征 **位置**	在用户的测量循环之内有意使用$TC_CARR35[m] $TC_CARR36[m] $TC_CARR37[m]		
	$TC_CARR38[m]	$TC_CARR39[m]	$TC_CARR40[m]
精密 **位移**	可以添加给基本参数中的**数值** 的参数。		
l_1 偏移矢量	$TC_CARR41[m]	$TC_CARR42[m]	$TC_CARR43[m]
l_2 偏移矢量	$TC_CARR44[m]	$TC_CARR45[m]	$TC_CARR46[m]
l_3 偏移矢量	$TC_CARR55[m]	$TC_CARR56[m]	$TC_CARR57[m]
l_4 偏移矢量	$TC_CARR58[m]	$TC_CARR59[m]	$TC_CARR60[m]
v_1 旋转轴	$TC_CARR64[m]		
v_2 旋转轴	$TC_CARR65[m]		

说明

有关参数的解释

使用 "m" 可相应说明需要描述的刀架的编号。

$TC_CARR47 ～ $TC_CARR54 以及 $TC_CARR61 ～ $TC_CARR63 未定义且当尝试对其进行读写访问时会导致报警。

轴上距离矢量的起始点和终点可以自由选择。 围绕两个轴的旋转角 α_1, α_2 在刀架的基本状态中被定义为 0° 。 刀架的运动关系可以有任意多种可能性来描述。

只有一个旋转轴或者没有旋转轴的刀架可以通过一个或者两个旋转轴方向矢量的零设置来描述。

如果是一个没有旋转轴的刀架，距离矢量的作用如同附加的刀具补偿，其分量在转换加工平面（G17 至 G19）° 时不受影响。

参数的扩展

旋转轴的参数

系统变量已经按照输入记录 $TC_CARR24[m] ～ $TC_CARR33[m] 扩展且描述如下：

旋转轴**偏移** v_1, v_2	当可定向的刀具处于基本位置时，旋转轴 v_1 或者 v_2 的位置变化。
角度偏移/角度增量 旋转轴 v_1, v_2	旋转轴 v_1 和 v_2 的圆周齿圈的偏移量或者角度增量。 倒圆编程设计的或者计算出的角度到下一个值，这个值从来自公式 phi = s + n * d 的整数的 n 得出。
最小和最大位置 旋转轴 v_1, v_2	旋转轴的最小位置/最大位置 旋转轴 v1 和 v2 的极限角度（软件极限值）。

用户参数

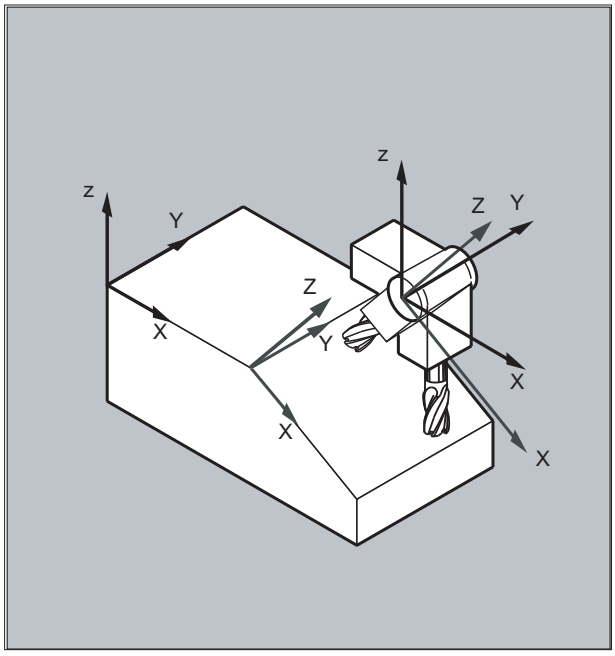
$TC_CARR34 至 $TC_CARR40 包含参数，这些参数可供用户任意使用并且软件版本 6.4 以下时默认在 NCK 之内不会被继续分析或者没有含义。

精偏移参数

$TC_CARR41 至 $TC_CARR65 包含精偏移参数，这些参数以数值方式可以添加到基本参数中。 当将数值 40 添加给参数编号时，得出分配给某个基本参数的精密位移值。

示例

下列示例中所使用的刀架可通过围绕 Y 轴旋转来完整描述。



程序代码	注释
N10 $TC_CARR8[1]=1	；定义刀架 1 第一个旋转轴的 Y 分量。
N20 $TC_DP1[1,1]= 120	;定义某个立铣刀。
N30 $TC_DP3[1,1]=20	；定义一个长度为 20 mm 的立铣刀。
N40 $TC_DP6[1,1]=5	；定义一个半径为 5 mm 的立铣刀。
N50 ROT Y37	；以围绕 Y 轴旋转 37°来定义框架。
N60 X0 Y0 Z0 F10000	；返回运行到出发位置。
N70 G42 CUT2DF TCOFR TCARR=1 T1 D1 X10	；在旋转后的框架中设置半径补偿、刀架长度补偿，选择刀架 1，刀具 1。
N80 X40	；在旋转 37°的情况进行加工。
N90 Y40	
N100 X0	
N110 Y0	
N120 M30	

其它信息

分解运动

对于具有分解运动的机床（刀具和工件均可旋转），系统变量均已经以输入记录 $TC_CARR18[m] ～ $TC_CARR23[m] 得到扩展且描如下：

可旋转的刀具台由以下几部分组成：

- 第二个旋转轴 V_2 相对于第三个旋转轴的一个可旋转刀具台参考点的矢量距离 I_4。

回转轴由以下几项组成：

- 两个用于旋转轴 V_1 和 V_2 参照点的通道标识符，在确定可定位刀架的定位时有可能要访问这些旋转轴的位置。

带有值 T，P 或者 M 其中之一的运动关系类型：

- 运动类型 T： 只有刀具是可旋转的。
- 运动类型 P： 只有工件是可旋转的。
- 运动类型 M： 工件和刀具均可以旋转

删除刀架数据

使用 $TC_CARR1[0] = 0 可以删除所有刀架数据记录的数据。

运动类型 $TC_CARR23[T] = T 必须配置三个允许大写或者小写字母 (T,P,M) 中的一个字母且处于这个原因不得被删除。

修改刀架数据

每个描述的值都可以通过零件程序中一个新的值的分配来改变。 每个其他不是 T，P 或者 M 的标志，在尝试激活可定向刀架时，会产生报警。

读取刀架数据

每个被描述的值可以通过对零件程序中变量的赋值来读取。

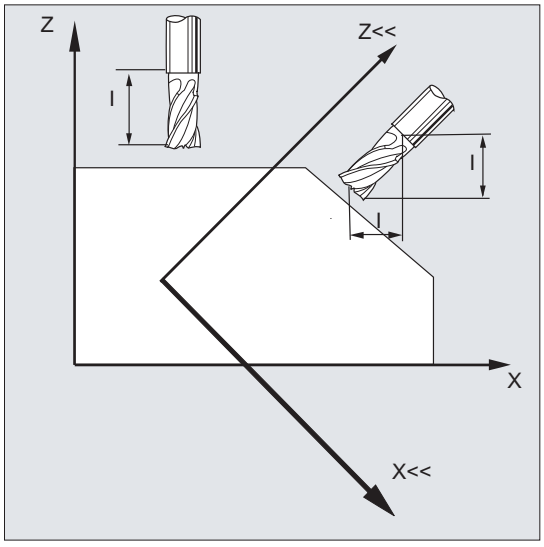
精密位移

当激活一个可定向的刀架时，才会识别一个非法的精密位移值，该刀架含有一个此类数值同时还有设置数据 SD42974 $SC_TOCARR_FINE_CORRECTION = TRUE。

允许的精偏移的量可以通过机床数据限制在一个最大允许值上。

11.9 用于可定向刀架的刀具长度补偿(TCARR, TCOABS, TCOFR, TCOFRX, TCOFRY, TCOFRZ)

刀具的空间方向改变后，刀具长度分量也一起变化。



重新安装后，比如使用固定的空间定向手动设定或者更换刀架，则必须重新计算刀具长度分量。这可以通过位移指令 TCOABS 和 TCOFR 进行。

在有效框架中，对于可以定向的刀架，在用 TCOFRZ、TCOFRY 和 TCOFRX 选择刀具时，可以确定刀具的方向。

句法

```
TCARR=[<m>]
TCOABS
TCOFR
TCOFRZ
TCOFRY
TCOFRX
```

含义

TCARR=[<m>]:	要求带“m”的刀架
TCOABS:	从当前刀架的方向计算刀具长度分量。
TCOFR:	从当前框架的方向确定刀具长度分量
TCOFRZ:	有效的框架中可定向的刀架，其刀具指向 Z 方向

TCOFRY:	有效的框架中可定向的刀架，其刀具指向 Y 方向
TCOFRX:	有效的框架中可定向的刀架，其刀具指向 X 方向

其它信息

从刀架方向确定刀具长度补偿（TCOABS）

TCOABS 从刀架当前的方向角计算刀具长度补偿；存储在系统变量 $TC_CARR13 和 $TC_CARR14 中。

有关使用系统变量定义刀架的运动性能，请参见“刀架运动关系 (页 487)”。

在框架更换后，为了重新计算刀具长度补偿，必须再次选择刀具。

当前框架的刀具方向

可以对可定向刀架进行设置，使得刀具指向下列方向：

- 通过 TCOFR 或 TCOFRZ 在 Z 方向
- 通过 TCOFRY 在 Y 方向
- 通过 TCOFRX 在 X 方向

在 TCOFR 和 TCOABS 之间进行转换，会引起刀具长度补偿的重新计算。

刀架要求（TCARR）

使用 TCARR，要求刀架号 m 连同其几何数据（补偿存储器）。

当 m=0 时，撤销选择当前的刀架。

只有在调用一个刀具之后，刀架的几何数据才有效。 在更换一个刀架后，所选择的刀具仍然有效。

刀架当前的几何数据也可以在零件程序中通过相应的系统变量进行定义。

在框架更换时重新计算刀具长度补偿 (TCOABS)

在框架更换后，为了重新计算刀具长度补偿，必须再次选择刀具。

说明

刀具方向必须手动匹配到当前的框架。

在计算刀具长度补偿时，可以在中间步骤计算刀架的转角。 带两个旋转轴的刀架通常有两个旋转角组，它们可以使刀具方向与当前的框架相适应。系统变量中存储的转角值至少必须与机械设定的转角相近似。

说明

刀具定向

控制系统不可以检查机床上旋转角的设定，旋转角可以通过框架定向进行计算。

如果刀架的旋转轴由于结构的原因，不能达到通过框架定向所计算得到的刀具方向，则发出一个报警。

刀具精确补偿不可以与运动刀架的刀具长度补偿功能相结合。 如果试图同时调用两个功能，则发出一个报警。

使用 TOFRAME 可以根据所选刀架的方向定义一个框架。 详细的信息参见章节“框架”。

在方向变换生效时（3、4、5 轴变换）可以选择一个与零位置方向偏离的刀架，而不产生一个报警。

标准循环和测量循环的传递参数

已定义的值域适用于标准循环和测量循环的传递参数。

角度值域如下：

- 围绕第 1 个几何轴旋转： -180 度 到 +180 度
- 围绕第 2 个几何轴旋转： -90 度 到 +90 度
- 围绕第 3 个几何轴旋转： -180 度 到 +180 度

参见章节框架，“可编程的旋转 （ROT，AROT，RPL）”

说明

将角度值传递到标准循环或测量循环时要注意：

小于 NC 计算精度的值将取整为零！

角度位置的 NC 计算精度在机床数据中确定：

MD10210 $MN_INT_INCR_PER_DEG

11.10 在线式刀具长度补偿 (TOFFON, TOFFOF)

通过系统变量 $AA_TOFF[<n>] 可根据三个刀具方向实时三维叠加有效的刀具长度。

三个几何轴标识符作为索引 **<n>** 使用。 这样，当前有效的补偿方向的数量便可以由同时有效的几何轴来确定。

所有的补偿可以同时有效。

功能“在线刀具长度补偿”可以在以下情况应用：

- 定向转换 TRAORI
- 可定向的刀架 TCARR

说明

在线刀具长度补偿是一个**选项**，必须事先已经激活。 只有在与一个激活的方向转换功能或者一个激活的可定向刀架配合使用时，该功能才会有效。

句法

```
TRAORI
TOFFON(<补偿方向>[,<偏移值>])
WHEN TRUE DO $AA_TOFF[<补偿方向>]                    ; 在同步动作中。
...
TOFFOF(<补偿方向>)
```

运动同步动作中的在线刀具长度补偿编程的更多相关信息请参见“同步动作 (页 655)”。

含义

TOFFON:	**激活**在线刀具长度补偿	
	<补偿方向>:	在线刀具长度补偿生效的刀具方向（X、Y、Z）。
	<偏移值>:	在激活时可以针对相应的补偿方向指定一个立即执行的偏移值。
TOFFOF:	**取消**在线刀具长度补偿 针对指定补偿方向上的补偿值被取消，并触发预处理停止。	

示例

示例 1： 选择刀具长度补偿

程序代码	注释
MD21190 $MC_TOFF_MODE =1	; 逼近绝对值。
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)	; 激活转换。
N20 TOFFON(Z)	; 激活用于 Z 轴刀具方向的在线刀具长度补偿。
N30 WHEN TRUE DO $AA_TOFF[Z]=10 G4 F5	; 为 Z 轴刀具方向插补一个大小为 10 的刀具长度补偿。
...	
N100 XOFFSET=$AA_TOFF_VAL[X]	; 在 X 方向分配当前补偿。
N120 TOFFON(X,-XOFFSET) G4 F5	; 将 X 轴刀具方向的刀具长度补偿复位为 0。

示例 2：取消刀具长度补偿

程序代码	注释
N10 TRAORI(1)	; 激活转换。
N20 TOFFON(X)	; 激活用于 X 轴刀具方向的在线刀具长度补偿。
N30 WHEN TRUE DO $AA_TOFF[X] = 10 G4 F5	; 为 X 轴刀具方向插补一个大小为 10 的刀具长度补偿。
...	
N80 TOFFOF(X)	; 删除 X 轴刀具方向的位置偏移： ...$AA_TOFF[X]=0 不运行轴。 根据当前的定位将位置偏移添加至 WCS 中的当前位置。

其它信息

程序段预处理

在程序段预处理时，会一同考虑主处理中生效的刀具长度偏移。 为了进一步充分利用允许的最大轴速度，需要用预处理停止 STOPRE 在产生刀具偏移的这段时间里来暂停预处理。

如果刀具长度补偿在程序启动之后不再改变，或者此之后又执行多个程序段，而这些程序段的数目已经超出了预处理和主处理之间 IPO 缓冲器的空间，在预处理时该偏移量始终会纳入计算。

变量$AA_TOFF_PREP_DIFF

当前插补器中生效的补偿，和程序段预处理时生效的补偿之间的差值可在变量 $AA_TOFF_PREP_DIFF[<n>] 中查询。

设置机床数据和设定数据

对于在线刀具长度补偿可以使用以下系统数据：

- MD20610 $MC_ADD_MOVE_ACCEL_RESERVE（叠加运行的加速度预留量）
- MD21190 $MC_TOFF_MODE
  系统变量 $AA_TOFF[<n>] 的内容作为绝对值处理或者相加。
- MD21194 $MC_TOFF_VELO（在线刀具长度补偿的速度）
- MD21196 $MC_TOFF_ACCEL（在线刀具长度补偿的加速度）
- 用于设定限值的设定数据：
  SD42970 $SC_TOFF_LIMIT（刀具长度补偿上限）

文献：

功能手册 特殊功能；F2：多轴转换

11.11 可旋转刀具的补偿数据修改

11.11.1 计算定向（ORISOLH）

预定义功能 ORISOLH 能够协助用户设置机床的回转轴位置，从而将车刀送入定义的、相对于工件的独立于运动系统的位置。前提条件是，一个通过运动链参数设置的 6 轴转换生效。

提供两个基本功能：

- 对准刀具
  设定两个角度 β 和 γ。该功能计算出三个定向轴的为此所需的角度。
- 直接对准刀具
  设定第二定向轴和第三定向轴的角度。该功能据此计算出对应的角度 β 和 γ，以及尚缺少的第一定向轴的角度。

说明

定向轴的顺序

若从工件起至刀具为止经过描述机床结构的运动链，则以下规定适用于 6 轴转换的三个定向轴的顺序：

- 最靠近**工件**的定向轴为**第一**定向轴。
- 最靠近**刀具**的定向轴为**第三**定向轴。

第一定向轴通常为主轴，故在此情形下通过旋转框架实现对应的旋转。

句法

```
<RetVal> = ORISOLH(<Cntrl>,<W1>,<W2>)
```

含义

ORISOLH:	功能调用		
<RetVal>:	功能的返回值		
	数据类型:	INT	
	值域:	0, -2, -3, ..., -17	
	数值:	0	该功能已无故障地结束。
		-2	无有效转换（6 轴定向转换）生效。
		-3	第一参数（<Cntrl>）为负。
		-4	第一参数（<Cntrl>）的个位无效。 仅允许值 0 和 1。
		-5	第一参数（<Cntrl>）的十位无效。 仅允许值 0 至 3。
		-6	第一参数（<Cntrl>）的百位无效。 仅允许值 0 和 1。
		-7	第一参数（<Cntrl>）的千位无效。 仅允许值 0 至 3。
		-8	在“直接对准刀具”功能中，角度 γ 过大。
		-9	在“直接对准刀具”功能中，设定的轴位置中的至少一个超出轴极限。
		-10	无刀具生效。
		-11	要求的定向不可设置。
		-12	对于第一或唯一解决方案，无法调整采用端面齿时的自由轴角。
		-13	对于第二解决方案，无法调整采用端面齿时的自由轴角。
		-14	对于两个解决方案中的任一个均无法调整采用端面齿时的自由轴角。
		-15	第一定向轴被参数设置为端面齿轴。
		-16	第二和第三回转轴均被参数设置为端面齿轴。这两个轴中最多仅有一个可以是端面齿轴。
		-17	在“直接回转”功能中，设定的轴位置中的至少一个与对应的端面齿不兼容。

<Cntrl>:	控制功能的特性		
	数据类型：	INT	
	参数 **<Cntrl>** 为十进制编码（个位至千位）：		
	个位：	个位控制故障时的特性。	
		xxx0	在故障情形下（返回值 **< 0**）终止程序执行，并输出报警 14106。 **提示：** 在参数 **<Cntrl>** 为负的情况下同样会输出报警，而与个位的值无关。
		xxx1	在故障情形下（返回值 **< 0**）**不**输出报警。用户可以在程序中作适当的响应。
	十位：	对存在带端面齿的定向轴时的特性进行控制。 **提示：** 仅在“对准刀具”功能中（即在百位值为“0”的情况下）才会分析这个参数。	
		xx0x	将轴位置取整至最靠近的位置。
		xx1x	如此将轴位置取整，从而将角度 β 与其编程值的偏差最小化。
		xx2x	如此将轴位置取整，从而使角度 β 等于：小于编程值的最大的值（将 β 向下取整）。
		xx3x	如此将轴位置取整，从而使角度 β 等于：大于编程值的最小的值（将 β 向上取整）。
	百位：	给定需要执行的功能，以及两个后续参数 **<W1>** 和 **<W2>** 有何含义。	
		x0xx	“对准刀具”功能 参数 **<W1>** 和 **<W2>** 具有以下含义： ● **<W1>** = β ● **<W2>** = γ 计算定向轴的对应角度。
		x1xx	“直接对准刀具”功能 **<W1>** 是针对 6 轴转换的第二定向轴的位置设定，**<W2>** 是针对第三定向轴的位置设定。确定第一定向轴的位置，以及与这两个位置设定兼容的角度 β 和 γ。

			若无故障，则在系统变量 $P_ORI_POS[<n>, <m>] 中总是输出两个解决方案。其中第一索引 <n>（0 或 1）参考解决方案，且第二索引 <m>（0 ... 2）参考定向轴： • $P_ORI_POS[0/1, **0**]:第一定向轴的位置 • $P_ORI_POS[0/1, **1**]:角度 β • $P_ORI_POS[0/1, **2**]:角度 γ 系统检查位置设定 <W1> 及 <W2> 是否与可能生效的端面齿或生效的软件限位兼容。若并非如此，则会反馈对应的故障编号（参见参数 <RetVal>）。 在任意选择这两个角度 <W1> 和 <W2> 时，刀具刀沿通常不位于加工平面中。刀沿自加工平面发生旋转的角度 γ 不允许大于通过设定数据 SD42999 $SC_ORISOLH_INCLINE_TOL 确定的限值。
	千位：	给定在百位的值为“0”，即采用“对准刀具”功能的情况下，如何视情况修改解决方案的位置。	
		0xxx	计算出的轴位置应尽可能接近当前机床轴位置。
		1xxx	就模态轴而言，计算出的轴位置应尽可能接近模态范围的中心，而就其他轴而言则应尽可能接近 0。对于非模态轴而言表示将轴位置减小至 -180° … +180° 的范围。
		2xxx	与轴类型无关，将计算出的轴位置减小至 -180° … +180° 的范围。
<W1>：	第一角度 含义取决于参数 <Cntrl> 的百位。		
	数据类型：	REAL	
<W1>：	第二角度 含义取决于参数 <Cntrl> 的百位。		
	数据类型：	REAL	

说明

未编程的参数具有缺省值“0”。

其它信息

在执行 ORISOLH 功能时，可通过以下系统变量读取找到的解决方案的数量以及其他状态信息：

系统变量	含义		
$P_ORI_POS [<n>, <m>]	提供在定向编程中得到的定向轴的角度。		
	<n>： 解决方案的索引		
		值域：	0, 1
	<m>： 定向轴的索引		
		值域：	0 ... 2
			定向轴的顺序（1 ... 3）基于 $NT_ROT_AX_NAME 中的轴定义。
	在“直接对准刀具”模式中调用 ORISOLH 功能时，变量 $P_ORI_POS[0/1, 1] 和 P_ORI_POS[0/1, 2] 包含两个角度 β 和 γ 的归属于两个解决方案的值。 $P_ORI_POS[<n>, <m>] 中记录的第一解决方案，即通过索引 <n> = 0 记录的解决方案，始终为在直接逼近要求的定向时由控制系统选择的解决方案。第二索引 <m> 基于定向轴，即基于 $NT_ROT_AX_NAME。 $P_ORI_POS[<n>, <m>] 中记录的轴位置将记录于 $NK_OFF 和 $NK_OFF_FINE 中的偏移考虑在内，亦即，可在后续程序段中将这些轴角用于设置所需的定向，而不作进一步修改。 若回转轴为端面齿轴，则将解决方案位置取整至最靠近的端面齿卡合位置。在系统变量 $P_ORI_DIFF 中，可为设端面齿的回转轴读取针对精确解决方案的轴位置的差值，以及与端面齿卡合对应的解决方案。		
$P_ORI_DIFF [<n>, <m>]	针对在定向编程中得到的定向轴，提供该定向轴的精确位置与在 $P_ORI_POS 中提供之位置的差值。		
	<n>： 解决方案的索引		
		值域：	0, 1
	<m>： 定向轴的索引		
		值域：	0 ... 2
			定向轴的顺序（1 ... 3）基于 $NT_ROT_AX_NAME 中的轴定义。
	仅当位置被栅格化时（端面齿），及当相关轴的系统数据 $NT_HIRTH_INCR 不等于零且这个轴为手动回转轴时，内容才可能不等于零。		

系统变量	含义		
$P_ORI_SOL	若在包含超过一个定向轴的定向转换中计算需导致预设定向的轴角，则通常有超过一个解决方案。在系统变量 $P_ORI_SOL 中包含了有效解决方案的数量以及额外的状态信息。 $P_ORI_SOL 的内容如下编码：		
	值 < 0	一般故障状态	
		-1	针对生效的转换尚未计算出解决方案（缺少 ORISOLH 调用）。
		-2	无转换生效，或者生效的转换不是能够提供设定的定向编程的位置的定向转换（6 轴转换）。
		-4	所需的定向无法通过给定的运动系统设置。
		-5	在“直接对准刀具”模式中调用 ORISOLH 功能时，未找到解决方案。
		-6	在“直接对准刀具”模式中调用 ORISOLH 功能时，角度 γ 过大。
		-7	在“直接对准刀具”模式中调用 ORISOLH 功能时，设定了一个因端面齿而无法设置的角度。
		-8	第一定向轴（框架轴）不允许被参数设置为端面齿轴。
		-9	第二和第三回转轴均被参数设置为端面齿轴。这两个轴中最多仅有一个可以是端面齿轴。
		-10	未找到根据端面齿作出的解决方案调整。
	值 > 0 个位	就数学而言可能的解决方案的数量，不考虑轴极限和可能存在的故障条件。	
		0	不存在解决方案，即要求的定向不可设置。 可能有三种不同的原因： ● 基于机床运动系统（并非直角布置的定向轴），即便在在定向轴的任意运行范围内，仍无法实现要求的定向。在此情形下，$P_ORI_SOL 的十位和百位均为零，与定向轴对应的状态变量 $P_ORI_STAT 的值为“-4”。 ● 计算出的解决方案无法实现，因为其会超出轴限制。定向轴的在无轴限制的情况下的位置可在 $P_ORI_POS 中读取。 ● 在“直接对准刀具”模式中调用 ORISOLH 功能时如此设定了轴位置，使得或是定向矢量，或是刀具的定向法线矢量平行于需要计算位置的第一定向轴。在这些情形下，该轴的位置未经定义。

系统变量	含义		
		1	存在一个解决方案。 可能有三种不同的原因： ● 基于设定的定向和机床运动系统，即便在不考虑轴极限的情况下，也仅有一个解决方案（就数学而言涉及两个重叠的解决方案）。在采用非直角运动系统时，在定向区域的边缘处会出现此种情形。$P_ORI_POS 包含两个（相同的）解决方案。 ● 仅有一个解决方案，因为第二解决方案因超出轴极限而无效。故有效解决方案始终为 $P_ORI_POS 中的第一解决方案。在不考虑轴极限的情况下得出的第二解决方案同样可在 $P_ORI_POS 中读取。 ● 在“直接对准刀具”模式中调用 ORISOLH 功能时，属正常情形。针对两个定向轴的设定的轴位置，待计算的缺少的定向轴的有效位置通常只有一个。
		2	存在两个解决方案。
		8	存在无穷多的解决方案，即定向轴（极轴）的位置任意。在其他轴的两个可能的位置中，则有一个因超出轴极限而被排除。
		9	存在无穷多的解决方案，即定向轴（极轴）的位置不确定。可从百位或从系统变量 $P_ORI_STAT 来获取不确定的轴。
	值 > 0 十位	针对超出的轴极限的位编码显示。可从系统变量 $P_ORI_STAT 获取确切的故障原因。	
		位 0（值 10）：	针对至少一个解决方案，超出第 1 定向轴的至少一个轴极限。
		位 1（值 20）：	针对至少一个解决方案，超出第 2 定向轴的至少一个轴极限。
		位 2（值 40）：	针对至少一个解决方案，超出第 3 定向轴的至少一个轴极限。
	值 > 0 百位	针对未定义的轴位置的位编码显示（仅当有无穷多的解决方案时，即当个位等于“9”时才会出现）	
		位 0（值 100）：	第 1 定向轴的位置未定义。
		位 1（值 200）：	第 2 定向轴的位置未定义。

系统变量	含义		
		位 2（值 400）：	第 3 定向轴的位置未定义。
	“第 1、第 2 和第 3 定向轴”这些表述基于 $NT_ROT_AX_NAME 中的轴定义。		

系统变量	含义		
$P_ORI_STAT [<n>]	针对最多 3 个定向轴中的任一个，提供调用 ORISOLH 后的状态。		
	<n>： 定向轴的索引 （对应相关定向轴在 $NT_ROT_AX_NAME 中的索引）		
		值域：	0 ... 2
			定向轴的顺序（1 ... 3）基于 $NT_ROT_AX_NAME 中的轴定义。
	$P_ORI_STAT 的内容如下编码：		
	值 < 0	一般故障状态	
		-1	状态未定义（缺少 ORISOLH 调用）。
		-2	无转换生效，或者生效的转换不是能够提供设定的定向编程的位置的定向转换（6 轴转换）。
		-3	在生效的转换中不包含此轴。
		-4	无法计算轴的位置，因为即便在任意假设轴的运行范围的情况下，通过给定的运动系统仍无法实现要求的定向。
		-5	在“直接对准刀具”模式中调用 ORISOLH 功能时如此设定了轴位置，使得或是定向矢量，或是刀具的定向法线矢量平行于需要计算位置的第一定向轴。在这些情形下，该轴的位置未经定义。
		-6	在“直接对准刀具”模式中调用 ORISOLH 功能时，角度 γ 过大。
		-7	在“直接对准刀具”模式中调用 ORISOLH 功能时，设定了一个因端面齿而无法设置的角度。
		-8	第一定向轴（框架轴）不允许被参数设置为端面齿轴。
		-9	第二和第三回转轴均被参数设置为端面齿轴。这两个轴中最多仅有一个可以是端面齿轴。
		-10	未找到根据端面齿作出的解决方案调整。
	值 > 0 个位	针对第一解决方案超出轴极限的位编码显示。	
		位 0（值 1）：	第一解决方案超出轴下限。
		位 1（值 2）：	第一解决方案超出轴上限。
	值 > 0 十位	针对第二解决方案超出轴极限的位编码显示。	
		位 0（值 10）：	第二解决方案超出轴下限。

系统变量	含义		
		位 1（值 20）：	第二解决方案超出轴上限。
	值 > 0 百位	显示未定义的轴位置。	
		位 0（值 100）：	定向轴的位置未经定义，亦即，要求的定向通过回转轴的任意设置实现（极位置）。此信息也包含在系统变量 $P_ORI_SOL 中。
	在这些显示超出轴极限的故障编号中，可能会同时出现多个。在超出轴极限时，系统会尝试通过叠加或减去 360° 的倍数来使位置处于允许的轴极限内，故（在无法实现的情况下）不明确定义是超出轴下限还是轴上限。 若针对要求的定向无解决方案（$P_ORI_SOL = 0），则转换中包含的定向轴的状态为“0”。		

说明

$NT_ROT_AX_NAME

通过此系统变量参考最多 3 个用于设置定向的轴。其包含链元素的名称（$NK_NAME），这些名称定义需要执行自运动转换得出的定向运动的机床轴（回转轴）。对于机床运动系统而言，最多三个回转轴在此系统变量中的顺序无意义，因为这个顺序是从运动链的结构推导出的。但该顺序定义了通过其他变量访问回转轴的顺序，故 $NT_ROT_AX_NAME 中的定向轴的顺序必须与运动系统描述一致。

说明

状态信息

例如显示“定向无法实现”或“仅在超出相关轴极限的情况下才能实现”的状态信息不会导致 NC 报警。用户需要负责对述及的条件作出适当的响应。

11.11.2 激活可旋转刀具的补偿数据修改（CUTMOD、CUTMODK）

在 NC 程序中通过语言指令 CUTMOD（结合可定向刀架）或 CUTMODK（针对借助运动链定义的定向转换）激活可旋转刀具的补偿数据修改。

说明

可定向刀架和借助运动链定义的定向转换不会同时生效，故不会发生两个方案之间的冲突。

句法

```
CUTMOD = <值>
```

或

```
CUTMODK = <指令>
```

含义

CUTMOD:	结合可定向刀架调用功能		
<值>:	赋值		
	数据类型:	INT	
	值:	0	该功能被取消激活。 由系统变量 $P_AD... 提供的值和相应的刀具参数相同。
		> 0	如果一个指定的可定向刀架激活，则该功能也激活，即该功能的启用和特定的可定向刀架绑定。 由系统变量 $P_AD... 提供的值不和相应的刀具参数相同，而是取决于当前有效的旋转发生变化。 禁用某个指定的可定向刀架会暂时锁定该功能，启用另一个刀架会持续锁定该功能。因此，在第一种情况下，再次选中指定刀架即可再次激活功能，而在第二种情况下，即使再次选中指定刀架，还需要重新选择。 该功能不受复位操作的影响。
		-1	如果一个可定向的刀架激活，该功能总是激活。 在更换刀架，或取消选择并接着再次选中时，CUTMOD 不必重新设置。
		-2	如果一个可定向刀架激活，而它的号码与当前的可定向刀架相同，该功能总是激活。 如果没有可定向刀架激活，则该设置相当于 CUTMOD=0。 如果一个可定向刀架激活，则该设置相当于直接给定当前刀架号。
		< -2	小于 2 的值被忽略，即该值相当于没有编程 CUTMOD。 **提示：** 该取值范围预留用于将来的功能扩展，请勿使用。
CUTMODK:	结合借助运动链定义的定向转换调用功能		

<指令>：	分配的指令		
	数据类型：	STRING	
	值：	"NEW"	针对生效的借助运动链定义的转换的与“补偿数据修改”相关的状态，将转换的名称和当前的轮廓框架保存。 **提示：** 仅当适用的转换（TRAORI_DYN、TRAORI_STAT 或 TRAANG_K）生效时，才允许使用此指令。
		"OFF"	将生效的“补偿数据修改”取消。先前通过“NEW”保存的数据保留。 **提示：** 即便在 CUTMODK 未生效的情况下，也允许使用此指令。在此情形下该指令无作用。可能存在的用于“补偿数据修改”的数据组保留。
		"ON"	借助这个指令，通过先前用“NEW”指令保存的数据组将“补偿数据修改”重新激活。 在执行此指令时，若包含保存的数据组的名称的转换生效，则“补偿数据修改”立即生效。否则激活延迟，直至激活生效的转换。
		"CLEAR"	像“OFF”指令那样将“补偿数据修改”取消，此外删除保存的数据组。 **提示：** 即便在 CUTMODK 未生效的情况下，也允许使用此指令。

说明

SD42984 $SC_CUTDIRMOD

可通过指令 CUTMOD 或 CUTMODK 激活的功能会替换可通过设定数据 SD42984 $SC_CUTDIRMOD 激活的功能。但原先的功能仍可供使用。但并行使用两个功能无意义，故仅当 CUTMOD 等于零且 CUTMODK 等于零字符串时，才能激活原先的功能。

更多信息

读取经修改的补偿数据

在下列系统变量和 OPI 变量中提供经修改的补偿数据：

含义	系统变量	OPI 变量
刀沿位置	$P_AD[2]	cuttEdgeParam2
主偏角	$P_AD[10]	cuttEdgeParam10
切削方向	$P_AD[11]	cuttEdgeParam11
后角	$P_AD[24]	cuttEdgeParam24

当通过指令 CUTMOD 或 CUTMODK 激活“可旋转刀具的补偿数据修改”功能，并通过可定向刀架或适当的定向转换将刀具旋转时，数据总是相对对应的刀具参数（$TC_DP2[..., ...], ...）经过修改。

其他功能相关的系统变量

系统变量	含义
$P_CUTMOD_ANG / $AC_CUTMOD_ANG	提供角度，其为刀具在生效的加工平面中的旋转角度，以及在 CUTMOD 或 CUTMODK 功能中测定经修改的刀沿数据所基于的角度。
$P_CUTMOD / $AC_CUTMOD	读取最后通过指令 CUTMOD 编写的当前有效的值（需要激活补偿数据修改的刀架的编号）。 若最后编写的值为 CUTMOD=-2（通过当前生效的可定向刀架激活），则在系统变量中不返回值“-2”，而是在编程时间点上生效的可定向刀架的编号。
$P_CUTMODK / $AC_CUTMODK	读取转换的名称，当前有效的用于“补偿数据修改”的数据组在该名称下创建。

系统变量	含义
$P_CUT_INV / $AC_CUT_INV	如果刀具旋转，使得主轴旋转方向必须反转，则值为 TRUE。为此在相关各个读取操作的程序段中必须满足下列四项条件： 1. 车刀或磨具生效 （刀具类型 400 到 599 并且/或者 SD42950 $SC_TOOL_LENGTH_TYPE = 2）。 2. 已通过指令 CUTMOD 或 CUTMODK 激活补偿数据修改。 3. 通过指令 CUTMOD 或 CUTMODK 选择的可定向刀架或借助运动链定义的定向转换生效。 4. 通过可定向刀架或运动定向转换如此将刀具旋转，使得刀具刀沿的法线相对起始位置旋转超过 90°（通常为 180°）。 若不满足所述四个条件中的至少一个，则变量提供值 FALSE。对于未定义刀沿位置的刀具，变量值总是为 FALSE。

系统变量	含义	
$P_CUTMOD_ERR	最后调用 CUTMOD 功能后的故障状态 CUTMOD 功能也可以在换刀中隐性调用。此变量在复位时恢复为零。在每次换刀时首先将此变量复位，并视情况重新描述。 此变量为位编码。位的含义如下：	
	位 0：	没有为生效刀具定义有效的切削方向。
	位 1：	生效刀具的刀沿角度（后角和主偏角）都为零。
	位 2：	生效刀具的后角具有不允许的值（< 0° 或 > 180°）。
	位 3：	生效刀具的主偏角具有不允许的值（< 0° 或 > 90°）。
	位 4：	生效刀具的刀片角度具有不允许的值（< 0° 或 > 90°）。
	位 5：	生效刀具的刀沿位置与主偏角的组合不允许使用（当刀沿位置为 1 到 4 时主偏角必须 ≤ 90° ， 当刀沿位置为 5 到 8 时必须 ≥ 90° ）。
	位 6：	生效刀具发生不允许的旋转。 刀具已自生效的加工平面转出 ± 90°（公差约为 1°）。因此在加工平面中无法定义刀沿位置。
	位 7：	刀片不位于加工平面中，且刀片与加工平面之间的角度超出通过设定数据 SD42998 $SC_CUTMOD_PLANE_TOL 设定的上限。
	位 8：	刀片不位于加工平面中。角度 α 的值大于 1°。角度 α 是围绕一个坐标轴的旋转角，该坐标轴既垂直于角度 β 的旋转轴又垂直于角度 γ 的旋转轴（在 G18 中为 X 轴）。

$P_...：预处理变量

$AC_...：主处理变量

所有主处理变量均可在同步动作中读取。预处理中的读取访问产生预处理停止。

平面切换

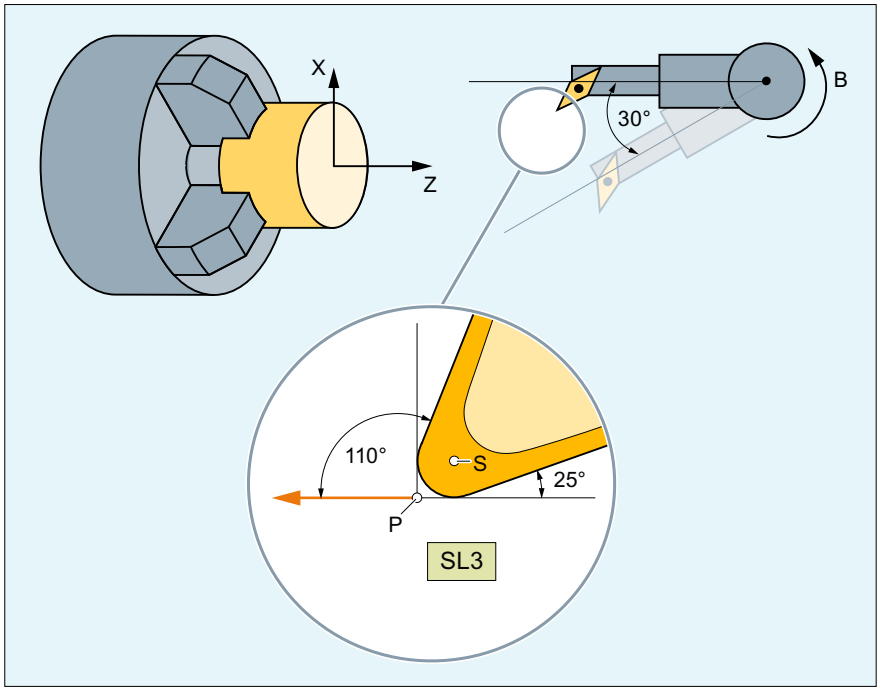
为了确定经修改的刀沿位置、切削方向以及主偏角或后角，在各生效平面（G17 - G19）中的刀沿观测起决定性作用。

但若设定数据 SD42940 $SC_TOOL_LENGTH_CONST（平面切换中刀具长度分量的切换）包含不等于零的有效值（正或负 17、18 或 19），则其内容决定用于观测相关尺寸的平面。

这个设定数据优先于 G 代码的规则可通过将机床数据 $MC_TOOL_PARAMETER_DEF_MASK 的位 18 置位来取消。亦即，当该位置位时，始终是通过 G 指令组 6 的 G 指令定义的平面生效。

修改的刀沿数据有效性

修改的刀沿位置和刀沿参考点在编程时也立即生效于已经激活的刀具。为此无需选择刀具。



S： 刀沿中心点

P： 刀沿参考点

SL 刀沿位置

：

图 11-4 具有刀沿位置 3 和可定向刀架的刀具，该刀架能够使刀具围绕 B 轴旋转

程序代码	注释
N10 $TC_DP1[1,1]=500	
N20 $TC_DP2[1,1]=3	；刀沿位置
N30 $TC_DP3[1,1]=12	
N40 $TC_DP4[1,1]=1	
N50 $TC_DP6[1,1]=6	
N60 $TC_DP10[1,1]=110	；主偏角

程序代码	注释
N70 $TC_DP11[1,1]=3	; 切削方向
N80 $TC_DP24[1,1]=25	; 后角
N90 $TC_CARR7[2]=0 $TC_CARR8[2]=1 $TC_CARR9[2]=0	; B 轴
N100 $TC_CARR10[2]=0 $TC_CARR11[2]=0 $TC_CARR12[2]=1	; C 轴
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	

注释中的数值给出的是机床坐标系（MCS）中的按照顺序 X → Y → Z 的程序段末尾位置。

说明

CUTMOD=0：程序段 N180 中，先选择刀具，没有选择可定向的刀架。由于可定向刀架的所有偏移坐标为 0，所以轴逼近某个符合 $TC_DP3[1,1] 和 $TC_DP4[1,1] 中规定刀具长度的位置。

在程序段 N200 中，激活绕着 B 轴旋转 30°可定向的刀架。因为 CUTMOD=0，刀沿长度没有被修改，所以还是参照以前的切削参考点。因此在程序段 N210 中，轴逼近某个零点中保留了旧切削参考点的位置（即矢量 (1, 12) 在 Z/X 平面内旋转 30° ）。

和程序段 N200 不同，在 N260 中，CUTMOD=2。由于可定向刀架的旋转，到达经过修改的切削刃位置 8。由此出现偏差的轴位置。

在程序段 N220 或 N270 中总是激活刀具半径补偿 (WRC)。两个程序块中不同的刀沿位置对 WCR 作用的程序段的最终位置没有影响，对应的位置因此相同。仅在程序段 N260 或 N300 中，不同的刀沿位置才有影响。

11.12 使用刀具环境工作

功能一览

- 保存刀具环境 (TOOLENV) (页 517)
- 删除刀具环境 (DELTOOLENV) (页 520)
- 读取 T 号、D 号和 DL 号 (GETTENV) (页 521)
- 读取刀具长度或刀具长度分量 (GETTCOR) (页 523)
- 修改刀具分量 (SETTCOR) (页 530)

系统变量一览

- 读取保存的刀具环境的信息 ($P_TOOLENVN, ($P_TOOLENV) (页 522)

11.12.1 保存刀具环境 (TOOLENV)

功能 TOOLENV 用来存储当前的所有状态，它们对于计算存储器所保存的刀具数据非常有意义。

详细地指下列数据：

- G 指令组中生效的 G 指令：
  - 6 (G17、G18、G19)
  - 56 (TOWSTD、TOWMCS、TOWWCS、TOWBCS、TOWTCS、TOWKCS)
- 有效的端面轴
- 机床数据：
  - MD18112 $MN_MM_KIND_OF_SUMCORR （TO 范围内总补偿的属性）
  - MD20360 $MC_TOOL_PARAMETER_DEF_MASK（定义刀具参数）

- 设定数据：
  - SD42900 $SC_MIRROR_TOOL_LENGTH （镜像时的刀具长度符号变换）
  - SD42910 $SC_MIRROR_TOOL_WEAR （镜像时的刀具磨损符号变换）
  - SD42920 $SC_WEAR_SIGN_CUTPOS （在使用刀沿位置的刀具上磨损的符号）
  - SD42930 $SC_WEAR_SIGN（磨损的符号）
  - SD42935 $SC_WEAR_TRANSFORM （用于刀具分量的转换）
  - SD42940 $SC_LENGTH_CONST（平面转换时刀具长度分量的转换）
  - SD42942 $SC_TOOL_LENGTH_CONST_T（平面转换时车刀刀具长度分量的转换）
  - SD42950 $SC_TOOL_LENGTH_TYPE（刀具长度分量的分配与刀具类型无关）
  - SD42954 $SC_TOOL_ORI_CONST_M（平面转换时铣刀刀具定向分量的转换）
  - SD42956 $SC_TOOL_ORI_CONST_T（平面转换时车刀刀具定向分量的转换）
- 当前总框架的定向部分 （旋转和镜像，没有零点偏移或比例缩放）
- 可定向刀架有效时的定向部分和生成的长度
- 转换有效时的定向部分和生成的长度

除了上述用来说明刀具环境的数据，还一同保存了有效刀具的 T 号码、D 号码和 DL 号码，这样以后在同样的环境中就能够象调用 TOOLENV 一样对该刀具进行存取，而不必重新对刀具进行标识。

句法

```
<Status> = TOOLENV(<Name>)
```

含义

TOOLENV(...)：	用于保存刀具环境的预定义功能	
	在单独程序段中编程：	支持

<Status>:	函数的返回值。负值代表故障状态。		
	数据类型：		INT
	值：	0	功能正常
		-1	没有预留用于刀具环境的存储位置： MD18116 $MN_MM_NUM_TOOL_ENV = 0 即“刀具环境”功能不存在。
		-2	不再有空的存储位置用于刀具环境。
		-3	不允许使用零字符串作为刀具环境的名称。
		-4	未给定参数 (<Name>)。
参数			
1 <Name>:	名称，在该名称下保存有当前数据组。 如果一个数据组与已有的数据组名称相同，则会覆盖原数据组。在这种情况下状态为“0”。		
	数据类型：		STRING

更多信息

基本尺寸/适配器尺寸 - 刀具长度补偿

适配器长度或者基础尺寸（刀沿专用参数 $TC_DP21、$TC_DP22 和 $TC_DP23）是否参与刀具长度的计算在刀库管理生效时由机床数据的值来决定（仅在带选件“刀具管理”的情况下可用！）：

MD18104 $MN_MM_NUM_TOOL_ADAPTER （TO 范围内的刀具适配器）

因为该机床数据的修改只能在启动控制系统时生效，所以不将其保存在刀具环境中。

由可定向刀架和转换所生成的长度

说明

不仅在可定向刀架上而且在转换时都有一些系统变量或机床数据，它们和其他刀具长度分量一样生效并且全部或部分服从于此时所进行的旋转。由此得出的、额外的刀具长度部分必须在调用 TOOLENV 时一起进行保存，因为它们是构成刀具使用环境的一部分。

适配器转换

适配器转换是刀具适配器以及整个刀具的一个属性。所以它不是刀具环境的组成部分，可以被用于其他刀具。

通过保存全部用来确定刀具总长度所需的数据能够在以后的时间点计算刀具的有效长度，即使刀具在该时间点不再有效或者环境条件（例如 G 指令或者设定数据）发生了变化。接受后同样能够用来计算其他刀具的有效长度，可以在同样的条件下与所保存的刀具一样来使用。

刀具环境数据组的最大数量

通过机床数据 MD18116 $MN_MM_NUM_TOOL_ENV 确定用于说明刀具环境的数据组最大可保存数。数据位于 TOA 范围内。在控制系统关闭时数值保持不变。

无法进行数据备份。这表示这些数据可能不是在不同控制系统之间传送的。

11.12.2 删除刀具环境 (DELTOOLENV)

使用 DELTOOLENV 功能可以删除用于说明刀具环境的数据组。删除意味着无法在特定名称下所保存的数据组上再进行存取（尝试存取会导致报警）。

说明

只能使用功能 DELTOOLENV、通过 INITIAL.INI 下载或者通过冷启动（使用标准机床数据进行 NC 引导启动）来删除数据组。没有其他的自动删除过程。

句法

```
<Status> = DELTOOLENV(<Name>)
<Status> = DELTOOLENV()
```

含义

DELTOOLENV(...):	用于删除刀具环境的预定义功能		
	在单独程序段中编程：		支持
<Status>:	函数的返回值。负值代表故障状态。		
	数据类型：		INT
	值：	0	功能正常
		-1	没有预留用于刀具环境的存储位置： MD18116 $MN_MM_NUM_TOOL_ENV = 0 即“刀具环境”功能不存在。
		-2	不存在使用给定名称的刀具环境。

参数			
1	<Name>:	待删除数据组的名称	
		数据类型：	STRING
DELTOOLENV():		DELTOOLENV() 未给定名称的情况下会删除用于说明刀具环境的数据段	

11.12.3 读取 T 号、D 号和 DL 号 (GETTENV)

GETTENV 功能用来读取保存在刀具环境中的 T、 D 和 DL 号码。

句法

```
<Status> = GETTENV(<Name>, <TDDL>)
```

含义

GETTENV(...):	用于读取用于说明刀具环境的数据组中的 T、 D 和 DL 号码的预定义功能		
	在单独程序段中编程：	支持	
<Status>:	函数的返回值。负值代表故障状态。		
	数据类型：	INT	
	值：	0	功能正常
		-1	没有预留用于刀具环境的存储位置： MD18116 $MN_MM_NUM_TOOL_ENV = 0 即“刀具环境”功能不存在。
		-2	不存在使用给定名称的刀具环境。

参数			
1	<Name>:	需要读取 T、 D 和 DL 号码的数据组名称	
		数据类型：	STRING

2	<TDDL>：	该结果参数区域中包含刀具的 T、 D 和 DL 号码，其刀具环境保存在指定数据组中： • **<TDDL>** [0]：T 号 • **<TDDL>** [1]：D 号 • **<TDDL>** [2]：DL 号	
		数据类型：	INT[3]
GETTENV(,<TDDL>), GETTENV("",<TDDL>) ：		允许在调用 GETTENV 功能时忽略第一个参数或者将零字符串作为第一个参数。在这两种特殊情况下，**<TDDL>** 中会返回**有效**刀具的 T、 D 和 DL 号码。	

11.12.4 读取保存的刀具环境的信息 ($P_TOOLENVN, ($P_TOOLENV)

保存的刀具环境的信息可通过以下系统变量读取：

$P_TOOLENVN ：	提供用于说明刀具环境的借助 TOOLENV 所定义的（且尚未删除的）数据组数			
	句法：	<n> = $P_TOOLENVN		
	含义：	<n>：	指定数据组的数量	
			数据类型：	INT
			值域：	0 ... MD18116 $MN_MM_NUM_TOOL_ENV
	如果刀具环境不可能使用（MD18116 = 0），则允许在该系统变量上进行存取。在这种情况下的返回值为 “0”。			

$P_TOOLENV:	提供用于说明刀具环境的第 **<i>** 个数据组的名称			
	句法：	<Name> = $P_TOOLENV[<i>]		
	含义：	<Name>：	使用编号 **<i>** 的数据组名称。	
			数据类型：	STRING
		<i>：	数据组的编号	
			数据类型：	INT
			值域：	1 ... $P_TOOLENVN
	编号不是固定分配给数据组，而是能够通过删除和新建数据组进行修改。在内部对数据组进行编号。 如果 **<i>** 未指向指定数据组，则返回零字符串。 如果索引**<i>** 无效，即 **<i>** 小于 1 或大于最大刀具环境数据组数 (MD18116 $MN_MM_NUM_TOOLENV)，则输出以下报警： 报警 “17020 （非法的数组索引 1）”			

11.12.5 读取刀具长度或刀具长度分量 (GETTCOR)

GETTCOR 功能用来读出刀具长度或刀具长度分量。

此时可以通过编程来设定，应当考虑哪些分量以及应当在什么样的使用条件下观察刀具。

句法

```
<Status> = GETTCOR(<Len>[, <Comp>, <Stat>, <T>, <D>, <DL>])
```

含义

GETTCOR(...):	用于读取刀具长度或刀具长度分量的预定义功能	
	在单独程序段中编程：	支持

<Status>:	函数的返回值。负值代表故障状态。		
	数据类型：		INT
	值：	0	功能正常
		-1	没有预留用于刀具环境的存储位置： MD18116 $MN_MM_NUM_TOOL_ENV = 0 即“刀具环境”功能不存在。
		-2	不存在使用 **<Stat>** 中给定名称的刀具环境。
		-3	参数 **<Comp>** 中无效的字符串。 造成该故障的原因可能是字符无效或者重复编程了字符。
		-4	无效的 T 号码
		-5	无效的 D 号码
		-6	无效的 DL 号码
		-7	在不存在的存储器模块上尝试存取。
		-8	在不存在的选项上（例如：可编程的刀具方向、刀具管理）尝试存取。
		-9	字符串 **<Comp>** 包含一个冒号（用于坐标系规范的标识），但没有跟随有效的字符用来说明所要求的坐标系。
参数			

1	<Len>：	结果矢量	
		数据类型：	REAL[11]
		矢量分量按下列顺序排列： • **<Len>** [0]：刀具类型 • **<Len>** [1]：刀沿位置 • **<Len>** [2]：横坐标 • **<Len>** [3]：纵坐标 • **<Len>** [4]：垂直坐标 • **<Len>** [5]：刀具半径 **<Comp>** 和 **<Stat>** 中所定义的坐标系用作长度分量的参考坐标系。如果在 **<Comp>** 中没有定义坐标系，则刀具长度显示在机床坐标系中。 此时横坐标、纵坐标和竖坐标在几何轴上的分配取决于所使用刀具环境中的有效平面。也就是说在使用 G17 时横坐标平行于 X，在使用 G18 时平行于 Z 等等。 分量 **<Len>**[6] 到 **<Len>**[10] 包含额外的参数，能够用于刀具的几何数据说明（例如 $TC_DP7 到 $TC_DP11 用于几何数据或者相应的磨损分量或者总补偿和安装补偿）。 这 5 个额外的元素以及刀具半径只能被定义用于分量 E、G、S 和 W。它们的计算与 **<Stat>** 无关。如果在刀具长度计算时上述四个分量有一个参与了计算，那么 **<Len>**[6] 到 **<Len>**[10] 中相应的值只能不等于零。其余的分量对结果没有影响。尺寸数据与控制器的初始系统相关 （英寸或者公制）。	

2	<Comp>:	刀具长度分量（可选）			
		数据类型：	STRING		
		字符串由两个部分字符串组成，它们通过冒号进行分隔。 常规形式："<SubStr_1> [: <SubStr_2]"			
		<SubStr_1>:	**第一个部分字符串**用来标识在计算刀具长度时应当考虑的刀具长度分量。 部分字符串中字符的顺序及其书写方式（大写或小写）都任意。在字符之间可以插入任意多个空格或制表键（white spaces 空白）。 **提示：** 在部分字符串中**不得重复**编程字符！		
			字符：	-	负号（只允许作为第一个字符！） 计算时会从整个刀具长度扣除后续字符串中所列出的分量。
				C	适配器尺寸或基础尺寸（在这两个二选一有效存在的分量中，选择对于所用刀具有效的那个）
				E	安装补偿
				G	几何尺寸
				K	运动转换 （只能在类 3、4 和 5 轴转换时运用）
				S	总补偿
				T	可定向刀架
				W	磨损
			如果第一个部分字符串（除了空白以外）为空，则表示应当在考虑所有分量的情况下计算整个刀具长度。当未给定参数 **<Comp>** 时，这也同样适用。		
		<Substr_2>:	**第二个部分字符串**用来标识输出刀具长度所在的坐标系。 它仅由一个唯一的相关字符构成。		
			字符：	A	可设定的坐标系 (ACS)
				B	基准坐标系 (BCS)
				K	运动转换的刀具坐标系 (KCS)
				M	机床坐标系 (MCS)

			T	刀具坐标系 (TCS)
			W	工件坐标系 (WCS)
			如果没有给定坐标系，则在机床坐标系（MCS）中进行计算。通过 **<Stat>** 中所定义的刀具环境来确定可能需要考虑的旋转。	
3	<_Stat>：	用来说明刀具环境的数据组名称（可选）		
		数据类型：	STRING	
		如果该参数的值为零字符串 ("") 或者值未给定，则使用当前状态。如未指定刀具，则使用当前刀具。		
4	<T>：	刀具的内部 T 号（可选）		
		数据类型：	INT	
		如果未给定该参数或者参数值为 “0”，则使用保存在 **<Stat>** 中的刀具。 如果该参数的值为 “-1”，则使用有效刀具的 T 号码。也允许用显性方式给出有效刀具的号码。 **提示：** 如果未给定 **<Stat>**，则使用当前状态作为刀具环境。因为使用 **<T> = 0** 指向了刀具环境中所储存的 T 号，所以在这里使用有效的刀具，即说明数据 **<T> = 0** 和 **<T> = -1** 在这种特殊情况中含义相同。		
5	<D>：	刀具的刀沿编号（可选）		
		数据类型：	INT	
		如果未给定该参数或者参数值为 “0”，则使用的 D 号码取决于 T 号码源。如果使用了来自刀具环境的 T 号码，则也读取刀具环境的 D 号码，否则使用当前有效刀具的 D 号码。		
6	<DL>：	与位置相关的补偿号码（可选）		
		数据类型：	INT	
		如果未给定该参数，则使用的 DL 号码取决于 T 号码源。如果使用了来自刀具环境的 T 号码，则也读取刀具环境的 D 号码，否则使用当前有效刀具的 D 号码。		

示例

`GETTCOR(_LEN)`	在机床坐标系中考虑所有分量的情况下计算当前有效刀具的刀具长度。
`GETTCOR(_LEN,"CGW:W")`	计算由适配器尺寸或基础尺寸、几何尺寸和磨损所组成的有效刀具的刀具长度。不考虑其他部分例如可定向刀架或者运动转换。在工件坐标系中进行输出。
`GETTCOR (_LEN,"-K:B")`	计算有效刀具的刀具总长度，不考虑可能有效的运动转换的长度分量。在基本坐标系中输出。
`GETTCOR (_LEN,":M","Testenv1",,3)`	在机床坐标系中计算刀具总长度，该刀具在刀具环境中使用名称“Testenv1”进行保存。然而计算与所保存的、用于刀沿编号 D3 的刀沿编号无关。

更多信息

适配器转换/可定向刀架/运动转换

可能由适配器转换、可定向刀架和运动转换所执行的旋转和分量交换是刀具环境的组成部分。因此它们总是被执行，即使在不用考虑相应长度分量的时候。如果不需要，则必须定义相应转换在其中无效的刀具环境。在许多情况中（即当机床上没有设置转换或可定向刀架时），所保存的刀具环境数据组会自动执行，以致于用户无需专门去注意这些。

车刀和磨具：根据机床数据 MD20360 $MC_TOOL_PARAMETER_DEF_MASK 计算刀具长度

如何在使用车刀和磨具时在可能存在的直径轴中计算磨损或刀具长度。

MD20360 $MC_TOOL_PARAMETER_DEF_MASK（定义刀具参数）

位	值	
0	对于车刀和磨具，端面轴的**磨损参数**作为直径值考虑：	
	= 0（缺省值）	不支持
	= 1	支持

位	值	
1	对于车刀和磨具，端面轴的**刀具长度分量**作为直径值考虑：	
	= 0（缺省值）	不支持
	= 1	支持

如果设置了相关的位，则使用系数 0.5 来计算相应的记录项。在由 GETTCOR 所提供的刀具长度中也会出现这种计算。

示例：

MD20360 $MC_TOOL_PARAMETER_DEF_MASK = 3

MD20100 $MC_DIAMETER_AX_DEF（带端面轴功能的几何轴）= “X”

X 为直径轴 （标准车床配置）

程序代码	注释
N30 $TC_DP1[1,1]=500	
N40 $TC_DP2[1,1]=2	
N50 $TC_DP3[1,1]=3.0	; 几何数据 L1
N60 $TC_DP4[1,1]=4.0	
N70 $TC_DP5[1,1]=5.0	
N80 $TC_DP12[1,1]=12.0	; 磨损 L1
N90 $TC_DP13[1,1]=13.0	
N100 $TC_DP14[1,1]=14.0	
N110 T1 D1 G18	
N120 R1=GETTCOR(_LEN,"GW")	
N130 R3=_LEN[2]	; 17.0 (= 4.0 + 13.0)
N140 R4=_LEN[3]	; 7.5 (= 0.5 * 3.0 + 0.5 * 12.0)
N150 R5=_LEN[4]	; 19.0 (= 5.0 + 14.0)
N160 M30	

运动转换和可定向刀架的长度分量

如果在计算刀具长度时需要考虑**可定向刀架**，则下列矢量会参与计算：

类型	矢量
M	I1 和 I2
T	I1、I2 和 I3
P	刀具长度不被可定向刀架影响。

在类 **5 轴转换** 中、当转换类型为 24 和 56 时下列机床数据会参与刀具长度计算：

转换类型	机床数据
24	MD24550/24650 $MC_TRAFO5_BASE_TOOL_1/2 MD24560/24660 $MC_TRAFO5_JOINT_OFFSET_1/2 MD24558/24658 $MC_TRAFO5_PART_OFFSET_1/2
56	MD24550/24650 $MC_TRAFO5_BASE_TOOL_1/2 MD24560/24660 $MC_TRAFO5_JOINT_OFFSET_1/2

转换类型 56（可移动刀具和可移动工件）相当于可定向刀架时的类型 M 。

在该 5 轴转换时，先前软件版本中的矢量 MD24560/24660 $MC_TRAFO5_JOINT_OFFSET_1/2（中第 1/2 个 5 轴转换的运动偏移矢量）与在类型 M 的可定向刀架上的两个矢量 l_1 和 l_3 的和一致。

在这两种情况下只有总和对于转换较为重要。由两个单个分量组成的方式没有意义。哪个部分分配给刀具与哪个分配给刀具台，这在计算刀具长度很有意义。因此，导入机床数据 MD24558/24658 $MC_TRAFO5_JOINT_OFFSET_PART_1/2 （平台中的运动偏移矢量）。它与矢量 l3 一致。此时，机床数据 MD24560/24660 $MC_TRAFO5_JOINT_OFFSET_1/2 不再与由 l1 和 l3 组成的和一致，而只与矢量 l1 一致。如果机床数据 MD24558/24658 $MC_TRAFO5_JOINT_OFFSET_PART_1/2 等于零，则其特性与之前版本的一致。

兼容性

此外 GETTCOR 功能和 TOOLENV 与 SETTCOR 功能一起用来替换原先在测量循环中外部实现的功能性部分。

在测量循环中只计算最终用来确定有效刀具长度的参数部分。可以编程上述的功能，使与刀具长度计算有关的测量循环特性能够重复。

11.12.6 修改刀具分量 (SETTCOR)

SETTCOR 用来在考虑所有参与单个分量计算的前提条件的情况下修改刀具分量。

说明

用于专业术语：如果在下面提到与刀具长度相关的刀具分量，则指的是矢量观点的分量，可以由它们组成刀具的总长度，例如几何尺寸或磨损。因此一个这样的分量由三个单值（L1、 L2、 L3）构成，它们在下面被称为坐标值。
所以例如刀具分量“几何尺寸”由三个坐标值 $TC_DP3 到 $TC_DP5 构成。

句法

```
<Status> = SETTCOR(<CorVal>, <Comp>, [<CorComp>, <CorMode>,
<GeoAx>, <Stat>, <T>, <D>, <DL>])
```

含义

SETTCOR(...):	用于修改刀具分量的预定义功能		
	在单独程序段中编程：		支持
<Status>：	函数的返回值。负值代表故障状态。		
	数据类型：		INT
	值：	0	功能正常
		-1	没有预留用于刀具环境的存储位置： MD18116 $MN_MM_NUM_TOOL_ENV = 0 即“刀具环境”功能不存在。
		-2	不存在使用 **<Stat>** 中给定名称的刀具环境。
		-3	参数 **<Comp>** 中无效的字符串。 造成该故障的原因可能是字符无效或者重复编程了字符。
		-4	无效的 T 号码。
		-5	无效的 D 号码。
		-6	无效的 DL 号码。
		-7	在不存在的存储器模块上尝试存取。
		-8	在不存在的选项上（例如：可编程的刀具方向、刀具管理）尝试存取。
		-9	禁止用于参数 **<CorComp>** 的数值。
		-10	禁止用于参数 **<CorMode>** 的数值。
		-11	参数 **<Comp>** 和 **<CorComp>** 的内容矛盾。
		-12	参数 **<Comp>** 和 **<CorMode>** 的内容矛盾。
		-13	参数 **<GeoAx>** 的内容没有标识几何轴。
		-14	在不存在的安装补偿上尝试写入。
参数			

1	<CorVal>：	补偿矢量 在通过 **<Stat>** 所定义的工件坐标系 (WCS) 进行分配： • **<CorVal> [0]**：横坐标 • **<CorVal> [1]**：纵坐标 • **<CorVal> [2]**：垂直坐标 如果只要修正一个刀具分量（也就是说没有矢量补偿，参见参数 **<CorMode>**），则补偿值始终在 **<CorVal>[0]** 中，它对哪根轴起作用无关紧要。不计算剩余两个分量的内容。 如果 **<CorVal>** 或者 **<CorVal>** 的一个分量与端面轴有关，则将说明数据当作**半径尺寸**来计算。这表示，例如一把刀具“伸长”了给定的尺寸，则相应的会导致两倍大小的工件直径变化。 尺寸数据与控制器的 **初始系统** 相关 （英寸或者公制）。	
		数据类型：	REAL[3]

2	<Comp>：	刀具分量 (n)			
		数据类型：	STRING		
		字符串由两个部分字符串组成，它们通过冒号进行分隔。 常规形式："<SubStr_1> [: <SubStr_2]"			
		<SubStr_1>：	**第一个部分字符串**必须始终存在且可由一个或两个字符组成。此时第一个或唯一一个字符代表第 1 个分量（Val_1）而第二个字符代表第 2 个分量（Val_2），这两个分量按照后面的参数 **<CorComp>** 和 **<CorMode>** 处理。		
			字符：	C	适配器尺寸或基础尺寸（在这两个二选一有效存在的分量中，选择对于所用刀具有效的那个）
				E	安装补偿
				G	几何尺寸
				S	总补偿
				W	磨损
		<Substr_2>：	**第二个部分字符串**为可选字符串。也可由（唯一）字母 “W” 或 “T” 组成。		
			字符：	W	如果第二个部分字符串为空或包含字母 “W”，则按照假设其在**工件**坐标系 (WCS) 中测量的方式计算补偿值。
				T	如果第二个部分字符串包含字母 “T”，则按照假设其在**刀具**坐标系（**T**ool Coordinate System, TCS）中测量的方式计算补偿值。
		在字符串中字符的书写方式（大写或小写）任意。可以插入任意多个空格或制表键（white spaces 空白）。			

3	<CorComp>:	详细说明需要进行说明的刀具数据组分量（可选）		
		数据类型：		INT
		值：	0	补偿值 **<CorVal>[0]** 与参数 **<GeoAx>** 中在工件坐标系或刀具坐标系中所传输的几何轴有关（参见参数 **<Comp>** 的说明）。即要将补偿值算进标识出的刀具分量中，使得在考虑所有会影响刀具长度计算的参数的情况下、得出在给定轴方向上刀具总长度相对预设值的变化值作为结果。 应当通过 **<Comp>** 中给定分量的补偿与 **<CorMode>** 中（参见后续参数）给定的符号计算方法来达到该变化。因此所生成的补偿可以在全部三根轴分量上生效。
			1	和 “0” 一样，但是为矢量模式。矢量 **<CorVal>** 的内容与工件坐标系或刀具坐标系中的横坐标、纵坐标、竖坐标有关（参见参数 **<Comp>** 的说明）。 不计算后续参数 **<GeoAx>**。
			2	矢量补偿，即 L1、 L2 和 L3 能够同时改变。 与方案 “0” 和 “1” 相反， **<CorVal>** 中所包含的补偿值与刀具 Val_1 分量的坐标有关（参见后续参数 **<CorMode>**）。 相对于工件坐标系，可能存在的刀具斜置设定对补偿没有影响。
			3 - 5	刀具长度补偿 L1 到 L3 （$TC_DP3 到 $TC_DP5）或者磨损、安装补偿或总补偿时相应的值。 补偿值包含在 **<CorVal>[0]** 中。在刀具 Val_1 分量的坐标中测量该值（参见后续参数 **<CorMode>**）。相对于工件坐标系，可能存在的刀具斜置设定对补偿没有影响。
			6	刀具半径补偿（$TC_DP6）或者磨损、安装补偿或总补偿时相应的值。考虑机床数据 MD20360 $MC_TOOL_PARAMETER_DEF_MASK 中的位 10 和 11（将直径或直径磨损数据作为半径或直径数据计算）。
			7 – 11	$TC_DP7 到 $TC_DP11 的补偿或者磨损、安装补偿或总补偿时相应的值。这些参数的处理和刀具半径时一样。
		如果未给定该参数，则值为 “0”。		

4	<CorMode>:	详细说明了所进行写操作的方式（可选）		
		数据类型：		INT
		值：	0	Val_{1neu} = <CorVal>
			1	Val_{1neu} = Val_{1alt} + <CorVal>
			2	Val_{1neu} = <CorVal> Val_{2neu} = 0
			3	Val_{1neu} = Val_{1alt} + Val_{2alt} + <CorVal> Val_{2neu} = 0
		书写方式 Val_{1alt} + Val_{2alt} 可以象征性理解。如果两个分量（基于 <_Stat> 状态）的计算有区别，也就是说在两个分量之间存在一个有效的旋转，则 Val_{2alt} 在累加前要进行转换，使得所生成的刀具长度在删除 Val_{2neu} 之后并加上 <CorVal> 之前保持不变。 _CORVAL 总是与 Val_1 有关。<CorVal> 是一个根据参数 <Comp> 的第二部分在工件坐标系 (WCS) 或刀具坐标系 (TCS) 中测量出的值。因此相对于需要算入的刀具分量，它也许已经进行了转换。所以它不能直接与保存的值一起计算，而必须在累加到 Val_1 或 Val_2 之前转换返回。这可能会导致补偿在不是在通过 <CorComp> 定义的其他轴上生效，或者在多根轴上生效。 对于这种情况 <CorComp> = 0，也就是说如果 <CorVal> 包含的不是矢量而只是一个单值，则在测量 <CorVal> 的坐标 (WKS/TCS) 中执行所描述的操作。尤其在将方案 2 和 3 中的 Val_{2neu} 归零时也适用。该结果会被转换返回到刀具坐标中。这可能会导致待归零的坐标值（L1、L2、 L3）不为零或者原先等于零的坐标值现在不等于零。逐步执行用于全部三根轴的相应操作，使要删除分量的全部三个坐标值始终为零。如果刀具相对于工件坐标系未旋转，或者旋转后所有刀具分量都平行于坐标轴（轴交换），则可以保证每次只有一个刀具坐标发生变化。 同样的操作 (<CorMode>) 使用 <CorComp> = 0 对全部三根坐标轴按任意顺序逐步执行与使用 <CorComp> = 1 一次性执行同样操作的效果相同。 对于参数值 “0” 和 “1”，参数 <Comp> 必须包含一个字符，对于参数值 “2” 和 “3”，参数 <Comp> 必须包含两个字符。 示例： <Comp> 包含字符串 “ES”，<CorMode> 包含值 “2” ⇒ 安装补偿$_{新}$ = <CorVal>，总补偿$_{新}$ = 0 如果未给定参数 <CorMode>，则值为 “0”。		

5	<GeoAx>:	给定几何轴的序号，在该轴上测量出了补偿值 <CorVal>[0]（可选）	
		数据类型：	INT
		取值范围：	0 ... 2
		序号 0 ... 2 相对于当前刀具环境下生效平面 (G17/G18/G19) 中的横坐标、纵坐标、竖坐标。 该参数的内容仅在参数 <CorComp> 的值为“0”时计算。	
6	<Stat>:	用来说明刀具环境的数据组名称（可选）	
		数据类型：	STRING
		如果该参数的值为零字符串 ("") 或者值未给定，则使用当前状态。如未指定刀具，则使用当前刀具。	
7	<T>:	刀具的内部 T 号（可选）	
		数据类型：	INT
		如果未给定该参数或者参数值为 “0”，则使用保存在 <Stat> 中的刀具。 如果该参数的值为 “-1”，则使用有效刀具的 T 号码。也允许用显性方式给出有效刀具的号码。 **提示：** 如果未给定 <Stat>，则使用当前状态作为刀具环境。因为使用 <T> = 0 指向了刀具环境中所储存的 T 号，所以在这里使用有效的刀具，即说明数据 <T> = 0 和 <T> = -1 在这种特殊情况中含义相同。	
8	<D>:	刀具的刀沿编号（可选）	
		数据类型：	INT
		如果未给定该参数或者参数值为 “0”，则使用的 D 号码取决于 T 号码源。如果使用了来自刀具环境的 T 号码，则也读取刀具环境的 D 号码，否则使用当前有效刀具的 D 号码。	
9	<TL>:	与位置相关的补偿号码（可选）	
		数据类型：	INT
		如果未给定该参数，则使用的 DL 号码取决于 T 号码源。如果使用了来自刀具环境的 T 号码，则也读取刀具环境的 D 号码，否则使用当前有效刀具的 D 号码。如果 T、 D 和 DL 详细说明了没有位置相关补偿的刀具，则在参数 <Comp> 中不允许说明总补偿或安装补偿（<Status> 中的故障代码）。	

说明

不是所有这三个参数 <Comp>、<CorComp> 和 <CorMode> 的可能组成都有意义。例如 <CorComp> 中的计算方法 3 需要 <Comp> 中两个字符的说明数据。如果给定了禁止的参数组合，则在 <Status> 中会返回一个相应的故障代码。

示例

示例 1

程序代码	注释
N10 DEF REAL _CORVAL[3]	
N20 $TC_DP1[1,1]=120	; 铣刀
N30 $TC_DP3[1,1]=10.0	; 几何数据 L1
N40 $TC_DP12[1,1]=1.0	; 磨损 L1
N50 _CORVAL[0]=0.333	
N60 T1 D1 G17 G0	
N70 R1=**SETTCOR(_CORVAL,"G",0,0,2)**	
N80 T1 D1 X0 Y0 Z0	; ==> MCS 位置 X0.000 Y0.000 **Z1.333**
N90 M30	

<CorComp> 等于 “0”，因此通过补偿值 0.333 来替代 Z 方向上有效的几何分量的坐标值。

由此生成的刀具总长度为：L1 = 0.333 + 1.000 = 1.333

示例 2

程序代码	注释
N10 DEF REAL _CORVAL[3]	
N20 $TC_DP1[1,1]=120	; 铣刀
N30 $TC_DP3[1,1]=10.0	; 几何数据 L1
N40 $TC_DP12[1,1]=1.0	; 磨损 L1
N50 _CORVAL[0]=0.333	
N60 T1 D1 G17 G0	
N70 R1=**SETTCOR(_CORVAL,"W",0,1,2)**	
N80 T1 D1 X0 Y0 Z0	; ==> MCS 位置 X0.000 Y0.000 Z11.333
N90 M30	

<CorComp> 等于 1，因此将 Z 方向上有效的补偿值 0.333 加到磨损值 1.0 上。

由此生成的刀具总长度为：L1 = 10.0 + 1.333 = 11.333

示例 3

程序代码	注释
N10 DEF REAL _CORVAL[3]	
N20 $TC_DP1[1,1]=120	; 铣刀
N30 $TC_DP3[1,1]=10.0	; 几何数据 L1
N40 $TC_DP12[1,1]=1.0	; 磨损 L1
N50 _CORVAL[0]=0.333	
N60 T1 D1 G17 G0	
N70 R1=**SETTCOR(_CORVAL,"GW",0,2,2)**	
N80 T1 D1 X0 Y0 Z0	; ==> MCS 位置 X0.000 Y0.000 Z0.333
N90 M30	

<CorComp> 为 2，因此 Z 方向上生效的补偿会被记录至几何数据分量（旧值会被覆盖），且磨损值会被删除。

由此生成的刀具总长度为：L1 = 0.333 + 0.0 = 0.333

示例 4

程序代码	注释
N10 DEF REAL _CORVAL[3]	
N20 $TC_DP1[1,1]=120	; 铣刀
N30 $TC_DP3[1,1]=10.0	; 几何数据 L1
N40 $TC_DP12[1,1]=1.0	; 磨损 L1
N50 _CORVAL[0]=0.333	
N60 T1 D1 G17 G0	
N70 R1=**SETTCOR(_CORVAL,"GW",0,3,2)**	
N80 T1 D1 X0 Y0 Z0	; ==> MCS 位置 X0.000 Y0.000 Z11.333
N90 M30	

<CorComp> 为 3，因此磨损值和补偿值会被累加至几何数据分量，且磨损分量会被删除。

由此生成的刀具总长度为：L1 = 11.333 + 0.0 = 11.333。

示例 5

程序代码	注释
N10 DEF REAL _CORVAL[3]	
N20 $TC_DP1[1,1]=120	; 铣刀
N30 $TC_DP3[1,1]=10.0	; 几何数据 L1
N40 $TC_DP12[1,1]=1.0	; 磨损 L1
N50 _CORVAL[0]=0.333	
N60 T1 D1 G17 G0	
N70 R1=**SETTCOR(_CORVAL,"GW",0,3,0)**	
N80 T1 D1 X0 Y0 Z0	; ==> MCS 位置 X0.333 Y0.000 Z11.000
N90 M30	

同上一示例一样，<CorComp> 为 3，但是补偿此时则是对序号为 0 的几何轴（X 轴）生效，对于铣刀刀具分量 L3 基于 G17 被指定给该轴。因此调用 SETTCOR 不会影响刀具参数 $TC_DP3 和 $TC_DP12。补偿值会被记录至 $TC_DP5。

示例 6

程序代码	注释
N10 DEF REAL _CORVAL[3]	
N20 $TC_DP1[1,1]=500	; 车刀
N30 $TC_DP3[1,1]=10.0	; 几何数据 L1
N40 $TC_DP4[1,1]=15.0	; 几何数据 L2
N50 $TC_DP12[1,1]=10.0	; 磨损 L1
N60 $TC_DP13[1,1]=0.0	; 磨损 L2
N70 _CORVAL[0]=5.0	
N80 ROT Y-30	
N90 T1 D1 G18 G0	
N100 R1=**SETTCOR(_CORVAL,"GW",0,3,1)**	
N110 T1 D1 X0 Y0 Z0	; ==> MCS 位置 X24.330 Y0.000 Z17.500
N120 M30	

刀具为车刀。在 N80 程序段中会激活一个框架旋转，从而使基本坐标系 (BCS) 相对于工件坐标系 (WCS) 旋转。补偿值 (N70) 在 WCS 中对序号为 “1” 的几何轴生效，即 G18 生效时作用于 X 轴。由于 <CorMode> = 3，执行 N100 后 WCS 的 X 轴方向上的刀具磨损必须变为零。

因此相关刀具参数的内容在程序末尾处为：

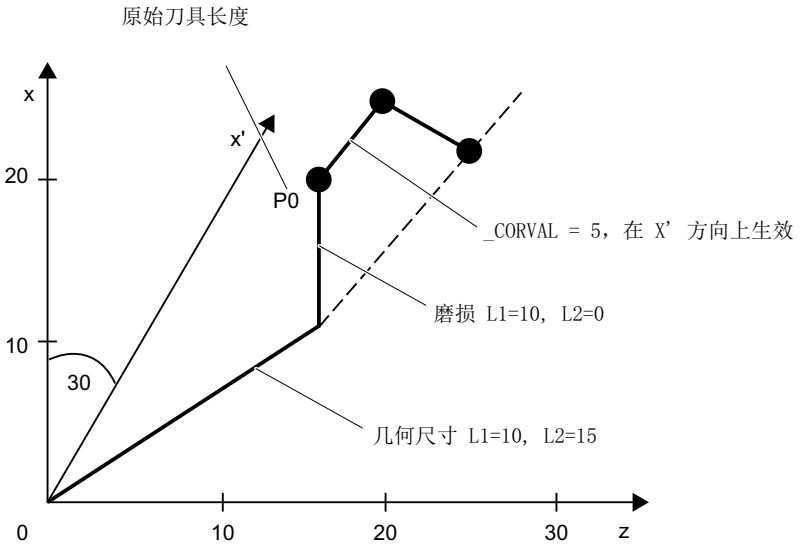
$TC_DP3[1,1]：21.830；几何数据 L1

$TC_DP4[1,1]：21.830；几何数据 L2

$TC_DP12[1,1]：2.500；磨损 L1

$TC_DP13[1,1]：-4.330；磨损 L2

几何关系如下图所示。包含 _CORVAL 在内的总磨损会映射至 WCS 中的 X' 方向。这样便得到点 P2。该点的坐标（以 X-Y 坐标测量）会被记录至刀具的几何分量。磨损中差值矢量 $P_2 - P_1$ 为多余。因此磨损不再包含 _CORVAL 方向的分量。



在 N110 后通过以下指令继续执行示例程序：这样一来剩余的磨损会被完全接收至几何数据中，因为补偿现在作用于 Z' 轴（参数 <GeoAx> 为 0）：

```
N120 _CORVAL[0]=0.0
N130 R1=SETTCOR(_CORVAL,"GW",0,3,0)
N140 T1 D1 X0 Y0 Z0 ; ==> MCS 位置 X24.330 Y0.000 Z17.500
```

由于新的补偿值为“0”，因此刀具总长度以及 N140 中逼近的位置不得更改。若 N120 中 _CORVAL 不等于“0”，则会得到新的刀具总长度，N140 中的位置也会因此变化，但刀具长度的磨损分量始终为零，也就是说刀具总长度在每种情形下都包含在刀具的几何分量中。

通过参数 <CorComp> = 1（矢量补偿）调用一次 SETTCOR 得到的结果与通过 <CorComp> = 0 调用两次时相同。

程序代码	注释
N10 DEF REAL _CORVAL[3]	
N20 $TC_DP1[1,1]=500	; 车刀
N30 $TC_DP3[1,1]=10.0	; 几何数据 L1
N40 $TC_DP4[1,1]=15.0	; 几何数据 L2
N50 $TC_DP12[1,1]=10.0	; 磨损 L1
N60 $TC_DP13[1,1]=0.0	; 磨损 L2
N70 _CORVAL[0]=0.0	
N71 _CORVAL[1]=5.0	
N72 _CORVAL[2]=0.0	
N80 ROT Y-30	
N90 T1 D1 G18 G0	
N100 R1=**SETTCOR(_CORVAL,"GW",1,3,1)**	
N110 T1 D1 X0 Y0 Z0	; ==> MCS 位置 X24.330 Y0.000 Z17.500

程序代码	注释
N120 M30	

在此情形下，N100 中首次调用 SETTCOR 后刀具的所有磨损分量都会立即归零。

示例 7

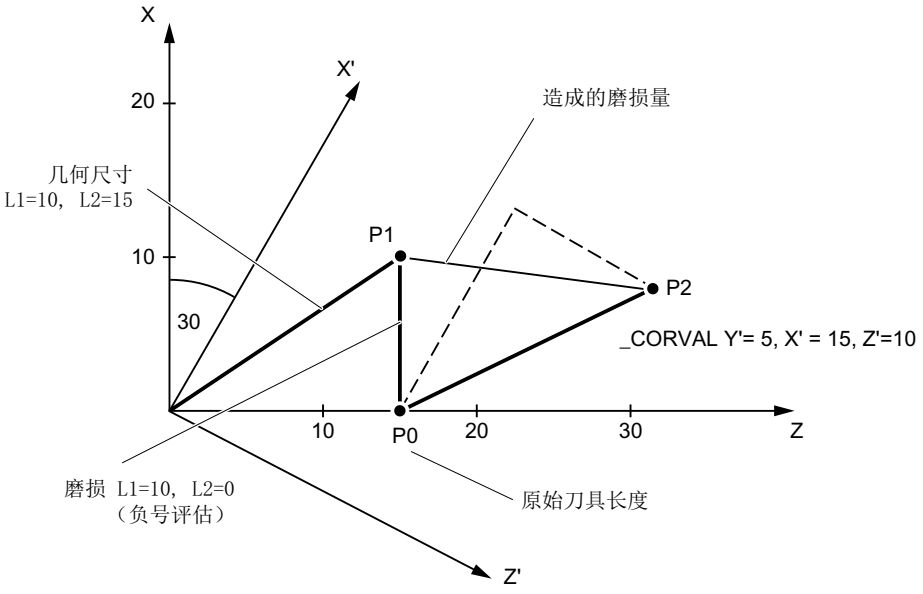
程序代码	注释
N10 DEF REAL _CORVAL[3]	
N20 $TC_DP1[1,1]=500	; 车刀
N30 $TC_DP3[1,1]=10.0	; 几何数据 L1
N40 $TC_DP4[1,1]=15.0	; 几何数据 L2
N50 $TC_DP12[1,1]=10.0	; 磨损 L1
N60 $TC_DP13[1,1]=0.0	; 磨损 L2
N70 _CORVAL[0]=5.0	
N80 ROT Y-30	
N90 T1 D1 G18 G0	
N100 R1=**SETTCOR(_CORVAL,"GW",3,3)**	
N110 T1 D1 X0 Y0 Z0	; ==> MCS 位置 X25.000 Y0.000 Z15.000
N120 M30	

与示例 6 相比，此处参数 **<CorComp> = 3**，因此可省去对参数 **<GeoAx>** 的设定。_CORVAL[0] 中包含的值现在直接对刀具分量 L1 生效，N80 中的旋转对结果无影响，$TC_DP12 中的磨损分量与 _CORVAL[0] 一同接收至几何数据分量，这样一来 $TC_DP13 使得刀具总长度在 N100 中首次调用 SETTCOR 后即已处于刀具的几何分量中。

示例 8

程序代码	注释
N10 DEF REAL _CORVAL[3]	
N20 $TC_DP1[1,1]=500	; 车刀
N30 $TC_DP3[1,1]=10.0	; 几何数据 L1
N40 $TC_DP4[1,1]=15.0	; 几何数据 L2
N50 $TC_DP5[1,1]=20.0	; 几何数据 L3
N60 $TC_DP12[1,1]=10.0	; 磨损 L1
N70 $TC_DP13[1,1]=0.0	; 磨损 L2
N80 $TC_DP14[1,1]=0.0	; 磨损 L3
N90 $SC_WEAR_SIGN=TRUE	
N100 _CORVAL[0]=10.0	
N110 _CORVAL[1]=15.0	
N120 _CORVAL[2]=5.0	
N130 ROT Y-30	
N140 T1 D1 G18 G0	
N150 R1=**SETTCOR(_CORVAL,"W",1,1)**	
N160 T1 D1 X0 Y0 Z0	; ==> MCS 位置 X7.990 Y25.000 Z31.160
N170 M30	

在 N90 中置位了设定数据 SD42930 $SC_WEAR_SIGN，也就是说磨损量必须带负号计算。补偿为矢量 (<CorComp> = 1)，且补偿矢量必须叠加至磨损 (<CorMode> = 1)。Z-X 平面中的几何关系如下图所示：



<CorMode> = 1 使得刀具的几何分量保持不变。WCS（围绕 y 轴旋转）中定义的补偿矢量接收至磨损分量时，必须确保图 3 中的刀具总长度指向点 P_2。因此得到的磨损分量由点 P_1 和 P_2 的距离确定。

但由于设置了设定数据 SD42930 $SC_WEAR_SIGN，磨损量必须为负值，因此得到的补偿必须带负号输入至补偿存储器。因此相关刀具参数的内容在程序末尾处为：

$TC_DP3[1,1]：10.000；几何数据 L1（不变）

$TC_DP4[1,1]：15.000；几何数据 L2（不变）

$TC_DP5[1,1]：10.000；几何数据 L3（不变）

$TC_DP12[1,1]：2.010；磨损 L1 (= 10 - 15 * cos(30) + 10 * sin(30))

$TC_DP13[1,1] : -16.160；磨损 L2 (= -15 * sin(30) - 10 * cos(30))

$TC_DP14[1,1] : -5.000；磨损 L3

在 Y 方向上的 L3 分量上，无需通过框架旋转另外编译便可识别出设定数据 SD42930 $SC_WEAR_SIGN 的作用。

更多信息

车刀/磨具：根据机床数据 MD20360 $MC_TOOL_PARAMETER_DEF_MASK 计算刀具长度

如何在使用车刀和磨具时在可能存在的直径轴中计算磨损或刀具长度，可通过以下机床数据确定：

MD20360 $MC_TOOL_PARAMETER_DEF_MASK.<位> = <数值>

<位>	<值>	含义
0	0	使用车刀和磨具时，端面轴的**磨损参数**是**半径值**：
	1	使用车刀和磨具时，端面轴的**磨损参数**是**直径值**：
1	0	使用车刀和磨具时，端面轴的**刀具长度分量**是**半径值**：
	1	使用车刀和磨具时，端面轴的**刀具长度分量**是**直径值**：

如果设置了相关的位，则使用系数 0.5 来计算相应的记录项。借助 SETTCOR 进行补偿，使得整个有效的刀具长度变化等于 <CorVal> 中所传输的值。基于机床数据 MD20360 $MC_TOOL_PARAMETER_DEF_MASK 在长度计算时使用系数 0.5 来计算其长度的分量补偿，必须使用双重传输值进行补偿：

示例

MD20360 $MC_TOOL_PARAMETER_DEF_MASK = 2（在直径轴中的刀具长度必须乘以系数 0.5）

X 轴是直径轴。

程序代码	注释
N10 DEF REAL _LEN[11]	
N20 DEF REAL _CORVAL[3]	
N30 $TC_DP1[1,1]=500	; 刀具类型
N40 $TC_DP2[1,1]=2	; 刀沿位置
N50 $TC_DP3[1,1]=3.	; 几何尺寸 - 长度 1
N60 $TC_DP4[1,1]=4.	; 几何尺寸 - 长度 2
N70 $TC_DP5[1,1]=5.	; 几何尺寸 - 长度 3
N80 _CORVAL[0]=1.	
N90 _CORVAL[1]=1.	
N100 _CORVAL[2]=1.	
N110 T1 D1 G18 G0 X0 Y0 Z0	; ==> MCS 位置 X1.5 Y5 Z4
N120 R1=SETTCOR(_CORVAL,"G",1,1)	
N130 T1 D1 X0 Y0 Z0	; ==> MCS 位置 X2.5 Y6 Z5
N140 R3=$TC_DP3[1,1]	; = 5. = (3.000 + 2.*1.000)
N150 R4=$TC_DP4[1,1]	; = 5. = (4.000 + 1.000)
N160 R5=$TC_DP5[1,1]	; = 6. = (5.000 + 1.000)
N170 M30	

每根轴中的刀具长度补偿应为 1 mm（N80 至 N100）。因此长度 L2 和 L3 中分别会在原始长度上添加 1 mm。对 L1 中的原始刀具长度则会添加双倍的补偿值 (2 mm)，从而使刀具总长度变化 1 mm。比较程序段 N110 和 N130 中逼近的位置时，可识别出每个轴位置变化了 1 mm。

11.13 读取刀具长度 L1、 L2、 L3 对坐标轴的指定关系 (LENTOAX)

“LENTOAX” 功能提供了关于刀具长度 L1、 L2 和 L3 当刀具**有效**时在横坐标、纵坐标和竖坐标上的分配信息。将横坐标、纵坐标和竖坐标分配到几何轴受框架和有效平面 (G17 - G19) 的影响。

B 此时仅观察刀具的几何尺寸部分（$TC_DP3[<t>,<d>] 到 $TC_DP5[<t>,<d>]），也就是说其他分量（例如磨损）可能出现偏移的轴分配对结果没有影响。

句法

```
<Status> = LENTOAX(<AxInd>, <Matrix>[, <Coord>])
```

原理

LENTOAX(...):	用于读取生效刀具 L1、L2 和 L3 与坐标轴的对应关系的预定义功能		
	在单独程序段中编程：		支持
<Status>:	函数的返回值。负值代表故障状态。		
	数据类型：		INT
	值：	0	功能正常 <AxInd> 中的信息足够用于描述（所有的刀具长度分量平行于几何轴）。
		1	功能正常，要进行正确的描述必须运用 <Matrix> 的内容 （刀具长度分量不平行于几何轴）。
		-1	参数 <Coord> 中无效的字符串。
		-2	没有刀具有效。
参数			

1	<AxInd>:	如果刀具长度分量平行于几何轴，则会将分配给长度分量 L1 到 L3 的轴指数在 **<AxInd>** 区域中返回提供。 • **<AxInd>** [0]：横坐标 • **<AxInd>** [1]：纵坐标 • **<AxInd>** [2]：垂直坐标		
		数据类型：		INT[3]
		值：	0	分配不存在（轴不存在）
			1 ... 3 或 -1 ... -3	长度编号，在相应坐标轴上有效。 符号为负，当刀具长度分量指向负的坐标方向时。
		如果不是所有的长度分量都平行或都不平行于几何轴，则会将包含有刀具长度分量最大部分的轴的指数返回到 **<AxInd>** 中。在这种情况下（假如功能没有由于其他原因发出故障）返回值为 **<Status> = 1**。刀具长度分量 L1 到 L3 和几何轴 1 到 3 对应关系可以通过第 2 个参数 _MATRIX 的内容进行完整说明。		
2	<Matrix>:	矩阵将刀具长度矢量（L1=1、L2=1、L3=1）映像到坐标轴（横坐标、纵坐标和竖坐标）矢量，也就是说 列 是按 L1、L2、L3 顺序分配的刀具长度分量， 行 是横坐标、纵坐标和竖坐标顺序的轴。		
		数据类型：		REAL
		在矩阵中所有元素始终有效，即使附属于坐标轴的几何轴不存在，即当 **<AxInd>** 中相应的记录项为零时。		

3	<Coord>：	标记分配生效的坐标系（可选）		
		数据类型：		STRING
		字符：	MCS M	刀具长度在机床坐标系中的映像
			BCS B	刀具长度在基本坐标系中的映像
			WCS W	刀具长度在工件坐标系中的映像（默认）
			KCS K	刀具长度在运动转换的刀具坐标系中的映像
			TCS T	刀具长度在刀具坐标系中的映像
		在字符串中字符的书写方式（大写或小写）任意。 如果未给定参数 **<Coord>**，则使用 WCS（默认）。		

说明

在 TCS 中所有的刀具长度分量始终都平行或不平行于轴。

只有当镜像有效并设置了下列设定数据时，分量才会不平行：

SD42900 $SC_MIRROR_TOOL_LENGTH （镜像时的刀具长度符号变换）

示例

G17 时铣刀的标准情况。

L1 在 Z（竖坐标）方向生效，L2 在 Y（纵坐标）方向生效，L3 在 X（横坐标）方向生效。

以下列形式调用功能：

```
<Status>=LENTOAX(<AxInd>,<Matrix>,"WCS")
```

结果参数 **<AxInd>** 包含有数值：

<AxInd>[0] = 3

<AxInd>[1] = 2

<AxInd>[2] = 1

或简短：(3, 2, 1)

从属的矩阵 (**<Matrix>**) 在这种情况下：

$$\text{<Matrix>} = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{pmatrix}$$

从 G17 转换到 G18 或 G19 结果不变，因为长度分量到几何轴的分配以同样的方式发生了改变，像横坐标、纵坐标和竖坐标的分配一样。

在 G17 有效时编程一个绕 Z 旋转 60 度的框架旋转，例如使用
```
ROT Z60
```

竖坐标的方向（Z 方向）不变， L2 的主体现在位于新的 X 轴方向上，L1 的主体则位于负的 Y 轴方向上。因此返回值 (<Status>) 为 1，<AxInd> 包含数值（2, -3, 1）。

从属的矩阵 (<Matrix>) 在这种情况下：

$$\text{<Matrix>} = \begin{pmatrix} 0 & \sin 60° & \cos 60° \\ 0 & \cos 60° & -\sin 60° \\ 1 & 0 & 0 \end{pmatrix}$$

轨迹特性

12

12.1 切向控制

12.1.1 定义耦合 (TANG)

通过预定义程序 TANG(...) 定义一根回转轴（跟随轴）与两根几何轴（引导轴）之间的切向耦合。此时，跟随轴连续对准引导轴的轨迹切线。

说明

耦合系数

不必精确编程耦合系数 1。
通过将耦合系数 -1 来旋转切向轴的方向。

句法

```
TANG(<跟随轴>、<引导轴_1>、<引导轴_2>、<耦合系数>、<坐标系>、<优化>)
```

含义

TANG(...):	定义切向耦合	
<跟随轴>:	跟随轴（回转轴）的名称	
	数据类型：	AXIS
	值域：	通道轴名称
<引导轴_1> <引导轴_2>:	引导轴（几何轴）的名称 [1)]	
	数据类型：	AXIS
	值域：	通道的几何轴
<耦合系数>:	用于修改引导轴轨迹切线的跟随轴角度变化的系数 n： 角度变化$_{跟随轴}$ = 角度变化$_{轨迹切线}$ * n	
	数据类型：	REAL
	缺省值：	1.0

<坐标系>：	生效坐标系 2)		
	数据类型：	CHAR	
	值：	"B"：	基本坐标系（缺省值）
		"W"：	工件坐标系（不可用）
<优化>：	优化方式		
	数据类型：	CHAR	
	值：	"S"：	标准（缺省值） 回转轴的动态响应不会对引导轴产生影响。如果回转轴的动态响应高于跟踪所需的动态响应，该运行则足够精确。如果回转轴的动态响应还不足以修改轨迹切线，回转轴沿着非指定的平滑距离的校准则会与设定校准产生偏差。
		"P"：	在计划引导轴的轨迹时会考虑回转轴的动态响应。 为此，在通过 TANGON() 激活切向耦合时，必须给定两个附加参数： ● 平滑距离 ● 角度公差 参见章节“激活耦合 (TANGON) (页 552)” **提示** 结合运动转换，建议使用优化方式“P”。
提示 缺省值不必精确编程。			

[1] **提示** 作为切向耦合的引导轴，必须以机床的初始位置为基准、在机床坐标系 (MCS) 中按编程的轨迹运行几何轴。如果在带旋转头的铣床上使用旋转循环 CYCLE800，则根据循环配置在工件坐标系中通过几何轴 X 和 Y 进行插补。但必须通过在机床坐标系 (MCS) 中按编程的轨迹运行的几何轴将切向耦合定义为引导轴。为此，作为引导轴，几何轴必须在机床处于**非旋转**状态下使用。
[2] **提示** 基准坐标系 (BCS) 不可相对于 MCS 旋转。如果通过指令 ROT 或通过旋转循环 CYCLE800 旋转 BCS，切向控制则无法再准确运行。

12.1.2 激活中间程序段生成 (TLIFT)

如果跟随轴在编程的引导轴轨迹的某个位置上的切线变化超出了机床数据 MD37400 $MA_EPS_TLIFT_TANG_STEP 中设置的限值，其他轨迹设计则须依据设置的拐角特性进行。如果不使用预定义程序 TLIFT(...)，则根据通过 TANG(...) (页 549) 和 TANGON(...) (页 552) 编程的平滑特性运行。

激活中间程序段生成

通过编程 TLIFT(...) 和 TANG(...)，从预处理开始，在该轨迹位置上发现拐角时系统会添加一个由控制系统自动生成的中间程序段。

执行程序时，引导轴在到达中间程序段时停止。在中间程序段中，具有最大轴动态响应的跟随轴按下一程序段的轨迹切线的方向旋转。之后，引导轴再次在编程的轨迹上运行。

取消中间程序段生成

为取消中间程序段生成，必须借助 TANG(...) 重新定义切向耦合，但不包括下一次借助 TLIFT(...) 激活中间程序段生成。

句法

```
TLIFT(<跟随轴>)
```

含义

TLIFT(...):	激活带中间程序段计算的拐角识别	
<跟随轴>:	跟随轴（回转轴）的名称	
	数据类型:	AXIS
	值域:	通道轴名称

跟随轴的转速

轨迹轴

如果跟随轴在激活切向耦合前已经作为轨迹轴运行，则作为轨迹轴在中间程序段中旋转。

通过 FGREF[<轴>]=0.001 设定参考半径，以设置的最大轴速度旋转：

MD32000 $MA_MAX_AX_VELO[<跟随轴>]

定位轴

如果跟随轴在激活切向耦合前尚未作为轨迹轴运行，则作为定位轴在中间程序段中旋转。

此时，以设置的定位轴速度旋转：

MD32060 $MA_POS_AX_VELO[<跟随轴>]

12.1.3 激活耦合 (TANGON)

通过预定义程序 TANGON(...) 激活之前通过 TANG(...) (页 549) 定义的切向耦合。之后，在下一次运行引导轴时，跟随轴会连续对准引导轴的轨迹切线。

跟随轴的角度

跟随轴同轨迹切线所形成的角度取决于 TANG(...) 中给定的传动比、机床数据 MD37402 $MA_TANG_OFFSET 中设置的偏移角和 TANGON(...) 时给定的额外生效的偏移角。

优化 “P”

如果在定义切向耦合 (TANG(...)) 时给定值 “P” 作为优化参数，则须在激活耦合时给定参数“平滑距离”和参数“角度公差”（可选）。

如果给定值 0 作为角度公差，则只有参数“平滑距离”生效。

如果给定一个大于 0 的值作为角度公差，有效平滑距离由设置的最小平滑距离和以设置的角度公差为基准的平滑距离得出。

如果跟随轴的动态响应不满足设定条件，则相应地减小跟随轴的轨迹速度。

句法

TANGON(<跟随轴>、<偏移角>、<平滑距离>、<角度公差>)

含义

TANGON(...):	激活切向控制	
<跟随轴>:	跟随轴（回转轴）的名称	
	数据类型：	AXIS
	值域：	通道轴名称
<偏移角>:	跟随轴与轨迹切线之间的偏移角 参考点为回转轴零点。	
	数据类型：	REAL
<平滑距离>:	允许的最大平滑距离 如果平滑距离就动态响应条件来说较大，则减小引导轴的轨迹速度。	
	数据类型：	REAL
<角度公差>:	跟随轴零位与轨迹切线之间允许的最大角度公差	
	数据类型：	REAL

12.1.4 取消耦合 (TANGOF)

通过预定义程序 TANGOF(...) 取消通过 TANG(...) (页 549) 定义和通过 TANGON(...) (页 552) 激活的切向耦合。之后，跟随轴不再对准引导轴的轨迹切线。取消后，跟随轴在引导轴上的耦合仍然保留，以避免触发以下功能：

- 平面切换
- 几何轴切换
- 为跟随轴定义新的切向耦合

只有在通过 TANGDEL(...) (页 554) 删除耦合后，才能完全取消跟随轴在引导轴上的耦合。

编程

```
TANGOF(<跟随轴>)
```

含义

TANGOF(...):	取消切向耦合	
<跟随轴>:	跟随轴（回转轴）的名称	
	数据类型:	AXIS
	值域:	通道轴名称

12.1.5 删除耦合 (TANGDEL)

在通过 TANGOF(...) (页 553) 取消切向耦合后，通过 TANG(...) (页 549) 定义的切向耦合仍然保留。现有切向耦合之后仍会避免触发以下功能：

- 平面切换
- 几何轴切换
- 为跟随轴定义新的切向耦合

在通过 TANGOF(...) 取消切向耦合后，通过预定义程序 TANGDEL(...) 删除现有切向耦合。

句法

```
TANGDEL(<跟随轴>)
```

含义

TANGDEL(...):	删除通过 TANG() 定义的切向耦合	
	生效方式:	逐段式
<跟随轴>:	应删除切向耦合的跟随轴的名称	
	数据类型:	AXIS
	值域:	通道轴名称

示例

引导轴切换

在通过另一根引导轴为跟随轴定义新的切向耦合前，必须先删除现有切向耦合。

程序代码	注释
N10 **TANG**(A, X, Y, 1)	; 定义跟随轴 A 的切向耦合：A 到 X 和 Y
N20 TANGON(A)	; 激活跟随轴 A 的切向耦合
N30 X10 Y20	
...	
N80 TANGOF(A)	; 取消跟随轴 A 的切向耦合
N90 **TANGDEL**(A)	; 删除跟随轴 A 的切向耦合
...	
N120 **TANG**(A, X, Z)	; 为跟随轴 A 定义新的切向耦合
N130 TANGON(A)	; 为跟随轴 A 激活新的切向耦合
...	

几何轴切换

在对现有耦合进行几何轴切换前，必须先删除耦合。

程序代码	注释
N10 GEOAX(2, Y1)	; 第 2 几何轴 = 机床轴 Y1
N20 TANG(A, X, Y)	; 定义跟随轴 A 的切向耦合
N30 TANGON(A, 90)	; 激活跟随轴 A 的切向耦合
N40 G2 F8000 X0 Y0 I0 J50	; 运动程序段
N50 TANGOF(A)	; 取消跟随轴 A 的切向耦合
N60 **TANGDEL**(A)	; 删除跟随轴 A 的切向耦合
N70 **GEOAX**(2, Y2)	; 第 2 几何轴 = 机床轴 Y2
N80 TANG(A, X, Y)	; 为跟随轴 A 定义新的切向耦合
N90 TANGON(A, 90)	; 为跟随轴 A 激活新的切向耦合
...	

12.2 进给曲线 (FNORM, FLIN, FCUB, FPO)

为了较为灵活的设定进给曲线，根据 DIN 66025 的规定对进给编程增加线性曲线和三次曲线。

三次曲线可以直接编程或作为插补样条编程。 从而可以根据待加工工件的曲度持续编程平滑的速度曲线。

这种速度曲线实现了平滑、没有急动的加速度变化，并进而完成了均匀的工件表面加工。

句法

```
F… FNORM
F… FLIN
F… FCUB
F=FPO(…,…,…)
```

含义

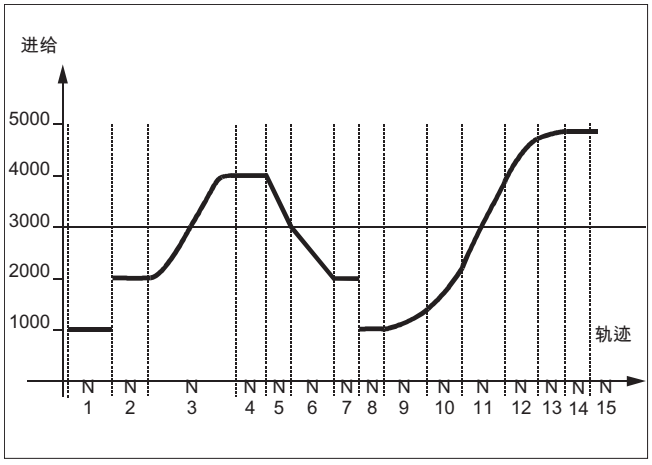
FNORM:	初始设置。 进给值通过程序段中的位移来规定，并且作为模态值有效。
FLIN:	轨迹速度曲线图**线性：** 从程序段开头到程序段末尾的范围内，轴从当前值开始以进给值进行线性位移，然后模态值生效。 这种属性可以和 G93 和 G94 组合使用。
FCUB:	轨迹速度曲线图**三次曲线：** 以程序段方式编程的 F 值－与程序段结束处有关－通过一个样条连接。 样条的开始和前一个进给值相切，结束和后一个进给值相切，并与 G93 和 G94 一起作用。 如在一个程序段中缺少 F 地址，在这里将使用最后编程的 F 值。
F=FPO… :	轨迹速度曲线图**通过多项式：** F 地址表示由多项式定义的、从当前值开始到程序段结尾的进给曲线。 此后终值将作为模态值有效。

在弯曲的轨迹中优化进给

进给多项式 F=FPO 和进给样条 FCUB 应当始终以恒定切削速度 CFC 执行完毕。 这样就可生成加速度恒定的设定进给曲线图。

示例： 不同的进给曲线图

在这个例子中列出了不同的进给曲线图的编程和图示。



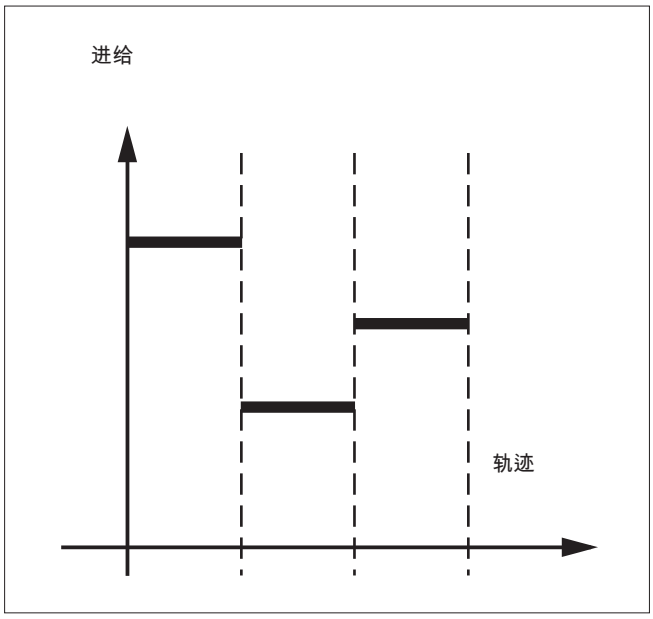
程序代码	注释
N1 F1000 FNORM G1 X8 G91 G64	; 恒定进给曲线图，增量数据
N2 F2000 X7	; 设定速度急动
N3 F=FPO(4000, 6000, -4000)	; 多项式进给曲线图，程序段结束处的进给为 4000
N4 X6	;多项式进给 4000 作为模态值。
N5 F3000 FLIN X5	;线性进给曲线图
N6 F2000 X8	;线性进给曲线图
N7 X5	;线性进给作为模态值
N8 F1000 FNORM X5	; 恒定的进给曲线图，带有加速度急动。
N9 F1400 FCUB X8	; 所有下列逐段编程的 F 值均使用样条联系在一起。
N10 F2200 X6	
N11 F3900 X7	
N12 F4600 X7	
N13 F4900 X5	;关闭样条曲线图。
N14 FNORM X5	
N15 X20	

其它信息

FNORM

进给地址 F 把轨迹进给表示为符合 DIN66025 的恒定值。

更多的信息，请参见编程手册“基础部分”。

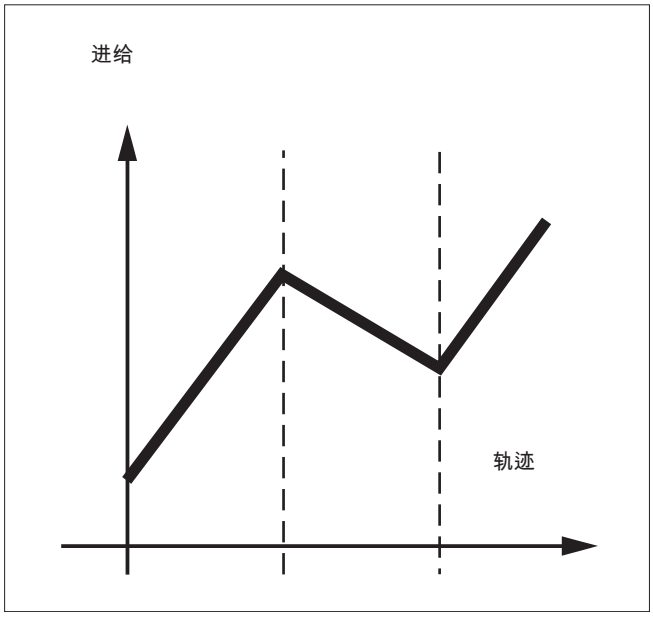


FLIN

进给进程通过实际的进给值到被编程的 F 值，线性运行到程序段末尾。

示例：

```
N30 F1400 FLIN X50
```

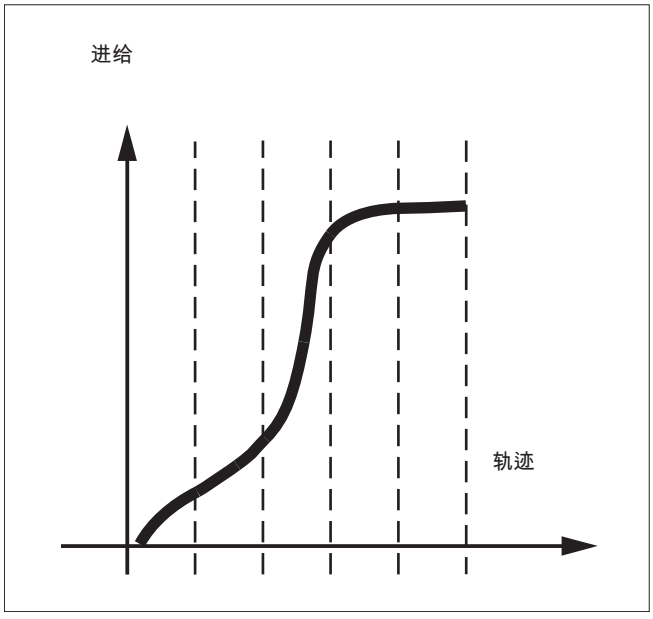


FCUB

从实际的进给值到编程的 F 值到程序段结尾，进给以空间的行程运行。 控制系统通过样条连接所有有效 FCUB 程序方式编程的进给值。 进给值作为计算样条插补的支点。

示例：

```
N50 F1400 FCUB X50
N60 F2000 X47
N70 F3800 X52
```



F=FPO(…,…,…)

进给进程通过一个多项式直接编程。 多项式系数 的参数类似于多项式插补。

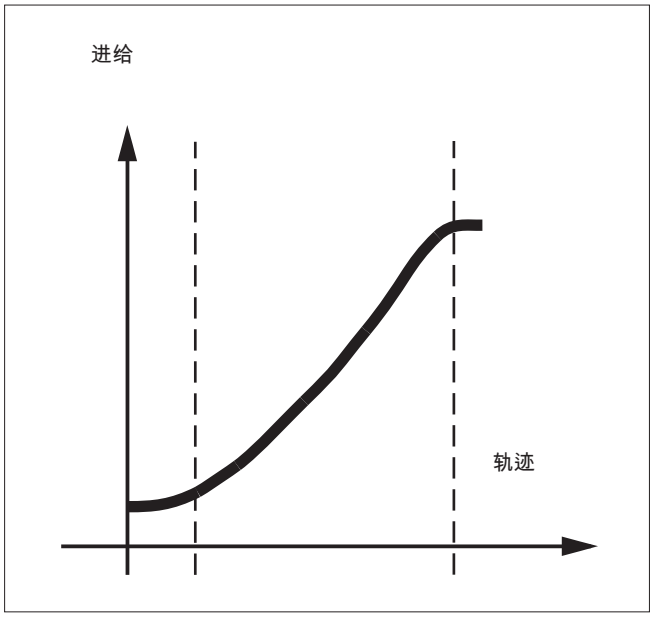
示例：

```
F=FPO(endfeed, quadf, cubf)
```

endfeed, quadf 和 cubf 为之前定义的变量。

endfeed:	程序段结束时的进给
quadf:	二次方的多项式系数
cubf:	立方的多项式系数

当 FCUB 激活时，程序段开始和结束处的样条与通过 FPO 设定的曲线相切。



边界条件

- 与编程的进给进程无关，轨迹运行方式的编程的功能有效。
- 可编程的进给曲线基本上绝对独立于 G90 或者 G91.
- 进给曲线 FLIN 和 FCUB 在 G93 和 G94 起作用，在 G95, G96/G961 和 G97/G971 不起作用。
- 当压缩器 COMPON 激活时，将多个程序段合并成一个样条线段时适用：

FNORM:	对于样条段，最后的程序段 F 字有效。
FLIN:	对于样条段，最后的程序段 F 字有效。 编程的 F 值在程序段的结束处有效，然后线性返回。
FCUB:	生成的进给样条最多按照机床数据 MD20172 $MC_COMPRESS_VELO_TOL 中所定义的值偏离以编程的终点。
F=FPO(…,…,…):	程序段不压缩。

12.3 加速性能

12.3.1 加速模式（BRISK，BRISKA，SOFT，SOFTA，DRIVE，DRIVEA）

关于加速模式的编程有下列零件程序指令可供使用：

- "BRISK, BRISKA"
  单轴或轨迹轴以最大加速度运行，直至达到编程的进给速度（**无急动限制的加速**）。
- "SOFT, SOFTA"
  单轴或轨迹轴以稳定的加速度运行，直至达到编程的进给速度（**有急动限制的加速**）。
- "DRIVE, DRIVEA"
  单轴或轨迹轴以最大加速度运行，直至达到所设置的速度极限（机床数据设置！）。 此后降低加速度（机床数据设置！），直至达到编程的进给速度。

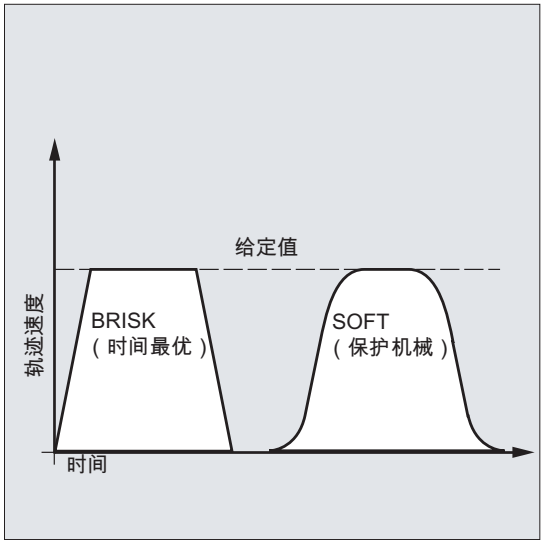


图 12-1 轨迹速度的走势，在 BRISK 和 SOFT 时

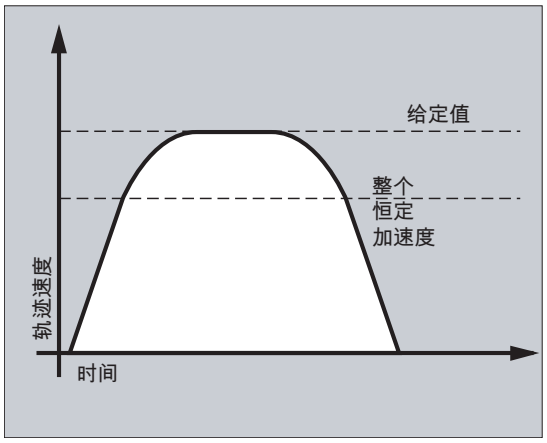


图 12-2　轨迹速度的走势，在 DRIVE 时

句法

```
BRISK
BRISKA(<轴 1>,<轴 2>,…)
SOFT
SOFTA(<轴 1>,<轴 2>,…)
DRIVE
DRIVEA(<轴 1>,<轴 2>,…)
```

含义

BRISK:	用于激活轨迹轴“无急动限制的加速”的指令。
BRISKA:	用于激活单轴运行（JOG，JOG/INC，定位轴，摆动轴，等）“无急动限制的加速”的指令。
SOFT:	用于激活轨迹轴“有急动限制的加速”的指令。
SOFTA:	用于激活单轴运行（JOG，JOG/INC，定位轴，摆动轴，等）“有急动限制的加速”的指令。
DRIVE:	超出速度上限（MD35220 $MA_ACCEL_REDUCTION_SPEED_POINT）时，激活轨迹轴降低加速度指令。
DRIVEA:	超出设置的速度上限（MD35220 $MA_ACCEL_REDUCTION_SPEED_POINT）时，激活单轴运行（JOG，JOG/INC，定位轴，摆动轴，等）降低加速度指令。
(<轴 1>,<轴 2>,…):	调用的加速模式适用的单轴。

边界条件

在加工时变换加速模式

如果加工时在一个零件程序中变换加速模式（BRISK ↔ SOFT），则在连续路径运行时也会在程序段结束的过渡处使用准停来更换程序段。

示例

示例 1： SOFT 和 BRISKA

程序代码
N10 G1 X… Y… F900 SOFT
N20 BRISKA(AX5,AX6)
...

示例 2： DRIVE 和 DRIVEA

程序代码
N05 DRIVE
N10 G1 X… Y… F1000
N20 DRIVEA (AX4, AX6)
...

文档

功能手册 基本功能 加速度 (B2)

12.3.2 跟随轴时的加速影响(VELOLIMA、ACCLIMA、JERKLIMA）

在轴耦合（切向跟踪、耦合运动、引导值耦合、电子齿轮；参见“轴耦合 (页 597)”）中，跟随轴/主轴的运行取决于一个或多个引导轴/引导主轴。

即使是在已激活的轴耦合中，也可通过零件程序或同步动作中的 VELOLIMA，ACCLIMA 和 JERKLIMA 指令对跟随轴/主轴的动态限值进行调节。

说明

JERLIMA 功能不能用于全部的耦合类型。

文档：

- 功能手册 特殊功能；轴耦合（M3）
- 功能手册 扩展功能；同步主轴（S3）。

说明

在 SINUMERIK 828D 上的可用性

VELOLIMA，ACCLIMA 和 JERKLIMA 在 SINUMERIK 828D 上只能和“耦合运动”功能一起使用！

句法

```
VELOLIMA(<轴>)=<值>
ACCLIMA(<轴>)=<值>
JERKLIMA(<轴>)=<值>
```

含义

VELOLIMA:	该指令用于修改参数设置的最大**速度**
ACCLIMA:	该指令用于修改参数设置的最大**加速度**
JERKLIMA:	该指令用于修改参数设置的最大**急动**
<轴>:	需要调节动态限值的跟随轴
<值>:	百分比补偿值

示例

示例 1： 修改跟随轴（AX4）的动态限值

程序代码	注释
...	
VELOLIMA[AX4]=75	；将限值修改为机床数据中存储的轴向最大速度的 75%。
ACCLIMA[AX4]=50	；将限值修改为机床数据中存储的轴向最大加速度的 50%。
JERKLIMA[AX4]=50	；将限值修改为机床数据中存储的轨迹运行最大轴向急动的 50%。
...	

示例 2：电子齿轮

轴 4 通过“电子齿轮”与 X 轴耦合。 将跟随轴的加速性能限制为最大加速度的 70 %。 将允许的最大速度限制为最大速度的 50 %。 耦合接通后，再将最大允许速度重新设为 100 %。

程序代码	注释
...	
N120 ACCLIMA[AX4]=70	；降低的最大加速度。
N130 VELOLIMA[AX4]=50	；降低的最大速度。

程序代码	注释
...	
N150 EGON(AX4,"FINE",X,1,2)	; 接通 EG 耦合。
...	
N200 VELOLIMA[AX4]=100	; 重设为最大速度。
...	

示例 3： 通过静态同步动作调节引导值耦合

轴 4 通过引导值耦合与 X 轴耦合。 通过静态同步动作 2，从位置 100 起将加速度特性限制为 80%。

程序代码	注释
...	
N120 IDS=2 WHENEVER $AA_IM[AX4] > 100 DO ACCLIMA[AX4]=80	; 同步动作
N130 LEADON(AX4, X, 2)	; 打开引导值啮合
...	

12.3.3 激活工艺专用动态值（DYNNORM，DYNPOS，DYNROUGH，DYNSEMIFIN，DYNFINISH）

借助于 G 功能组“工艺”可以为 5 个不同的工艺加工步骤激活匹配的动态性能。

动态值和 G 指令可设计，因此受机床数据设置的影响（→ 机床制造商!）。

文档：
功能手册 基本功能；连续路径运行，准停，预读（B1）

句法

激活动态值：
```
DYNNORM
DYNPOS
DYNROUGH
DYNSEMIFIN
DYNFINISH
```

说明

动态值已在程序段中生效，在该程序段中编程了相应的 G 指令。 无法停止加工。

读取或写入特定数组元素：：
```
R<m>=$MA...[n,X]
$MA...[n,X]=<值>
```

含义

DYNNORM:	用于激活**一般动态**的 G 指令	
DYNPOS:	用于激活**定位运行，攻丝**的动态的 G 指令	
DYNROUGH:	用于激活**粗加工**动态的 G 指令	
DYNSEMIFIN:	用于激活**精加工**动态的 G 指令	
DYNFINISH:	用于激活**精修整**动态的 G 指令	
R<m>:	编号为<m>的计算参数	
$MA...[n,X]:	带动态特定数组元素的机床数据	
<n>:	数组索引	
	取值范围:	0 ... 4
	0	一般动态（DYNNORM）
	1	定位运行动态（DYNPOS）
	2	粗加工动态（DYNROUGH）
	3	精加工动态（DYNSEMIFIN）
	4	精修整动态（DYNFINISH）
<X>:	轴地址	
<值>:	动态值	

示例

示例 1： 激活动态值

程序代码	注释
DYNNORM G1 X10	; 默认设置
DYNPOS G1 X10 Y20 Z30 F…	; 定位运行，攻丝
DYNROUGH G1 X10 Y20 Z30 F10000	; 粗加工
DYNSEMIFIN G1 X10 Y20 Z30 F2000	; 精加工
DYNFINISH G1 X10 Y20 Z30 F1000	; 精细加工

示例 2： 读写特定数组元素

粗加工最大加速度，X 轴

程序代码	注释
R1=$MA_MAX_AX_ACCEL[2, X]	;读
$MA_MAX_AX_ACCEL[2, X]=5	; 写

12.4 带预控制运行（FFWON, FFWOF）

通过前馈控制可以使得受速度影响的超程长度在轨迹运行时逐渐降低到零。 使用带前馈控制的加工可以提高轨迹精度，改善加工质量。

句法

```
FFWON
FFWOF
```

含义

FFWON:	用于**激活**前馈控制的指令
FFWOF：	用于**取消**前馈控制的指令

说明

通过机床数据可以确定前馈控制方式，并且确定哪些轨迹轴必须进行前馈控制运行。

标准： 由速度决定的前馈控制

选项： 由加速度决定的前馈控制

示例

程序代码

```
N10 FFWON
N20 G1 X… Y… F900 SOFT
```

12.5 可编程轮廓精度（CPRECON, CPRECOF）

“可编程轮廓精度”功能通过自动调整速度减小弯曲轮廓上的轨迹误差。

所遵循的轮廓精度取决于机床配置（MD20470 $MC_MC_CPREC_WITH_FFW；参见机床制造商的说明），可通过设定数据 $SC_CONTPREC 或编程轮廓公差设定。此数值和几何轴的 K_V 系数越小，轨道在弯曲轮廓上的进给就下降得越剧烈。

“可编程轮廓精度”功能通过 CPRECON 和 CPRECOF 在 NC 程序中激活和取消。

句法

```
CPRECON
...
CPRECOF
```

含义

CPRECON：	G 代码调用：**激活**“可编程轮廓精度”	
	生效方式：	模态
CPRECOF：	G 代码调用：**关闭**“可编程轮廓精度”	
	生效方式：	模态

CPRECON 和 CPRECOF 共同构成 G 功能组 39（可编程轮廓精度）。

说明

用户可通过设定数据 $SC_MINFEED（CPRECON 时的最小进给）为轨迹进给设定最小速度。

进给不会受到此数值的限制，除非已编程了更小的 F 数值或者轴的动态特性限制强制要求更低的轨迹速度。

说明

“可编程轮廓精度”功能仅涉及轨迹的几何轴。其不会对定位轴的速度产生影响。

示例

程序代码	注释
N10 G0 X0 Y0	
N20 CPRECON	；激活“可编程轮廓精度”。
N30 G1 G64 X100 F10000	；在连续路径运行中以 10 m/min 的速度加工。
N40 G3 Y20 J10	；在圆弧程序段中自动的进给限制。
N50 G1 X0	；无限制进给率（10 m/min）。
...	
N100 CPRECOF	；取消“可编程轮廓精度”。
N110 G0 ...	

文档

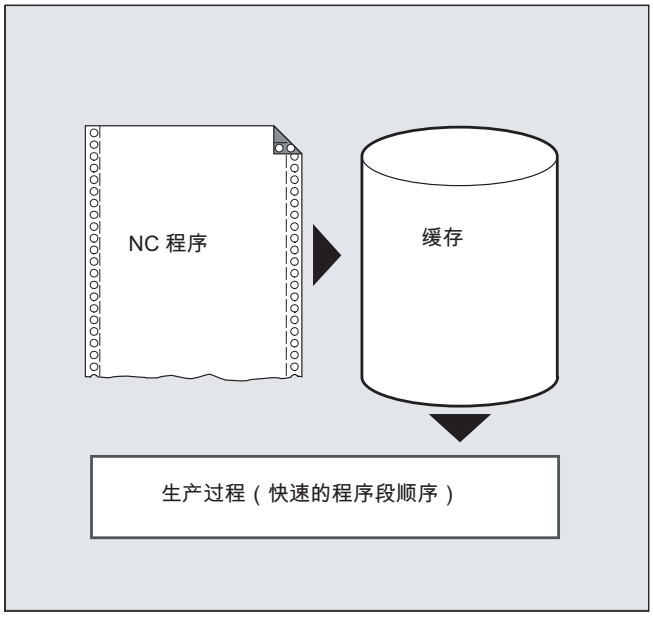
CTOL 的编程请见“轮廓公差/定向公差的编程 (CTOL、OTOL、ATOL) (页 591)”

“可编程轮廓精度”功能的更多详细信息请见：

功能手册 特殊功能；轮廓道监控（K6），章节：“可编程轮廓精度”

12.6 带有缓存的程序运行过程 (STOPFIFO, STARTFIFO, FIFOCTRL, STOPRE)

根据扩展级，控制系统具有一定数量的缓存存储器，它们会在加工前存储预处理的程序段，并在加工过程中作为快速程序段输出。 借此可以以较高速度运行较短的行程。 只要控制系统的剩余时间允许，原则上缓存都会被载满。



标记缓存段

零件程序中需要在缓存中存储的程序段会标有"STOPFIFO"（开始）以及"STARTFIFO"（结束）。 当缓存已满或者执行指令"STARTFIFO"之后，才会开始执行经过预处理并处于缓存中的程序段。

缓存的自动控制

缓存的自动控制通过指令"FIFOCTRL"调用。 "FIFOCTRL"首先像"STOPFIFO"一样起作用。在每次编程时都会等待缓存被载满，然后才开始执行程序。 只是缓存在空运行时的属性有所不同： 编程"FIFOCTRL"后，当缓存容量达到 2/3 时，轨迹速度会不断降低，从而避免出现完全空运行和突然停止过程。

预处理停止

如果在程序段中编程了"STOPRE"指令，程序段预处理和缓存过程将被终止。 只有当全部执行了所有预处理并缓存的程序段后，才开始执行后面的程序段。 之前的程序段会以准停方式停止（如 G9）。

注意
程序中断 当刀具补偿已启用且进行样条插补时，不应编程"STOPRE"，否则会中断相关的程序段顺序。

句法

表格 12-1　标记缓存段：

```
STOPFIFO
...
STARTFIFO
```

表格 12-2　缓存的自动控制：

```
...
FIFOCTRL
...
```

表格 12-3　预处理停止：

```
...
STOPRE
...
```

说明

指令"STOPFIFO", "STARTFIFO", "FIFOCTRL"和"STOPRE"必须在各自的程序段编程。

含义

STOPFIFO:	"STOPFIFO"标出了需要存储在缓存中的程序段的开始。 "STOPFIFO"指令会停止处理并载满缓存，直至： ● 识别"STARTFIFO"或"STOPRE" 或者 ● 缓存已满 或者 ● 达到程序末尾。
STARTFIFO:	"STARTFIFO"会启动程序段的快速处理，同时会加载缓存
FIFOCTRL:	启用缓存的自动控制
STOPRE:	停止预处理

说明

当缓存程序段中包含的指令需要强制进行非缓存运行，如回参考点、测量功能等，系统就不会执行或者中断缓存加载。

说明

当访问机床的状态数据时（$SA...），控制系统生成会内部预处理停止。

示例： 停止预处理

程序代码	注释
...	
N30 MEAW=1 G1 F1000 X100 Y100 Z50	；带有第一个测量输入端的测量探头和直线插补的测量程序段。
N40 STOPRE	；预处理停止。
...	

12.7 定义停止延迟区 (DELAYFSTON, DELAYFSTOF)

停止延迟区是在零件程序中通过预定义程序 DELAYFSTON 和 DELAYFSTOF 定义的。

说明

DELAYFSTON 和 DELAYFSTOF 不可用于同步动作！

句法

```
DELAYFSTON
...
DELAYFSTOF
```

含义

DELAYFSTON:	定义一个停止延迟区的开始	
	在单独程序段中编程:	选择
DELAYFSTOF:	定义一个停止延迟区的结尾	
	在单独程序段中编程:	选择

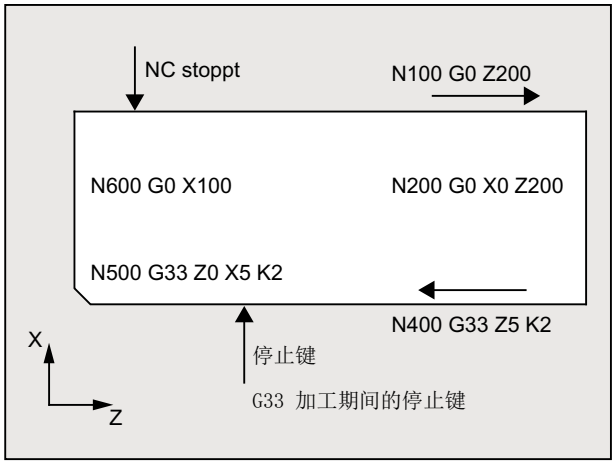
编程示例

在一个循环中重复下列程序块：

程序代码

```
...
N99 MY_LOOP:
N100 G0 Z200
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
...
```

下图可见用户在停止延迟区中按下“停止”，NC 开始执行停止延迟区范围之外的制动过程，即在程序段 N100 中。这样 NC 就在 N100 的前端区域停住。



更多信息

子程序结束

当子程序结束时，DELAYFSTON 已经在其中被调用，DELAYFSTOF 就被隐式激活。

嵌套

如子程序 1 在停止延迟区中调用子程序 2，那么子程序 2 就是完整的停止延迟区。特别的是 DELAYFSTOF 在子程序 2 中无效。

示例：

程序代码	注释
N10010 **DELAYFSTON**	; 带 N10xxx 的程序级面 1 的程序段。
N10020 R1 = R1 + 1	
N10030 G4 F1	; 开始停止延迟区。
...	
N10040 子程序 2	
...	
...	; 子程序 2 的说明。
N20010 **DELAYFSTON**	; 无效，程序级 2 重复的开始。
...	
N20020 **DELAYFSTOF**	; 另一个程序级的结束，无效。
N20030 RET	
N10050 **DELAYFSTOF**	; 相同程序级中停止延时区的结束。
...	
N10060 R2 = R2 + 2	
N10070 G4 F1	; 结束停止延迟区。停止从现在起立即有效。

系统变量

可通过以下系统数据查询零件程序执行当前是否位于停止延迟区中：

- 在带 $P_DELAYFST 的零件程序中
- 在带 $AC_DELAYFST 的同步动作中

值	含义
0	停止延迟区未生效
1	停止延迟区生效

12.8 阻止 SERUPRO 的程序位置 (IPTRLOCK, IPTRUNLOCK)

对于某些机械复杂的机床配置，要求阻止程序段查找 SERUPRO。

使用一个可编程的中断指示，可以使“查找中断点”时在不可查找的位置之前停止。

也可以在零件程序范围中定义不可查找的区域，在其中 NC 不可以再次进入。使用程序中断，NC 记下最后加工的程序段，通过操作界面 HMI 可以查找到该程序段。

句法

```
IPTRLOCK
IPTRUNLOCK
```

这些示例单独处于某个零件程序行中并且可实现一个可编程的中断向量。

含义

IPTRLOCK:	开始不可查找的程序段
IPTRUNLOCK:	结束不可查找的程序段

两个指令仅允许在零件程序中，但**不可** 在同步动作中。

示例

在两个带有隐式"IPTRUNLOCK"的程序层中嵌套不可查找的程序段。子程序 1 中的隐式"IPTRUNLOCK"结束不可查找的程序段。

程序代码	注释
N10010 IPTRLOCK()	
N10020 R1 = R1 + 1	
N10030 G4 F1	；停止程序段，开始不可查找的程序段。
...	
N10040 子程序 2	
...	；子程序 2 的说明。
N20010 IPTRLOCK ()	；无效，重复的开始。
...	
N20020 IPTRUNLOCK ()	；另一个程序级的结束，无效。
N20030 RET	
...	
N10060 R2 = R2 + 2	
N10070 RET	；结束不可查找的程序段。
N100 G4 F2	；继续主程序。

中断到 100，重新提供了中断指示。

其它信息

采集和查找不可查找的区域

不可查找的程序段使用语言指令"IPTRLOCK"和"IPTRUNLOCK"进行标识。

指令"IPTRLOCK"将中断向量冻结成一个在主过程中可以执行的单程序段 (SBL1) 。 该程序段在以下所述中被作为停止程序段。 如果在"IPTRLOCK"之后出现一个程序中断，就可以在操作界面 HMI 上查找该停止程序段。

再次停止在当前的程序段

使用后续程序段的"IPTRUNLOCK"将中断向量设置给中断点的当前程序段。

在找到一个查找目标后，可以用该停止程序段重复一个新的查询目标。

被用户编辑过的中断向量必须通过 HMI 重新删除。

嵌套时调节

下列各项用来调节语言指令"IPTRLOCK"和"IPTRUNLOCK"与嵌套和子程序结束之间的相互作用：

1. 当子程序结束时， "IPTRLOCK"已经在其中被调用， "IPTRUNLOCK"就被隐式激活。
2. "IPTRLOCK"在一个不可查找的程序段中保持无效。
3. 如果子程序 1 在一个不可查找的区域调用子程序 2，则子程序 2 保持不可查找。 特别是"IPTRUNLOCK"在子程序 2 中不起作用。

其它信息，参见
/FB1/ 功能手册基本功能；BAG、通道、程序运行（K1）。

系统变量

可使用"$P_IPTRLOCK"在零件程序中识别一个不可查找的程序段。

自动的中断指示

自动的中断指示的功能自动将一个先前确定的耦合方式确定为无法搜索。 借助机床数据

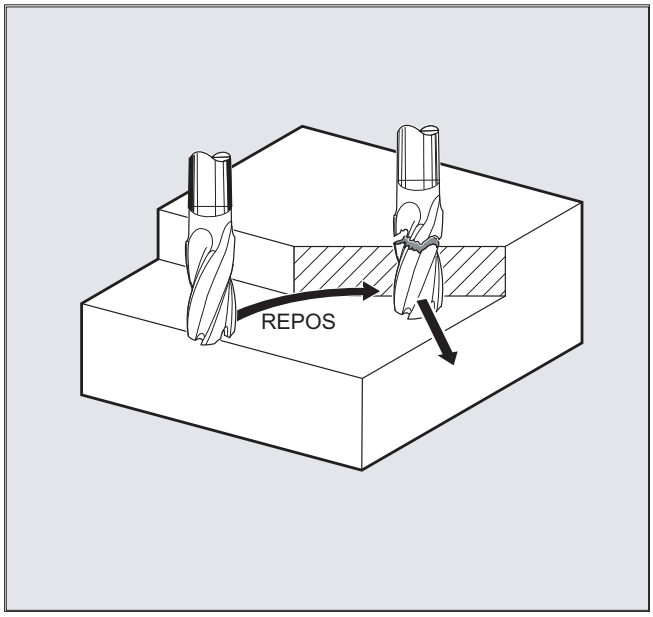
- "EGON"的电子驱动装置
- "LEADON"的轴向引导值耦合

激活自动中断指针。 如果已编程的中断向量和可通过机床数据激活的自动中断向量相互交迭，就会形成最有可能的不可查找程序段。

12.9 返回轮廓（REPOSA, REPOSL, REPOSQ, REPOSQA, REPOSH, REPOSHA, DISR, DISPR, RMIBL, RMBBL, RMEBL, RMNBL）

若在加工过程中中断了当前运行的程序并执行了退刀（例如由于刀具断裂或需要测量工件），可使刀具在程序的控制下返回到轮廓上某个选中的点。

REPOS 指令的作用如同一个子程序返回指令（例如 M17）。后续程序段将不再执行。程序运行的中断也请见“中断程序（ASUP） (页 136)”。



句法

```
REPOSA RMIBL DISPR=…
REPOSA RMBBL
REPOSA RMEBL
REPOSA RMNBL
REPOSL RMIBL DISPR=…
REPOSL RMBBL
REPOSL RMEBL
REPOSL RMNBL
REPOSQ RMIBL DISPR=… DISR=…
REPOSQ RMBBL DISR=…
REPOSQ RMEBL DISR=…
REPOSQA DISR=…
REPOSH RMIBL DISPR=… DISR=…
REPOSH RMBBL DISR=…
REPOSH RMEBL DISR=…
REPOSHA DISR=…
```

含义

选择逼近行程

REPOSA:	通过几何轴沿一条直线再次逼近轮廓。 所有其他轴也全部重新定位。
REPOSL:	通过几何轴沿一条直线再次逼近轮廓。 所有其他轴必须进行精确编程。
REPOSQ DISR=… :	通过几何轴沿半径为 DISR 的四分之一圆弧再次逼近轮廓。 所有其他轴必须进行精确编程。
REPOSQA DISR=… :	通过几何轴沿半径为 DISR 的四分之一圆弧再次逼近轮廓。 所有其他轴也全部重新定位。
REPOSH DISR=… :	通过几何轴沿直径为 DISR 的半圆弧再次逼近轮廓。 所有其他轴必须进行精确编程。
REPOSHA DISR=… :	通过几何轴沿半径为 DISR 的半圆弧再次逼近轮廓。 所有其他轴也全部重新定位。

选择重定位点

RMIBL:	逼近中断点
RMIBL DISPR=…:	逼近和中断点相距 DISPR（毫米/英寸）的某个点
RMBBL:	逼近程序段起点
RMEBL:	逼近程序段终点
RMEBL DISPR=… :	逼近和程序段终点相距 DISPR 的某个点。
RMNBL:	逼近下一个轨迹点
A0 B0 C0 :	需要执行逼近的轴

说明

兼容性

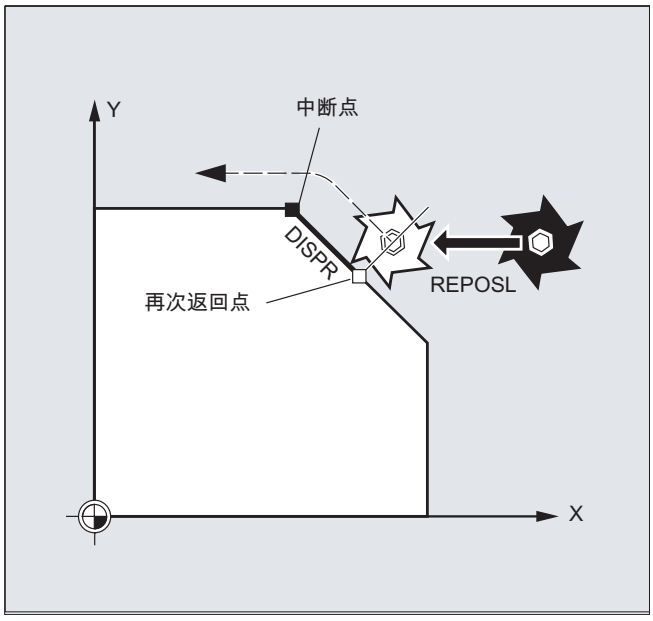
为保持与旧版本的兼容性，您仍可通过模态指令 RMI、RMB、RME 和 RMN 编程 REPOS 逼近模式。在 ASUP 内应用时应该在 PROC 说明中提供属性 SAVE。否则，当在 ASUP 使用的模态 REPOS 逼近模式与预设置 RMI 有偏差时有效，同样在以下 REPOS 过程。

沿一条直线再次逼近轮廓，REPOSA，REPOSL

刀具沿一条直线直接运行到重定位点。

示例

```
REPOSL RMIBL DISPR=6 F400
```

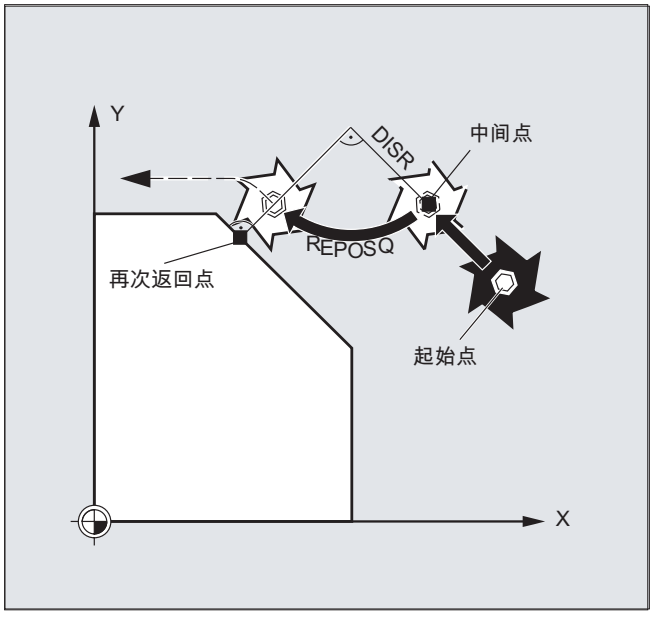


沿四分之一圆弧再次逼近轮廓，REPOSQ，REPOSQA

刀具沿四分之一圆弧运行至重定位点，半径通过 DISR=...设定。控制系统自动计算起点和重定位点之间的必要中间点。

示例

```
REPOSQ RMIBL DISR=10 F400
```

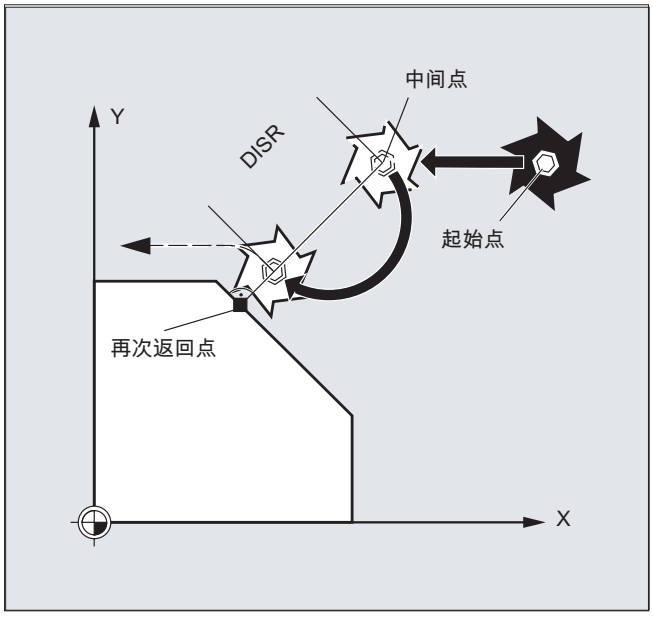


沿半圆弧再次逼近轮廓，REPOSH，REPOSHA

刀具沿半圆运行至重定位点，直径通过 DISR=...设定。控制系统自动计算起点和重定位点之间的必要中间点。

示例

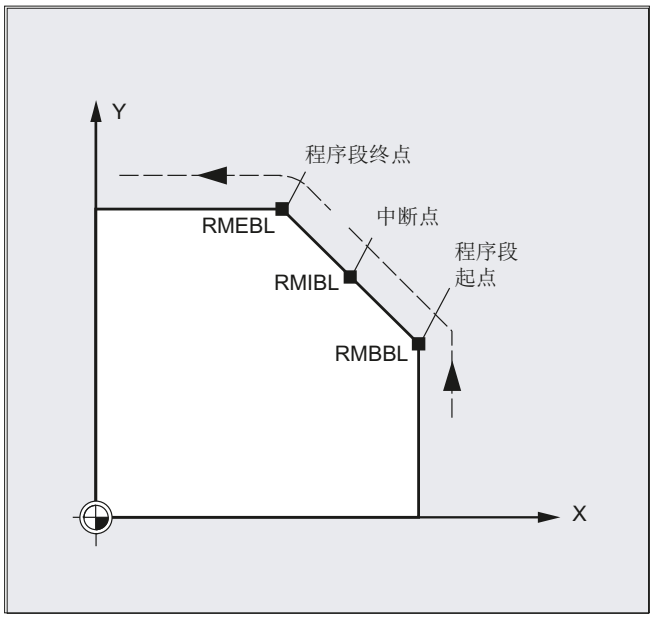
```
REPOSH RMIBL DISR=20 F400
```



确定重定位点（不适用于采用 RMNBL 的 SERUPRO 逼近）

针对中断处理的程序段，您可以在三个重定位点之间进行选择：

- RMIBL，中断点
- RMBBL，程序段起点或者上一个终点
- RMEBL，程序段终点



RMIBL DISPR=... 或 RMEBL DISPR=... 可确定重定位点是位于中断点前还是位于程序段终点前。

DISPR=... 可以毫米或英寸为单位确定重定位点和中断点或终点之间的间距。 该间距最大可设为程序段起点和这两个点之间的间距。

若未编程 DISPR=...，则 DISPR 自动为 0，中断点（编写 RMIBL 时）或程序段终点（编写 RMEBL 时）成为重定位点。

DISPR 的符号

DISPR 的符号一同被计算。 其符号为正号时，特性和现有版本一样。

符号为负号时，重定位点在中断点后，编写 RMBBL 时重定位点在起点后。

中断点和重定位点之间的间距为 DISPR 的绝对值。 该间距可最大为程序段终点和这两个点之间的间距。

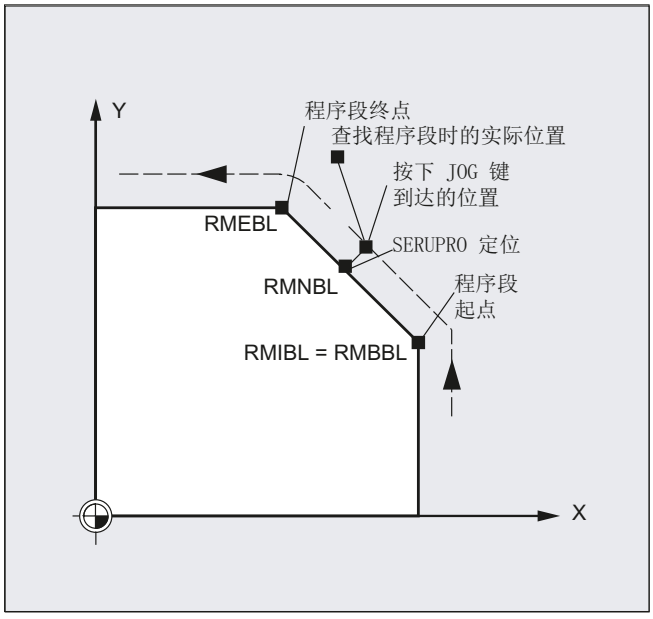
应用示例：

传感器检测到轴靠近虎钳。 此时系统会触发一个异步子程序，绕过虎钳。

然后使轴再定位到一个虎钳后方相距 DISPR 的点上，继续执行程序。

SERUPRO 和 RMNBL 的综合使用

如果是在任意位置上强制中断加工，可综合使用 SERUPRO 和 RMNBL，使轴以最短行程返回断点，接着走完余程。 此时用户需启动 SERUPRO，找到中断程序段，然后用 JOG 键将轴定位到断点前。



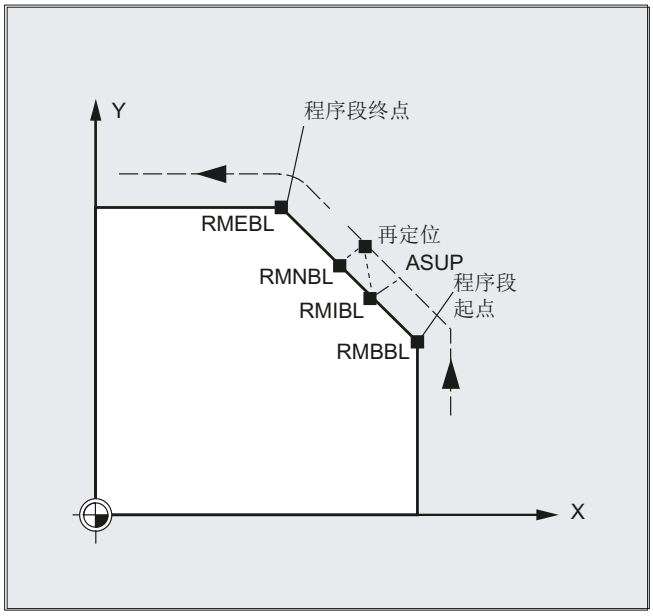
说明

SERUPRO

RMIBL 和 RMBBL 对于 SERUPRO 而言作用相同。 RMNBL 并非限定用于 SERUPRO，可普遍使用。

逼近轨迹上的下一个点 RMNBL

编写了 REPOSA RMNBL 时，系统不会再次从头执行中断程序段，而是只执行余程。 此时轴会逼近中断程序段的下一个轨迹点。



有效 REPOS 模式的状态

中断程序段的有效 REPOS 模式可通过同步动作和变量 $AC_REPOS_PATH_MODE 读取：

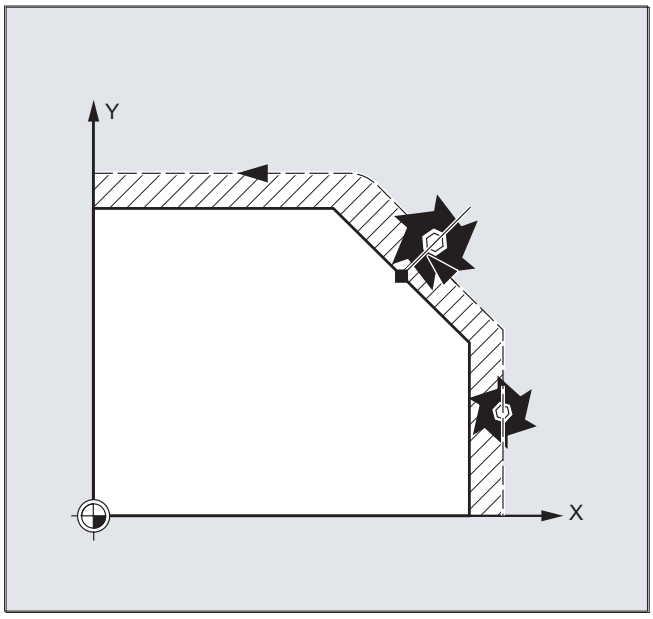
0 逼近未定义
1 RMBBL: 逼近到开始处
2 RMIBL: 逼近到中断点
3 RMEBL: 逼近到程序段终点
4 RMNBL: 向已中断程序段的下一个轨迹点运动

使用新刀具逼近

如果程序由于刀具损坏而中断：

通过编程新的 D 号，该程序自重定位点起以修改后的刀具补偿值继续进行。

刀具补偿值修改后可能无法再逼近中断点。 在这种情况下轴会逼近新轮廓上该中断点的下一个点（可能相距 DISPR ）。



逼近轮廓

可对刀具重新逼近轮廓的运动进行编程。 用值零设定待运行轴的地址。

REPOSA、REPOSQA 和 REPOSHA 指令会自动对所有轴进行重新定位。 此时不需要指定轴。

当编程 REPOSL、REPOSQ 和 REPOSH 指令时，所有几何轴均会自动逼近轮廓，即便在指令中未进行设定。 所有其它轴必须在指令中指定。

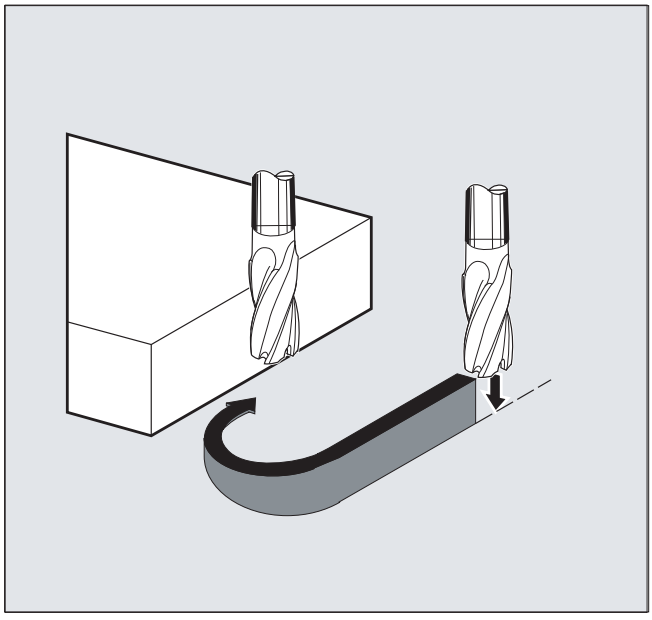
针对 REPOSH 和 REPOSQ 圆弧运动：

刀具在指定的工作平面 G17 至 G19 中沿圆弧运动。

若在逼近程序段中指定了第三个几何轴（进给方向），而进给方向的刀具位置和编程位置不一致，则刀具会以螺线逼近重定位点。

在以下情况下刀具会自动转换为线性逼近 REPOSL：

- 没有设定 DISR 的值。
- 没有定义逼近方向（程序在一条无运行信息的程序段中中断）。
- 逼近方向垂直于当前工作平面。



12.10 对运动控制的影响

12.10.1 百分比式急冲修正 (JERKLIM)

使用 NC 指令"JERKLIM"，可在重要程序段落中降低或升高原先由机床数据设置的、路径运行允许的最大轴急动。

前提条件

加速模式 SOFT 必须已激活。

生效方式

此功能在以下情况下生效：

- 在 AUTO 运行方式中。
- 仅对于路径轴生效。

句法

```
JERKLIM[<轴>]=<值>
```

含义

JERKLIM:	急动补偿指令	
<轴>:	需要调整急动限值的机床轴。	
<值>:	百分比的补偿值，以设置的路径运行中最大轴急动为基准（MD32431 $MA_MAX_AX_JERK）。	
	取值范围：	1 ... 200
	取值为 100 时对急动没有影响。	

说明

零件程序结束和通道复位时 JERKLIM 的特性通过机床数据 MD32320 $MA_DYN_LIMIT_RESET_MASK 的位 0 配置。

- 位 0 = 0：
  编程的 JERKLIM 的值在通道复位/M30 时复位为 100 %。
- 位 0 = 1：
  编程的 JERKLIM 的值在通道复位/M30 后保持不变。

示例

程序代码	注释
...	
N60 JERKLIM[X]=75	；轴溜板在 X 方向以最大 75% 的轴急动进行加速/减速。
...	

12.10.2 百分比式速度修正 (VELOLIM)

使用指令 VELOLIM 可在零件程序或同步动作中降低通过机床数据设置的最大轴速度,以及取决于齿轮级的最大主轴转速。

生效方式

此功能在以下情况下生效：

- 在 AUTO 运行方式中。
- 对路径轴和定位轴生效。
- 对主轴模式/进给轴模式中的主轴生效

句法

VELOLIM[<进给轴/主轴>]=<值>

含义

VELOLIM:	速度补偿指令
<进给轴/主轴>：	需要调整速度或转速限值的轴或主轴。 **VELOLIM 用于主轴** 通过机床数据（MD30455 $MA_MISC_FUNCTION_MASK，位 6）可为零件程序中的编程设置，"VELOLIM" 针对进给轴/主轴作用相同（位 6 = 1），还是作用不同而需要分别编程（位 6 = 0）。 如果设置了分别编程，则在编程时通过标识符进行选择轴： ● 主轴标识符 S<n> 用于主轴运模式 ● 进给轴标识符，例如“C”，用于进给轴模式

<值>：	百分比补偿值 补偿值的基准： • 进给轴模式中的进给轴/主轴（MD30455, 位 6 == 0）： 以配置的最大轴速度为基准 （MD32000 $MA_MAX_AX_VELO）。 • 主轴模式或进给轴模式中的主轴（MD30455 位 6 = 1）： 以生效的齿轮级的最大转速为基准 （MD35130 $MA_GEAR_STEP_MAX_VELO_LIMIT[<n>]）	
	取值范围：	1 ... 100
	取值为 100 时对速度或转速没有影响。	

说明

零件程序结束和通道复位时的特性

零件程序结束和通道复位时"VELOLIM"行为通过机床数据可调节： MD32320 $MA_DYN_LIMIT_RESET_MASK, 位 0

在主轴模式中识别生效的转速限制

在主轴模式中可使用以下系统变量识别通过"VELOLIM"进行的转速限制（小于 100 %）：

- $AC_SMAXVELO（最大主轴转速）
- $AC_SMAXVELO_INFO（指示了限制转速的原因）

示例

示例 1： 机床轴的速度限制

程序代码	注释
...	
N70 VELOLIM[X] = 80	; 轴溜板在 X 方向应以 80% 轴允许的最大速度运行。
...	

示例 2： 主轴的转速限制

程序代码	注释
N05 VELOLIM[S1]=90	; 将主轴 1 的最大转速限制为 1000 rpm 的 90 %。
...	
N50 VELOLIM[C]=45	; 将转速限制为 1000 rpm 的 45 %，C 为 S1 的进给轴标识符。
...	

主轴 1 的机床数据设置（AX5）

- 齿轮级 1 的最大转速 = 1000 rpm：
  MD35130 $MA_GEAR_STEP_MAX_VELO_LIMIT[1, AX5] = 1000
- "VELOLIM"的编程对主轴模式和进给轴模式共同生效，与编程的标识符无关：
  MD30455 $MA_MISC_FUNCTION_MASK[AX5], 位 6 = 1

12.10.3 JERKLIM 和 VELOLIM 的程序举例

下面的程序说明了百分比情况下突变和速度极限值的应用示例：

程序代码	注释
N1000 G0 X0 Y0 F10000 SOFT G64	
N1100 G1 X20 RNDM=5 ACC[X]=20	
ACC[Y]=30	
N1200 G1 Y20 VELOLIM[X]=5	；轴溜板在 X 方向应以最大 5% 轴的允许速度进行运行。
JERKLIM[Y]=200	；轴溜板在 X 方向应以最大 200% 轴的允许突变值进行加速/延迟。
N1300 G1 X0 JERKLIM[X]=2	；轴溜板在 X 方向应以最大 2% 轴的允许突变值进行加速/延迟。
N1400 G1 Y0	
M30	

12.11 轮廓公差/定向公差的编程 (CTOL、OTOL、ATOL)

使用地址 CTOL、OTOL 和 ATOL，可以调整通过机床数据和设定数据设置的加工公差，以便用于零件程序中的压缩器功能、精磨和定向平滑。

这些编程的公差值会持续生效，直至被新的编程值取代，或由于分配了一个负值而被删除。此外，在编程结束或复位时也会被删除。删除后，编程设置的公差值会再次生效。

句法

```
CTOL=<Value>
OTOL=<Value>
ATOL[<Axis>]=<Value>
```

含义

CTOL:	用于编程**轮廓公差**的地址			
	应用范围：	• 所有的压缩器功能 • 所有的精磨方式，除了 G641 和 G644		
	预处理停止：	不选择		
	生效方式：	模态		
	<Value>:	轮廓公差值是长度数据。		
		类型：	REAL	
		单位：	英寸/毫米（根据当前单位系统的设置）	
		值域：	≥ 0:	公差值
			< 0:	清除编写的公差值 ⇒ 机床数据或设定数据中设置的公差值重新生效。

OTOL:	用于编程**定向公差**的地址			
	应用范围：	• 所有的压缩器功能 • 定向平滑 ORISON • 所有的精磨方式，除了 G641, G644 和 OSD		
	预处理停止：	不选择		
	生效方式：	模态		
	<Value>:	定向公差值是角度数据。		
		类型：	REAL	
		单位：	度	
		值域：	≥ 0:	公差值
			< 0:	清除编写的公差值 ⇒ 机床数据或设定数据中设置的公差值重新生效。
ATOL:	用于编程**轴专用公差**的地址			
	应用范围：	• 所有的压缩器功能 • 定向平滑 ORISON • 所有的精磨方式，除了 G641, G644 和 OSD		
	预处理停止：	不选择		
	生效方式：	模态		
	<Axis>:	编写的公差所生效于的通道轴的名称		
	<Value>:	取决于轴的类型（线性轴或回转轴），轴公差的值为长度数据或角度数据。		
		类型：	REAL	
		单位：	用于线性轴：	英寸/毫米（根据当前单位系统的设置）
			用于回转轴：	度
		值域：	≥ 0:	公差值
			< 0:	清除编写的公差值 ⇒ 机床数据或设定数据中设置的公差值重新生效。

说明

通过 CTOL 和 OTOL 编写的轴专用公差值较通过 ATOL 编写的轴专用公差值具有更高优先级。

说明

缩放框架

缩放框架对编程公差的影响和对轴位置的影响一样，即：相对公差保持不变。

示例

程序代码	注释
COMPCAD G645 G1 F10000	；激活压缩器功能 COMPCAD。
X...Y...Z...	；此处机床数据和设定数据生效。
X...Y...Z...	
X...Y...Z...	
CTOL=0.02	；从此处开始，0.02 毫米的轮廓公差生效。
X...Y...Z...	
X...Y...Z...	
X...Y...Z...	
ASCALE X0.25 Y0.25 Z0.25	；从此处开始，0.005 毫米的轮廓公差生效。
X...Y...Z...	
X...Y...Z...	
X...Y...Z...	
CTOL=-1	；从此处开始，机床数据和设定数据再次生效。
X...Y...Z...	
X...Y...Z...	
X...Y...Z...	

系统变量

带预处理停止的读取

可通过以下系统变量在零件程序和同步动作中读取当前生效的公差：

- $AC_CTOL
  处理当前主运行程序段时生效的通道专用轮廓公差。
  若无轮廓公差生效，$AC_CTOL 会返回将各几何轴公差的平方相加后求得的平方根值。
- $AC_OTOL
  处理当前主运行程序段时生效的通道专用定向公差。
  若无定向公差生效，定向转换生效期间 $AC_OTOL 会返回由各定向轴公差的平方相加后求得的平方根值，否则返回“-1”值。
- $AA_ATOL[<轴>]
  处理当前主运行程序段时生效的轴专用轮廓公差。
  如果轮廓公差生效，$AA_ATOL[<几何轴>] 会返回由该轮廓公差除以几何轴数量的平方根所得到的值。
  如果定向公差和定向转换生效，$AA_ATOL[<定向轴>] 会返回由定向公差除以定向轴数量的平方根所得到的值。

说明

若未编写公差值，那么 $A 变量将无法区分各功能的公差。

当机床数据和设定数据中确定了不同的公差值时，即压缩器功能、精磨和定向平滑的公差，会出现上述情况。此时系统变量会返回一个出现在当前生效功能中的最大值。例如，如果压缩器功能的定向公差为 0.1，而定向平滑 ORISON 的定向公差为 1°，那么 $AC_OTOL 会返回值“1”。如果关闭了定向平滑功能，$AC_OTOL 将返回值“0.1”。

无预处理停止的读取

可通过以下系统变量在零件程序中读取当前生效的公差：

- $P_CTOL
  当前生效的通道专用轮廓公差。
- $P_OTOL
  当前生效的通道专用定向公差。
- $PA_ATOL
  当前生效的轴专用轮廓公差。

前提条件

通过 CTOL、OTOL 和 ATOL 编写的公差同样对间接关联的功能生效：

- 设定值计算中的切线误差限制
- 任意形状表面模式：基本功能

CTOL、OTOL 和 ATOL 的编程不影响以下平滑功能：

- OSD 定向精磨
  OSD 不使用公差，而是使用到程序段过渡处的间距。
- G644 精磨
  G644 不用于加工，而是用于优化换刀和其他无加工运动。
- G645 精磨
  G645 的特性和 G642 几乎一样，也使用编程公差。只有在曲率变化的相切程序段过渡中（比如：圆弧到直线的相切过渡），才使用机床数据 MD33120 $MA_PATH_TRANS_POS_TOL 的公差值。因为在这种条件下平滑距离也可以位于编程轮廓的外侧，而大多数应用不太允许。另外通常一个很小的固定设置的公差足以平衡曲率变化，编程员无需再加以考虑。

12.12 耦合生效时的程序段切换特性（CPBC）

CPBC 指令用于设定须遵循的程序段切换标准，从而在耦合生效的情况下在零件程序中执行程序段切换。

句法

```
CPBC[<跟随轴>] = <标准>
```

含义

CPBC：	耦合生效时的程序段切换标准	
<跟随轴>:	跟随轴的轴名称	
<标准>：	程序段切换标准	
	类型：	STRING
	值	**含义：** 执行程序段切换
	"NOC"	与耦合状态无关
	"IPOSTOP"	在设定值侧同步运行时进行
	"COARSE"	在实际值侧“粗”同步运行时进行
	"FINE"	在实际值侧“精”同步运行时进行

示例

程序代码

```
; 在以下情况下进行程序段切换：
;  - 与跟随轴 X2 的耦合 == 生效
;  - 设定值侧同步 == 生效
CPBC[X2]="IPOSTOP"
```

轴耦合 13

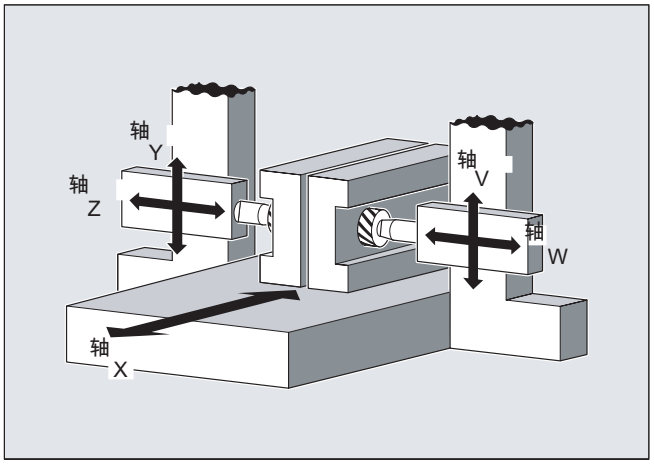
13.1 联动 (TRAILON, TRAILOF)

当一个已定义的引导轴运动时，指定给该轴的耦合轴（＝跟随轴）会在参照某个耦合系数的情况下，开始运行引导轴所引导的位移。

引导轴和跟随轴共同组成耦合组合。

应用范围

- 通过一个模拟轴进行轴运行。 引导轴是一个模拟轴，而耦合轴是一个真正的轴。 从而可以使得真实轴可以参照耦合系数运行。
- 用 2 个耦合组合进行两面加工：
  第 1 引导轴 Y，耦合轴 V
  第 2 引导轴 Z，耦合轴 W



句法

```
TRAILON(<跟随轴>,<引导轴>,<耦合系数>)
TRAILOF(<跟随轴>,<引导轴>,<引导轴 2>)
TRAILOF(<跟随轴>)
```

含义

TRAILON:	用于启用和定义耦合轴组合的指令	
	生效方式：	模态

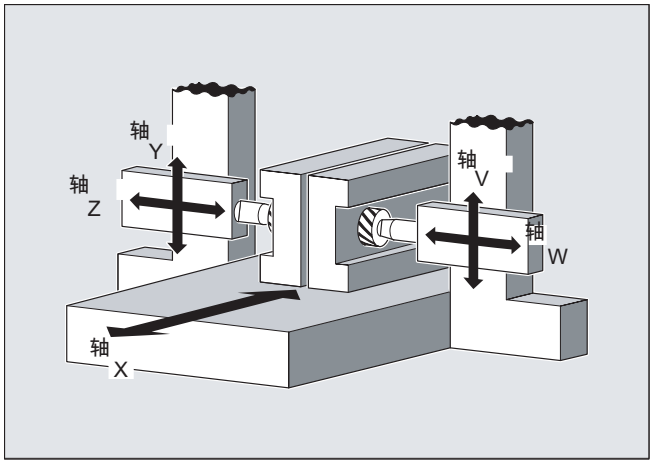
<跟随轴>：	参数 1： 耦合轴的名称 **提示：** 一个耦合轴也可以是其余耦合轴的引导轴。 以这种方式可以建立不同的耦合组合。	
<引导轴>：	参数 2： 引导轴的名称	
<耦合系数>:	参数 3： 耦合系数 耦合系数说明了耦合轴和引导轴位移之间的关系。 **<**耦合系数**>**= 耦合轴位移/ 引导轴位移	
	类型：	REAL
	缺省设置：	1
	负值表明引导轴和耦合轴在相反方向运行。 如在编程中未指定耦合系数，则耦合系数 1 自动生效。	
TRAILOF:	关闭耦合组合的指令	
	生效方式：	模态
	有 2 个参数的 TRAILOF 只会关闭指定引导轴的耦合： TRAILOF(<跟随轴>,<引导轴>) 如果一个耦合轴拥有 2 个引导轴，可调用带有 3 个参数的 TRAILOF 来关闭这两个耦合： TRAILOF(<跟随轴>,<引导轴>,<引导轴 2>) 如果编程了 TRAILOF，而没有指定引导轴，也会给出相同的结果： TRAILOF(<跟随轴>)	

说明

耦合运动始终在基准坐标系（BCS)中进行。
可同时激活的耦合组合的数量只由机床上现有的轴的组合方法限制。

示例

须根据展示的轴结构加工工件两面。 据此应构成 2 个耦合组合。



程序代码	注释
…	
N100 TRAILON(V,Y)	;启用第 1 个联动组合
N110 TRAILON(W,Z,-1)	; 启用第 2 个耦合组合。 耦合系数为负： 耦合轴以与引导轴相反的方向作相应运动。
N120 G0 Z10	; Z 轴和 W 轴以相反的轴向进给。
N130 G0 Y20	; Y 轴和 V 轴以相同的轴向进给。
…	
N200 G1 Y22 V25 F200	; 叠加耦合轴"V"轴的某个相关和不相关的运动。
…	
TRAILOF(V,Y)	; 关闭第 1 个耦合组合。
TRAILOF(W,Z)	; 关闭第 2 个耦合组合。

其它信息

轴类型

一个耦合组合可以由线性轴和回转轴的任意组合构成。 一个模拟轴也可在此被定义为引导轴。

耦合轴

一个耦合轴最多可同时指定 2 个引导轴。 在不同的耦合组合中指定引导轴。

可以为耦合轴编程所有系统提供的运行指令(G0, G1, G2, G3, ...).。 除了单独定义的位移，耦合轴还会按照耦合系数运行从引导轴导出的位移。

动态性能限制

动态性能的限制取决于激活耦合组合的方式：

- 在零件程序中激活
  如果在零件程序中激活耦合，而所有的引导轴被用作当前生效的编程轴，那么在引导轴运行时会考虑所有耦合轴的动态性能，避免出现过载。
  如果在零件程序中激活了耦合，而其中的引导轴没有被用作当前生效通道中的编程轴 ($AA_TYP ≠ 1)，那么在引导轴运行时不会考虑耦合轴的动态性能。 因此，如果某个耦合轴的动态性能稍稍低于耦合要求的水平，会使该轴出现过载。
- 在同步中激活
  如果在同步中激活耦合，那么在引导轴运行时不会考虑耦合轴的动态性能。 因此，如果某个耦合轴的动态性能稍稍低于耦合要求的水平，会使该轴出现过载。

⚠ 小心
轴过载 如果一个耦合组合 ● 在同步中 ● 或在零件程序中被激活，其中的引导轴不是耦合轴通道中的编程轴， 那么用户和机床制造商应负责采取相应的措施，避免引导轴的运行导致耦合轴出现过载。

耦合状态

在零件程序中可以采用以下系统变量查询轴的耦合状态：

$AA_COUP_ACT[<轴>]

值	含义
0	无耦合有效
8	耦合运行生效

使用模数回转轴时的联动轴剩余行程显示

若引导轴和联动轴为模数回转轴，则引导轴的运行会以 n * 360°（n = 1, 2, 3...）累加，从而显示联动轴的剩余行程，直至取消耦合。

示例： 含 TRAILON 的程序段，引导轴为 B，跟随轴为 C

程序代码	注释
TRAILON(C,B,1)	; 激活耦合
G0 B0	; 起始位置
	; 程序段起始处的剩余路径显示：
G91 B360	; B=360, C=360

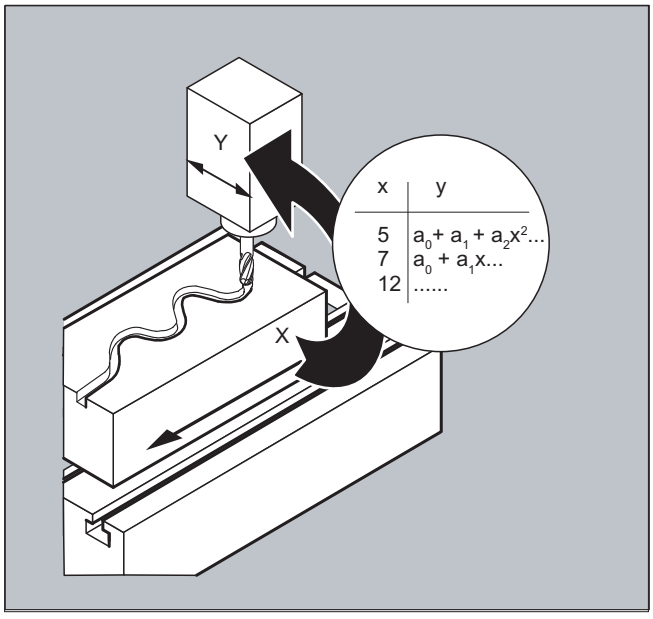
程序代码	注释
G91 B720	; B=720, C=1080
G91 B360	; B=360, C=1440

13.2 曲线图表 (CTAB)

借助曲线图表可以编程两个轴（引导轴和跟随轴）之间的位置关系和速度关系。 曲线图表的定义在零件程序中进行。

应用

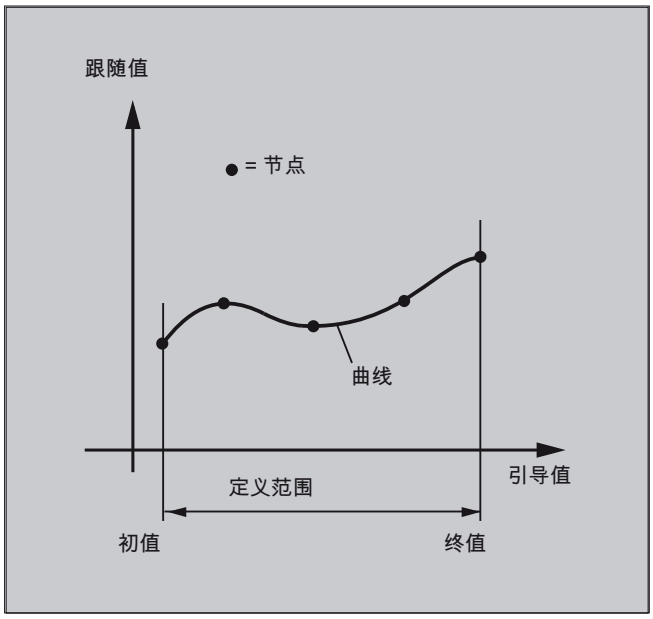
曲线图表替代了机械凸轮。 通过实现引导值和跟随值之间的函数关联，曲线图表构成了轴向引导值耦合的基础。 在相应的编程中，控制系统从相互所属的引导轴和跟随轴的位置中计算出一个与凸轮相应的多项式。



13.2.1 定义曲线图表(CTABDEF, CATBEND)

一个曲线图表所描述的是一个零件程序或者一个零件程序段，其特点是前面插入 CTABDEF 且使用指令 CTABEND 结束。

在该程序段范围内，通过运动指令将引导轴的各个位置一一指定给跟随轴的位置，这些跟随值位置用来作为计算曲线的节点，曲线的形式至多为 5 阶多项式。



前提条件

在定义曲线图表前，必须通过相应的机床数据定义来预留足够的存储容量(→ 机床制造商！）。

句法

```
CTABDEF(<跟随轴>,<引导轴>,<n>,<周期性>[,<存储地点>])
...
CTABEND
```

含义

CTABDEF( ):	曲线图表定义的开始
CTABEND:	曲线图表定义的结束
<跟随轴>:	需要通过曲线图表计算其运行的轴
<引导轴>:	提供引导值以计算跟随轴运行的轴
<n>:	曲线图表的编号(ID) 曲线图表的编号是唯一的，和存储地点无关。 在静态和动态 NC 存储器中不能出现带有相同编号的图表。

<周期性>:	图表周期性	
	0	图表不具有周期性，即使是回转轴也只执行一次
	1	引导轴上图表具有周期性
	2	引导轴和跟随轴上，图表具有周期性
<存储地点>:	存储地点的说明（可选）	
	"SRAM"	曲线图表保存在**静态** NC 存储器中。
	"DRAM"	曲线图表保存在**动态** NC 存储器中。
	提示： 如果没有为该参数编程任何值，机床数据 MD20905 $MC_CTAB_DEFAULT_MEMORY_TYPE 中设置的默认存储地点会生效。	

说明

覆盖

只要一个新曲线图表定义时其编号(<n>)被使用，这个曲线图表将被覆盖。（特例：曲线图表在某个轴耦合中被激活或已被 CTABLOCK 禁用）。 **在覆盖曲线图表时不会给出相应的警告！**

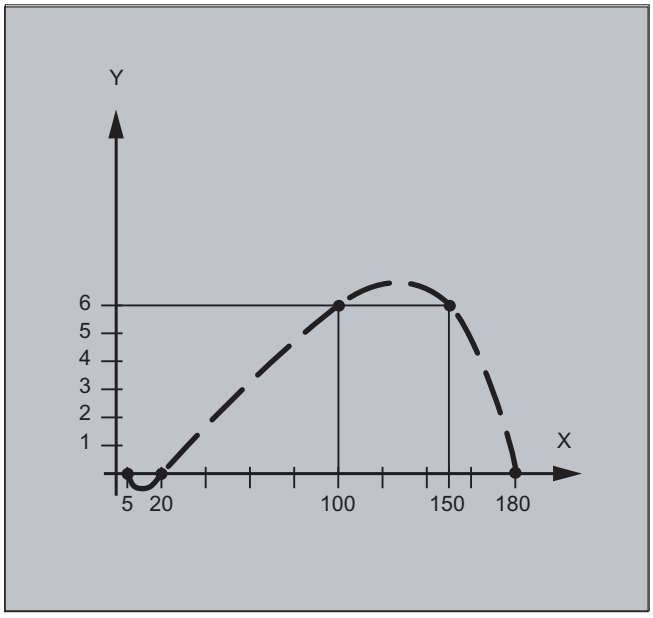
示例

示例 1： 程序段，作为曲线图表定义

不改变一个程序段，用于定义一个曲线图表。 其中所出现的预处理指令 STOPRE 会被保留，在该程序段不再用作图表定义，而 CTABDEF 和 CTABEND 也被删除后，立即恢复生效。

程序代码	注释
…	
CTABDEF(Y,X,1,1)	;定义一个曲线图表。
…	
IF NOT ($P_CTABDEF)	
STOPRE	
ENDIF	
…	
CTABEND	

示例 2： 定义一个非周期性的曲线图表



程序代码	注释
N100 CTABDEF(Y,X,3,0)	；开始定义一个带有编号 3 的非周期性曲线图表。
N110 X0 Y0	；第 1 个运动指令，确定起始值和第 1 个节点：
	引导值： 0，跟随值： 0
N120 X20 Y0	；第 2 个节点：
	引导值： 0…20，跟随值： 起始值…0
N130 X100 Y6	；第 3 个节点：
	引导值： 20…100，跟随值： 0…6
N140 X150 Y6	；第 4 个节点：
	引导值： 100…150，跟随值： 6…6
N150 X180 Y0	；第 5 个节点：
	引导值： 150…180，跟随值： 6…0
N200 CTABEND	；结束定义。 曲线图表在其内部示意图中会生成为最大 5 阶多项式。 视模态选择的零件程序状态插补方式（圆弧插补、直线插补、样条插补）而定，用规定的节点对曲线段再次进行计算。 再次恢复到定义开始前的零件程序状态。

示例 3：定义一个周期性曲线图表

定义一个周期性曲线图表，带有编号 2，引导值范围 0 ～ 360，跟随轴运动从 0 到 45 并且返回到 0：

程序代码	注释
N10 DEF REAL DEPPOS	
N20 DEF REAL GRADIENT	
N30 CTABDEF(Y,X,2,1)	;开始定义。

程序代码	注释
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	; 结束定义。
; 通过耦合 Y 到 X 对曲线进行测试	
N120 G1 F1000 X0	
N130 LEADON(Y,X,2)	
N140 X360	
N150 X0	
N160 LEADOF(Y,X)	
N170 DEPPOS=CTAB(75.0,2,GRADIENT)	; 在引导值为 75.0 时读图表功能。
N180 G0 X75 Y=DEPPOS	; 引导轴和跟随轴的定位。
; 启用耦合之后，无需对跟随轴进行同步。	
N190 LEADON(Y,X,2)	
N200 G1 X110 F1000	
N210 LEADOF(Y,X)	
N220 M30	

其它信息

曲线图表的初值和终值

曲线图表定义范围开始的初值是曲线图表定义之内相关轴位置的说明（第一个运动指令）。曲线图表的定义范围的终值相应地由最后的运行指令决定。

可用的语言范围

在曲线图表的定义内，可使用整个 NC 语言范围。

说明

在曲线图表定义中不允许以下指令：

- 预处理停止
- 引导轴运动过程中的跳转（例如当切换坐标转换时）
- 单独的跟随轴的运动指令
- 引导轴的反向运动，即引导轴的位置必须始终唯一
- 不同程序级的 CTABDEF 和 CTABEND 指令

模态指令的有效性

所有在曲线图表定义之内激活的模态指令均在曲线图表定义结束处失效。 因此，图表定义所位于的零件程序，在图表定义的前后处于相同的状态。

R 参数赋值

编程 CTABEND 后，图表定义范围内的 R 参数赋值被复位。

示例：

程序代码	注释
...	
R10=5 R11=20	;R10=5
...	
CTABDEF	
G1 X=10 Y=20 F1000	
R10=R11+5	;R10=25
X=R10	
CTABEND	
...	;R10=5

ASPLINE, BSPLINE, CSPLINE 的激活

如果在一个曲线图表定义 CTABDEF( ) ... CTABEND 内部激活一个 ASPLINE， BSPLINE 或 CSPLINE , 则在激活该样条前至少应当编程一个起始点。 要避免在 CTABDEF 之后立即激活，否则样条会依赖于曲线图表定义之前的当前轴位置。

示例：

程序代码
...
CTABDEF(Y,X,1,0)
X0 Y0
ASPLINE
X=5 Y=10
X10 Y40
...
CTABEND

重复使用曲线图表

如果图表存储在 NC 静态存储器（SRAM）中，则通过曲线图表计算出的、主动轴和跟随轴的函数关系会保留在所选择的图表号之下，即使零件程序结束或断电。

保存在动态存储器（DRAM）中的图表会在上电时被删除，必须再次创建。

已建立的曲线图表可用到引导轴和跟随轴的任意轴组合上，而和建立曲线图表时使用的轴没有关系。

曲线图表的覆盖

只要一个新曲线图表定义时其编号被使用，这个曲线图表将被覆盖。

特例： 曲线图表已在某个轴耦合中激活或者已被 CTABLOCK 禁用。

说明

在覆盖曲线图表时中不给出相应的警告。

曲线图表定义生效?

使用系统变量$P_CTABDEF 可随时从零件程序中查询曲线图表定义是否已激活。

取消曲线图表定义

将定义曲线图表的语句用括号括起来后，零件程序段就可重新作为真实的零件程序使用。

通过“从外部执行”载入曲线图表

通过“从外部执行”载入曲线图表时，必须通过机床数据 MD18360 $MN_MM_EXT_PROG_BUFFER_SIZE 正确选择加载缓存器(DRAM)的容量，从而可以同时在加载缓存器中保存完整的曲线表定义。 否则将发出报警，停止零件程序的处理。

跟随轴的跳转

根据机床数据
MD20900 $MC_CTAB_ENABLE_NO_LEADMOTION
的设置，在缺少引导轴运动时允许跟随值跳转。

13.2.2 检查曲线图表的存在性(CTABEXISTS)

通过指令 CTABEXISTS 可以检查，NC 存储器中是否存在某个曲线图表号。

句法

```
CTABEXISTS(<n>)
```

含义

CTABEXISTS:	检查，在静态或动态 NC 存储器中是否存在编号为<n>的曲线图表	
	0	图表不存在
	1	图表存在
<n>:	曲线图表的编号(ID)	

13.2.3 删除曲线图表(CTABDEL)

使用 CTABDEL 可以删除曲线图表。

说明

在轴耦合中生效的曲线图表不能被删除。

句法

```
CTABDEL(<n>)
CTABDEL(<n>,<m>)
CTABDEL()(<n>,<m>,<存储地点>)
CTABDEL()
CTABDEL(,,<存储地点>)
```

含义

CTABDEL:	用于删除曲线图表的指令
<n>:	待删除的曲线图表的编号(ID) 在删除曲线图表范围 CTABDEL(<n>,<m>)时，用<n>指定范围内第一个曲线图表的编号。
<m>:	在删除曲线图表范围 CTABDEL(<n>,<m>)时，用<m>指定范围内最后一个曲线图表的编号。 <m>必须大于<n>!

<存储地点>:	存储地点的说明（可选） 如果删除时**没有指定**存储地点，则删除静态和动态 NC 存储器中指定的曲线图表。 如果删除时**指定了**存储地点，则只删除指定存储器中指定的曲线图表。 其他曲线图表保持不变。	
	"SRAM"	删除**静态** NC 存储器中的曲线图表
	"DRAM"	删除**动态** NC 存储器中的曲线图表

如果编程 CTABDEL 时没有指定需要删除的曲线图表，则删除**所有**或指定存储器中的所有曲线图表。

CTABDEL():	删除静态和动态 NC 存储器中的所有曲线图表
CTABDEL(,,"SRAM"):	删除静态存储器中的所有曲线图表
CTABDEL(,,"DRAM"):	删除动态存储器中的所有曲线图表

说明

需要删除多个曲线图表 CTABDEL(<n>,<m>)或 CTABDEL()时，如果至少其中有一个在耦合运动中生效，则不执行删除指令，即：**不**删除指定的曲线图表。

13.2.4 禁止删除和覆盖曲线图表(CTABLOCK, CTABUNLOCK)

可以设置“禁止删除和覆盖曲线图表”来保护曲线图表。 该禁止可以随时被取消。

句法

激活锁定功能：

```
CTABLOCK(<n>)
CTABLOCK(<n>,<m>)
CTABLOCK(<n>,<m>,<存储地点>)
CTABLOCK()
CTABLOCK(,,<存储地点>)
```

取消锁定功能：

```
CTABUNLOCK(<n>)
CTABUNLOCK(<n>,<m>)
CTABUNLOCK(<n>,<m>,<存储地点>)
CTABUNLOCK()
CTABUNLOCK(,,<存储地点>)
```

含义

CTABLOCK:	**激活**禁止删除/覆盖的指令
CTABUNLOCK:	**取消**禁止删除/覆盖的指令 CTABUNLOCK 会再次激活被 CTABLOCK 锁定的曲线图表。 在激活的耦合中生效的图表继续保持锁定状态并不能被删除。 一旦由于耦合失效而取消了禁止，CTABLOCK 的锁定功能就被取消。 从而可以删除此图表。 而无需再次调用 CTABUNLOCK。
<n>:	待锁定/解除锁定的曲线图表的编号(ID) 在锁定/解除锁定曲线图表范围 CTABLOCK(<n>,<m>)/ CTABUNLOCK(<n>,<m>)时，用<n>指定范围内第一个曲线图表的编号。
<m>:	在锁定/解除锁定曲线图表范围 CTABLOCK(<n>,<m>)/ CTABUNLOCK(<n>,<m>)时，用<m>指定范围内最后一个曲线图表的编号。 <m>必须大于<n>!
<存储地点>:	存储地点的说明（可选） 如果在锁定/解除锁定时**没有指定**存储地点，则该指令对静态和动态 NC 存储器中指定的曲线图表生效。 如果在锁定/解除锁定时**指定了**存储地点，则该指令只对指定存储器中指定的曲线图表生效。 其他的曲线图表不会受影响。
	"SRAM" 锁定/解除锁定**静态** NC 存储器中的曲线图表
	"DRAM" 锁定/解除锁定**动态** NC 存储器中的曲线图表

如果编程 CTABLOCK/CTABUNLOCK 时没有指定需要锁定或解除锁定的曲线图表，则该指令对**所有**或指定存储器中的所有曲线图表生效。

CTABLOCK():	锁定静态和动态 NC 存储器中的所有曲线图表
CTABLOCK(,,"SRAM"):	锁定静态存储器中的所有曲线图表
CTABLOCK(,,"DRAM"):	锁定动态存储器中的所有曲线图表
CTABUNLOCK():	解除锁定静态和动态 NC 存储器中的所有曲线图表
CTABUNLOCK(,,"SRAM"):	解除锁定静态存储器中的所有曲线图表
CTABUNLOCK(,,"DRAM"):	解除锁定动态存储器中的所有曲线图表

13.2.5 曲线图表： 确定图表属性(CTABID, CTABISLOCK, CTABMEMTYP, CTABPERIOD)

通过该指令可以查询曲线图表的重要属性，如图表编号、锁定状态、存储地点和周期性。

句法

```
CTABID(<p>)
CTABID(<p>,<存储地点>)
CTABISLOCK(<n>)
CTABMEMTYP(<n>)
TABPERIOD(<n>)
```

含义

CTABID:	返回一个**图表号**，该图表号在指定的存储器中被存为第<p>个曲线图表。 示例： CTABID(1,"SRAM")返回静态 NC 存储器中第一个曲线图表的编号。 此处，第一个曲线图表相当于最高编号的图表。 **提示：** 如果在前后两个 CTABID 的调用期间，存储器中的图表顺序发生变化，例如，由于用 CTABDEL 删除了某个图表，则 CTABID(<p>,...)会在 <p>号相同时返回另一个图表，而不是之前的一个。	
CTABISLOCK:	返回编号为<n>的曲线图表的**锁定状态**：	
	0	图表未锁定
	1	图表被 CTABLOCK 锁定
	2	图表被激活的耦合锁定
	3	图表被 CTABLOCK 和激活的耦合锁定
	-1	图表不存在
CTABMEMTYP:	返回编号为<n>的曲线图表的**存储地点**：	
	0	静态 NC 存储器中的曲线图表
	1	动态 NC 存储器中的曲线图表
	-1	图表不存在

CTABPERIOD:	返回编号为<n>的曲线图表的**周期性**：	
	0	图表为非周期性
	1	引导轴上图表具有周期性
	2	引导轴和跟随轴上，图表具有周期性
	-1	图表不存在
<p>:	存储器中的输入号	
<n>:	曲线图表的编号(**ID**)	
<存储地点>:	存储地点的说明（可选）	
	"SRAM"	**静态** NC 存储器
	"DRAM"	**动态** NC 存储器
	提示： 如果没有为该参数编程任何值，机床数据 MD20905 $MC_CTAB_DEFAULT_MEMORY_TYPE 中设置的默认存储地点会生效。	

13.2.6 读取曲线图表值 (CTABTSV, CTABTEV, CTABTSP, CTABTEP, CTABSSV, CTABSEV, CTAB, CTABINV, CTABTMIN, CTABTMAX)

在零件程序中可以读取以下曲线图表值：

- 曲线图表开头和结尾上的引导轴值和跟随轴值
- 曲线段开头和结尾的跟随轴值
- 一个引导轴值的跟随轴值
- 一个跟随轴值的引导轴值
- 跟随轴的最大值和最小值
  - 在图表的整个定义范围内
    或者
  - 在定义的图表间隔内

句法

```
CTABTSV(<n>,<斜率>[,<跟随轴>])
CTABTEV(<n>,<斜率>[,<跟随轴>])
CTABTSP(<n>,<斜率>[,<引导轴>])
CTABTEP(<n>,<斜率>[,<引导轴>])
CTABSSV(<引导值>,<n>,<斜率>[,<跟随轴>])
CTABSEV(<引导值>,<n>,<斜率>[,<跟随轴>])
CTAB(<引导值>,<n>,<斜率>[,<跟随轴>,<引导轴>]
```

```
CTABINV(<跟随值>,<近似值>,<n>,<斜率>[,<跟随轴>,<引导轴>]
CTABTMIN(<n>[,<跟随轴>])
CTABTMAX(<n>[,<跟随轴>])
CTABTMIN(<n>,<a>,<b>[,<跟随轴>,<引导轴>])
CTABTMAX(<n>,<a>,<b>[,<跟随轴>,<引导轴>])
```

含义

CTABTSV:	读取编号为<n>的图表**开头**的跟随轴值
CTABTEV:	读取编号为<n>的图表**结尾**的跟随轴值
CTABTSP:	读取编号为<n>的图表**开头**的引导轴值
CTABTEP:	读取编号为<n>的图表**结尾**的引导轴值
CTABSSV:	指定的引导轴值，即<引导值>对应的曲线段**开头**的跟随轴值
CTABSEV:	指定的引导轴值，即<引导值>对应的曲线段**结尾**的跟随轴值
CTAB:	读取指定的引导轴值，即<引导值>对应的跟随轴值
CTABINV:	读取指定的跟随轴值，即<跟随值>对应的引导轴值
CTABTMIN:	确定跟随轴的**最小值**： ● 在图表的整个定义范围内 或者 ● 在一个定义的间隔<a> ... <b>内
CTABTMAX:	确定跟随轴的**最大值**： ● 在图表的整个定义范围内 或者 ● 在一个定义的间隔<a> ... <b>内
<n>：	曲线图表的编号(**ID**)
<斜率>:	在参数<斜率>内返回了测定位置上曲线图表函数的**斜率**。
<跟随轴>:	需要通过曲线图表计算其运行的轴（可选）
<引导轴>:	提供引导值以计算跟随轴运行的轴（可选）
<跟随值>:	CTABINV 中，用于读取相应引导值的跟随值
<引导值>:	引导值： ● CTAB 中，用于读取相应跟随值 或者 ● CTABSSV/CTABSEV 中，用于选择曲线段
<近似值>:	CTABINV 中，引导值轴和跟随轴值的对应关系并不强制是唯一的。 CTABINV 为此需要一个参数，即用于所求引导轴值的近似值。

<a>:	CTABTMIN/CTABTMAX 中引导值间隔的**下限**
<b>:	CTABTMIN/CTABTMAX 中引导值间隔的**上限**
	提示： 引导值间隔<a> ... <b>必须在曲线图表的定义范围内。

示例

示例 1：

确定曲线图表开头和末尾的跟随轴值和引导轴值，以及整个定义范围内跟随轴的最大值和最小值。

```
程序代码                                    注释
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)                       ; 图表定义开始
N110 X0 Y10                                 ; 第 1 个图表曲线段的开始位置
N120 X30 Y40                                ; 第 1 个图表曲线段的结束位置 =
                                            第 2 个图表曲线段的开始位置
N130 X60 Y5                                 ; 第 2 个图表曲线段的结束位置= ...
N140 X70 Y30
N150 X80 Y20
N160 CTABEND                                ; 结束图表定义。
...
N200 STARTPOS=CTABTSV(1,GRADIENT)           ; 曲线图表开头的跟随轴值 = 10
N210 ENDPOS=CTABTEV(1,GRADIENT)             ; 曲线图表结尾的跟随轴值 = 20
N220 STARTPARA=CTABTSP(1,GRADIENT)          ; 曲线图表开头的引导轴值 = 0
N230 ENDPARA=CTABTEP(1,GRADIENT)            ; 曲线图表结尾的引导轴值 = 80
N240 MINVAL=CTABTMIN(1)                     ; 确定 Y=5 时跟随轴的最小值
N250 MAXVAL=CTABTMAX(1)                     ; 确定 Y=40 时跟随轴的最大值
```

示例 2：

确定引导轴值 X=30 所属的曲线段开头和结尾的跟随轴值。

```
程序代码                                    注释
N10 DEF REAL STARTPOS
N20 DEF REAL ENDPOS
```

程序代码	注释
N30 DEF REAL GRADIENT	
...	
N100 CTABDEF(Y,X,1,0)	; 图表定义开始
N110 X0 Y0	; 第 1 个图表曲线段的开始位置
N120 X20 Y10	; 第 1 个图表曲线段的结束位置 = 第 2 个图表曲线段的开始位置
N130 X40 Y40	; 第 2 个图表曲线段的结束位置= ...
N140 X60 Y10	
N150 X80 Y0	
N160 CTABEND	; 结束图表定义。
...	
N200 STARTPOS=CTABSSV(30.0,1,GRADIENT)	; 曲线段 2 的开始位置 Y = 10
N210 ENDPOS=CTABSEV(30.0,1,GRADIENT)	; 曲线段 2 的结束位置 Y = 40

其它信息

同步中的应用

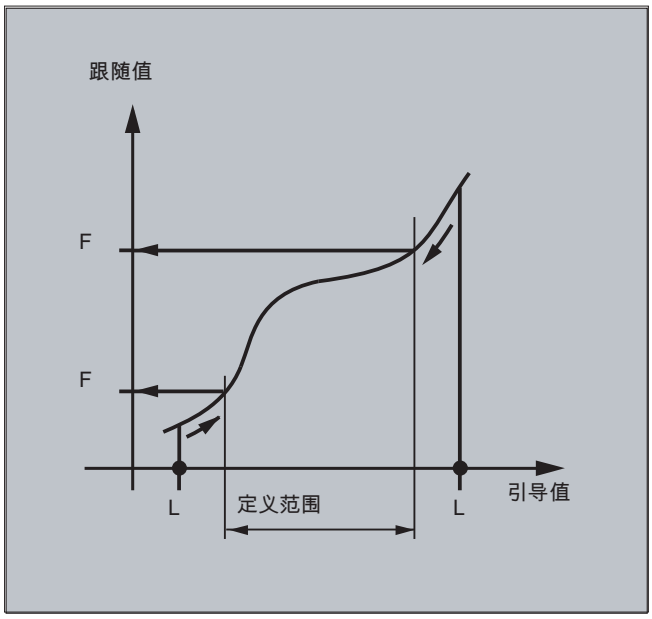
所有用于读取曲线图表值的指令也可以应用在同步中，参见章节“运行同步”。

在使用指令 CTABINV, CTABTMIN 和 CTABTMAX 时应注意：

- 在执行指令时应具有足够的 NC 性能
  或者
- 在调用前应询问曲线图表段的数量，从而可以划分相关的图表。

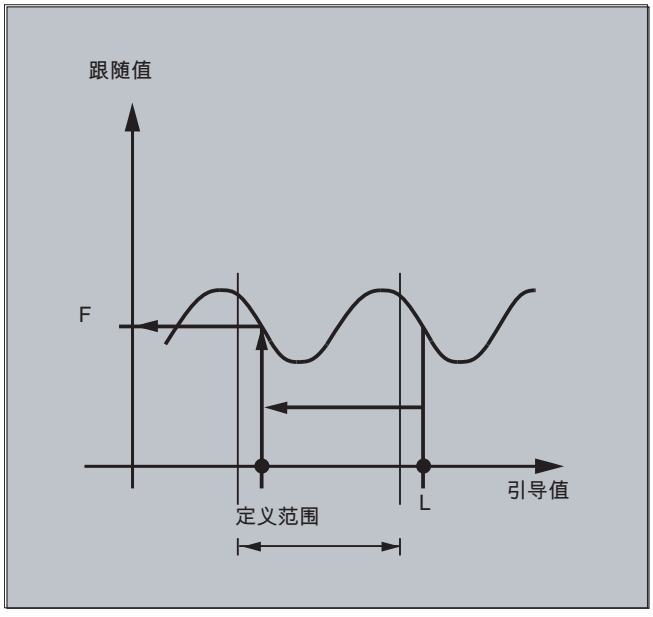
CTAB，非周期性曲线图表

如果指定的<引导值>在定义范围外，则输出上限值或下限值作为跟随值：



CTAB，周期性曲线图表

如果指定的<引导值>在定义范围之外，则取模计算定义范围内的引导值，并输出相应的跟随值。



用于 CTABINV 的近似值

CTABINV 指令需要一个用于所求引导值的近似值。 CTABINV 返回与近似值最接近的主值。例如，近似值可以是上一个插补周期中的引导值。

曲线图表函数的斜率

斜率的输出值(<斜率>)可以计算相应位置上的引导轴或跟随轴的速度。

引导轴或跟随轴的说明

如果以不同的长度单位定义了引导轴和跟随轴，引导轴和/或跟随轴的可选说明就比较重要。

CTABSSV, CTABSEV

指令 CTABSSV 和 CTABSEV 在下列情况下**不适合**，查询对已编程的曲线段：

- 编程了圆弧或者渐开线。
- 倒角或倒圆已使用 CHF, RND 激活。
- 使用 G643 进行精磨的功能已激活。
- NC 程序段压缩器已使用例如 COMPON/COMPCURV/COMPCAD 激活。

13.2.7 曲线图表： 检查资源使用率(CTABNO, CTABNOMEM, CTABFNO, CTABSEGID, CTABSEG, CTABFSEG, CTABMSEG, CTABPOLID, CTABPOL, CTABFPOL, CTABMPOL)

使用该指令，编程人员可以了解目前被曲线图表、图表段和多项式占用的资源情况。

句法

```
CTABNO
CTABNOMEM(<存储地点>)
CTABFNO(<存储地点>)
CTABSEGID(<n>,<存储地点>)
CTABSEG(<存储地点>,<段类型>)
CTABFSEG(<存储地点>,<段类型>)
CTABMSEG(<存储地点>,<段类型>)
CTABPOLID(<n>)
CTABPOL(<存储地点>)
CTABFPOL(<存储地点>)
CTABMPOL(<存储地点>)
```

含义

CTABNO:	确定**已定义**的曲线图表的总数量（在静态和动态 NC 存储器中）
CTABNOMEM:	确定指定的<存储地点>中**已定义**的曲线图表的数量。
CTABFNO:	确定指定的<存储地点>中**还可以定义**的曲线图表的数量。

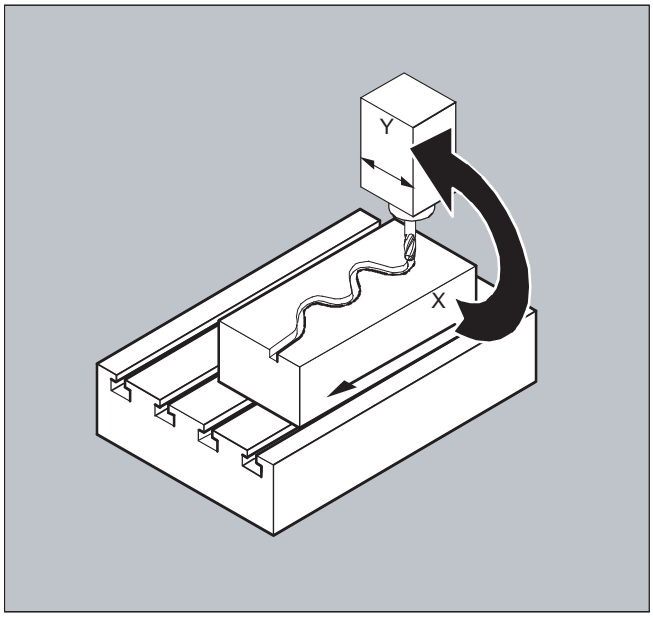
CTABSEGID:	确定编号为<n>的曲线图表、指定<段类型>的段数量。	
CTABSEG:	确定指定的<存储地点>中、指定<段类型>的**已使用**的段数量	
CTABFSEG:	确定指定的<存储地点>中、指定<段类型>的**还可使用**的段数量	
CTABMSEG:	确定指定的<存储地点>中、指定<段类型>的**允许的最大**曲线段数量	
CTABPOLID:	确定编号为<n>的曲线图表的多项式数量。	
CTABPOL:	确定指定的<存储地点>中**已使用**的曲线多项式的数量。	
CTABFPOL:	确定指定的<存储地点>中**还可以使用**的曲线多项式的数量。	
CTABMPOL:	确定指定的<存储地点>中**允许的最大**曲线多项式的数量。	
<n>：	曲线图表的编号(**ID**)	
<存储地点>:	存储地点的说明（可选）	
	"SRAM"	**静态** NC 存储器
	"DRAM"	**动态** NC 存储器
	提示： 如果没有为该参数编程任何值，机床数据 MD20905 $MC_CTAB_DEFAULT_MEMORY_TYPE 中设置的默认存储地点会生效。	
<段类型>:	段类型的说明（可选）	
	"L"	线性段
	"P"	多项式段
	提示： 如果没有为该参数编程任何值，则输出所有线性段和多项式段。	

13.3 轴向引导值耦合 (LEADON, LEADOF)

说明

该功能不能用于 SINUMERIK 828D。

在轴的引导值耦合时同步运行一个引导轴和一个跟随轴。 同时，跟随轴的相应位置通过一个曲线图表或者通过一个从该图表算得的多项式已明确分配给引导轴的一个（有可能是模拟的）位置。



引导轴 是给曲线图表发送输入值的那个轴。 **跟随轴** 是接收通过曲线图表算得的位置的那个轴。

实际值和设定值耦合

作为引导值，即用于计算位置的输出值可以使用：

- 引导轴位置的实际值： 实际值同步
- 引导轴位置的设定值： 设定值同步

引导值耦合一直在基准坐标系中有效。

曲线图表的建立请见“曲线图表”一章。

句法

```
LEADON(<跟随轴>,<引导轴>,<n>)
LEADOF(<跟随轴>,<引导轴>,<n>)
```

或者在不给定引导轴的情况下关闭：
LEADOF(<跟随轴>)

引道值耦合的激活和取消即可通过零件程序进行，也可在运行中通过同步动作进行。

含义

LEADON:	启用引导值耦合
LEADOF:	关断引导值耦合
<跟随轴>:	跟随轴
<引导轴>:	引导轴
<n>:	曲线图表编号
$SA_LEAD_TYPE:	在设定值和实际值耦合之间转换

关闭引导值耦合， LEADOF

如关闭引导轴耦合，跟随轴可重新成为正常的指令轴！

轴向引导值耦合和各种运行状态，RESET

与机床参数的设定有关，引导值耦合由 RESET 关断。

同步动作所构成的引导值耦合举例

在冲压设备中，在一个引导轴（冲杆轴）和由传输轴与辅助轴构成的传输系统的轴之间，这种传统地机械耦合应由一个电子的耦合系统代替。

这表明了，在一个冲压设备中一个机械的传输系统如何被一个电子的传输系统所代替。 耦合和去耦合过程作为**静态同步动作** 实现。

传输轴和辅助轴可通过曲线图表作为跟随轴被引导轴 LW(冲杆轴)控制。

跟随轴

X 进给轴或者纵向轴
YL 闭合轴或者横向轴
ZL 升降轴
U 辊进给，辅助轴
V 校正头，辅助轴
W 润滑装置，辅助轴

动作

作为动作出现在同步动作中的例如有：

- 耦合，LEADON(<跟随轴>,<引导轴>,<曲线图表编号>)
- 解耦，LEADOF(<跟随轴>,<引导轴>)
- 设定实际值，PRESETON(<轴>,<值>)
- 设定标记，$AC_MARKER[i]=<值>
- 同步方式： 实际/虚拟的引导值
- 轴位置的逼近，POS[<轴>]=<值>

条件

作为条件，对数字快速输入、实时变量 $AC_MARKER 和与逻辑运算子 AND 有关联的位置对比进行分析。

说明

下列示例中将行交错变化、行首空格和**粗体**字仅用于提高编程的可读性。 为了控制，所有在标志行数下的都占一行。

注释

程序代码	注释
	；定义所有静态同步动作。
	; **** 复位标记器
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 $AC_MARKER[7]=0	
	; **** E1 0=>1 启用传送偶合
N10 IDS=1 EVERY ($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) $AC_MARKER[0]=1	
	;**** E1 0=>1 启用辊进给偶合
N20 IDS=11 EVERY ($A_IN[1]==1) AND ($A_IN[5]==0) AND ($AC_MARKER[5]==0) DO LEADON(U,LW,4) PRESETON(U,0) $AC_MARKER[5]=1	
	; **** E1 0->1 启用校正头偶合
N21 IDS=12 EVERY ($A_IN[1]==1) AND ($A_IN[5]==0) AND ($AC_MARKER[6]==0) DO LEADON(V,LW,4) PRESETON(V,0) $AC_MARKER[6]=1	
	; **** E1 0->1 启用润滑装置偶合
N22 IDS=13 EVERY ($A_IN[1]==1) AND ($A_IN[5]==0) AND ($AC_MARKER[7]==0) DO LEADON(W,LW,4) PRESETON(W,0) $AC_MARKER[7]=1	
	; **** E2 0=>1 断开偶合
N30 IDS=3 EVERY ($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	
....	

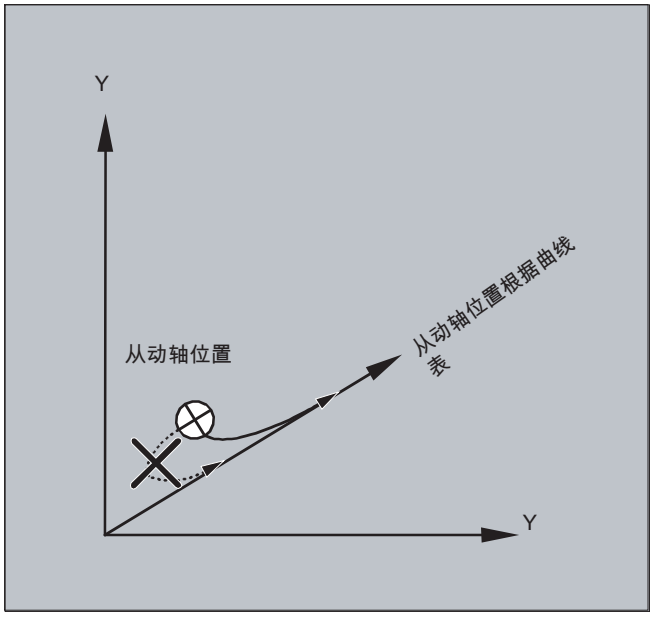
程序代码	注释
N110 G04 F01	
N120 M30	

说明

引导值耦合要求引导轴和跟随轴的同步动作。 只有跟随轴在根据曲线图表计算出的曲线的公差范围内，引导值耦合打开的情况下，才能达到同步动作。

跟随轴位置的允差范围通过机床数据 MD 37200: COUPLE_POS_POL_COARSE A_LEAD_TYPE 定义。

如跟随轴在引导值耦合接通时还不位于相应的位置上，同步运动自动产生，只要计算所得的跟随轴位置的额定值与其实际的跟随轴位置接近。 跟随轴在同步过程中沿着通过跟随轴的额定速度（从引导轴速度中且根据曲线图表 CTAB 算得）所定义的方向运动。

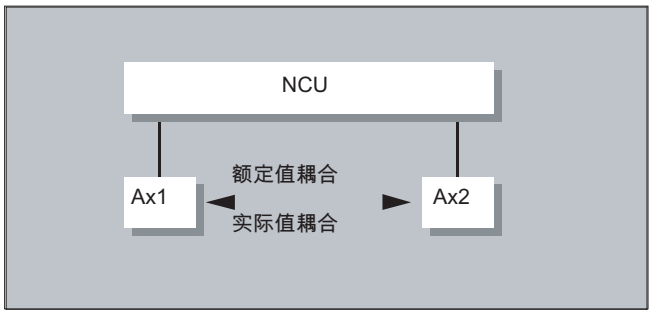


没有同步运动

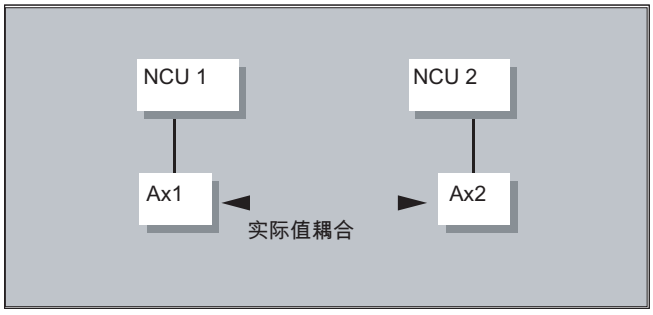
如计算所得的、在引导值耦合打开时跟随轴的额定位置与其实际跟随轴位置相距很远，则不产生同步运动。

实际值和额定值耦合

与实际值耦合相比，额定值耦合提供更好的引导轴和跟随轴之间的同步运动，因此符合预置符合标准。



只有当引导轴和跟随轴插补由相同的 NCU 进行时，才可能进行额定值耦合。 一个外部的引导轴上仅可以通过实际值，使跟随轴与引导轴耦合。



可通过设定数据 $SA_LEAD_TYPE 进行**转换**。

实际值和额定值间的转换应一直在跟随轴静止状态下完成。 因为只有在静止状态时，转换后才能重新同步。

应用示例

额定值的读取在大的机床振荡时不能完成。 在启动引导值耦合时，如在压力传输时有大的振荡，则有必要从实际值耦合切换到额定值耦合。

引导值模拟，当进行额定值偶合时

通过机床数据可将引导轴插补器与伺服器分开。 从而额定值耦合时额定值可以在引导轴不运动的情况下得到。

例如用在同步动作中的通过额定值耦合生成的引导值可以从下列变量中读取：

- $AA_LEAD_P 引导值位置
- $AA_LEAD_V 引导值速度

生成引导值

引导值可有选择性地在其他自动编程的程序中产生。 这样产生的引导值将写入变量

- $AA_LEAD_SP 引导值位置
- $AA_LEAD_SV 引导值速度

并从中读取。 必须设置设定数据 $SA_LEAD_TYPE = 2，以使用这些变量。

耦合的状态

在 NC 零件程序中可以用以下系统变量询问耦合的状态：

$AA_COUP_ACT[轴]
0:没有偶合激活
16: 引导值耦合有效

同步动作的状态管理

开关和耦合过程通过实时变量

$AC_MARKER[i] = n
进行管理，使用:
i 标记编号
n 状态参数

13.4 电子齿轮箱 (EG)

使用辅助功能“电子齿轮”可以控制 **跟随轴**运动，使之按照线性运动偏移与最多与五个 **引导轴**相关联运动。 引导轴和跟随轴之间的关联按照每个引导轴通过耦合系数进行定义。

算出的跟随轴运动分量是由单个引导轴运动分量乘各自的耦合系数通过加法构成的。 激活一个 EG 轴组时，可以使跟随轴在某定义的位置上同步。 一个齿轮组可以由零件程序：

- 定义，
- 接通，
- 关闭，
- 删除

。

跟随轴的运动可以有选择的被

- 引导轴的额定值以及
- 引导轴的实际值来引导。

作为扩展功能，引导轴和跟随轴之间的非线性关联也可以通过 **曲线列表** (参见章节轨迹特性)来实现。 电子齿轮可以串联，即：电子齿轮的跟随轴可以成为另一个电子齿轮的引导轴。

13.4.1 定义电子齿轮 (EGDEF)

一副 EG 轴组可以通过跟随轴数据和最少 1 个最多五个引导轴带各自耦合类型来确定。

前提条件

EG 轴组定义的前提：

对于跟随轴还不允许定义轴耦合（有的话，必须提前用 EGDEL 删除现有的）。

句法

```
EGDEF（跟随轴，引导轴 1，耦合类型 1，引导轴 2，耦合类型 2，...）
```

含义

EGDEF:	电子齿轮定义	
跟随轴:	由引导轴影响的轴	
引导轴 1 ,..., 引导轴 5	影响跟随轴的轴	
耦合类型 1 ,..., 耦合类型 5	耦合类型 耦合类型不必对所有引导轴都相同，因此必须为每个引导轴单独标注。	
	值：	**含义：**
	0	跟随轴受相应引导轴的 **实际值** 影响。
	1	跟随轴受相应引导轴的 **额定值** 影响。

说明

对 EG 耦合组进行定义时，必须预设耦合系数为零。

说明

EGDEF 触发进给停止。 用 EGDEF 进行齿轮定义，在使用时必须保持不变，如果系统中有一个或者多个引导轴通过**曲线图表**影响跟随轴。

示例

程序代码	注释
EGDEF(C,B,1,Z,1,Y,1)	；定义一个 EG 轴组。 引导轴 B，Z，Y 通过额定值影跟随轴 C。

13.4.2 接通电子齿轮（EGON，EGONSYN，EGONSYNE）

有 3 种型式用于接通 EG 轴组。

句法

型式 1：

在无同步的情况下选择性接通 EG 轴组，通过：
EGON（FA，“程序段转换模式”，LA1，Z1，N1，LA2，Z2，N2，...，LA5，Z5，N5）

型式 2：

在同步的情况下选择性接通 EG 轴组，通过：
EGONSYN（FA，“程序段转换模式”，SynPosFA，[，LAi，SynPosLAi，Zi，Ni]）

型式 3：

在同步的情况下选择性接通 EG 轴组并规定返回模式，通过：
EGONSYNE（FA，“程序段转换模式”，SynPosFA，返回模式[，LAi，SynPosLAi，Zi，Ni]）

含义

型式 1：

FA	跟随轴	
程序段转换模式:	可以用下列模式：	
	"NOC"	立即进行程序段转换
	"FINE"	在“精确同步运行”时进行程序段转换
	"COARSE"	在“近似同步运行”时进行程序段转换
	"IPOSTOP"	当额定值同步运行时进行程序段转换
LA1, ... LA5	引导轴	
Z1, ... Z5	耦合系数 i 的分子	
N1, ... N5	耦合系数 i 的分母 耦合系数 i = 分子 i /分母 i	

只允许对先前用 EGDEF 进行详细说明的引导轴编程。 必须至少编程一个引导轴。

型式 2：

FA	跟随轴	
程序段转换模式:	可以用下列模式：	
	"NOC"	立即进行程序段转换
	"FINE"	在“精确同步运行”时进行程序段转换
	"COARSE"	在“近似同步运行”时进行程序段转换
	"IPOSTOP"	当额定值同步运行时进行程序段转换
[,LAi,SynPosLAi,Zi,Ni]	(不写方括号) 最少 1 个，最多 5 个跟随由：	
LA1, ... LA5	引导轴	

SynPosLAi	i.引导轴的同步位置
Z1, ... Z5	耦合系数 i 的分子
N1, ... N5	耦合系数 i 的分母 耦合系数 i = 分子 i /分母 i

只允许对先前用 EGDEF 进行详细说明的引导轴编程。 通过为跟随轴(SynPosFA) 和引导轴 (SynPosLA) 编程的“同步定位”，对位置进行定义，其中耦合组当作*同步*有效。 一旦接通时电子齿轮不处于同步状态，跟随轴就运行到它定义的同步位置。

型式 3：

该参数符合型式 2 中的参数，包括：

逼近模式:	可以用下列模式：	
	"NTGT"	在最佳时间返回下一个齿间隙
	"NTGP"	以最佳路径返回下一个齿间隙
	"ACN"	在负旋转方向上绝对运行回转轴
	"ACP"	在正旋转方向上绝对运行回转轴
	"DCT"	编程同步位置时间最佳
	"DCP"	编程同步位置路径最佳

方案 3 只对与模数引导轴耦合的模数跟随轴有影响。 最佳时间考虑了跟随轴的速度极限。

其它信息

描述接通型式

型式 1：

将引导轴以及跟随轴启动时刻的位置作为“同步位置”保存。可以使用系统变量 $AA_EG_SYN 读入“同步位置”。

型式 2：

如果模态轴处于耦合关联状态，则其位置值模态降低。这样就保证了向可能最接近的同步位置运动（所谓的*相对同步*：例如最接近的齿隙）。如果没有将“释放跟随轴叠加”共生面信号 DB(30 +轴编号), DBX 26 位 4 发送给跟随轴，就不会向同步位置运动。与此相反，程序在 EGONSYN 程序段处停止，并且只要上面的信号设置，就给出自清除的报警 16771。

型式 3：

齿间距（度）由下面公式产生：360 * Zi/Ni.对于跟随轴处于调用时刻的情况而言，行程优化与时间优化一样，可提供相同的特性。

如果跟随轴已经运动，可使用 NTGP 向下一个齿隙同步运动，不受跟随轴当前速度的影响。如果跟随轴已经运动，可使用 NTGT 向下一个齿隙同步运动，受跟随轴当前速度的影响。有时轴也会制动。

曲线图表

要将某个**曲线图表** 用于引导轴中的某一个引导轴时，就必须：

Ni　线性耦合耦合系数的分母设置为 0。（分母 0 对于线性耦合是不允许的）。分母零对于控制系统而言表示

Zi　应解释成待使用的曲线表的编号。带有给定编号的曲线图表在启用时刻必须已经定义。

LAi　引导轴的参数与通过耦合系数耦合时的引导轴参数一样（线性耦合）。

有关使用曲线图表和电子齿轮的级联及其同步的其它提示，请参见

文献：

功能手册 特殊功能；轴耦合和 ESR（M3），章节“联动和引导值耦合”。

上电、RESET、运行方式转换、查找时的电子齿轮箱的特性

- 在上电之后**没有** 偶合激活。
- 在复位和运行方式转换之后有效的耦合仍保持。
- 在程序段搜索时，有关开关、删除、定义电子齿轮的指令不予执行和考虑，而是直接跳过。

电子齿轮箱的系统变量

利用电子齿轮的系统变量，零件程序可以求值一个 EG 轴关联的当前状态，有时并做出反应。

电子齿轮的系统变量标记如下：

$AA_EG_ ...

或者

$VA_EG_ ...

文献：

系统变量手册

13.4.3 关闭电子齿轮（EGOFS, EGOFC）

有 3 种型式用于关闭 EG 轴组。

编程

型式 1：

句法	含义
EGOFS(跟随轴)	关闭电子齿轮。 跟随轴制动到停止。 此调用触发预处理停止。

型式 2：

句法	含义
EGOFS（跟随轴，引导轴 1， …，引导轴 5）	指令的这种参数设定允许**有选择性地**排除各个引导轴对跟随轴的运动的影响。

必须至少指定一个引导轴。 有针对性地中止指定引导轴对跟随轴的影响。 此调用触发预处理停止。 如果尚有引导轴保持激活状态，则跟随轴将在其影响下继续运行。 如果所有的引导轴影响都以这种方式关闭，则跟随轴被制动到停止。

型式 3：

句法	含义
EGOFC（跟随主轴 1）	关闭电子齿轮。 跟随主轴以关闭时刻有效的转速/速度继续运行。 此调用触发预处理停止。

说明

该型式仅允许用于主轴。

13.4.4 删除某个电子齿轮箱的定义（EGDEL）

在可以删除电子齿轮箱轴组合的定义之前，必须先将其关闭。

编程

句法	含义
EGDEL（跟随轴）	轴关联的耦合定义被删除。 在到达同时激活的轴关联最大个数之前，又可以用 EGDEF 重新定义其它的轴关联。 此调用触发预处理停止。

13.4.5 旋转进给（G95）/电子齿轮箱（FPR）

使用 FPR 指令也可以将一个电子齿轮箱的跟随轴设定成旋转进给的轴。 对于这种情况，以下的特性适用：

- 进给取决于电子齿轮跟随轴的给定速度。
- 给定速度可以由引导轴和取模－引导轴（不是轨迹轴）的速度和相应的耦合系数计算出来。
- 线性或者非模量引导轴的速度比例和跟随轴的叠加运动不予考虑。

13.5 同步主轴

在同步运行中有一个引导主轴（LS）和一个跟随主轴（FS），即**同步主轴对**。 跟随主轴根据确定的功能关系在激活耦合（同步运行）时跟随引导主轴。

可以借助通道专用的机床数据为每个机床固定同步主轴对，或通过用户专用的 CNC 零件程序定义同步主轴对。 根据 NC 通道，可以最多同时运行 2 个同步主轴对。

耦合可以在零件程序中

- 定义以及修改
- 打开
- 关闭
- 删除

。

由此，可以根据软件版本

- 等待同步运行条件
- 修改程序段转换特性
- 选择给定值耦合或实际值耦合的耦合方式，或者规定引导主轴和跟随主轴之间的角度偏差
- 在打开耦合时接受上一个跟随主轴的编程
- 修改一个测量的或一个已知的同步运行偏差

。

13.5.1 同步主轴： 编程（COUPDEF, COUPDEL, COUPON, COUPONC, COUPOF, COUPOFS, COUPRES, WAITC）

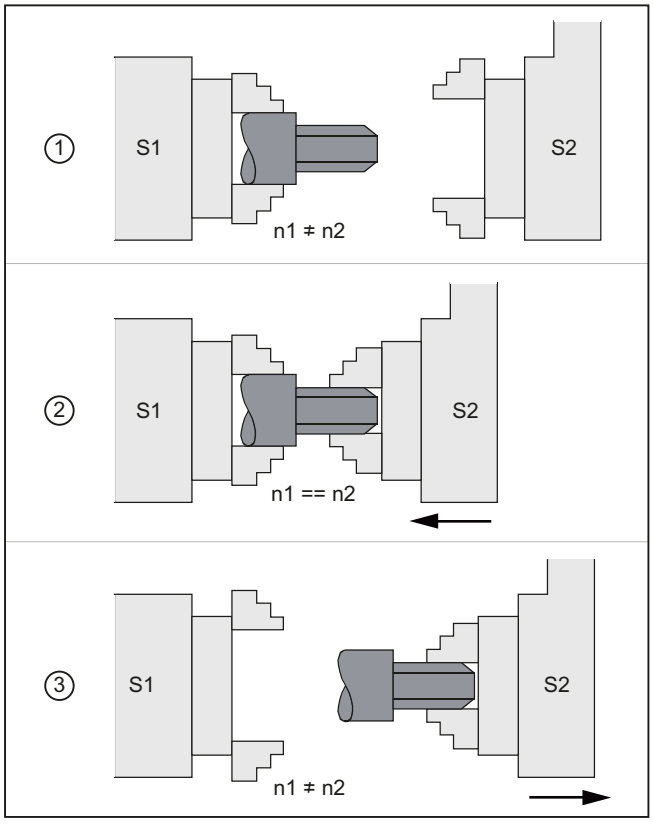
“主轴同步”功能可使主主轴(FS) 和副主轴(LS)以设定的转速比同步运行。

该功能具有以下模式：

- 转速同步($n_{FS} = n_{LS}$)
- 位置同步（$\phi_{FS} = \phi_{LS}$）
- 含角度偏移的位置同步（$\phi_{FS} = \phi_{LS} + \Delta\phi$）

应用示例：

- 实时递交工件，例如用于背面加工，传动比：1:1

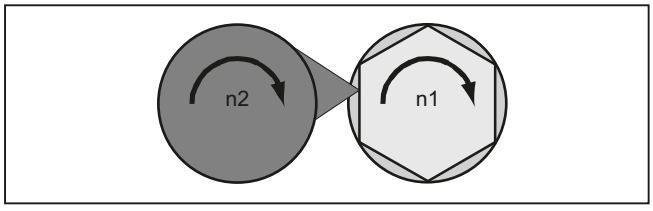


① 转速同步

② 递交工件

③ 背面加工

- 多面体加工（多面体车削）、转速同步、传动比：n_1:n_2



句法

```
COUPDEF(<副主轴>,<主主轴>,<传动比分子>,<传动比分母>,<程序段切换>,<同步方式>)
COUPON(<副主轴>,<主主轴>,<副主轴位置>)
COUPONC(<副主轴>,<主主轴>)
COUPOF(<副主轴>,<主主轴>,<副主轴位置>,<主主轴位置>)
COUPOFS(<副主轴>,<主主轴>)
COUPOFS(<副主轴>,<主主轴>,<副主轴位置>)
```

```
COUPRES(<副主轴>,<主主轴>)
COUPDEL(<副主轴>,<主主轴>)
WAITC(<副主轴>,<程序段切换>,<主主轴>,<程序段切换>)
```

说明

简化的写入方法

在使用 COUPOF, COUPOFS, COUPRES 和 COUPDEL 时可以使用缩写，不用说明主主轴。

含义

COUPDEF:	定义/更改用户自定义的同步
COUPON:	激活同步。从当前转速开始，副主轴与主主轴同步
COUPONC:	使主轴继续采用零件程序中之前写入的方向和转速 M3 S...或 M4 S...。 此时主主轴和副主轴会以差速运行。
COUPOF:	解除同步。 ● 立即开始程序段切换： COUPOF(<S2>,<S1>) ● 等副主轴越过指定位置 <副主轴位置> 或 <主主轴位置>后才进行程序段切换: COUPOF(<S2>,<S1>,<POSFS>) COUPOF(<S2>,<S1>,<POSFS>,<POSLS>)
COUPOFS:	撤销主轴同步，并停止副主轴。 立即进行程序段切换。 COUPOFS(<主轴 2>,<主轴 1>) 副主轴越过指定位置后才使能程序段切换。 COUPOFS(<主轴 2>,<主轴 1>,<副主轴位置>)
COUPRES:	将同步参数复位到定义的 MD 和 SD
COUPDEL:	删除用户定义的同步
WAITC:	等待同步运行条件 （切换程序段时取消 IPO 上的 NOC）
<副主轴>:	副主轴名称
可选参数：	

<主主轴>:	主主轴名称 主轴号：z. B. S2, S1	
<ZFS>, <NLS>:	副主轴和主主轴之间的传动比。 <ZFS> / <NLS> = 分子/分母 缺省设置：<ZFS> / <NLS> = 1.0；分母选择性设定	
<程序段切换>:	程序段切换特性 程序段切换方式有：	
	"NOC"	立即
	"FINE"	达到“精同步”
	"COARSE"	达到“粗同步”
	"IPOSTOP"	在满足 IPOSTOP 条件时进行程序段切换(即：在设定值同步后)，缺省值
	程序段切换特性模态有效。	
<同步方式>:	同步方式：副主轴和主主轴之间的同步方式	
	“DV”	给定值耦合（预设置）
	“AV”	实际值同步
	“VV”	速度同步
	同步方式模态有效。	
<副主轴位置>:	主主轴和副主轴之间的角度偏差	
	取值范围：	0 °… 359.999 °
<副主轴位置>,<主主轴位置>:	副主轴和主主轴解除同步的位置 “在越过 POS_{FS}（副主轴位置）, POS_{LS}（主主轴位置）后使能程序段切换”	
	取值范围：	0 °… 359.999 °

示例

采用主主轴和副主轴工作

程序代码	注释
	主主轴 = 主轴 1 副主轴 = 主轴 2
N05 M3 S3000 M2=4 S2=500	主主轴转速为 3000 rpm, 副主轴转速为 500 rpm。
N10 COUPDEF(S2,S1,1,1,"NOC","Dv")	定义同步（也可以通过机床数据设置）。
...	

程序代码	注释
N70 SPCON	主主轴进入位置闭环控制（设定值同步）。
N75 SPCON(2)	副主轴进入位置闭环控制
N80 COUPON(S2,S1,45)	实时同步，两者之间的角度偏移为 45 度。
...	
N200 FA[S2]=100	定位速度 = 100 deg/min
N205 SPOS[2]=IC(-90)	负方向上 90 度增量运行。
N210 WAITC(S2,"Fine")	等待“精同步”。
N212 G1 X...Y...F...	加工
...	
N215 SPOS[2]=IC(180)	正方向上 180 度增量运行。
N220 G4 S50	暂停时间 = 主主轴 50 转
N225 FA[S2]=0	激活设定的速度 (MD)
N230 SPOS[2]=IC(-7200)	20 转。在负方向上以配置的速度运行。
...	
N350 COUPOF(S2,S1)	实时撤销同步，S=S2=3000
N355 SPOSA[2]=0	副主轴停止在零度位置上
N360 G0 X0 Y0	
N365 WAITS(2)	等待主轴 2。
N370 M5	停止副主轴。
N375 M30	

差速编程

程序代码	注释
	主主轴 = 主轴 1
	副主轴 = 主轴 2
N01 M3 S500	主主轴转速为 500 rpm。
N02 M2=3 S2=300	副主轴转速为 300 rpm。
...	
N10 G4 F1	主主轴停留时间。
N15 COUPDEF (S2,S1,-1)	传动比-1:1
N20 COUPON(S2,S1)	激活同步。副主轴转速由主主轴转速和传动比得出。
...	
N26 M2=3 S2=100	差速编程。

示例：差速运动

1. 用 COUPON 激活主轴同步，副主轴已编程

程序代码	注释
	主主轴 = 主轴 1
	副主轴 = 主轴 2
N05 M3 S100 M2=3 S2=200	主主轴转速为 100 rpm，副主轴转速为 200 rpm。
N10 G4 F5	暂停时间 = 主主轴 5 秒
N15 COUPDEF(S2,S1,1)	副主轴和主主轴之间的传动比为 1.0（缺省设置）。

程序代码	注释
N20 COUPON(S2,S1)	与主主轴实时同步。
N10 G4 F5	副主轴转速为 100 rpm。

2. 用 COUPONC 激活主轴同步，副主轴已编程

程序代码	注释
	主主轴 = 主轴 1
	副主轴 = 主轴 2
N05 M3 S100 M2=3 S2=200	主主轴转速为 100 rpm，副主轴转速为 200 rpm。
N10 G4 F5	暂停时间 = 主主轴 5 秒
N15 COUPDEF(S2,S1,1)	副主轴和主主轴之间的传动比为 1.0（缺省设置）。
N20 COUPONC(S2,S1)	与主主轴实时同步，副主轴恢复之前的转速 S2。
N10 G4 F5	S2 以 100 rpm + 200 rpm = 300 rpm 的转速转动

3. 用 COUPON 激活主轴同步，副主轴静止

程序代码	注释
	主主轴 = 主轴 1
	副主轴 = 主轴 2
N05 SPOS=10 SPOS[2]=20	副主轴 S2 处于定位模式。
N15 COUPDEF(S2,S1,1)	副主轴和主主轴之间的传动比为 1.0（缺省设置）。
N20 COUPON(S2,S1)	与主主轴实时同步。
N10 G4 F1	同步已关闭，S2 保持在 20 度。

4. 用 COUPONC 激活主轴同步，副主轴静止

说明

定位模式或进给轴模式

如果副主轴在同步前处于定位模式或者进给轴模式，则 COUPON(<副主轴>,<主主轴>) 和 COUPONC(<副主轴>,<主主轴>) 中副主轴的工作方式是相同的。

说明

主主轴和进给轴模式

如果主主轴在定义同步前处于进给轴模式中，在启用同步后，以下机床数据中的速度极限值也会生效：

MD32000 $MA_MAX_AX_VELO（最大轴速度）

为避免出现此特性，必须在定义同步前将进给轴切换到主轴模式中(M3 S...或 M4 S...)。

其它信息

由机床数据定义的同步

在由机床数据定义的同步中，主主轴和副主轴通过机床数据定义。 在零件程序中无法修改由机床数据定义的主轴。 但可以通过 COUPDEF 在零件程序中设定其参数（前提条件： 未写保护）。

用户定义的同步：

通过 COUPDEF 可以在零件程序中重新定义或修改同步。 如果已有一个同步生效，则在重新定义前必须先通过 COUPDEL 删除该同步。

通过以下指令可以完整重新定义一个同步：

COUPDEF(<副主轴>,<主主轴>,<传动比分子>,<传动比分母>, 程序段切换特性, 同步方式)

副主轴（FS）和主主轴（LS）

通过“副主轴名称”和“主主轴名称”可以明确确定同步。 在每个 COUPDEF 指令后都必须编写轴名称。 改变其他同步参数模态有效，仅当需要改变时，才必须编写这些同步参数。

示例：

```
COUPDEF(S2,S1)
```

传动比

传动比是副主轴和主主轴之间的转速比：

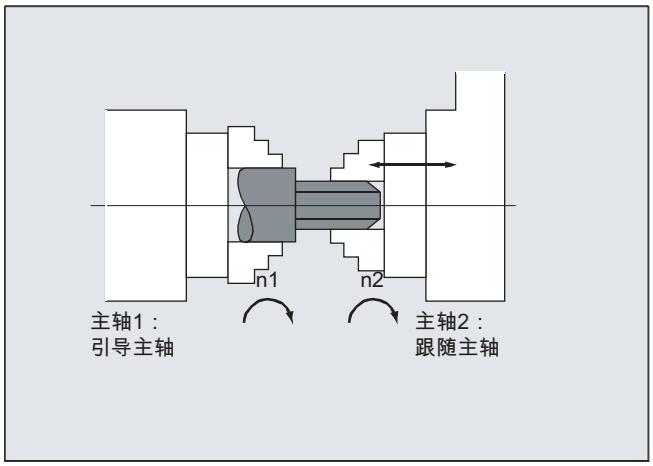
副主轴/主主轴 = 分子/分母

必须写入分子。 分母不强制写入。 不写入分母时，分母自动为 1.0。

示例：

副主轴 S2 和主主轴 S1，传动比 = 1/1

```
COUPDEF(S2, S1, 1.0)
```



说明

在启用了同步并且主轴旋转时也可以修改传动比。

程序段切换特性 NOC，FINE，COARSE，IPOSTOP

在编程程序段切换特性时，可以采用简化的写入方式：

- "NO": 立即（缺省设置）
- "FI": 达到“精同步”
- "CO": 达到“粗同步”
- "IP": 在满足 IPOSTOP 条件时进行程序段切换(即：在设定值同步后)

同步方式

说明

仅当关闭同步后才能更改同步方式。

激活主轴同步 COUPON，<POSFS>

- 激活主轴同步，同时主主轴和副主轴之间存在指定的角度偏移
  - COUPON(S2,S1)
  - COUPON(S2)
- 激活主轴同步，同时主主轴和副主轴之间存在指定的角度偏移“<副主轴位置>”
  <副主轴位置> 以副主轴正转、主主轴 0°位置为基准
  取值范围：0°… 359.999°
  - COUPON(S2,S1,30)

说明

角度偏移在同步激活时仍可修改。

副主轴的定位

即使是在同步激活时，副主轴也可以在 ±180° 的范围内定位，而不受主主轴的影响。

- 采用 SPOS 的主轴定位
  示例： SPOS[2]=IC(-90)
  SPOS 的更多相关信息请参见：
  文档：
  编程手册 基本原理

差速

差速的形成途径有:在主轴同步期间在转速开环模式中重新编程副主轴的旋转方向和转速,另一种方法是通过编写 COUONC 指令来激活主轴同步，该指令会时使副主轴继续采用之前编程的转速：

- COUPONC 激活主轴同步
- S<副主轴>=<转速> [M<副主轴>=<旋转方向>]

  说明

  边界条件
  - 指定旋转方向 M3/M4 时，转速 S...必须要一同重新指定。
  - 只有在使能了叠加后，上述两个指令（M<旋转方向> S<副主轴>、COUPONC）才能形成差速。
  - 主主轴的动态性能应限制在一定范围内，从而避免在副主轴叠加时，超出副主轴的动态性能极限。

  差速的更多相关信息请参见：
  文档：
  功能手册 扩展功能；同步主轴 (S3)

速度、加速度： FA, ACC, OVRA, VELOLIMA

可以采用以下指令来编写副主轴的速度和加速度：

- FA[SPI(S<n>)] 或 FA[S<n>]（轴速度）
- ACC[SPI(S<n>)] 或 ACC[S<n>]（轴加速度）
- OVRA[SPI(S<n>)] 或 OVRA[S<n>]（轴倍率）
- VELOLIMA[SPI(S<n>)] 或 VELOLIMA[S<n>]（轴速度提升或降低）

其中，<n> = 1, 2, 3, ...（副主轴的编号）

文档：
编程手册之基本原理分册

说明

主轴最大急动度的降低或升高不生效。
轴动态性能的更多相关信息请参见：
文档：
功能手册之扩展功能分册；回转轴（R2）

可编程的程序段切换特性 WAITC

通过 WAITC 可以借助不同的同步条件，如粗同步、精同步、IPOSTOP 来规定程序段切换的特性，如：在修改了同步参数或定位过程后。 如果没有规定同步条件，则定义 COUPDEF 时指定的程序段切换特性生效。

示例

- 等待达到副主轴 S2 上的同步条件 FINE，和副主轴 S4 上的 COARSE：
  WAITC(S2,"FINE",S4,"COARSE")
- 等待达到 COUPDEF 中指定的同步运行条件： WAITC( )

撤销同步 COUPOF

通过 COUPOF 可以设定同步的撤销特性：

- 撤销同步，并立即切换程序段：
  - COUPOF(S2,S1)（指定主主轴）
  - COUPOF(S2)（未指定主主轴）
- 在越过指定位置后撤销同步。 越过指定位置后切换程序段。
  - COUPOF(S2, S1, 150)（副主轴位置：150°）
  - COUPOF(S2,S1,150,30)（副主轴位置：150°，主主轴位置：30°）

撤销同步并停止副主轴 COUPOFS

通过 COUPOFS 可撤销同步，并停止副主轴。

- 撤销同步，并停止副主轴，立即切换程序段：
  - COUPOFS(S2,S1)（指定主主轴）
  - COUPOFS(S2)（未指定主主轴）
- 在越过指定位置后撤销同步，并停止副主轴。 越过指定位置后切换程序段。
  - COUPOFS(S2,S1,150)（副主轴位置：150°） 150°)

删除同步 COUPDEL

通过 COUPDEL 可以删除同步：

- COUPDEL(S2,S1)（指定主主轴）
- COUPDEL(S2)（未指定主主轴）

复位同步参数 COUPRES

通过 COUPRES 可以激活机床数据和设定数据中设置的同步数据值：

- COUPRES(S2,S1)（指定主主轴）
- COUPRES(S2)（未指定主主轴）

系统变量

- 副主轴当前的同步状态
  通过以下系统变量可以读取副主轴当前的同步状态，值采用位编码：
  <值> = $AA_COUP_ACT[<副主轴>]

位	<值>	含义
-	0	无耦合有效
2	4	主轴同步有效
提示 ● 其它所有值均针对进给轴模式 ● 如果主轴包含在多个同步组内，则此处返回所有同步组的状态值总和。		

- 当前角度偏移
  通过以下系统变量可以读取副主轴和主主轴之间的角度偏差：
  – $AA_COUP_OFFS[<副主轴>]（设定值角度偏移）
  – $VA_COUP_OFFS[<副主轴>]（实际值角度偏移）
  应用示例
  跟踪运行取消后对 NC 程序中的角度偏移差进行补偿：
  角度偏移差 = 编程的角度偏移 - 系统变量

文档

系统变量的更多详细信息请见：

参数手册之系统变量

13.6 同类耦合 (CP...)

“同类耦合”为通用的耦合功能，已有耦合类型（联动、电导耦合、电子齿轮箱和同步主轴）的所有耦合属性都涵盖其中。

该功能可实现灵活的编程方式：

- 用户可根据自身应用自由选择必要的耦合属性（模块原理）。
- 每个耦合属性都可独立编程。
- 已定义耦合的属性（例如耦合系数）可以修改。
- 日后仍可添加其他耦合属性。
- 跟随轴的参考坐标系（基准坐标系或机床坐标系）也可编程。
- 某些耦合属性也可以在同步动作中编程。
  文档：功能手册 同步动作

说明

到目前为止，联动(TRAIL*)、电导耦合(LEAD*)、电子齿轮箱(EG*)和同步主轴(COUP*)的调用还继续由匹配循环支持。

所有关键字和耦合属性一览

下表中列出了所有同类耦合的关键字和可编程的耦合属性一览：

关键字	耦合属性/含义	句法
CPDEF	创建耦合模块	CPDEF=(<FAx>)
CPDEL	删除耦合模块	CPDEL=(<FAx>)
CPLA	定义一个引导轴	CPLA[<FAx>]=(<LAx>)
CPLDEF	定义引导轴并创建耦合模块 （也可使用 CPDEF + CPLA）	CPLDEF[<FAx>]=(<LAx>) 或者 CPDEF=(<FAx>) CPLA[<FAx>]=(<LAx>)
CPLDEL	删除耦合模块中的引导轴 （也可使用 CPDEL + CPLA）	CPLDEL[<FAx>]=(<LAx>) 或者 CPDEL=(<FAx>) CPLA[<FAx>]=(<LAx>)
CPON	激活耦合模块	CPON=(<FAx>)
CPOF	关闭耦合模块	CPOF=(<FAx>)

关键字	耦合属性/含义	句法		
CPLON	激活耦合模块中的引导轴	CPLON[<FAx>]=<LAx>		
CPLOF	关闭耦合模块中的引导轴	CPLOF[<FAx>]=<LAx>		
CPLNUM	耦合系数分子	CPLNUM[FAx,LAx]=<值>		
CPLDEN	耦合系数分母	CPLDEN[FAx,LAx]=<值>		
CPLCTID	曲线图表的编号	CPLCTID[FAx,LAx]=<值>		
CPLSETVAL	耦合基准	CPLSETVAL[FAx,LAx]="<耦合基准>"		
		"<耦合基准>":	"CMDPOS"	设定值同步
			"CMDVEL"	速度同步
			"ACTPOS"	实际值同步
CPFRS	参考坐标系	CPFRS[FAx]="<参考坐标>"		
		"<参考坐标>":	"BCS"	基准坐标系
			"MCS"	机床坐标系
CPBC	程序段切换标准	CPBC[FAx]="<程序段切换标准>"		
		"<程序段切换标准>":	"NOC"	程序段切换与耦合状态无关。
			"IPOSTOP"	当设定值同步运行时进行程序段切换。
			"COARSE"	当“粗略”实际值同步运行时进行程序段切换。
			"FINE"	当“精细”实际值同步运行时进行程序段切换。
CPFPOS + CPON	激活时跟随轴的同步位置	CPON=FAx CPFPOS[FAx]=<值>		
CPLPOS + CPON	激活时引导轴的同步位置	CPLPOS[FAx,LAx]=<值>		

关键字	耦合属性/含义	句法		
CPFMSON	同步模式	CPFMSON[FAx]="<同步模式>"		
		"<同步模式>":	"CFAST"	在最短的时间内完成耦合。
			"CCOARSE"	当耦合规则所要求的跟随轴位置在当前跟随轴位置的范围内时，耦合才会被激活。
			"NTGT"	在最短的时间内逼近下一个齿槽。
			"NTGP"	以最短的行程逼近下一个齿槽。
			"NRGT"	根据螺纹头数与齿数之比，在最短的时间内逼近下个轮廓段。
			"NRGP"	根据螺纹头数与齿数之比，以最短的行程逼近下个轮廓段。
			"ACN"	仅针对回转轴！ 回转轴以负轴向逼近同步位置。同步立即执行。
			"ACP"	仅针对回转轴！ 回转轴以正轴向逼近同步位置。同步立即执行。
			"DCT"	仅针对回转轴！ 回转轴在最短的时间内逼近所编程的同步位置。同步立即执行。
			"DCP"	仅针对回转轴！ 回转轴以最短的行程逼近所编程的同步位置。同步立即执行。

关键字	耦合属性/含义	句法		
CPFMON	激活时跟随轴的响应方式	CPFMON[FAx]="<激活响应方式>"		
		"<激活响应方式>":	"STOP"	仅针对主轴！ 激活前跟随轴的当前运动将停止。
			"CONT"	仅针对主轴和主运行轴！ 跟随轴/主轴的当前运动将在耦合时被作为起始运动。
			"ADD"	仅针对主轴！ 耦合运动会额外叠加在当前的运动上，即跟随轴/主轴的当前运动会保留为叠加运动。
CPFMOF	完全关闭时跟随轴的响应方式	CPFMOF[FAx]="<关闭响应方式>"		
		"<关闭响应方式>":	"STOP"	跟随轴/主轴停止。 激活的叠加运动将被停止。之后将断开耦合
			"CONT"	仅针对主轴和主运行轴！ 跟随主轴以关闭时刻有效的转速/速度继续运行。
CPFPOS + CPOF	关闭时跟随轴的关闭位置	CPOF=(FAx) CPFPOS[FAx]=<值>		

关键字	耦合属性/含义	句法		
CPMRESET	复位时耦合的响应方式	CPMRESET[FAx]="<复位响应方式>"		
		"<复位响应方式>":	"NONE"	耦合的当前状态不受影响。
			"ON"	如果创建了相应的耦合模块，则会激活耦合。定义的全部引导轴关系都会激活。当全部这些或部分引导轴关系都有效时仍会如此，也就是说即使是完全激活的耦合也会重新进行同步。
			"OF"	激活的叠加运动将被停止。之后将关闭耦合。如果创建相应的耦合模块时未进行明确的定义 (CPDEF)，则该耦合模块将被删除。否则将会保留，即可继续进行使用。
			"OFC"	仅针对主轴！ 跟随主轴以关闭时刻有效的转速/速度继续运行。耦合会被关闭。如果创建相应的耦合模块时未进行明确的定义 (CPDEF)，则该耦合模块将被删除。否则将会保留，即可继续进行使用。
			"DEL"	激活的叠加运动将被停止。此后耦合将被取消激活，然后删除。
			"DELC"	仅针对主轴！ 跟随主轴以关闭时刻有效的转速/速度继续运行。

关键字	耦合属性/含义	句法		
				耦合将被取消激活，然后删除。
CPMSTART	零件程序启动时耦合的响应方式	CPMSTART[FAx]="<启动响应方式>"		
		"<启动响应方式>":	"NONE"	耦合的当前状态不受影响。
			"ON"	激活耦合。定义的全部引导轴关系都会激活。当全部这些或部分引导轴关系都有效时仍会如此，也就是说即使是完全激活的耦合也会重新进行同步。
			"OF"	耦合会被关闭。如果创建相应的耦合模块时未进行明确的定义 (CPDEF)，则该耦合模块将被删除。否则将会保留，即可继续进行使用。
			"DEL"	耦合将被取消激活，然后删除。
CPMPRT	通过程序测试进行程序段查找的情况下零件程序启动时的耦合响应方式	CPMPRT[FAx]="<启动响应方式>"		
		"<启动响应方式>":	参见 CPMSTART	
CPLINTR	引导轴输入值的偏移值	CPLINTR[FAx,LAx]=<值>		
CPLINSC	引导轴输入值的缩放系数	CPLINSC[FAx,LAx]=<值>		
CPLOUTTR	耦合输出值的偏移值	CPLOUTTR[FAx,LAx]=<值>		
CPLOUTSC	耦合输出值的缩放系数	CPLOUTSC[FAx,LAx]=<值>		
CPSYNCOP	“粗略”位置同步运行的阈值	CPSYNCOP[FAx]=<值>		
CPSYNFIP	“精细”位置同步运行的阈值	CPSYNFIP[FAx]=<值>		

关键字	耦合属性/含义	句法		
CPSYNCOP2	“粗略”位置同步运行的第二个阈值	CPSYNCOP2[FAx]=<值>		
CPSYNFIP2	“精细”位置同步运行的第二个阈值	CPSYNFIP2[FAx]=<值>		
CPSYNCOV	“粗略”速度同步运行的阈值	CPSYNCOV[FAx]=<值>		
CPSYNFIV	“精细”速度同步运行的阈值	CPSYNFIV[FAx]=<值>		
CPMBRAKE	特定停止信号和指令下跟随轴的响应方式	CPMBRAKE[FAx]=<位编码的值>		
CPMVDI	对特定 NC/PLC 接口信号的跟随轴响应方式	CPMVDI[FAx]=<位编码的值>		
CPMALARM	抑制耦合专用报警输出	CPMALARM[FAx]=<位编码的值>		
CPSETTYPE	耦合类型	CPSETTYPE[FAx]="<耦合类型>"		
		"<耦合类型>":	"CP"	可自由编程
			"TRAIL"	耦合类型“联动”
			"LEAD"	耦合类型“电导耦合”
			"EG"	耦合类型“电子齿轮箱”
			"COUP"	耦合类型“同步主轴”

FAx:跟随轴/主轴

LAx:引导轴/主轴

说明

未明确编程设置（在零件程序或同步动作中）的耦合属性将使用默认设置。

根据关键字 CPSETTYPE 的设置，也可以取代默认设置(CPSETTYPE="CP")而使用预设的耦合属性。

文档

同类耦合的详细信息请参见：

- 功能手册之特殊功能；M3：轴耦合，章节：“同类耦合”

13.7 主/从耦合 (MASLDEF, MASLDEL, MASLON, MASLOF, MASLOFS)

“主从耦合”如下进行：

- 仅在相关轴为静止状态时，从动轴才可与引导轴耦合。
- 耦合和分离**旋转、已进行转速控制的主轴。**
- 动态配置

说明

定位方式

如果是定位运行方式中的轴和主轴，仅在停止状态中闭合和断开耦合。

句法

```
MASLON(<从动轴_1>,<从动轴_2>,...)
MASLOF(<从动轴_1>,<从动轴_2>,...)
MASLOFS(<从动轴_1>,<从动轴_2>,...)
```

动态配置：

```
MASLDEF(<从动轴_1>,<从动轴_2>,...,<引导轴>)
MASLDEL(<从动轴_1>,<从动轴_2>,...)
```

含义

MASLON:	激活临时主从耦合	
	<从动轴_x>,...:	从动轴 1 ... n
MASLOF:	解除生效的主从耦合	
	<从动轴_1>,...:	从动轴 1 ... n
MASLOFS:	解除主从耦合，并自动制动从动主轴（参见提示“转速控制运行方式中的主轴耦合特性”！）	
	<从动轴_1>,...:	从动轴 1 ... n
MASLDEF:	从零件程序创建/修改主从耦合	
	<从动轴_1>,...:	从动轴 1 ... n
	<引导轴>:	引导轴

MASLDEL:	解除主从耦合，删除主从组合定义	
	<从动轴_1>,...:	从动轴 1 ... n
	提示： 机床数据中配置的主从数据将保留。	

说明

转速控制运行方式中的主轴耦合特性

对于转速控制运行方式中的主轴，其 MASLON、MASLOF、MASLOFS 和 MASLDEL 这些耦合特性通过以下机床数据确定：

MD37263 $MA_MS_SPIND_COUPLING_MODE

在 MD37263 = 0 的默认设置中，仅在参与轴的停止状态中进行从动轴的耦合和脱离。MASLOFS 相当于 MASLOF。

当 MD37263 = 1 时会直接执行耦合指令，在运动中也会执行。使用 MASLON 时会立即建立耦合，使用 MASLOFS 或者 MASLOF 时会立即解耦。使用 MASLOF 时，旋转的从动主轴维持其转速，直至重新编程转速。使用 MASLOFS 时其会自动制动。

说明

当 MASLOF/MASLOFS 时隐式进给停止会消失。受缺少进刀停止的限制，用于从动轴的 $P-系统变量不提供更新值，直至重新编程为止。

说明

对于从动轴而言，可通过 PRESETON 使实际值同步到与引导轴的值相同。为此必须瞬间断开持续的主/从耦合，以便在上电时将尚未找零的从动轴的实际值设定成引导轴的值。然后重新建立持续的耦合。

通过设置以下机床数据激活持续的主从耦合：
MD37262 $MA_MS_COUPLING_ALWAYS_ACTIVE = 1
其对于临时耦合的语言命令没有效用。

示例

示例 1：主/从耦合从动轴的实际值设置

在永久主/从耦合用 PRESETON 将从动轴的实际值设置为主动轴的实际值。

程序代码	注释
$MA_MS_COUPLING_ALWAYS_ACTIVE[AX2]=0	；关闭从动轴的永久耦合
NEWCONF	；激活机床数据更改
STOPRE	

程序代码	注释
MASLOF(Y1)	; 取消临时耦合
PRESETON(AX2,$VA_IM(M_AX))	; 从动轴的实际值=主动轴的实际值
$MA_MS_COUPLING_ALWAYS_ACTIVE[AX2]=1	; 开启从动轴的永久耦合
NEWCONF	; 激活机床数据更改

示例 2：动态配置主/从耦合

为了能在轴容器旋转之后使用另一个主轴来闭合耦合，原有的耦合必须实现断开、删除设计并且设计新的耦合。

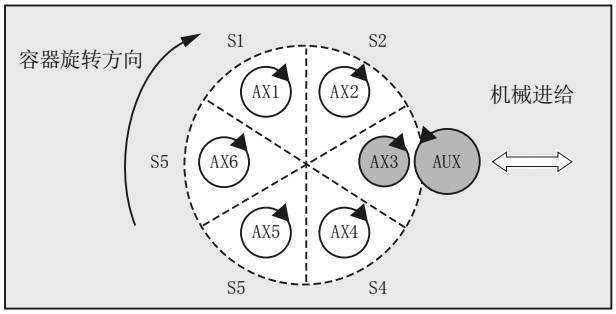


图 13-1　在轴容器旋转之前

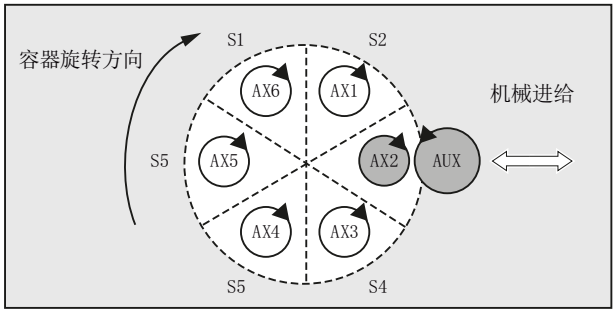


图 13-2　在以某个槽轴容器旋转之后

程序代码	注释
MASLDEF(AUX,S3)	; AUX:从动, S3:主动 = AX3
MASLON(AUX)	; 激活耦合
M3=3 S3=4000	; 主动旋转
MASLDEL(AUX)	; 耦合分离并删除
AXCTSWE(CT1)	; 使能轴容器旋转
MASLDEF(AUX,S3)	; AUX:从动, S3:主动 = AX2

文档

- 功能手册 特殊功能，章节"TE3:主从转速/转矩耦合，主/从连接 “
- 功能手册之扩展功能分册，章节"B3:分散系统 - 仅 840D sl" > "NCU-链接" > "轴容器"

同步动作 14

14.1 定义同步动作

同步动作在零件程序的程序段中定义。在此程序段中不可编写其它不为此同步动作组成部分的指令。

同步动作包含以下部分：

有效性，识别号（可选）	条件部分（可选）			动作部分		
	频率	G 代码（可选）	条件	关键字	G 代码（可选）	动作
--- [1] ID=<编号> IDS=<编号>	--- [1] WHENEVER FROM WHEN EVERY	G...	逻辑表达式	DO	G...	动作 1 ... 动作 n

[1] 未编程

句法

```
DO <动作 1> ... <动作 n>
<频率> [<G 功能>] <条件> DO <动作 1> ... <动作 n>
ID=<编号> <频率> [<G 功能>] <条件> DO <动作 1> ... <动作 n>
IDS=<编号> <频率> [<G 功能>] <条件> DO <动作 1> ... <动作 n>
```

文档

同步动作功能的详细说明请参见：

功能手册 同步动作

摆动 15

15.1 异步摆动(OS, OSP1, OSP2, OST1, OST2, OSCTRL, OSNSC, OSE, OSB)

一个摆动轴在两个换向点 1 和 2 之间以给定的进给来回摆动，直至摆动运动关断。

在摆动运行期间其它的轴可以任意插补。 通过一个轨迹运动或者用一个定位轴，可以达到一个连续进给。 此时在摆动运动和进给运动之间**不存在关系** 。

异步摆动特性

- 在特定的轴上，异步摆动会超出程序段范围生效。
- 通过零件程序可以确保摆动运动和程序段同时启动。
- 几个轴的共同插补和摆动距离的叠加无法进行。

编程

通过以下指令可以从零件程序中启用并控制异步摆动，使它和 NC 程序的执行一致。

编程的值会在主运行中、和程序段同步地输入到相应的设定数据中，在下一次修改前一直保持有效。

句法

```
OSP1[<轴>]=<值> OSP2[<轴>]=<值>
OST1[<轴>]=<值> OST2[<轴>]=<值>
FA[<轴>]=<值>
OSCTRL[<轴>]=(<设置选项>,<复位选项>)
OSNSC[<轴>]=<值>
OSE[<轴>]=<值>
OSB[<轴>]=<值>
OS[<轴>]=1
OS[<轴>]=0
```

含义

<轴>:	摆动轴的名称		
OS:	启动/关闭摆动		
	值:	1	**启动**摆动
		0	**关闭**摆动

OSP1:	确定换向点 1 的位置		
OSP2:	确定换向点 2 的位置 **提示：** 如果处于一个增量运行中，则按照上次 NC 程序中写入的相应换向点增量计算出该位置。		
OST1:	确定换向点 1 的 停止时间，单位秒		
OST2:	确定换向点 2 的 停止时间，单位秒		
	<值>：	-2	继续插补，无需等待准停
		-1	等待粗准停
		0	等待精准停
		>0	等待精准停，并且接着等候指定的停止时间 **提示：** 停止时间的单位与通过 G4 编程的停止时间相同。
FA:	确定进给速度 进给速度指定位轴的定义进给速度。 如果没有定义进给速度，则机床数据中存储的值生效。		

OSCTRL:	指定设置选项和复位选项 选项 0 - 3 规定了取消摆动时换向点上的属性。可以选择 0 - 3 其中的一个类型。 可以根据需要结合所选类型进行其他设置。 通过正号(+)将多个选项相加在一起。		
	<值>：	0	取消摆动运动时停止在下一个换向点处（预设置） **提示：** 只允许通过值 1 和 2 复位。
		1	取消摆动运动时停止在换向点 1 处
		2	取消摆动运动时停止在换向点 2 处
		3	如果没有编程修光行程，则在取消摆动运动时不返回换向点。
		4	在修光后运行到终点位置
		8	由于删除剩余行程而中断了摆动运动后，会紧接着执行修光，并运行到终点位置。
		16	由于删除剩余行程而中断了摆动运动后，会如同取消摆动一样，运行到相应的换向位置。
		32	修改的进给率从下一个换向点开始才有效
		64	FA 等于 0, FA = 0: 位移叠加有效 FA 不等于 0, FA <> 0: 速度叠加有效
		128	在回转轴 DC 时（最短的行程）
		256	两次修光。（标准） 1 = 单次修光。
		512	首先运行到起始位置
OSNSC:	确定修光次数		
OSE:	确定 WCS 中的终点位置，在取消摆动时将运行到该位置 **提示：** 当编程"OSE"时，"OSCTRL"的选项 4 会隐式有效。		
OSB:	确定 WCS 中的起始位置，在开始摆动前轴会运行到该位置 首先到达起始位置，然后是换向点 1。 如果起始位置和换向点 1 相同，则接着运行到换向点 2。 即使起始位置和换向点 1 相同，在达到起始位置时也不会等待“停止时间”，而是等待精准停。 即满足设置的准停条件。 **提示：** 必须在设定数据 SD43770 $SA_OSCILL_CTRL_MASK 中置位数位 9，才能够运行到起始位置。		

示例

示例 1： 摆动轴在两个换向点之间摆动

摆动轴 Z 应该在位置 10 和 100 之间摆动。 换向点 1 应以精准停逼近，换向点 2 以粗准停逼近。 摆动轴的进给为 250。 在加工结束处应当执行 3 次修光行程，并且使得摆动轴达到终点位置 200。 进给轴的进给是 1，在 X 方向的进给在位置 15 处结束。

程序代码	注释
WAITP(X,Y,Z)	; 起始位置。
G0 X100 Y100 Z100	; 转换到定位轴运行方式。
WAITP(X,Z)	
OSP1[Z]=10 OSP2[Z]=100	; 换向点 1，换向点 2。
OSE[Z]=200	; 终点位置。
OST1[Z]=0 OST2[Z]=-1	; 换向点 1 的停留时间： 精准停 ;返回点 2 上的停止时间： 粗准停
FA[Z]=250 FA[X]=1	; 摆动轴进给，进给轴进给。
OSCTRL[Z]=(4,0)	; 设置选项。
OSNSC[Z]=3	; 3 次修光行程。
OS[Z]=1	; 启动摆动。
WHEN $A_IN[3]==TRUE DO DELDTG(X)	; 剩余行程删除。
POS[X]=15	; X 轴初始位置。
POS[X]=50	; X 轴终点位置。
OS[Z]=0	;停止摆动。
M30	

说明

也可以在一个程序段中编程指令串 "OSP1[Z]=..."至"OSNCS[Z]=..."。

示例 2： 带换向位置在线修改的摆动

异步摆动所需的设定数据可以在零件程序中设置。

如果在零件程序中直接定义设定数据，则修改在预处理时就已经生效。 可以通过预处理停止 STOPRE 来实现同步特性。

程序代码	注释
$SA_OSCILL_REVERSE_POS1[Z]=-10	
$SA_OSCILL_REVERSE_POS2[Z]=10	
G0 X0 Z0	
WAITP(Z)	
ID=1 WHENEVER $AA_IM[Z] < $$AA_OSCILL_REVERSE_POS1[Z] DO $AA_OVR[X]=0	; 如果摆动轴的实际值超出换向点，则进给轴停止。
ID=2 WHENEVER $AA_IM[Z] < $$AA_OSCILL_REVERSE_POS2[Z] DO $AA_OVR[X]=0	

程序代码	注释
OS[Z]=1 FA[X]=1000 POS[X]=40	；激活往复。
OS[Z]=0	；取消往复。
M30	

其它信息

摆动轴

对于摆动轴而言：

- 每个轴都可以作为摆动轴使用。
- 而且可以同时有几个摆动轴有效（最大为 定位轴个数）。
- 对于摆动轴而言，直线插补 G1 始终有效，而与程序中当前有效的 G 指令无关。

摆动轴可以作为：

- 动态坐标转换的输入轴
- 使用龙门轴和耦合轴时的引导轴
- 运行：
  - 不带加加速度限制 "BRISK"
    或者
  - 带加加速度限制 "SOFT"
    或者
  - 有曲折的加速特性曲线（如同定位轴）

摆动换向点

在确定摆动位置时必须考虑当前的偏移：

- 绝对尺寸
  "OSP1[Z]=<值>"
  换向点位置 = 偏移 + 编程值之和
- 相对尺寸
  "OSP1[Z]=IC(<值>)"
  换向点位置 = 换向点 1 + 编程值

示例：

程序代码
N10 OSP1[Z] = 100 OSP2[Z] = 110
...
N40 OSP1[Z] = IC(3)

WAITP

如果要用几何轴进行摆动，则必须使用"WAITP"释放该轴进行摆动。

在摆动结束之后，用"WAITP"指令把摆动轴再次定义为定位轴，恢复正常使用。

带有运动同步动作和停止时间的摆动

在已设置的停止时间届满之后，就会在摆动时发生内部程序段转换（可在轴的新剩余行程上看出来）。 在转换程序段时检查关闭功能。 此时会根据已设置的运动过程控制设置(OSCTRL)来确定关闭功能。 *可通过进给修调率来调节这种时间特性。*

在启动修光行程或者向终点位置运动之前，有时还会执行一次摆动行程。 *此时会产生关闭特性发生变化的印象。 但实际上不是这种情况。*

15.2 由同步动作控制的摆动(OSCILL)

就这种摆动方式而言，仅允许在返回点上或者在定义的返回范围内有进给运动。

根据具体的要求，可以在进给期间

- 继续执行摆动运动，或者
- 停止摆动运动，直至进给完全执行。

句法

1. **确定摆动参数**
2. **定义运动同步动作**
3. **分配轴，确定进给**

含义

OSP1[<摆动轴>]=	换向点 1 的位置
OSP2[<摆动轴>]=	换向点 2 的位置
OST1[<摆动轴>]=	在换向点 1 处的保持时间，单位秒
OST2[<摆动轴>]=	在换向点 2 处的保持时间，单位秒
FA[<摆动轴>]=	摆动轴的进给率
OSCTRL[<摆动轴>]=	设定或取消选件
OSNSC[<摆动轴>]=	修光次数
OSE[<摆动轴>]=	终点位置
WAITP(<摆动轴>)	使能用于摆动的轴

轴分配，进给
OSCILL[<摆动轴>]=(<进给轴 1>,<进给轴 2>,<进给轴 3>)
POSP[<进给轴>]=(<终点位置>,<分段长度>,<方式>)

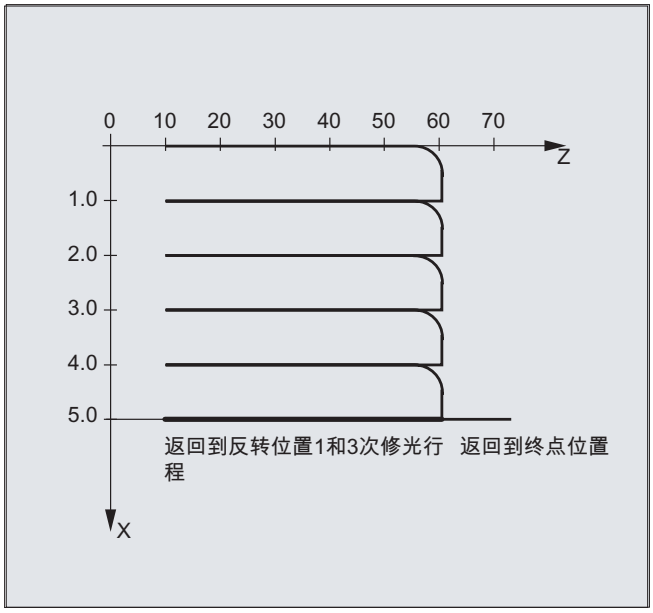
OSCILL:	给摆动轴分配进给轴
POSP：	确定总进给和分段进给（参见文件和程序管理一章）
终点位置:	在所有分段进给结束之后的进给轴的最终位置
分段长度:	换向点/换向区处的分段长度
方式：	总进给运动分为若干个分段进给 = 两个同样大小的剩余步距（缺省设置）； = 所有分段进给同样大

运动同步动作

WHEN… … DO	当…, 则…
WHENEVER … DO	始终当…, 则…

示例

在换向点 1 上，无进给运动。在换向点 2 上，在间隔换向点 2 ii2 处便应开始进给 ，并且在换向点处摆动轴不等待分段进给的结束。轴 Z 为摆动轴且轴 X 为进给轴。



1. 摆动的参数

程序代码	注释
DEF INT ii2	；定义变量，用于换向区 2
OSP1[Z]=10 OSP2[Z]=60	；定义换向点 1 和 2
OST1[Z]=0 OST2[Z]=0	；换向点 1：精准停
	换向点 2：精准停
FA[Z]=150 FA[X]=0.5	；摆动轴 Z 的进给率，进给轴 X 的进给率
OSCTRL[Z]=(2+8+16,1)	；在换向点 2 处取消摆动运动；在 RWL 之后修光并返回终点位置；在 RWL 之后返回相应的换向位置
OSNC[Z]=3	；修光行程
OSE[Z]=70	；终点位置＝ 70
ii2=2	；设定换向区
WAITP(Z)	；允许的摆动，用于 Z 轴

2. 运动同步动作

程序代码	注释
WHENEVER $AA_IM[Z]<$SA_OSCILL_REVERSE_POS2[Z] DO -> $AA_OVR[X]=0 $AC_MARKER[0]=0	; 总是当 MCS 中摆动轴 Z 的当前位置小于换向区 2 的起点时，进给轴 X 的轴向倍率设为 0%，带有索引 0 的标记器设为 0。
WHENEVER $AA_IM[Z]>=$SA_OSCILL_REVERSE_POS2[Z] DO $AA_OVR[Z]=0	; 总是当 MCS 中摆动轴 Z 的当前位置大于等于换向位置 2 时，摆动轴 Z 的轴向倍率设为 0%。
WHENEVER $AA_DTEPW[X]==0 DO $AC_MARKER[0]=1	; 总是当分段进给剩余行程等于 0 时，索引为 0 的标志器设为 1。
WHENEVER $AC_MARKER[0]==1 DO $AA_OVR[X]=0 $AA_OVR[Z]=100	; 总是当索引为 0 的标志器等于 1 时，进给轴 X 的轴向倍率设为 0%。从而防止过早进给（摆动轴 Z 尚未再次离开换向区 2，而进给轴 X 已准备就绪）。将摆动轴 Z 的轴向倍率从 0%（第 2 个同步动作）重新设为 100%。

-> 必须在一个程序段中编程

3. 启动摆动

程序代码	注释
OSCILL[Z]=(X) POSP[X]=(5,1,1)	;启动轴 轴 X 被作为进给轴分配给摆动轴。 轴 X 应以 1 为步距运动至终点位置 5。
M30	; 程序结束

其它信息

1. **确定摆动参数**
   在包含进给轴和摆动轴以及确定进给的运动程序段之前应确定摆动所需的参数（参见“异步摆动”）。
2. **确定运动同步动作**
   通过同步条件实现：
   抑制进给, 直到摆动轴在某个返回范围之内 (ii1, ii2) 或者在某个返回点 (U1, U2) 上时为止。
   摆动运动 在进给过程中**停止**在返回点内。
   摆动运动 在结束部分进给之后重新**启动**. 确定**启动下一个部分进给**。
3. 确定**配置摆动轴和进给轴**以及 **最大进给和部分进给**。

确定摆动参数

配置摆动轴和进给轴： OSCILL

OSCILL[<摆动轴>]=(<进给轴 1>,<进给轴 2>,<进给轴 3>)

使用指令"OSCILL"实现轴配置和启动摆动运动。

一个摆动轴最多分配 3 个横向进给轴。

说明

在启动摆动之前必须已经确定轴性能的同步条件。

确定进给： POSP

POSP[<进给轴>]=(<终点位置>,<分段长度>,<方式>)

使用指令"POSP" 向控制系统发送信息：

- 总的横向进给（通过终点位置）
- 在换向点或者在换向区处其分度横向进给的大小。
- 在到达终点位置时（通过方式）分度横向进给特性

方式＝ 0	对于两个最后的分度横向进给，划分剩余的位移，直至目标点在 2 个相同大小的剩余步（预设定）。
方式＝ 1	所有分度横向进给大小相同。 它们由总的横向进给计算。

确定运动同步动作

后面所执行的运动同步动作完全用于摆动。

您可以找到方案举例，用于满足您作为编制用户专用的摆动运动的模块而提出的各个要求。

说明

在单个情况下，同步条件也可以另外编程。

关键字

WHEN … DO …	当..., 则…
WHENEVER … DO	始终当..., 则…

功能

使用下列详细描述的语言手段可以实现下列功能：

1. 在换向点处的横向进给。
2. 在换向区的横向进给。
3. 在两个换向点处的横向进给。
4. 在换向点处停止摆动运动。

5. 再次启动摆动运动。
6. 分度横向进给不要启动太早。

所有这里举例说明的同步动作适用于以下设定：

- 换向点 1 < 换向点 2
- Z = 摆动轴
- X = 横向进给轴

说明

更详细的解释可参阅运动同步动作一章。

确定配置摆动轴和进给轴以及确定最大进给和部分进给。

在换向区的横向进给。

在到达返回点之前，进给运动应当在某个返回范围内开始。

该同步动作阻止横向进给运动，直至摆动轴位于一个换向区之内。

在所给定的假设条件下（见上面）产生以下的指令：

换向区 1：

```
WHENEVER $AA_IM[Z]>$SA_OSCILL_RESERVE_POS1[Z]+ii1 DO $AA_OVR[X] = 0
```

总是当 MCS 中的摆动轴的当前位置大于等于换向区 1 开始时，设置摆动轴的轴向倍率为 0%。

换向区 2：

```
WHENEVER $AA_IM[Z]<$SA_OSCILL_RESERVE_POS2[Z]+ii2 DO $AA_OVR[X] = 0
```

总是当 MCS 中的摆动轴的当前位置小于换向区 2 开始时，设置进给轴的轴向倍率为 0%。

在换向点处的横向进给

只要摆动轴没有到达换向点，就不进行横向进给轴的运动。

在所给定的假设条件下（见上面）产生以下的指令：

换向区 1：

```
WHENEVER $AA_IM[Z]<>$SA_OSCILL_RESERVE_POS1[Z] DO $AA_OVR[X]=0
$AA_OVR[Z]=100
```

总是当 MCS 中的摆动轴 Z 的当前位置大于或小于换向点 1 的位置时，进给轴 X 的轴向倍率设置为 0%，摆动轴 Z 的轴向倍率设置为 100%。

换向区 2：

用于换向点 2：

```
WHENEVER $AA_IM[Z]<>$SA_OSCILL_RESERVE_POS2[Z] DO $AA_OVR[X]=0
$AA_OVR[Z]=100
```

总是当 MCS 中的摆动轴 Zu 的当前位置大于或小于换向点 2 的位置时，进给轴 X 的轴向倍率设置为 0%，摆动轴 Z 的轴向倍率设置为 100%。

在换向点处停止摆动运动

摆动轴在返回点上停住，同时开始进给运动。 如果横向进给运动完全执行，则继续执行摆动运动。

如果横向进给运动通过一个事先进行的、仍然有效的同步动作停止，则可以同时使用该同步动作，启动横向进给运动。

在所给定的假设条件下（见上面）产生以下的指令：

换向区 1：

```
WHENEVER $SA_IM[Z]==$SA_OSCILL_RESERVE_POS1[Z] DO $AA_OVR[X]=0
$AA_OVR[Z]=100
```

总是当 MCS 中的摆动轴当前位置等于换向位置 1 时，摆动轴的轴向倍率设置为 0%，进给轴的轴向倍率设置为 100%。

换向区 2：

```
WHENEVER $SA_IM[Z]==$SA_OSCILL_RESERVE_POS2[Z] DO $AA_OVR[X]=0
$AA_OVR[Z]=100
```

总是当 MCS 中的摆动轴 Zu 当前位置等于换向位置 2 时，摆动轴的轴向倍率设置为 0%，进给轴的轴向倍率设置为 100%。

换向点的在线计算

如果在对比的右侧有一个使用 `$$` 标识的主过程变量，那么就会在 IPO 节拍中对这两个变量进行连续分析和相互比较。

说明

对此更多的信息参见“运动同步动作”章节。

再次启动摆动运动

如果分度横向进给运动已经结束，则使用同步动作，继续摆动轴的运动。

在所给定的假设条件下（见上面）产生以下的指令：

```
WHENEVER $AA_DTEPW[X]==0 DO $AA_OVR[Z] = 100
```

总是当 WCS 中进给轴 X 的零件进给的剩余行程等于零时，摆动轴的轴向倍率设置为 100％。

下一个分度横向进给

在结束进给之后，必须防止过早启动下一个部分进给。

为此可使用特定通道的标记 ($AC_MARKER[Index])，该标记在部分进给结束处 (部分剩余行程 ≡ 0) 被设定并且在离开返回范围时被删除。 然后用一个同步动作阻止下一个横向进给运动。

在所给定的假设条件下（见上面）产生给返回点 1 的以下指令：

1. 设定标记：

```
WHENEVER $AA_DTEPW[X]==0 DO $AC_MARKER[1]=1
```

总是当 WCS 中进给轴 X 的零件进给的剩余行程等于零时，带索引 1 的标记设置为 1。

2. 删除标记

```
WHENEVER $AA_IM[Z]<> $SA_OSCILL_RESERVE_POS1[Z] DO $AC_MARKER[1] = 0
```

总是当 MCS 中摆动轴 Z 的当前位置大于或小于换向点 1 的位置时，标记 1 设置为 0。

3. 阻止进给

```
WHENEVER $AC_MARKER[1]==1 DO $AA_OVR[X] = 0
```

总是当标记 1 相等时，进给轴 X 的轴向倍率设置为 0％。

冲裁和步冲 16

16.1 激活/取消

16.1.1 激活或取消冲压和步冲 (SPOF, SON, PON, SONS, PONS, PDELAYON, PDELAYOF, PUNCHACC)

激活/取消冲压或步冲

通过 PON 和 SON 可以激活冲压或步冲功能。而 SPOF 会结束所有与此相关的功能。模态有效的指令 PON 和 SON 相互关闭，即：PON 取消 SON，且 SON 取消 PON。

带预应力的冲压/步冲

SONS 和 PONS 功能同样启用冲压和步冲功能。

与 SON/PON 生效时插补平面上冲程控制相反，这两个功能采用了信号技术来控制伺服平面上的冲程释放。由此可以用更高的冲程频率和更高的冲压功率加工。

在分析预应力中的信号时，所有会引起冲压/步冲轴位置改变的功能都被锁定，如使用手轮手动运行、通过 PLC 修改框架、测量功能等。

冲压延迟

PDELAYON 会延迟冲压冲程的输出。模态有效的指令具有经过预处理的功能，一般在 PON 之前。在 PDELAYOF 之后，继续正常冲压。

说明

延迟时间在设定数据 SD42400 $SC_PUNCH_DWELLTIME 中设置。

位移决定的加速度

通过 PUNCHACC 可以确定一条加速度特性曲线，它根据孔距定义了不同的加速度。

第二个冲压接口

如果机床需要换用第二个冲压接口（第二个冲压单元或类似的媒介），可以切换到控制系统上第二对高速数字输入输出端（I/O 对）。所有的冲压功能可用于这两个接口。通过指令 SPIF1 和 SPIF2 可以在第一个冲压接口和第二个冲压接口之间转换。

说明

前提条件：必须已经通过机床数据定义了第二个用于冲压功能的 I/O 对（(→ 参见机床制造商的说明！）。

句法

```
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
```

含义

PON:	激活冲压。	
SON:	激活步冲	
PONS:	激活带预应力的冲压	
SONS:	激活带预应力的步冲	
SPOF:	取消冲压/步冲	
PDELAYON:	激活冲压延迟	
PDELAYOF:	取消冲压延迟	
PUNCHACC:	激活行程控制式加速度 参数：	
	<Smin>	最小孔距
	<Amin>	开始加速度 <Amin> 可以大于 <Amax>。
	<Smax>	最大孔距
	<Amax>	结束加速度 <Amax> 可以大于 <Amin>。

SPIF1:	激活**第一个**冲压接口 冲程控制由第一对高速 I/O 实现。
SPIF2:	激活**第二个**冲压接口 冲程控制由第二对高速 I/O 实现。
	提示： 复位或控制系统启动后，首个冲压接口始终有效。如果只使用一个冲压接口，则无需编程。

示例

示例 1：激活步冲

程序代码	注释
...	
N70 X50 SPOF	；定位，不释放冲压。
N80 X100 SON	；激活步冲，在运行前(X=50)和编程的运行结束(X=100)后释放冲程
...	

示例 2：冲压延迟

程序代码	注释
...	
N170 PDELAYON X100 SPOF	；不带冲程释放的定位，激活延迟的冲程释放
N180 X800 PON	；激活冲裁。到达终点位置后，冲程延迟释放。
N190 PDELAYOF X700	；取消延迟冲裁，标准冲程释放生效
...	

示例 3：带两个冲压接口的冲压

程序代码	注释
...	
N170 SPIF1 X100 PON	；程序段结束时释放第一个高速输出端上的冲程。监控第一个输入端上的信号“冲程有效”。
N180 X800 SPIF2	；第二次冲程释放在第二个高速输出端上进行。监控第二个输入端上的信号“冲程有效”。
N190 SPIF1 X700	；所有后续的冲程控制都由第一个接口实现。
...	

更多信息

冲压和步冲，带预应力(PONS/SONS)

不可以同时在几个通道中进行带预应力的冲压和步冲。PONS 或 SONS 仅可以在各自的通道中激活。

位移控制式加速度(PUNCHACC)

示例：

```
PUNCHACC(2,50,10,100)
```

2 毫米以下的孔距：

以最大加速度的 50% 运行。

孔距在 2 毫米到 10 毫米之间：

该加速度与距离成正比提高到 100%。

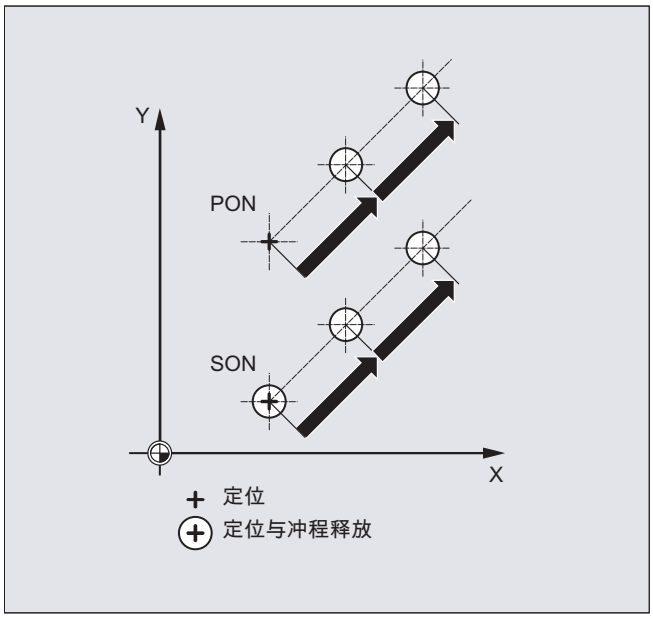
孔距大于 10 毫米：

以 100%的加速度运行。

释放第一个冲程

在步冲和冲压时，在激活该功能后释放第一个冲程在时间上不同。

- PON/PONS:
  - 所有的冲程 — 即使在激活之后第一个程序段的冲程 — 在程序段结束处发生。
- SON/SONS:
  - 在激活步冲之后，在程序段开始处已经发生第一个冲程。
  - 所有的其它冲程在程序段结束处释放。



立即进行冲压和步冲

只有当程序段中包含冲压轴或者步冲轴（有效平面的轴）的运行信息时，才释放一个冲程。

为了要在相同的地点释放一个冲程，用运行位移 0 编程一个冲压轴/步冲轴。

用可旋转的刀具工作

说明

为了可以在编程的轨迹处使用可旋转的刀具，请使用切线控制。

M 指令的应用

利用宏指令技术，仍可以用专用 M 功能替代语言指令使用（兼容性）。其中，和旧系统的对应关系如下：

```
M20, M23  ≙  SPOF
M22       ≙  SON
M25       ≙  PON
M26       ≙  PDELAYON
```

宏指令文件的示例：

程序代码	注释
DEFINE M25 AS PON	; 冲裁 开

程序代码	注释
DEFINE M125 AS PONS	；激活带预应力的冲压
DEFINE M22 AS SON	；步冲 开
DEFINE M122 AS SONS	；激活带预应力的步冲
DEFINE M26 AS PDELAYON	；激活延迟冲压
DEFINE M20 AS SPOF	；关闭冲压,步冲
DEFINE M23 AS SPOF	；关闭冲压,步冲

编程示例：

程序代码	注释
...	
N100 X100 M20	；定位，不释放冲压。
N110 X120 M22	；激活步冲，在运行前后释放冲程。
N120 X150 Y150 M25	；激活步冲，在运行结束后释放冲程。
...	

16.2 自动划分位移

部分区间再分

冲裁或步冲激活时，SPP 和 SPN 都会导致将轨迹轴的总运行区间划分为等长的部分区间（等长位移划分）。在内部每个部分区间都对应一个程序段。

冲程数量

冲裁时首个冲程在首个部分区间的终点时执行，相反步冲时在首个部分区间的起始点。从总运行区间会得到以下数量：

冲裁：冲程数 = 部分区间数

步冲：冲程数 = 部分区间数 + 1

辅助功能

辅助功能在生成的首个程序段中执行。

句法

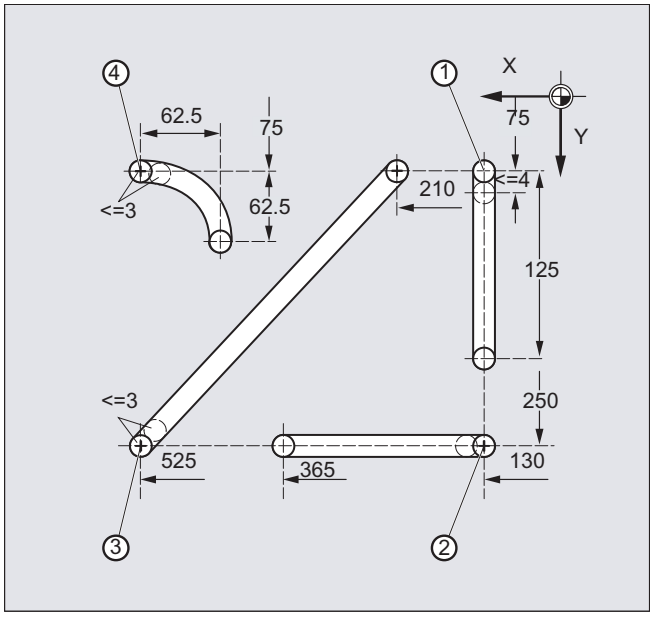
```
SPP=
SPN=
```

含义

SPP:	部分区间大小（最大冲程间距）；模态有效
SPN:	每个程序段的部分区间数；程序段方式有效

示例 1

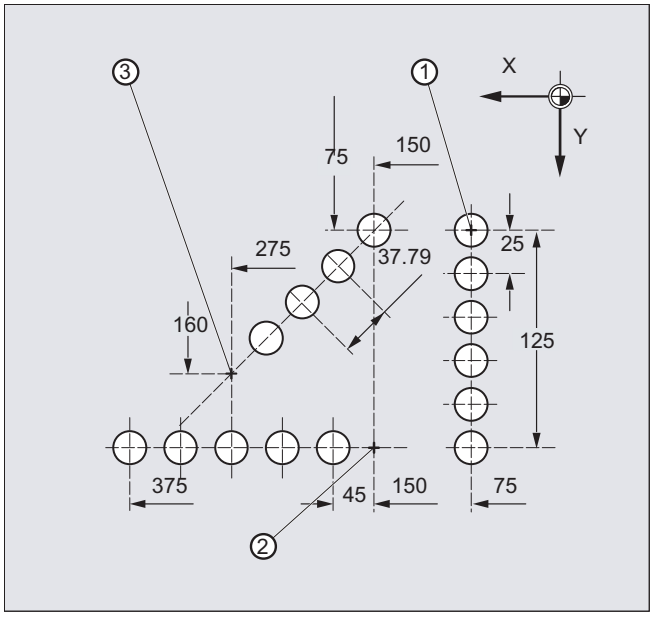
已编程的步冲区间要自动划分为等长的部分区间。



程序代码	注释
N100 G90 X130 Y75 F60 SPOF	；定位在起始点 1
N110 G91 Y125 SPP=4 SON	；打开步冲；自动位移分配最大部分区间位移：4 mm
N120 G90 Y250 SPOF	；关闭步冲；定位在起始点 2
N130 X365 SON	；打开步冲；自动位移分配最大部分区间位移：4 mm
N140 X525 SPOF	；关闭步冲；定位在起始点 3
N150 X210 Y75 SPP=3 SON	；打开步冲；自动位移分配最大部分区间位移：3 mm
N160 X525 SPOF	；关闭步冲；定位在起始点 4
N170 G02 X-62.5 Y62.5 I J62.5 SPP=3 SON	；打开步冲；自动位移分配最大部分区间位移：3 mm
N180 G00 G90 Y300 SPOF	；关闭步冲

示例 2

对于独立的排孔要进行自动位移划分。划分时要分别定义最大部分区间长度 (SPP 值）。



程序代码	注释
N100 G90 X75 Y75 F60 PON	；定位在起始点 1； 打开冲裁，冲裁单个孔
N110 G91 Y125 SPP=25	； 自动位移分配最大部分区间长度：25 mm
N120 G90 X150 SPOF	；关闭冲裁；定位在 起始点 2
N130 X375 SPP=45 PON	；打开冲裁；自动位移分配最大部分区间位移 ：45 mm
N140 X275 Y160 SPOF	；关闭冲裁；定位在 起始点 3
N150 X150 Y75 SPP=40 PON	；打开冲裁；使用计算的部分区间长度 37.79 mm 替代编程的部分区间长度 40 mm。
N160 G00 Y300 SPOF	；关闭冲裁；定位

16.2.1 轨迹轴位移划分

分度距离 SPP 的长度

使用 SPP 说明最大的冲程距离和分度距离的最大长度，按照该分度距离对总的运行距离进行划分。通过 SPOF 或者 SPP=0 关断该指令。

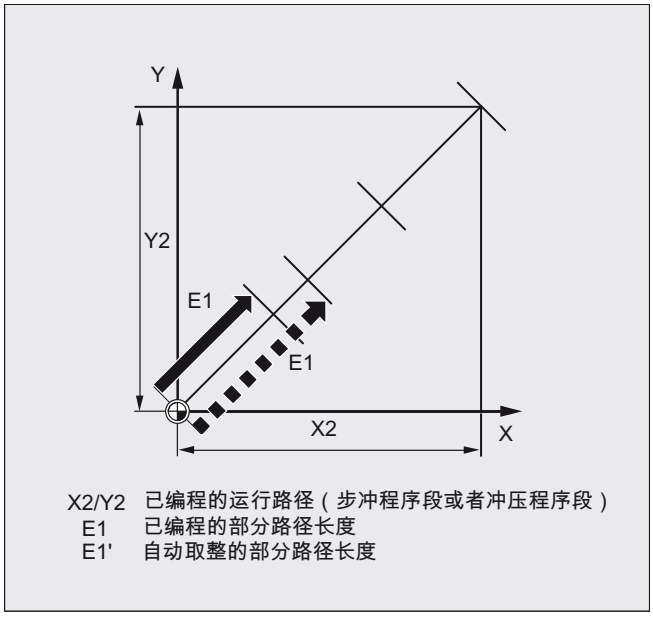
示例：

```
N10 SON X0 Y0
N20 SPP=2 X10
```

10 毫米的总的运行距离划分为 5 个分度距离，每段 2 毫米(SPP=2) 。

说明

用 SPP 划分位移总是进行等距离地划分：所有的分度距离长度相同。也就是说，只有当总的运行距离与 SPP 值的商为整数时，编程的分度距离大小（ SPP 值）才有效。如果不是这种情况，则分度距离的大小在内部减少，从而产生一个整数的商。



X2/Y2 已编程的运行路径（步冲程序段或者冲压程序段）
E1 已编程的部分路径长度
E1' 自动取整的部分路径长度

示例：

```
N10 G1 G91 SON X10 Y10
N20 SPP=3.5 X15 Y15
```

如果总的运行距离为 15 毫米，并且分度距离的一个长度为 3.5 毫米，则不会产生一个整数商（4.28）。因此降低 SPP 值，直至下一个可能的整数商。在这种情况下产生一个 3 毫米的分度距离长度。

分度距离 SPN 的个数

使用 SPN 您可以定义分度距离的个数，它应该由总的运行距离产生。分度距离的长度会自动计算出来。因为 SPN 为程序段方式生效，所以在事先必须激活冲压或者步冲，使用 PON 或者 SON 。

16.2.2 在单个轴时的位移划分

如果除了轨迹轴之外也有单个轴作为冲压－步冲－轴定义，则它们可能也会进行自动位移划分。

SPP 时单轴的特性

编程的部分区间长度（SPP）完全以轨迹轴为参考。因此在一个程序段中，除了单轴运行和 SPP 值之外没有编程轨迹轴时，则忽略 SPP 值。

如果不仅编程了单轴，而且在程序段中也编程了轨迹轴，则单轴的特性取决于相应机床数据的设定。

1. 默认设置
   单轴的位移均匀地划分到用 SPP 生成的中间程序段中。

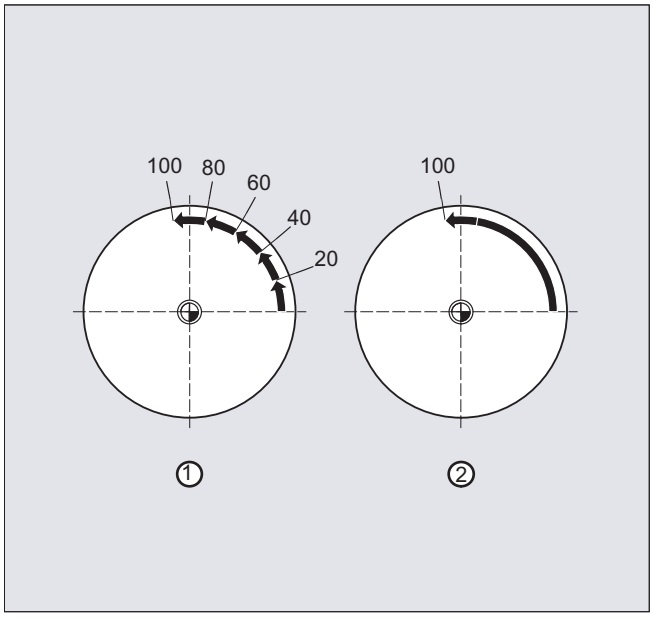
示例：

```
N10 G1 SON X10 A0

N20 SPP=3 X25 A100
```

按照 3 毫米的冲程距离，在 X 轴（轨迹轴）15 毫米的总运行区间中生成 5 个程序段。

因此 A 轴在每个程序段中转动 20 度。



1. 无位移划分的单轴
   单轴在生成的首个程序段中运行其总位移。
2. 不同的位移划分
   单轴的特性与轨迹轴的插补相关：

- 圆弧插补：位移划分
- 线性插补：无位移划分

SPN 时的特性

即使不是同时编程一个轨迹轴，则编程的部分区间数也有效。

前提条件：单轴定义为冲裁－步冲轴。

磨削 17

17.1 激活/取消磨削专用的刀具监控 (TMON, TMOF)

通过预定义程序 TMON(...) 和 TMOF(...) 激活或取消磨削专用的刀具监控（几何和转速监控）。

前提条件

必须设置刀具专用参数 $TC_TPG1 到 $TC_TPG9。

句法

```
TMON(<T 编号>)
...
TMOF(<T 编号>)
```

含义

TMON(...):	激活磨削专用的刀具监控 该指令必须在应激活磨削专用的刀具监控的通道中编程。
TMOF(...):	取消磨削专用的刀具监控 该指令必须在应取消磨削专用的刀具监控的通道中编程。
<T 编号>:	T 编号 **提示：** 仅当激活或关闭未生效的磨削砂轮，而非生效且在使用中的刀具监控时，才需要给定该数据。
TMOF(0):	取消所有刀具的监控

其它功能 18

18.1 轴功能 (AXNAME, AX, SPI, AXTOSPI, ISAXIS, AXSTRING, MODAXVAL)

当未识别轴的名称时，例如，在设置一般有效循环时使用"AXNAME" 。

"AX"用于几何轴和同步轴的间接编程。 此时轴名称存放在一个类型 AXIS 的变量中或者由指令如 "AXNAME" 或 "SPI" 提供。

当轴功能用于一个主轴，例如同步主轴编程时，使用"SPI" 。

使用"AXTOSPI"，可将一个轴名称转换到另一个主轴索引中 (转换功能针对"SPI")。

使用"AXSTRING"，可将一个轴名称(数据类型 AXIS) 转换到一个字符串中 (转换功能针对 "AXNAME")。

在一般有效循环中使用"ISAXIS" ，以确保某个指定的几何轴存在，并由 $P_AXNX 安全中断随后的调用。

使用"MODAXVAL"，可以在模数回转轴时确定模数位置。

句法

```
AXNAME("字符串")
AX[AXNAME("字符串")]
SPI（n）

AXTOSPI(A) 或 AXTOSPI(B) 或 AXTOSPI(C)
AXSTRING（SPI（n））
ISAXIS(<几何轴号>)
<模数位置>=MODAXVAL(<轴>,<轴位置>)
```

含义

AXNAME:	如果将输入字符串转换为轴标识符；输入字符串必须包含一个有效的轴名称。
AX:	可变轴标识符
SPI:	将主轴编号转换为轴名称；转换参数必须包含一个有效的主轴编号。
n:	主轴号
AXTOSPI:	将轴标识符转换为一个整数型主轴索引。 "AXTOSPI"相当于 "SPI" 的转换功能。

X, Y, Z：	AXIS 型的轴标识符作为变量或常量
AXSTRING:	输出带所分配主轴号的字符串。
ISAXIS:	检查是否存在规定的几何轴。
MODAXVAL:	在模数回转轴时确定模数位置：这符合模数余数，与参数化的模数范围有关（在标准设置下为 0 至 360 度；通过 MD30340 MODULO_RANGE_START 和 MD30330 $MA_MODULO_RANGE 可以改变模数范围的起始值和大小）。

说明

SPI 扩展

轴功能 SPI（n）也可用于读取和写入框架组件。 为此框架可以例如通过句法 $P_PFRAME[SPI(1),TR]=2.22 写入。

通过附加编程轴位置，通过地址 AX[SPI(1)]=<轴位置> 可以运行一根轴。 前提是主轴位于定位运行或者轴运行。

示例

示例 1： AXNAME, AX, ISAXIS

程序代码	注释
OVRA[AXNAME（“横向轴”）]=10	；横行轴倍率
AX[AXNAME（“横向轴”）]=50.2	；横向轴的终点位置
OVRA[SPI（1）]=70	；主轴 1 的倍率
AX[SPI(1)]=180	；主轴 1 的终点位置
IF ISAXIS(1) == FALSE GOTOF WEITER	；有横坐标?
AX[$P_AXN1]=100	；运行横坐标
继续:	

示例 2：AXSTRING

在使用 AXSTRING[SPI(n)] 编程时，不再将分配给主轴的轴索引作为主轴号输出，而是输出字符串“Sn”。

程序代码	注释
AXSTRING[SPI(2)]	；输出字符串“S2”。

示例 3：MODAXVAL

应该确定模数回转轴的模数位置 A。

计算输出值是轴位置 372.55。

参数化的模数范围为 0 至 360 度：

MD30340 MODULO_RANGE_START = 0

MD30330 $MA_MODULO_RANGE = 360

程序代码	注释
R10=MODAXVAL(A,372.55)	；计算的模数位置 R10 = 12.55。

示例 4：MODAXVAL

如果编程的轴名称不涉及到模数回转轴，则保持不变返回要转换的值 (<轴位置>)。

程序代码	注释
R11=MODAXVAL(X,372.55)	；X 是直线轴；R11 = 372.55。

18.2 可转换的几何轴 (GEOAX)

“可转换的几何轴”功能可使用其他通道轴替代通过机床数据设定的几何轴。

句法

```
GEOAX(<n>,<通道轴>,<n>,<通道轴>,<n>,<通道轴>)
GEOAX()
```

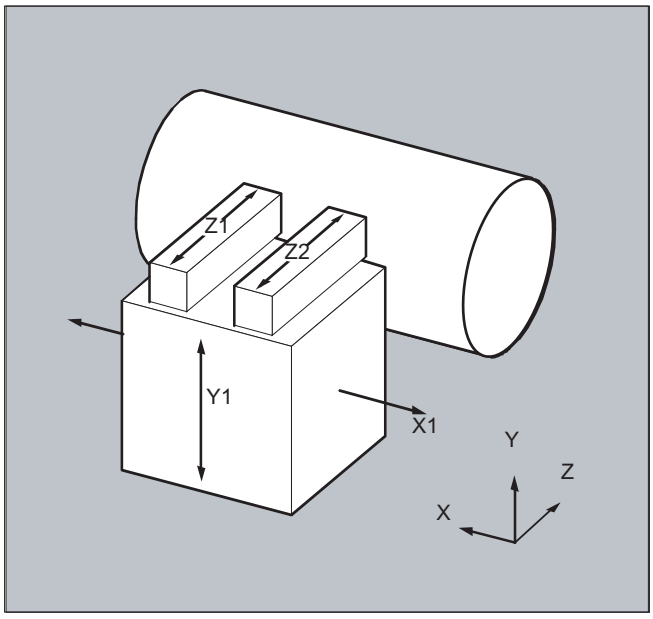
含义

GEOAX(...)	几何轴切换功能。 **提示：** 不设置参数的 GEOAX() 可再次激活在机床数据中设定的几何轴的基本配置。
<n>	由指定通道轴替代的几何轴的编号。 取值范围： 0, 1, 2, 3 **提示：** 0: 指定的通道轴不进行替换，从几何轴组中删除 1: 1. 几何轴 ≙ WCS 的坐标轴 X (横坐标) 2: 2. 几何轴 ≙ WCS 的坐标轴 Y (纵坐标) 3: 3. 几何轴 ≙ WCS 的坐标轴 Z (第三轴)
<通道轴>	通道轴的名称，这个通道轴应被纳入几何轴组中去。

示例

示例 1： 以切换方式切换两根轴作为几何轴

刀具溜板可以通过通道轴 X1，Y1，Z1，Z2 来运行：



这样设计几何轴，在打开后首先使 Z1 作为第 3 几何轴以几何轴名称“Z”有效，并与 X1 和 Y1 形成几何轴组合。

在零件程序中可以使用轴 Z1 和 Z2 交替地作为几何轴 Z：

程序代码	注释
...	
N100 GEOAX (3,Z2)	; 通道轴 Z2 作为第 3 几何轴 (Z) 起作用。
N110 G1 ...	
N120 GEOAX (3,Z1)	; 通道轴 Z1 作为第 3 几何轴 (Z) 起作用。
...	

示例 2： 在 6 通道轴时切换几何轴

机床具有 6 通道轴，名称分别是 XX, YY, ZZ, U, V, W。

几何轴配置基本设置通过机床数据实现：

通道轴 XX = 第 1 几何轴（X 轴）

通道轴 YY = 第 2 几何轴（Y 轴）

通道轴 ZZ = 第 3 几何轴（Z 轴）

程序代码	注释
N10 GEOAX()	;几何轴的基本配置有效。
N20 G0 X0 Y0 Z0 U0 V0 W0	;所有轴快速运动到位置 0.
N30 GEOAX(1,U,2,V,3,W)	;通道轴 U 成为第一个 (X), V 成为第二个 (Y), ;W 成为第三个 (Z)。

程序代码	注释
N40 GEOAX(1,XX,3,ZZ)	;通道轴 XX 成为第一个 (X), ZZ 成为第三个
	; 几何轴 (Z)。 通道轴 V 仍然是第二个
	; 几何轴(Y)。
N50 G17 G2 X20 I10 F1000	; 在 X/Y 平面中的整圆。 运行
	; 通道轴 XX 和 V。
N60 GEOAX(2,W)	;通道轴 W 成为第二个几何轴 (Y).
N80 G17 G2 X20 I10 F1000	; 在 X/Y 平面中的整圆。 运行
	; 通道轴 XX 和 W。
N90 GEOAX()	; 返回到初始状态。
N100 GEOAX(1,U,2,V,3,W)	; 通道轴 U 成为第一个几何轴 (X), V 成为第二个
	; (Y), W 成为第三个 (Z)。
N110 G1 X10 Y10 Z10 XX=25	; 通道轴 U、V、W 各自向
	; 位置 10 行驶。作为附加轴的 XX 向位置 25 行驶。
N120 GEOAX(0,V)	; 从几何轴组中去掉 V。
	; U 和 W 仍然为第一个 (X) 和第三个
	; 几何轴 (Z)。
	; 第二个几何轴 (Y) 保持未配置状态。
N130 GEOAX(1,U,2,V,3,W)	; 通道轴 U 仍然为第一个几何轴 (X), V 为第二个
	; (Y), W 为第三个 (Z)。
N140 GEOAX(3,V)	; V 为第三个几何轴 (Z), 同时
	; 覆盖 W 并从几何轴组中
	; 删除。 第二个几何轴 (Y)
	; 保持之前的未配置状态。

机床数据

轴配置

几何轴、附加轴和机床轴与通道轴之间的对应关系：

- MD10000 $MN_AXCONF_MACHAX_NAME_TAB
- MD20050 $MC_AXCONF_GEOAX_ASIGN_TAB
- MD20060 $MC_AXCONF_GEOAX_NAME_TAB
- MD20070 $MC_AXCONF_MACHAX_USED
- MD20080 $MC_AXCONF_CHANAX_NAME_TAB
- MD35000 $MA_SPIND_ASSIGN_TO_MACHAX

复位特性

更改过的几何轴分配的复位特性：

- MD20110 $MC_RESET_MODE_MASK，位 12
- MD20118 $MC_GEOAX_CHANGE_RESET

NC 启动特性

- MD20112 $MC_START_MODE_MASK, 位 12

通知 PLC 用户程序

几何轴切换时，NC/PLC 接口上输出的 M 指令的参数设置方式：

- MD22532 $MC_GEOAX_CHANGE_M_CODE

边界条件

无法进行几何轴切换

- 以下功能中有一个生效时，都无法进行几何轴切换：
  - 坐标转换
  - 样条插补
  - 刀具半径补偿
  - 刀具精补偿
- 几何轴与另一个通道轴名称相同。
- 几何轴切换的其中一个轴参与了另一个动作，该动作的持续时间超出了程序段范围。 例如跨程序段的定位轴或耦合轴对中的跟随轴。

回转轴

回转轴不能作为几何轴。

更换后轴状态

一个由几何轴组合中的转换替代的轴在转换过程之后，通过它们通道轴名称作为附加轴可编程。

框架，保护范围，工作区域限制

所有的框架，保护范围和工作范围限制都可以用几何轴转换来删除。

极坐标

使用 GEOAX 交换几何轴会向一个平面转换一样 (用 G17-G19) 将模态极坐标设定成数值 0。

DRF, NPV

一个可能发生的手轮偏移（DRF）或者一个外部的零点偏移（NPV）在转换之后依旧有效。

几何轴基本配置

指令 GEOAX() 用来调用几何轴组合的基本配置。

在上电后并且在转换到“参考点运行方式”时将自动转换回基本配置。

刀具长度补偿

一个当前有效的刀具长度补偿在转换过程之后也是有效的。 尽管如此，它对新接纳或者交换位置的几何轴仍然有效，当它们还没有运行时。 使用第一个针对这些几何轴的运行指令时，生成的运动行程相应的由刀具长度补偿的总和与编程设计的运动行程组成。

在转换时在轴组合中保持自身位置的几何轴，也保持其状态，包括刀具长度补偿。

激活转换时几何轴配置

- 在一个有效的转换中所适用的几何轴配置（通过转换机床数据确定），无法通过功能“可切换的几何轴”来更改。
- 适用于一个转换的不同的几何轴配置必须在转换机床数据的不同数据组中进行参数设置。
- 一个通过 GEOAX 更改的几何轴配置可通过激活一个转换来删除。
- 对于几何轴而言，激活的转换的专用几何轴参数设置优先于几何轴切换相关的参数设置。
  示例： 转换生效。 根据机床数据，转换在通道复位时仍然保留。 但同时在通道复位时还会生成几何轴的基本配置。 为转换所确定的几何轴配置会被保留。
- 关闭转换后，几何轴的基本配置会再次生效。

运行方式 JOG，机床功能 REF

切换到运行方式 JOG，机床功能 REF（回参考点运行）时，机床数据中设置的几何轴配置将生效。

18.3 轴容器 (AXCTSWE， AXCTSWED，AXCTSWEC)

使用指令"AXCTSWE" 及 "AXCTSWED"可使能指令轴容器的旋转。

通过"AXCTSWEC"指令可撤销轴容器旋转使能。

句法

```
AXCTSWE(<ID>)
AXCTSWED(<ID>)
AXCTSWEC(<ID>)
```

含义

AXCTSWE:	使能轴容器旋转 "AXCTSWE"指令不会中断程序处理。 轴容器所涉及的通道均进行使能后，会立即执行旋转。	
AXCTSWED:	使能轴容器旋转，不考虑其它轴容器涉及的通道 **提示** ● 此指令用于简化零件程序或同步动作的调试。 ● 与轴容器涉及的其它通道相关的特性可通过以下机床数据设定：MD12760 $MN_ AXCT_FUNCTION_MASK，位 0	
AXCTSWEC:	撤销轴容器旋转使能 **提示** 只有在轴容器还没有开始旋转时，才能撤销旋转使能： $AN_AXCTSWA[<轴容器>] == 0 系统变量参见“轴容器 (AXCTSWE， AXCTSWED，AXCTSWEC) (页 693)”	
<ID>:	轴容器或容器轴的名称：	
	CT<编号>:	轴容器的缺省名称： MD12750 $MN_AXCT_NAME_TAB 示例： "CT1"
	<容器>:	轴容器的用户定义名称： MD12750 $MN_AXCT_NAME_TAB 示例： "CONTAINER_1"
	<轴>:	通道中已知的容器轴的名称

说明

增量

轴容器旋转的增量通过以下设定数据设置：

SD41700 $SN_AXCT_SWWIDTH

其它信息

诊断

通过以下系统变量可读取轴容器的当前状态：

系统变量	类型	说明
$AC_AXCTSWA[<名称>]	BOOL	轴容器特定通道的状态
$AN_AXCTSWA[<轴容器>]	BOOL	轴容器特定 NCU 的状态
$AN_AXCTSWE[<轴容器>]	INT	轴容器旋转特定插槽的状态 系统变量对轴容器插槽的状态以 **位方式**进行说明， 每个位对应一个插槽。
$AN_AXCTAS[<轴容器>]	INT	轴容器当前已经旋转的插槽数。

隐含了 GET/GETD 的轴容器旋转

通过以下机床数据可设置，使用"AXCTSWE"指令时通过隐性 "GET / GETD" 将通道的所有容器轴取出。 在容器旋转后才可进行轴交换。

MD10722 $MN_AXCHANGE_MASK，位 1 = 1

说明

对于“主运行轴”状态下的轴（例如 PLC 轴）**不会**执行带隐性 "GET / GETD" 的轴容器旋转，因为其退出此状态后方可进行轴容器旋转。

18.4 等待有效的轴位置（WAITENC）

在 NC 程序中可编写 "WAITENC"指令等待，由 MD34800 $MA_WAIT_ENC_VALID = 1 配置的轴获得经过同步或补偿的位置。

在等待状态下可执行中断，例如启动一个异步子程序，或切换到 JOG 模式。 必要时可通过继续执行程序重新进入等待状态。

说明

等待状态在操作界面中通过停止状态“等待测量系统”显示。

句法

可在任意 NC 程序部分编程"WAITENC"指令。

必须在单独的程序段中进行编程。

```
...
WAITENC
...
```

示例

"WAITENC" 例如可用于事件控制的用户程序 .../_N_CMA_DIR/_N_PROG_EVENT_SPF，如下面的例子所示。

应用示例： 断电后通过定向转换回退刀具

带刀具定向的加工已由于电源故障中断。
在之后的启动中调用事件控制用户程序 .../_N_CMA_DIR/_N_PROG_EVENT_SPF。

在事件控制用户程序中使用"WAITENC"等待经过同步或补偿的轴位置，从而计算出框架，按刀具方向校准 WCS。

程序代码	注释
...	
IF $P_PROG_EVENT == 4	; 启动。
IF $P_TRAFO <> 0	; 已选择坐标转换。
WAITENC	; 等待有效的方向轴位置。
TOROTZ	; 将 WCS 的 Z 轴转到刀具轴方向。
ENDIF	
M17	
ENDIF	

程序代码	注释
...	

之后可在运行方式 JOG 中，通过回程运行将刀具沿刀具轴的方向退回。

18.5 可编程参数组切换（SCPARA）

使用"SCPARA"指令可为轴请求向特定参数组的切换。

说明

螺纹加工期间不可进行参数组切换

在螺纹切削 G33 和攻丝 G331 / G332 时，参数组已被控制系统选取，故无法修改。

禁用参数组切换

通过 NC/PLC 接口也可请求参数组切换。 为避免切换冲突，可通过 NC/PLC 接口禁用 NC 的参数组切换（SCPARA）：

DB31, ... DBX9.3（通过 NC 进行的参数组设定被禁用）

说明

若在通过 NC/PLC 接口禁用参数组切换功能期间使用 "SCPARA" 请求了该功能，则切换请求会被拒绝，且无故障信息。

句法

```
SCPARA[<轴>]=<值>
```

含义

SCPARA：	指令： 切换参数组	
<轴>：	轴名称（通道轴）	
	类型：	AXIS
<值>：	参数组号： 1, 2, 3, ... 最大参数组号	

示例

程序代码	注释
...	
N110 SCPARA[X]= 3	; 选择： X 轴，第 3 参数组
...	

其它信息

使能参数组切换

轴的参数组切换必须显性使能：

MD35590 $MA_PARAMSET_CHANGE_ENABLE[<轴>]

读取参数组编号

可通过系统变量 $AA_SCPAR 读取所选参数组（设定参数组）的编号。

文档

参数组的更多详细信息请见：

功能手册 基本功能；“速度、设定值/实际值系统、闭环控制（G2）”> “闭环控制” > “位置控制器的参数组”

18.6 检查现有的 NC 语言范围（STRINGIS）

使用"STRINGIS(...)" 功能可以检查，指定的字符串能否在当前语言集中作为 NC 编程语言元素使用。

定义

```
INT STRINGIS(STRING <名称>)
```

句法

```
STRINGIS(<名称>)
```

含义

STRINGIS:	带返回值的函数
<名称>：	待检查的 NC 编程语言元素的名称
返回值：	返回值的格式为 yxx（十进制）。

NC 编程语言元素

可检查以下 NC 编程语言元素：

- 所有现有 G 代码组的 G 代码，例如"G0", "INVCW", "POLY", "ROT", "KONT", "SOFT", "CUT2D", "CDON", "RMBBL", "SPATH"
- DIN 或 NC 地址，例如 "ADIS", "RNDM", "SPN", "SR", "MEAS"
- 功能，例如"TANG(...)" 或"GETMDACT"
- 步骤，例如"SBLOF"。
- 关键字，例如"ACN", "DEFINE" 或"SETMS"
- 系统数据，例如机床数据 $M...，设定数据 $S... 或选项数据 $O...
- 系统变量$A... , $V... , $P...
- 计算参数 R...
- 激活的循环的循环名称
- GUD 和 LUD 变量
- 宏名称
- 标签名称

返回值

返回值仅与前 3 个十进制位相关。返回值的格式为 yxx，其中 y 为基本信息，xx 为详细信息。

返回值	含义	
000	字符串“名称”在现有系统中未知 1)	
100	字符串“名称”是 NC 编程语言元素，但是当前不可编程（选项/功能未生效）	
2xx	字符串“名称”是可编程的 NC 编程语言元素（选项/功能生效）。详细信息 xx 包含了更多元素类型的相关信息：	
	xx	**含义**
	01	DIN 地址或 NC 地址 2)
	02	G 代码（例如 G04、INVCW）
	03	带返回值的函数
	04	无返回值的函数
	05	关键字（例如 DEFINE）
	06	机床数据（$M...）、设定数据（$S...）或选项数据（$O...）
	07	系统参数，例如系统变量（$...）或计算参数（R...）
	08	循环（循环必须在 NC 中加载，并且循环程序生效 3)）
	09	GUD 变量（GUD 变量必须在 GUD 定义文件中定义，且必须被激活）
	10	宏名称（宏必须在宏定义文件中定义，且必须被激活）4)
	11	当前零件程序的 LUD 变量
	12	ISO G 代码（ISO 语言模式必须生效）
400	字符串“名称”是未识别为 xx == 01 或 xx == 10，不是 G 或者 R 的 NC 地址。2)	
y00	无专用分配	

1) 某些控制系统可能只能识别西门子 NC 语言指令中的一部分，例如 SINUMERIK 802D sl。在这些控制系统上，对于这些原则上为西门子语言指令的字符串会返回值 0。可通过 MD10711 $MN_NC_LANGUAGE_CONFIGURATION 修改此特性。MD10711 = 1 时，对于西门子 NC 语言指令总是返回值 100。

2) NC 地址为以下字母：A, B, C, E, I, J, K, Q, U, V, W, X, Y, Z。NC 地址也可通过地址扩展编程。在使用 STRINGIS 进行检查时可设定地址扩展。示例：201 == STRINGIS("A1").
字母：D, F, H, L, M, N, O, P, S, T 为用户自定义的 NC 地址或辅助功能。对其总是返回值 400。示例：400 == STRINGIS("D").在使用 STRINGIS 检查这些 NC 地址时不可设定地址扩展。
示例：000 == STRINGIS("M02")，但 400 == STRINGIS("M")。

3) 不可使用 STRINGIS 检查循环参数的名称。

4) 定义为宏的地址，例如 G, H, M, L，也识别为宏

示例

在下面的示例中假设设定为字符串的 NC 语言元素在控制系统中可编程（若无特别说明）。

1. 字符串“T”定义为辅助功能：
   ```
   400 == STRINGIS("T")
   000 == STRINGIS ("T3")
   ```
2. 字符串“X”定义为进给轴：
   ```
   201 == STRINGIS("X")
   201 == STRINGIS("X1")
   ```
3. 字符串“A2”定义为带扩展的 NC 地址：
   ```
   201 == STRINGIS("A")
   201 == STRINGIS("A2")
   ```
4. 字符串 “INVCW” 定义为所命名的 G 代码：
   ```
   202 == STRINGIS("INVCW")
   ```
5. 字符串“$MC_GCODE_RESET_VALUES”定义为机床数据：
   ```
   206 == STRINGIS("$MC_GCODE_RESET_VALUES")
   ```
6. 字符串“GETMDACT”定义为 NC 语言功能：
   ```
   203 == STRINGIS("GETMDACT ")
   ```
7. 字符串“DEFINE”定义为关键字：
   ```
   205 == STRINGIS("DEFINE")
   ```
8. 字符串“$TC_DP3”定义为系统参数（刀具长度分量）：
   ```
   207 == STRINGIS("$TC_DP3")
   ```
9. 字符串“$TC_TP4”为系统参数（刀具尺寸）：
   ```
   207 == STRINGIS("$TC_TP4")
   ```
10. 字符串“$TC_MPP4”为系统参数（刀库刀位状态）：
    - 刀具刀库管理生效：`207 == STRINGIS("$TC_MPP4")` ；
    - 刀具刀库管理未生效：`000 == STRINGIS("$TC_MPP4")`

    另见章节：刀具刀库管理。
11. 字符串“MACHINERY_NAME”定义为 GUD 变量：
    ```
    209 == STRINGIS("MACHINERY_NAME")
    ```
12. 字符串“LONGMACRO”定义为轴：
    ```
    210 == STRINGIS("LONGMACRO")
    ```
13. 字符串“MYVAR”定义为 LU 变量：
    ```
    211 == STRINGIS("MYVAR")
    ```
14. 字符串“XYZ”不是 NC 中已知的指令、GUD 变量、宏名称或循环名称：
    ```
    000 == STRINGIS("XYZ")
    ```

刀具刀库管理

如果刀具刀库管理功能未生效，则与机床数据

- MD10711 $MN_NC_LANGUAGE_CONFIGURATION 无关，

STRINGIS 总是对刀具刀库管理的系统参数输出值 000。

ISO 模式

若“ISO 模式”功能生效：

- MD18800 $MN_MM_EXTERN_LANGUAGE（激活外部 NC 语言）
- MD10880 $MN_ MM_EXTERN_CNC_SYSTEM（待匹配的控制系统）

STRINGIS 会首先将指定字符串作为 SINUMERIK G 代码检查。如果字符串不是 SINUMERIK G 代码，则之后会将其作为 ISO G 代码检查。

编程的切换（G290 （SINUMERIK 模式），或 G291 （ISO 模式））对 STRINGIS 没有影响。

示例

STRINGIS(...) 功能相关的机床数据有以下值：

- MD10711 $MN_NC_LANGUAGE_CONFIGURATION = 2（只有设置了选件的 NC 语言指令才能被识别）
- MD19410 $ON_TRAFO_TYPE_MASK = 'H0'（选件：转换）
- MD10700 $MN_PREPROCESSING_LEVEL='H43'（循环的预处理生效）

执行以下示例程序时不输出故障信息：

程序代码	注释
N1 R1=STRINGIS("TRACYL")	; R1 == 0，由于缺少坐标转换选件， TRACYL 被视为 无法识别
N2 IF STRINGIS("TRACYL") == 204	
N3 TRACYL(1,2,3)	; 跳过 N3
N4 ELSE	
N5 G00	; 而是执行 N5
N6 ENDIF	
N7 M30	

18.7 交互式调用零件程序 (MMC) 窗口

通过预定义的子程序 MMC(...)，可在操作界面上，如 SINUMERIK Operate，显示来源于数控程序的用户专用对话框。

可对以下对话框类型进行对话框设置：

- Run MyScreens
- Easy XML
- 用户 XML

文档

- Run MyScreens 编程手册
- Easy XML 编程手册

句法

```
MMC("<ADDRESS>,<COMMAND>,<FILE>,<DIALOG>","<QUIT>")
```

含义

MMC(...):	子程序名称 参数进行了位置编码并在两个字符串、指令字符串和应答字符串之间使用逗号隔开。	
指令字符串中的参数：		
<ADDRESS>:	操作区，在此执行所设计的用户会话	
	功能	**操作区域**
	用户对话框“Run MyScreens”	CYCLES
	用户对话框“Easy XML”	CYCLES
	用户 XML	XML
	弹窗“Run MyScreens”	POPUPDLG
	弹窗“Easy XML”	POPUPDLG

<COMMAND>:	要执行的指令	
	功能	**指令**
	用户对话框“Run MyScreens”	PICTURE_ON, PICTURE_OFF
	用户对话框“Easy XML”	PICTURE_ON, PICTURE_OFF
	用户 XML	XML_ON, XML_OFF
	弹窗“Run MyScreens”	PICTURE_ON, PICTURE_OFF
	弹窗“Easy XML”	PICTURE_ON, PICTURE_OFF
<FILE>:	文件名，在其中对要显示的对话框进行编程	
	功能	**文件**
	用户对话框“Run MyScreens”	<name>.com
	用户对话框“Easy XML”	<name>.xml
	用户 XML	<name>.xml
	弹窗“Run MyScreens”	<name>.com
	弹窗“Easy XML”	<name>.xml
	弹窗“Easy XML”，可直接在数控程序中进行设置（参见示例 2）	xmldial_emb.xml
<DIALOG>:	要显示的对话框的名称	
	功能	**对话框名称**
	除可直接在数控程序中进行设置的弹窗“Easy XML”外的所有功能	在文件<FILE>中设置的对话框名称
	弹窗“Easy XML”，可直接在数控程序中进行设置（参见示例 3）	main
应答字符串中的参数：		

<QUIT>:	应答类型	
	N:	无应答。 指令发送后，程序将继续执行。若无法成功执行指令，则不进行反馈。 **提示** 如果在数控程序中编写了显示时间（停留时间），则必须使用应答类型“N”（参见下面的示例 2）
	A:	异步应答 指令发送后，程序将继续执行。返回值会保存在对话框设置中定义的用户专用应答变量（GUD 变量）中并可在数控程序中读取。

示例

示例 1

显示对话框并响应数控程序中的用户操作。

```
; 应答变量 QUIT 已经作为 STRING 型的全局用户变量 (GUD)
; 在对话框设置时进行了创建：
; DEF NCK STRING[20] QUIT
QUIT = "XXX"                                    ; 初始化应答变量
G4 F5
MMC("CYCLES,PICTURE_ON,test.com,test1","A       ; 显示对话框
")                                              ; - 操作区域：CYCLES
                                                ; - 图片状态：PICTURE_ON (显示)
                                                ; - 对话框图片文件：test.com
                                                ; 对话框图片：test1
INPUT:                                          ; 等待用户操作
   STOPRE                                       ;      预处理停止
   IF MATCH (QUIT,"RUN") >= 0 GOTOF WORK        ;      软键“RUN”
   IF MATCH (QUIT,"CHK") >= 0 GOTOF CHECK       ;      软键“CHK”
GOTOB INPUT                                     ; => 等待

WORK:                                           ;      软键“RUN”已按下
MSG("继续加工 -> NC 启动")                      ; 输出信息
MMC("CYCLES,PICTURE_OFF","N")                   ; 关闭对话框
M0                                              ; 等待 NC 启动
GOTOF END                                       ; => 程序结束

CHECK:                                          ;      软键“CHK”已按下
```

程序代码	注释
MSG("逼近位置 -> NC 启动")	; 输出信息
MMC("CYCLES,PICTURE_OFF","N")	; 关闭对话框
M0	; 等待 NC 启动
GOTOF END	; => 程序结束
END:	
...	

示例 2

对话框的显示时间在数控程序中确定，例如通过停留时间设置。

程序代码	注释
F1000 G94	
...	
MMC("POPUPDLG,PICTURE_**ON**,xmldial_emb.xml,main","N")	; 显示对话框
X200	
Z40	
MMC("POPUPDLG,PICTURE_**OFF**","N")	; 关闭对话框

示例 3

将 PopUp 脚本嵌入数控程序及其应用中

程序代码

```
PROC POPUP_TEST
; ------------------------------ 脚本 ------------------------------
; <main_dialog entry="rpara_main">
;    <let name="xpos" />
;    <let name="ypos" />
;    <let name="field_name" type="string" />
;    <let name="num" />
;    <menu name="rpara_main">
;       <open_form name="rpara_form"/>
;       <softkey_back>
;          <close_form />
;       </softkey_back>
;    </menu>
;    <form name="rpara_form">
;       <init>
;          <caption>mask from NC part program</caption>
;          <let name="count" >0</let>
;          <op>
;             xpos = 120;
```

程序代码

```
;             ypos = 34;
;             "nck/Channel/Parameter/R[10]" = 10;
;          </op>
;          <!-- load the number of controls -->
;          <op>
;             num = "nck/Channel/Parameter/R[10]";
;          </op>
;          <while>
;             <condition> count < num</condition>
;             <print name="field_name" text="edit%d">count</print>
;             <op>
;                ypos = ypos + 24;
;                count = count + 1;
;             </op>
;          </while>
;       </init>
;       <paint>
;          <op>
;             xpos =  8;
;             ypos = 36;
;             count = 0;
;          </op>
;          <while>
;             <condition>count < num</condition>
;             <print name="field_name" text="R-Parameter%d">count</print>
;             <text xpos = "$xpos" ypos = "$ypos" >$$$field_name</text>
;             <op>
;                ypos = ypos + 24;
;                count = count + 1;
;             </op>
;          </while>
;       </paint>
;    </form>
; </main_dialog>
; ========================= 程序部分 ============================
...
G94 F100
MMC("POPUPDLG,PICTURE_ON,xmldial_emb.xml,main","N")
G4 F4
X200
MMC("POPUPDLG,PICTURE_OFF","N")
```

程序代码

```
G4 F2
X0
...
```

边界条件

- 对话框的定义文件 *.com 应保存在文件夹“proj”中。
- 对话框的 Easy XML 定义文件 *.xml 应保存在文件夹“appl”中。
  如果将定义文件保存在其他的目录下，则路径应间接地位于目录“appl”下。
- 用户定义的对话框无法同时从不同的通道中显示。

18.8 程序执行时间/工件计数器

为了对机床操作人员提供支持，系统提供程序运行时间和工件数量的相关信息。

这些信息可以作为系统变量在 NC 和/或 PLC 程序中处理。同时这些信息也可在操作界面上显示。

18.8.1 程序运行时间

功能“程序运行时间”提供了 NC 内部计时器用于监控工艺过程，它可以通过 NC 和通道专用的系统变量在零件程序和同步动作中读取。

用于运行时间测量的触发器 ($AC_PROG_NET_TIME_TRIGGER) 是一个唯一可写的功能系统变量，用于选择性测量程序步骤。即通过在 NC 程序中触发器写入可以激活并再次关闭时间测量。

系统变量	含义	活动
NC 专用		
$AN_SETUP_TIME	从上一次使用缺省值启动控制系统（冷启动）到现在的时间，单位分 在每次使用缺省值启动控制系统时都将自动复位为“0”。	• 总是激活
$AN_POWERON_TIME	从上一次控制系统正常启动（热启动）到现在的时间，单位分 在每次正常启动控制系统时都将自动复位为“0”。	
通道专用		

系统变量	含义	活动
$AC_OPERATING_TIME	在自动方式时 NC 程序的总运行时间，单位秒 在每次控制系统启动时都将自动复位为“0”。	• 通过 MD27860 激活 • 仅自动运行方式
$AC_CYCLE_TIME	所选择的 NC 程序的运行时间，单位秒 在每次启动一个新的 NC 程序时都将自动复位为“0”。	
$AC_CUTTING_TIME	加工时间，单位秒 即测得的 NC 启动和程序结束/NC 复位之间、所有 NC 程序中轨迹轴（至少一条）的运行时间，不包含快速移动当暂停时间生效时，计算被中断。 在每次使用缺省值启动控制系统时该值都将自动复位为“0”。	
$AC_ACT_PROG_NET_TIME	当前 NC 程序的当前净运行时间，单位秒 在每次启动一个 NC 程序时都将自动复位为“0”。	• 总是激活 • 仅自动运行方式
$AC_OLD_PROG_NET_TIME	正确用 M30 结束程序的净运行时间，以秒为单位	
$AC_OLD_PROG_NET_TIME_COUNT	更改到 $AC_OLD_PROG_NET_TIME 在接通电源后 $AC_OLD_PROG_NET_TIME_COUNT 置“0”。 当控制系统 $AC_OLD_PROG_NET_TIME 重新写入时，$AC_OLD_PROG_NET_TIME_COUNT 总是升高。	

系统变量	含义		活动
$AC_PROG_NET_TIME_TRIGGER	触发器用于运行时间测量：		● 仅自动运行方式
	0	中央状态 触发器未激活。	
	1	结束 结束测量并从 $AC_ACT_PROG_NET_TIME 复制值到 $AC_OLD_PROG_NET_TIME。 $AC_ACT_PROG_NET_TIME 置 “0” 并继续运行。	
	2	Start 启动测量并设置 $AC_ACT_PROG_NET_TIME 为 “0”。 $AC_OLD_PROG_NET_TIME 不变。	
	3	停止 停止测量。不改变 $AC_OLD_PROG_NET_TIME 并保持 $AC_ACT_PROG_NET_TIME 直至继续。	
	4	继续 继续测量，即再次接受一个以前停止的测量。$AC_ACT_PROG_NET_TIME 继续运行。$AC_OLD_PROG_NET_TIME 不变。	
通过上电将所有系统变量复位为“0”！			

说明

机床制造商

可激活的定时器通过机床数据 MD27860 $MC_PROCESSTIMER_MODE 激活。

使用特定功能（例如 GOTOS，倍率 = 0%，生效的空运行进给，程序测试，ASSP，PROG_EVENT 等）时生效的时间测量特性通过机床数据 MD27850 $MC_PROG_NET_TIMER_MODE 和 MD27860 $MC_PROCESSTIMER_MODE 设置。

文献：

功能手册 基本功能；BAG，通道，程序运行，复位特性（K1），章节：程序运行时间

说明

工件的剩余时间

如果需要依次加工相同的工件，可通过计时器计算该工件的剩余时间。

- 上次加工该工件的时间，参见 $AC_OLD_PROG_NET_TIME
- 当前加工时间（参见 $AC_ACT_PROG_NET_TIME）

除了当前加工时间，还会在操作界面上显示剩余时间。

说明

STOPRE 应用

系统变量 $AC_OLD_PROG_NET_TIME 和 $AC_OLD_PROG_NET_TIME_COUNT 不会产生隐含的预处理停止。当系统变量值来自于预定的程序运行时，预处理停止在零件程序中无关紧要。但是如果用于运行时间测量的触发器 ($AC_PROG_NET_TIME_TRIGGER) 高频写入，并且由此导致 $AC_OLD_PROG_NET_TIME 改变频繁，则零件程序中应使用一个明确定义的 STOPRE。

边界条件

- **程序段搜索**
  在程序段搜索时不会计算程序运行时间。
- **REPOS**
  REPOS 过程的时间会计入当前的加工时间($AC_ACT_PROG_NET_TIME)。

示例

示例 1：测量“mySubProgrammA”的时间

程序代码

```
...
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
```

在程序处理行 N80 后，在 $AC_OLD_PROG_NET_TIME 中有“mySubProgrammA”的净运行时间。

$AC_OLD_PROG_NET_TIME 值：

- 在 M30 后保持不变。
- 在每次完整运行循环后更新。

示例 2：测量“mySubProgrammA”和“mySubProgrammC”的时间

程序代码

```
...
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
```

18.8.2 工件计数器

使用“工件计数器”功能可提供各种不同的计数器，它们专用于在控制系统内部计算工件数量。

这些计数器作为通道专用的系统变量存在，带读写存取，值范围为 0 到 999 999 999。

系统变量	含义
$AC_REQUIRED_PARTS	待加工工件的数量（设定工件数量） 在此计数器中可以定义工件的个数，在到达这个数值之后，实际工件的个数($AC_ACTUAL_PARTS)复位为“0”。
$AC_TOTAL_PARTS	所有已加工工件的数量（总工件数量实际值） 该计数器给出所有自开始时刻起所加工的工件数量。只有在使用缺省值启动控制系统时该值才会自动复位为“0”。

系统变量	含义
$AC_ACTUAL_PARTS	所有已加工工件的数量（工件数量实际值） 在这种计数器中记录自开始时刻起所加工的所有工件数量。当达到设定工件数量时($AC_REQUIRED_PARTS)，该计数器就会自动归“0” ($AC_REQUIRED_PARTS > 0 是前提条件)。
$AC_SPECIAL_PARTS	用户计算的工件数量 该计数器允许用户根据自定义来对工件计数。达到设定工件数量($AC_REQUIRED_PARTS)时可以定义一个报警输出。用户必须自行将该计数器归零。

说明

在控制系统按照缺省值启动时，所有的工件计数器都会归“0”，而且不管是否激活，都可以被读写。

说明

使用通道专用的机床数据可以对计数器激活、归零时刻和计数算法进行设置。

说明

带用户定义 M 指令的工件计数

通过机床数据可以确定，通过用户定义的 M 指令来触发用于不同工件计数器的计数脉冲，而不是通过程序结束指令 M2/M30。

18.9 Process DataShare - 数据输出到外部设备/文件上（EXTOPEN，WRITE， EXTCLOSE）

将数据从零件程序写入外部设备/文件需要三步：

1. 打开外部设备/文件
   通过 EXTOPEN 指令打开外部设备/文件。
2. 写入数据
   可以用 NC 语言的字符串函数（“”）来处理输出数据，例如 SPRINT 函数。而写入过程本身通过 WRITE 指令执行。
3. 关闭外部设备/文件
   通过指令 EXTCLOSE、达到程序结束 M30 或通道复位，再次关闭通道中的外部设备/文件。

句法

```
DEF INT <Result>
DEF STRING[<n>] <Output>
…
EXTOPEN(<Result>,<ExtDev>,<SyncMode>,<AccessMode>,<WriteMode>)
…
<Output>=“输出数据”
WRITE(<Result>,<ExtDev>,<Output>)
…
EXTCLOSE(<Result>,<ExtDev>)
```

含义

EXTOPEN:	打开外部设备/文件的预定义步骤		
<Result>:	**参数 1**：结果变量 通过结果变量值可以检查程序中指令的执行情况。		
	类型：	INT	
	数值：	0	无错误
		1	外部设备无法打开
		2	没有配置外部设备
		3	外部设备配置了无效路径
		4	缺少对外部设备的存储权限
		5	使用模式：外部设备被设为“独占”
		6	使用模式：外部设备被设为“共享”
		7	文件长度大于“LOCAL_DRIVE_MAX_FILESIZE”
		8	超过允许的外部设备最大数量
		9	没有勾选选项“LOCAL_DRIVE"
		11	V.24 端口已经被 EasyMessage 占用（仅限 828D）
		12	写入模式：输入和“extdev.ini”矛盾
		16	写入了无效的外部路径
		22	外部设备没有安装

<table>
<tr><td rowspan="9"><ExtDev>:</td><td colspan="2">参数 2：需要打开的外部设备/文件的标识符</td></tr>
<tr><td>类型：</td><td>STRING</td></tr>
<tr><td colspan="2">标识符由以下字符组成：
1. 逻辑设备名称
2. 必要时也包含文件路径（前面带“/”）。
定义了以下逻辑设备名称：</td></tr>
<tr><td>“LOCAL_DRIVE”:</td><td>本地 CF 卡（预定义）</td></tr>
<tr><td>“CYC_DRIVE”:</td><td>预留给西门子循环使用的驱动器（预定义）</td></tr>
<tr><td>"/dev/ext/1", ...
"/dev/ext/9":</td><td>可用的网络驱动
注：
必须在文件 extdev.ini 中进行配置！</td></tr>
<tr><td>"/dev/cyc/1",
"/dev/cyc/2":</td><td>预留给西门子循环使用的驱动器
注：
必须在文件 extdev.ini 中进行配置！</td></tr>
<tr><td>“/dev/v24”:</td><td>V.24 接口
注：
必须在文件 extdev.ini 中进行配置！</td></tr>
<tr><td colspan="2">文件路径：
• 必须为 “LOCAL_DRIVE” 和 “CYC_DRIVE” 指定文件路径，例如：“LOCAL_DRIVE/my_dir/my_file.txt”
• 逻辑设备名称 “/dev/ext/1...9” 和 “/dev/cyc/1...2” 可以通过配置：
– 链接到一个文件，但是只允许输入逻辑设备名称，如：“/dev/ext/4”
– 链接到一个目录，但是必须输入文件路径：“/dev/ext/5/my_dir/my_file.txt”
• “/dev/v24” 无需补充文件路径。</td></tr>
<tr><td colspan="2">注：
逻辑设备名称 “/dev/ext/1...9”，“/dev/v24” 和 “/dev/cyc/1...2” 不区分大小写，但文件路径区分大小写。“LOCAL_DRIVE” 和 “CYC_DRIVE” 只允许大写字母。</td></tr>
</table>

<SyncMode>:	**参数 3**：WRITE 指令的处理模式，向该设备/文件输入数据		
	类型：	STRING	
	数值：	"SYN"：	同步写入 数据写入结束后，才继续处理程序， 查看 WRITE 指令中的错误变量，可以检查同步写入是否顺利结束。
		"ASYN"：	异步写入 WRITE 指令不会中断程序处理。 **注：** WRITE 指令的结果变量在该模式下无效，而且一直显示为 0（无错误）。因此，在该模式下无法确保 WRITE 指令成功执行。
<AccessMode>:	**参数 4**：设备/文件的使用模式		
	类型：	STRING	
	数值：	"SHARED":	设备/文件进入“分享”模式，其他通道也可以使用该设备，即同样也在该模式下打开。
		"EXCL"：	设备/文件在一个通道中单独使用，其他通道不可使用该设备。
<WriteMode>:	**参数 5**：WRITE 指令的写入模式，向设备/文件输出数据（可选）		
	类型：	STRING	
	数值：	"APP"：	添加 文件内容保持不变，输出的数据添加到结尾处。
		"OVR"：	覆盖 输出的数据覆盖旧文件内容。
	注： 通过该参数， 不能覆盖 extdev.ini 文件中的写入模式。在出现冲突的情况下，EXTOPEN 指令会报告错误。		

WRITE:	写入输出数据的预定义步骤

EXTCLOSE:	关闭已打开的外部设备/文件的预定义步骤		
<Result>:	**参数 1**：结果变量		
	类型：	INT	
	数值：	0	无错误
		16	写入了无效的外部路径
		21	关闭外部设备出错
<ExtDev>:	**参数 2**：需要关闭的外部设备/文件的标识符，详细信息参见 EXTOPEN！ **注：** 标识必须与 EXTOPEN 调用中输入的标识一致！		

示例

程序代码

```
N10       DEF INT RESULT
N20       DEF BOOL EXTDEVICE
N30       DEF STRING[80] OUTPUT
N40       DEF INT PHASE
N50       EXTOPEN(RESULT，“LOCAL_DRIVE/my_file.txt”，“SYN”，“SHARED”)
N60       IF RESULT > 0
N70          MSG（“EXTOPEN 出错：” << RESULT)
N80       ELSE
N90          EXTDEVICE=TRUE
N100      ENDIF
…
N200      PHASE=4
N210      IF EXTDEVICE
N220         OUTPUT=SPRINT（“结束相位：%D”，PHASE)
N230         WRITE(RESULT,"LOCAL_DRIVE/my_file.txt",OUTPUT)
N240      ENDIF
…
```

参见

字符串运算 (页 91)

写入文件（WRITE） (页 156)

18.10 报警（SETAL）

在一个 NC 程序中可以设置报警。 报警在操作界面中的一个特殊栏内显示， 每个报警都会触发一个对应类别的控制系统反应。

文档：
有关报警反应的其他信息请参见“调试手册”。

句法

```
SETAL（<报警号>[，<字符串>]）
```

含义

SETAL:	用于编程报警的指令字。 SETAL 必须在一个 NC 程序段中编程。	
<报警号>:	整型变量，包括报警号。 报警号的有效范围在 60000 和 69999 之间，其中 60000 到 64999 用于西门子循环，65000 到 69999 供用户使用。	
<字符串>:	在编写用户循环的报警时，可以另外输入一个字符串，最多含 4 个参数。 在这些参数中，可以定义用户文本。 但还提供下列预定义参数：	
	参数	**含义**
	%1	通道号
	%2	程序段号，标签
	%3	用于循环报警的文本索引
	%4	补充的报警参数

说明
报警文本必须在操作界面中设计。

说明
如果希望采用操作界面上当前激活的语言来输出报警，用户需要了解 HMI 上当前激活的语言。在零件程序和同步动作中，可以查看系统变量“$AN_LANGUAGE_ON_HMI”，获得语言信息，参见“HMI 上的当前语言 (页 1038)”。

示例

程序代码	注释
...	
N100 SETAL (65000)	；设置报警号 65000
...	

18.11 扩展停止和退回（ESR）

通过“扩展停止和退回”功能（亦称为 ESR 功能）可在故障状态下根据进程进行灵活响应：

- **扩展停止**
  在所允许的特定故障状况下，对所用为“扩展停止“功能使能的轴执行有序停止。
- **退回**
  所涉及刀具会以最快的速度从工件退回。
- **回馈式运行（SINAMICS 驱动功能"Vdc 控制“）**
  若未达到可设置的直流母线电压值（例如由于掉电），将会通过预设驱动回馈的制动能提供退回所需的电能（回馈式运行）。

触发源

一般源（NC 外部/全局，或者 BAG/通道特定）

- 数字输入（例如 NCU 模块上），或控制系统内部的、可回读的数字输出映象（$A_IN，$A_OUT）
- 通道状态（$AC_STAT）
- VDI 信号（$A_DBB）
- 一定数量报警的汇总信息（$AC_ALARM_STAT）

轴向源

- 跟随轴的紧急退回阈值（电子耦合的同步运动, $VC_EG_SYNCDIFF[<跟随轴>]）
- 驱动： 直流母线警告阈值（欠压危险），$AA_ESR_STAT[<轴>]
- 驱动： 馈电最小转速阈值（不再有可回馈的旋转能）， $AA_ESR_STAT[<轴>]。

静态同步动作的逻辑关系： 源/响应关系

使用静态同步动作的灵活逻辑方法，从而根据源尽快触发特定响应。

用户可借助静态同步动作将所有有关的源联系起来。 其可将源系统变量作为整体分析，也可通过位掩码进行选择性分析，并且与所需的响应关联。 静态同步动作可以在所有运行方式中生效。

文档：
功能手册 同步动作

激活

功能使能

通过设置相应的控制信号 $AA_ESR_ENABLE 来启用回馈式运行、停止和退回功能。 该控制信号可以由同步动作修改。

功能触发

ESR 通过置位系统变量 $AC_ESR_TRIGGER 针对所有使能的轴整体触发。

在识别出直流母线欠压危险时，驱动中会“自动”激活回馈式运行。

在驱动中识别出通讯故障（NC 和驱动之间），或者识别出直流母线欠压时（前提条件是进行了配置和使能），停止和/或退回功能会以驱动自主方式生效。

此外，通过设置相应的控制信号 $AN_ESR_TRIGGER (向所有驱动主轴播发指令)，也可以从 NC 端来触发自主驱动的停止和/或退回。

文档

ESR 的更多详细信息请见：

功能手册，特殊功能；扩展停止和退回（R3）

18.11.1 NC 控制的 ESR

18.11.1.1 NC 控制的退回 (POLF, POLFA, POLFMASK, POLFMLIN)

NC 控制的退回需要特定初始条件（参见”NC 控制的退回 (POLF, POLFA, POLFMASK, POLFMLIN) (页 723)“）。 若满足了这些前提条件，则会通过置位系统变量 $AC_ESR_TRIGGER（或 $AA_ESR_TRIGGER 用于单个轴）为通道中配置的退回轴激活快速提起（LIFTFAST）功能。

句法

```
POLF(<轴>)=<位置>
POLFA(<轴>,<类型>,<位置>)
POLFMASK(<轴_1>,<轴_2>,...)
POLFMLIN(<轴_1>,<轴_2>,...)
```

POLFA 允许采用以下缩略形式：
POLFA(<轴>,<类型>)；适用于单个轴退回的缩略形式
POLFA(轴,0/1/2)；快速取消或激活
POLFA(轴,0,$AA_POLFA[轴])；触发预处理停止

POLFA(轴,0)；不触发预处理停止

含义

POLF:	地址，用于给定退回的目标位置 POLF 为模态有效。			
	<轴>:	需退回的几何轴或通道/机床轴的名称		
	<位置>:	退回位置		
		类型:	REAL	
		对于几何轴 WCS 生效，其它情况下则是 MCS 生效。几何轴和通道/机床轴名称相同的情况下会在 WCS 中退回。		
POLFA:	预定义子程序调用，用于给定单个轴的退回位置			
	<轴>:	通道轴名称		
	<类型>:	位置给定模式		
		类型:	INT	
		值:	0:	位置值标记为无效
			1:	位置值为绝对的
			2:	位置值为增量的（距离）
		提示： 若该轴非单个轴，或缺少类型以及类型=0，则会输出相应报警。		
	<位置>:	退回位置（s. o.）		
		提示： 类型=0 也可接收位置值。 只是此值会被标记为无效，并且需要重新编程用于退回功能。		
POLFMASK:	预定义子程序调用，用于选择触发快速退刀后需要**独立**退回的轴。			
	<轴_1>,…:	轴名称，这些轴需要在快速退刀时通过 POLF 运行至定义的位置。 所有说明的轴必须处于同一个坐标系中。		
	使用未指定轴的 POLFMASK() 将取消所有独立退回轴的快速退刀。			

POLFMLIN:	预定义子程序调用，用于选择触发快速退刀后需要**以线性关联**退回的轴。	
	<轴_1>,...:	S. O.
	使用未指定轴的 POLFMLIN() 将取消所有以线性关联退回的轴的快速退刀。	

说明

在通过 POLFMASK 或者 POLFMLIN 使能向某个固定位置的快速退刀动作前，必须通过 POLF 为选定的轴编程一个位置。

说明

当使用 POLFMASK, POLFMLIN 或者 POLFMLIN, POLFMASK 将轴依次释放时，相应轴的上一次的设置有效。

说明

使用 POLF 编程的位置和通过 POLFMASK 或者 POLFMLIN 编程的激活指令在零件程序开始执行时被删除。 这就是说，用户必须在每个零件程序中对 POLF 的值和在 POLFMASK 或者 POLFMLIN 中所选择的轴重新编程。

说明

如果在使用缩写型式 POLFA 时仅改变类型，则用户就必须保证退回位置或者退回行程含有一个有效的值。 特别是在上电以后必须重新设置退回位置和退回位移。

示例

回位一个单轴：

程序代码	注释
MD37500 $MA_ESR_REACTION[AX1]=21	; NC 控制的退回。
...	
$AA_ESR_ENABLE[AX1] = 1	
POLFA(AX1,1,20.0)	; 轴向退回位置 20.0 (绝对值) 被分配给 AX1。
$AA_ESR_TRIGGER[AX1] = 1	; 从这里开始退回。

其它信息

使用 NC 控制的退回的前提条件

- 在通道中为 NC 控制的退回配置了一个退回轴：
  MD37500 $MA_ESR_REACTION = 21
- 必须为此轴使能了 ESR 功能：
  $AA_ESR_ENABLE = 1
- 定义了延时：
  MD21380 $MC_ESR_DELAY_TIME1
  MD21381 $MC_ESR_DELAY_TIME2
- 在零件程序中通过 POLF 编程了针对特定轴的退回位置。
- 通过 POLFMASK/POLFMLIN 为 NC 控制的退回选择了轴。
- 对于退回运动必须已经设置使能信号，并且保持设置。

使能和启动 NC 控制的退回

当系统变量 $AC_ESR_TRIGGER = 1，在此通道中配置了退回轴（即 MD37500 $MA_ESR_REACTION = 21）且为此轴设置了 $AA_ESR_ENABLE = 1 时，则在此通道中激活快速退刀（LIFTFAST）功能。

为通过 POLFMASK 或 POLFMLIN 选中的轴使用 POLF（或 LFPOS）配置的提升运行将取代在零件程序中为这些轴定义的轨迹运行。

退回时间最多可为 MD21380 $MC_ESR_DELAY_TIME1 和 MD21381 $MC_ESR_DELAY_TIME2 的和。 在该段时间结束之后，对于退回轴也引入快速制动，接着跟随运行。

说明

扩展退回运动（即通过 $AC_ESR_TRIGGER 所触发的 LIFTFAST/LFPOS）**无法中断**，且只能通过急停提前结束。

说明

通过 $AC_ESR_TRIGGER 触发的退回不会发生多次退回的状况。

单个轴退回

单个轴退回时必须使用 POLFA 编程单个轴的退回位置，并须遵循以下条件：

- $AA_ESR_ENABLE = 1
- <轴> 在触发时间点（$AA_ESR_TRIGGER = 1）必须为单个轴。
- <类型> 必须为 1 或 2。

快速退刀时的退回方向

在激活快速退刀时会考虑到有效的框架。

说明

带有旋转的框架会通过 POLF 影响退刀方向。

交换轴

总是在一个 NC 通道中精确分配退回轴，通道之间不允许交换。 若尝试将退回轴切换至另一个通道，则会触发警告。 只有用 $AA_ESR_ENABLE[AX] = 0 取消该轴后，才能将其切换至一个新通道。 在进行轴交换后可以再次用 $AA_ESR_ENABLE[AX] = 1 给轴加压。

中性轴

中性轴不能够执行 NC 控制的 ESR 。

18.11.1.2 NC 控制的停止

通过置位系统变量 $AC_ESR_TRIGGER（或 $AA_ESR_TRIGGER，用于单个轴），可为通道中配置的停止轴激活 NC 控制的停止。

前提条件

- 在通道中为 NC 控制的停止配置了一个停止轴：
  MD37500 $MA_ESR_REACTION = 22
- 必须为此轴使能了 ESR 功能：
  $AA_ESR_ENABLE = 1
- 定义了延时：
  MD21380 $MC_ESR_DELAY_TIME1（ESR 轴延时）
  MD21381 $MC_ESR_DELAY_TIME2（插补制动的 ESR 时间）

操作步骤

轴不受干扰地继续插补在 MD21380 中的时间段，如同编程一样。 在 MD21380 中时间段结束之后，引入插补控制的制动（斜坡停止）。 插补控制的制动的时间最多可为 MD21381 中设定的时间值。 在此时间结束后会进行快速制动，接下来为跟随运行。

示例

单个轴的停止：

程序代码	注释
MD37500 $MC_ESR_REACTION[AX1] = 22	; NC 控制的停止。
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	; 从这里开始停止。

18.11.2 驱动自控 ESR

18.11.2.1 配置驱动自控的停止（ESRS）

通过 ESRS(...)功能可以设置驱动自控 ESR 停止功能的参数。

句法

ESRS（<轴_1>，<停止时间_1>[，...，<轴_n>，<停止时间_n>]）

含义

ESRS（...）：	此功能用于写入 ESR 功能“停止”的驱动参数。 该功能： ● 必须位于单独的程序段中。 ● 会触发预处理停止。 ● 不可在同步动作中使用。	
<轴_1>， ...， <轴_n>：	指定需要配置驱动自控停止功能的轴 在驱动中，参数 p0888 （配置）用于指定轴： p0888 = 1	
	类型：	AXIS
	取值范围：	通道轴名称
<停止时间_1>， ...， <停止时间_n>：	出现故障后，驱动以当前转速设定值持续运转的时间 在驱动中，参数 p0892 用于设置指定轴的延时段： p0892 = <停止时间>	
	单位：	s
	类型：	REAL
	取值范围：	0.00 - 20.00
在一个函数指令中，最多可以编写 5 根轴，即 n = 5		

18.11.2.2 配置驱动自控的退回（ESRR）

通过 ESRR(...)函数可以设置驱动自控 ESR 退回功能的参数。

句法

ESRR（<轴_1>，<退回行程_1>，<退回速度_1>[，...，<轴_n>，<退回行程_n>，<退回速度_n>]）

含义

ESRR（...）：	此功能用于写入 ESR 功能“退回”的驱动参数 该功能： ● 必须位于单独的程序段中。 ● 会触发预处理停止。 ● 不可在同步动作中使用。	
<轴_1>， ...， <轴_n>：	指定需要配置驱动自控退回功能的轴 在驱动中，参数 p0888 （配置）用于指定轴： p0888 = 2	
	类型：	AXIS
	取值范围：	通道轴名称
<退回行程_1>， ...， <退回行程_n>：	退回行程，在驱动中被转换为退回转速。 驱动参数 p0893（转速）用于设置指定轴的转速： p0893 = (<退回行程 _n> 换算为退回转速）	
	单位：	mm/min，inch/min，Grad/min（取决于轴的单位）
	类型：	REAL
	取值范围：	最小 - 最大
<退回 速度_1>， ...， <退回 速度_n>：	退回速度，在驱动中被转换为持续时间。 驱动参数 p0892 用于设置指定轴的延时段[s]： p0892 = <退回行程_n> / <退回速度 _n>	
	单位：	mm/min，inch/min，Grad/min（取决于轴的单位）
	类型：	REAL
	取值范围：	0.00 - 最大
在一个函数指令中，最多可以编写 5 根轴，即 n = 5		

18.12 定义毛坯 (WORKPIECE)

控制系统必须知道毛坯的形状和大小，以便显示图形化模拟。因此，用户要通过操作界面或直接在数控程序中进行定义。毛坯定义在（程序结束/通道/运行方式组）复位时保持不变。在下次启动控制系统时会被自动清除。

句法

```
WORKPIECE("<WP>", "<RefP>", "<ZeroOffset>", "<Type>", <Par5>,
<Par6>, ..., <Par12>)
```

含义

WORKPIECE(...):		预定义步骤，用于定义毛坯		
		预处理停止：	是	
		在单独程序段中编程：	选择	
参数：				
1	"<WP>":	工件名称（可选）		
		数据类型：	STRING	
		此数据只需要在一个通道中有多个工件时给定。如不给定，“WORKP<n>”会自动将<n>设为已声明通道的编号。		
2	"<RefP>":	夹紧（可选；只用于铣床）		
		数据类型：	STRING	
		值域：	“工作台”	在固定工作台上夹紧
			“A”	在回转轴 A 上夹紧
			“B”	在回转轴 B 上夹紧
			“C”	在回转轴 C 上夹紧
		前提条件： 工作台或回转轴应通过相应的机床数据启用毛坯夹紧功能（参见调试手册 SINUMERIK Operate）。		
3	"<ZeroOffset>":	用于毛坯定位的可设定零点偏移（不可编程设置） 用于毛坯定位的可设定零点偏移的选择只能通过操作界面在毛坯输入时给定。但是直接在零件程序中定义毛坯时，毛坯始终会以当前有效的零点偏移为基准。		

4	"<Type>":	毛坯外形		
		数据类型：	STRING	
		值域：	"CYLINDER":	圆柱体
			"PIPE":	管形
			"RECTANGLE":	中心六面体
			"BOX":	六面体
			"N_CORNER":	多边形
5 ... 12	<Par5> ... <Par12>:	描述毛坯外形的参数		
		数据类型：	实数	
		所需参数的数量及其含义取决于各毛坯形状以及位参数的值。 参见： ● “描述毛坯外形的参数”表 ● “位参数”表		
WORPIECE():		无参数的 WORKPIECE 调用会删除所有毛坯定义。		
WORPIECE(<WP>):		带工件名称的 WORKPIECE 调用只会删除该毛坯定义。		

表格 18-1　描述毛坯外形的参数

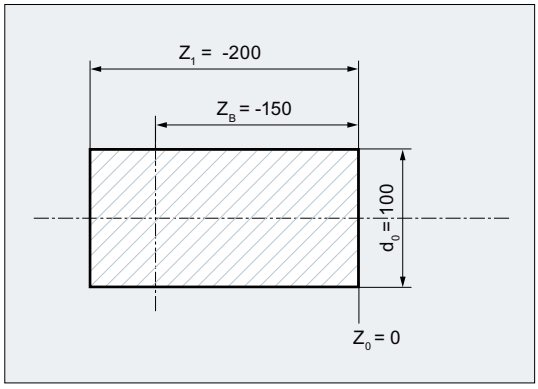
毛坯外形	参数							
	<Par5>	<Par6>	<Par7>	<Par8>	<Par9>	<Par10>	<Par11>	<Par12>
圆柱体	位参数 实值，编译为位编码的整数。这些位定义以下参数的含义（参见“位参数”表）。	参考点 Z_0	长度 Z_1	加工尺寸 Z_B	外径 d_0	-	围绕回转轴旋转	-
管形		参考点 Z_0	长度 Z_1	加工尺寸 Z_B	外径 d_0	壁厚（增量）/ 内径 d_1（绝对）	围绕回转轴旋转	-
中心六面体		参考点 Z_0	长度 Z_1	加工尺寸 Z_B	宽度 W	长度 L	围绕回转轴旋转	-
六面体		参考点 Z_0	长度 Z_1	加工尺寸 Z_B	X_0	Y_0	X_1	Y_1
多边形		参考点 Z_0	长度 Z_1	加工尺寸 Z_B	角的数量	对边宽度	围绕回转轴旋转	-

表格 18-2　位参数

位	含义	
4 (0x0010)	六面体：X_1	
	= 0	inc
	= 1	abs
5 (0x0020)	六面体：Y_1	
	= 0	inc
	= 1	abs
6 (0x0040)	长度 Z_1（最终尺寸）	
	= 0	inc
	= 1	abs
位 7 (0x0080)	加工尺寸 Z_B	
	= 0	inc
	= 1	abs
位 8 (0x0100)	管形：壁厚 / 内径	
	= 0	inc
	= 1	abs
9 (0x0200)	多边形	
	= 0	对边宽度
	= 1	边沿长度
12 (0x1000)	车床的夹紧	
	= 0	主主轴
	= 1	副主轴
13 (0x2000)	副主轴	
	= 0	带镜像
	= 1	无镜像

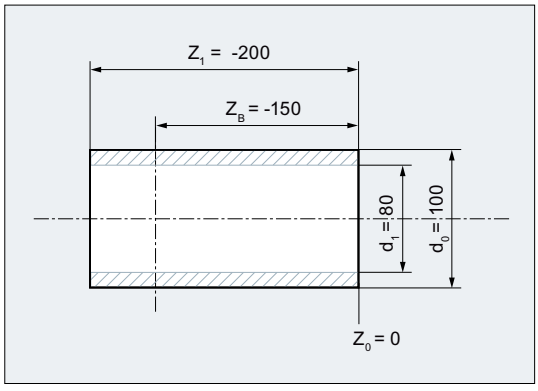
示例

示例 1：车床上的圆柱体毛坯



程序代码	注释
...	
WORKPIECE(,,,"CYLINDER",0,0,-200,-150,100)	；**毛坯定义：** ；毛坯外形：圆柱体 ；位参数=0(未置位) → 长度和加工尺寸为增量值，毛坯在主主轴上 ；参考点(Z0)=0 ；长度(Z1)=-200 ；加工尺寸(ZB)=-150 ；外径(d0)=100
...	

示例 2：车床上的管形毛坯



程序代码	注释
...	

程序代码	注释
WORKPIECE(,,,"PIPE",256,0,-200,-150,100,80)	；**毛坯定义：**
	；毛坯外形：管形
	；位参数=256(位 8=1) → 内径为绝对值，长度和加工尺寸为增量值，毛坯在主主轴上
	；参考点(Z0)=0
	；长度(Z1)=-200
	；加工尺寸(ZB)=-150
	；外径(d0)=100
	；内径(d1)=80
...	

18.13 切换语言模式 (G290, G291)

控制系统可从数控系统外部读取和执行零件程序。前提是，在调试时确定了相应的 NC 语言模式（ISO 语言）。

文档：
ISO 语言功能手册

ISO 语言模式可为每个通道分别激活。例如，通道 1 在 ISO 语言模式下运行，同时通道 2 在 SINUMERIK 模式下。

SINUMERIK 模式和 ISO 语言模式的切换在 NC 程序中通过 G 指令组 47 中的指令进行。运行方式的切换不会影响生效的刀具、刀具补偿和零点偏移。

句法

```
G291
...
G290
```

含义

G290:	激活 SINUMERIK 语言指令	
	在单独程序段中编程：	选择
	生效方式：	模态
G291:	激活 ISO 语言指令	
	在单独程序段中编程：	选择
	生效方式：	模态

条件

SINUMERIK 模式

- 在每个通道中都可以通过机床数据定义 G 指令的缺省设置。
- 在 SINUMERIK 模式下不能编写由 ISO 语言组成的语言指令。

ISO 语言模式

- 可以通过机床数据将 ISO 语言模式设为控制系统的基本设置。标准情况下控制系统随后在 ISO 语言模式中启动。
- 只能编写由 ISO 语言组成的 G 指令。在 ISO 语言模式下无法进行 SINUMERIK 功能的编程。
- 在同一个 NC 程序段中不允许混用 ISO 语言和 SINUMERIK 语言。
- 无法通过 G 指令在 ISO 语言 M（铣削）和 ISO 语言 T（车削）之间切换。
- 但在该模式下可以调用 SINUMERIK 模式下编写的子程序。
- 如要使用 SINUMERIK 功能，应先切换到 SINUMERIK 模式（参见示例）。

示例

ISO 语言模式下线性程序段的压缩

程序代码	注释
N5 G290	; 激活 SINUMERIK 语言模式
N10 COMPON	; COMPON 是西门子语言命令，可激活压缩功能，通过带有尽可能最大轨迹长度的多项式程序段替代连续的线性程序段。
N15 G291	; 激活 ISO 语言模式。
N20 G01 X100 Y100 F1000	; 因为已在 SINUMERIK 模式下激活了 COMPON，所以也可以在 ISO 语言模式下使用它来压缩线性程序段。
...	

自有切割程序 19

19.1 用于切割的支持性功能

您可以获得一个完整的加工循环用于切削。 由此您可以用以下所叙述的功能编制自身的切削程序：

- 设置轮廓表（CONTPRON）
- 设置轮廓表（CONTDCON）
- 断开轮廓预处理（EXECUTE）
- 计算两个轮廓元素之间的交点（INTERSEC）。
  （仅用于通过 CONTPRON 建立的表格。）
- 逐段执行某个图表的轮廓元素（EXECTAB）
  （仅用于通过 CONTPRON 建立的表格。）
- 计算圆的数据 (CALCDAT)

说明

您不仅可以在切削时用这些功能，而且也可以用于其它场合。

前提条件

在调用功能 CONTDCON 或 CONTDCON 之前必须：

- 返回到一个可以无轮廓冲突进行加工的起始点。
- 关断带 G40 的刀尖半径补偿。

19.2 设置轮廓表（CONTPRON）

使用 CONTPRON 打开轮廓预处理。 不处理下列调用的 NC 程序段，而分布在各个运动中并存放在轮廓表格内。 每个轮廓单元相当于轮廓表格中二维数组的一个表格行。 所计算出的咬边个数送回。

句法

启用轮廓预处理：
CONTPRON(<轮廓表>,<处理类型>,<底切>,
<加工方向>)

断开轮廓预处理并且在正常处理模式中重新接通：
EXECUTE(<FEHLER>)

参见“断开轮廓预处理（EXECUTE） (页 756)”

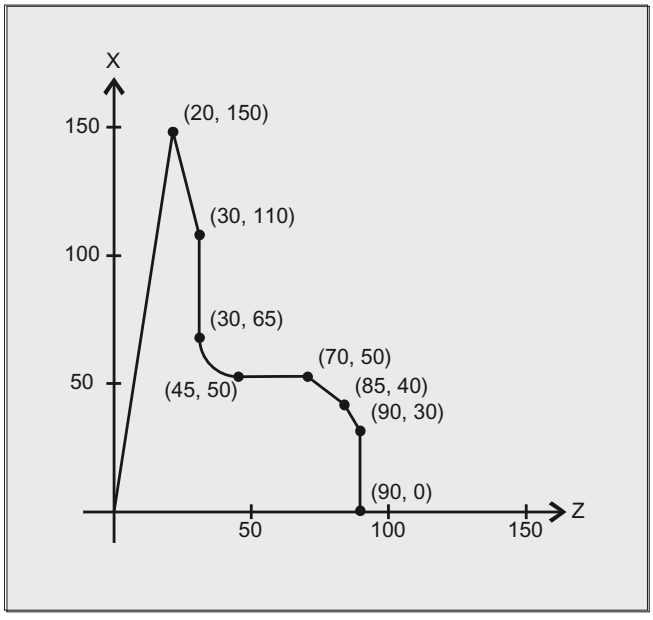
含义

CONTPRON:	启用轮廓预处理的预定义程序，用于创建轮廓表		
<轮廓表>：	轮廓表名称		
<加工方式>：	加工方式参数		
	类型：	CHAR	
	值：	"G":	纵向车削： 内部加工
		"L":	纵向车削： 外部加工
		"N":	端面车削： 内部加工
		“P”:	端面车削： 外部加工
<底切>：	出现的底切元素数目结果变量		
	类型：	INT	
<加工方向>：	加工方向参数		
	类型：	INT	
	值：	0	向前轮廓预处理（标准值）
		1	轮廓预处理在两个方向上

示例 1

编制一个轮廓表格，包括：

- 名称“KTAB”
- 最多 30 个轮廓单元（圆弧，直线）
- 一个变量，表明所出现的底切元素数量
- 用于故障信息的一个变量



NC 程序：

程序代码	注释
N10 DEF REAL KTAB[30,11]	；轮廓表包括名称 KTAB 和最多 30 个轮廓元素，参数值 11 (表格列数) 是一个固定值。
N20 DEF INT ANZHINT	；名称为 ANZHINT 用于底切元素数量的变量。
N30 DEF INT FEHLER	；故障反馈信息变量（0=没有故障，1=故障）。
N40 G18	
N50 CONTPRON (KTAB,"G",ANZHINT)	；启用轮廓预处理。
N60 G1 X150 Z20	；N60 至 N120： 轮廓说明
N70 X110 Z30	
N80 X50 RND=15	
N90 Z70	
N100 X40 Z85	
N110 X30 Z90	
N120 X0	
N130 EXECUTE(FEHLER)	；结束填写轮廓表，转换到正常程序运行方式。
N140	；图表的其它处理。

轮廓表 KTAB：

索引 行	列									
(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

各列内容说明：

(0) 指针到下一个轮廓单元（同一个行号）

(1) 指针到前一个轮廓单元

(2) 编码用于运动的轮廓模式

X = abc 可能值

a = 10^2 G90 = 0 G91 = 1

b = 10^1 G70 = 0 G71 = 1

c = 10^0 G0 = 0 G1 = 1 G2 = 2 G3 = 3

(3), (4) 轮廓元素的始点

(3) = 横坐标， (4) = 当前平面中的纵坐标

(5), (6) 轮廓单元终点

(5) = 横坐标， (6) = 当前平面中的纵坐标

(7) 最大/最小指针： 标记轮廓中局部的最大和最小

(8) 轮廓元素和横坐标（当纵向加工时）或者纵坐标（当端面加工时）之间的最大值。 角度取决于所编程的加工方式。

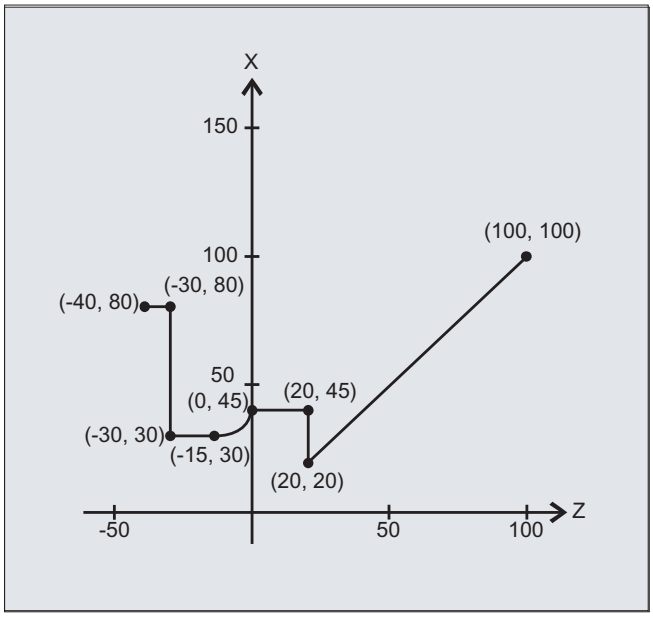
(9), (10) 如果是圆弧段，则轮廓单元的圆心坐标

(9) = 横坐标， (10) = 纵坐标

示例 2

编制一个轮廓表格，包括

- 名称 KTAB
- 最多 92 个轮廓单元（圆弧，直线）
- 工作方式 纵向车削，外侧加工
- 预处理，前进和后退



NC 程序：

程序代码	注释
N10 DEF REAL KTAB[92.11]	；轮廓表包括名称 KTAB 和最多 92 个轮廓元素，参数 11 是一个固定量。
N20 DEF CHAR BT="L"	；CONTPRON 运行方式： 纵向车削，外侧加工
N30 DEF INT HE=0	；底切元素的数量=0
N40 DEF INT MODE=1	；预处理，前进和后退
N50 DEF INT ERR=0	；故障反馈
...	
N100 G18 X100 Z100 F1000	
N105 CONTPRON(KTAB,BT,HE,MODE)	；启用轮廓预处理。

程序代码	注释
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)	；结束填写轮廓表，转换到正常程序运行方式。
...	

轮廓表 KTAB：

在结束轮廓预处理之后，可以在两个方向使用轮廓。

索引	列										
行	(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

栏内容说明和行 0、1、6、8、83、85 和 91 的备注

示例 1 中所述的栏内容说明有效。

始终在表格行 0：

1) 上一个： 行 n 包含向前的轮廓结束

2) 下一个： 行 n 是向前的轮廓表格结束

每一次在轮廓单元之内向前：

3) 上一个： 轮廓开始（向前）

4) 下一个： 轮廓结束（向前）

始终在轮廓表格行（向前）＋ 1：

5) 上一个： 咬边向前个数

6) 下一个： 咬边向后个数

每一次在轮廓单元之内向后：

7) 下一个： 轮廓结束（向后）

8) 上一个： 轮廓开始（向后）

始终在最后的表格行：

9) 上一个： 行 n 是轮廓表格起始（向后）

10) 下一个： 行 n 包含轮廓起始（向后）

其他信息

允许的运行指令，坐标系

下列 G 指令允许用于轮廓编程：

- G-组 1: G0, G1, G2, G3

最大可以是：

- 倒圆和倒角
- 圆编程通过 CIP 和 CT

样条、多项式和螺纹功能会导致出错。

不允许通过接通框架在 CONTPRON 和 EXECUTE 之间改变坐标系。 同时用于在 G70 和 G71 或 G700 和 G710 之间切换。

在预处理轮廓表格期间如果用 GEOAX 更换几何轴会导致报警。

咬边单元

单个的咬边单元的轮廓描述既可以在一个子程序中进行，也可以在单个程序段中进行。

与已编程的轮廓方向没有关系的切削

轮廓预处理通过 CONTPRON 已被扩展成在调用之后有独立于已编程方向的轮廓图表可供使用的型式。

19.3 设置轮廓表（CONTDCON）

对于通过 CONTDCON 启用的轮廓预处理，下列调用的 NC 程序段以编码方式有效存放在一个 6 栏轮廓表中。每个轮廓单元相当于轮廓表格中的一个表格行。基于对下述编码规则的认识，例如来自图表行中的循环，可以组成 DIN 代码程序。在号码 0 的表格行中，存储输出点的数据。

句法

启用轮廓预处理：
CONTDCON(<轮廓表>,<加工方向>)

断开轮廓预处理并且在正常处理模式中重新接通：
EXECUTE(<FEHLER>)

参见“断开轮廓预处理（EXECUTE） (页 756)”

含义

CONTDCON:	启用轮廓预处理的预定义程序，用于创建编码的轮廓表		
<轮廓表>:	轮廓表名称		
<加工方向>:	加工方向参数		
	类型：	INT	
	值：	0	轮廓预处理根据轮廓程序段结果（标准值）
		1	不允许

说明

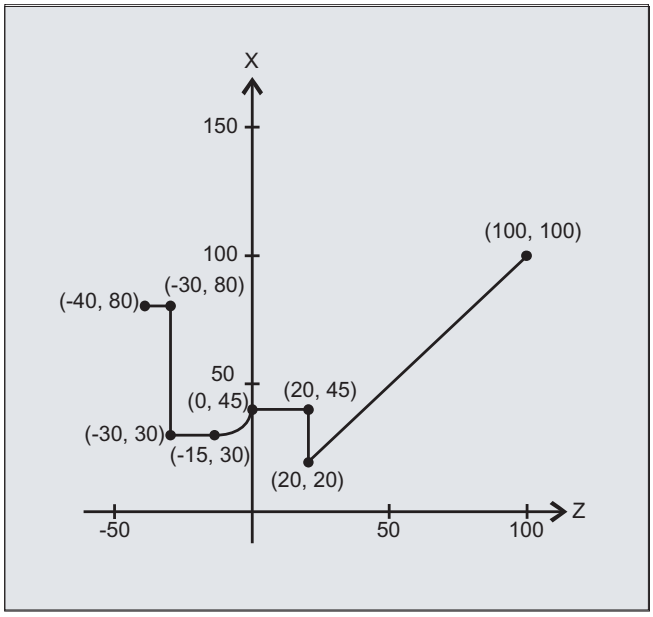
在待列表的程序块中，允许用于 CONTPRON 的 G 代码比功能 CONTPRON 中的范围更广。除此之外，还将同时保存每个轮廓的进给和进给类型。

示例

编制一个轮廓表格，包括：

- 名称“KTAB”
- 轮廓单元（圆弧，直线）

- 工作方式车削
- 加工方向：向前



NC 程序：

程序代码	注释
N10 DEF REAL KTAB[9,6]	；名称为 KTAB 且有 9 个图表行的轮廓图表。这些行允许 8 个轮廓程序段。参数值 6（表的栏数）为固定值。
N20 DEF INT MODE = 0	；加工方向变量。默认值 0:仅在已编程的轮廓方向中。
N30 DEF INT ERROR = 0	；反馈信息变量。
...	
N100 G18 G64 G90 G94 G710	
N101 G1 Z100 X100 F1000	
N105 CONTDCON (KTAB, MODE)	；调用轮廓预处理（MODE 允许省略）。
N110 G1 Z20 X20 F200	；轮廓说明。
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)	；结束填写轮廓表，转换到正常程序运行方式。
...	

轮廓表 KTAB：

	栏序号					
	0	1	2	3	4	5
行序号	轮廓模式	终点 横坐标	终点 纵坐标	中点 横坐标	中点 纵坐标	进给率
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

各列内容说明：

行 0：编码**始点：**

列 0： 10^0 (个位):G0 = 0

10^1 (十位):G70 = 0, G71 = 1, G700 = 2, G710 = 3

列 1： 起始点横坐标

列 2： 起始点纵坐标

列 3－4： 0

列 5： 表格中最后轮廓段的行序号

行 1-n:**轮廓段**的记录

列 0： 10^0 (个位):G0 = 0, G1 = 1, G2 = 2, G3 = 3

10^1 (十位):G70 = 0, G71 = 1, G700 = 2, G710 = 3

10^2 (百位):G90 = 0, G91 = 1

10^3 (千位):G93 = 0, G94 = 1, G95 = 2, G96 = 3

10^4 (万位):G60 = 0, G44 = 1, G641 = 2, G642 = 3

10^5 (十万):G9 = 1

列 1： 终点横坐标

列 2： 终点纵坐标

列 3： 圆心横坐标，在圆弧插补时

列 4： 圆心纵坐标，在圆弧插补时

列 5： 进给率

其他信息

允许的运行指令，坐标系

下列 G 组和 G 指令允许用于轮廓编程：

G-组 1: G0, G1, G2, G3

G-组 10: G60, G64, G641, G642

G-组 11: G9

G-组 13: G70, G71, G700, G710

G-组 14: G90, G91

G-组 15: G93, G94, G95, G96, G961

最大可以是：

- 倒圆和倒角
- 圆编程通过 CIP 和 CT

样条、多项式和螺纹功能会导致出错。

不允许通过接通框架在 CONTDCON 和 EXECUTE 之间改变坐标系。 同时用于在 G70 和 G71 或 G700 和 G710 之间切换。

在预处理轮廓表格期间如果用 GEOAX 更换几何轴会导致报警。

加工方向

使用 CONTDCON 生成的轮廓表可用于在已编程的轮廓方向中进行切削。

19.4 计算两个轮廓元素之间的交点（INTERSEC）。

INTERSEC 用来从使用 CONTPRON 生成的轮廓表中计算出两个已经过标准化处理的轮廓元素的交点。

句法

```
<Status>=INTERSEC(<轮廓表_1>[<轮廓元素_1>],
<轮廓表_2>[<轮廓元素_2>],<交点>,<加工方式>)
```

含义

INTERSEC:	预定义功能用于从通过 CONTPRON 生成的轮廓表中确定第二个轮廓元素交点		
<状态>:	用于交点状态的变量		
	类型：	BOOL	
	值：	TRUE	找到交点
		FALSE	没有找到交点
<轮廓表_1>:	第一个轮廓表名称		
<轮廓元素_1>:	第一个轮廓表轮廓元素编号		
<轮廓表_2>:	第二个轮廓表名称		
<轮廓元素_2>:	第二个轮廓表轮廓元素编号		
<交点>:	激活平面中的交点坐标(G17 / G18 / G19)		
	类型：	REAL	
<加工方式>:	加工方式参数		
	类型：	INT	
	值：	0	在通过参数 2 激活的平面中计算交点（标准值）
		1	交点计算与分配的平面无关

说明

请注意：变量必须在使用之前已经定义。

轮廓转换要求遵守用 CONTPRON 定义的值：

参数	含义
2	编码轮廓方式，用于运动
3	轮廓起始点横坐标
4	轮廓起始点纵坐标
5	轮廓终点横坐标
6	轮廓终点纵坐标
9	横坐标的中点坐标（仅对于圆弧轮廓）
10	纵坐标的中点坐标（仅对于圆弧轮廓）

示例

计算表格 TABNAME1 中的轮廓元素 3 与表格 TABNAME2 中轮廓元素 7 的交点。 活动平面中的交点坐标保存在变量 ISCOORD（第 1 项 = 横坐标，第 2 项= 纵坐标）中。 如果没有交点，则跳跃到 KEINSCH（没有找到交点）。

```
程序代码                                                    注释
DEF REAL TABNAME1[12,11]                                    ;轮廓图表 1
DEF REAL TABNAME2[10,11]                                    ;轮廓图表 2
DEF REAL ISCOORD [2]                                        ; 交点坐标变量。
DEF BOOL ISPOINT                                            ; 交点状态变量。
DEF INT MODE                                                ; 加工方式变量。
…
MODE=1                                                      ; 计算与激活平面无关。
N10 ISPOINT=INTERSEC(TABNAME1[3],TABNAME2[7],               ; 调用轮廓元素的交点。
ISCOORD,MODE)
N20 IF ISPOINT==FALSE GOTOF KEINSCH                         ; 跳转到 KEINSCH。
…
```

19.5 逐段执行某个图表的轮廓元素（EXECTAB）

使用 EXECTAB 可以逐段执行某个图表（例如用 CONTPRON 生成的图表）的轮廓元素。

句法

EXECTAB(<轮廓表>[<轮廓元素>])

含义

EXECTAB:	用于执行轮廓元素的预定义程序
<轮廓表>：	轮廓表名称
<轮廓元素>：	轮廓元素编号

示例

表格 KTAB 的轮廓元素 0 至 2 应该逐段执行。

程序代码	注释
N10 EXECTAB (KTAB[0])	; 执行表格 KTAB 的元素 0。
N20 EXECTAB (KTAB[1])	; 执行表格 KTAB 的元素 1。
N30 EXECTAB (KTAB[2])	; 执行表格 KTAB 的元素 2。

19.6 计算圆的数据 (CALCDAT)

通过 CALCDAT 可以从三个或者四个已知的圆弧点计算半径和圆心坐标。 所给出的点必须不同。

如果是 4 个点，它们不是精确的在圆弧上，则生成一个平均值用于圆心和半径。

说明

平均值的计算规则

圆弧计算为 4 x：

1. 圆弧点 1、2、3
2. 圆弧点 1、2、4
3. 圆弧点 1、3、4
4. 圆弧点 2、3、4

通过将四次圆弧计算的横坐标值或纵坐标值相加并除以 4 来计算圆心的横坐标和纵坐标。

通过对计算出的四个圆弧半径之和进行开方并将结果乘以 0.5 来计算半径。

句法

```
<Status>=CALCDAT(<圆弧点>[<数目>,<类型>],<数目>,<结果>)
```

含义

CALCDAT:	用于从 3 个或 4 个点计算一个圆弧的半径和圆心坐标的预定义功能		
<状态>：	用于圆弧计算状态的变量		
	类型：	BOOL	
	值：	TRUE	规定的点都位于圆弧上。
		FALSE	规定的点都不在圆弧上。
<圆弧点>[]：	变量用于 通过下列参数规定圆弧点：		
	<数量>：	圆弧点数目（3 或 4）	
	<类型>：	坐标数据类型， 例如 2 用于 2 点坐标系	
<数量>：	参数用于计算所使用点的数目（3 或 4）		

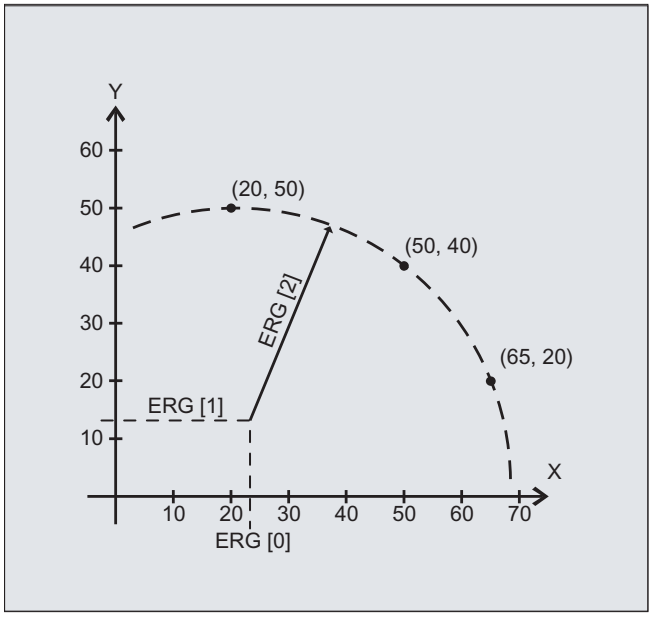
<结果>[3]：	用于结果的变量： 规定圆心坐标和半径	
	0	圆心： 横坐标值
	1	圆心： 纵坐标值
	2	半径

说明

请注意：变量必须在使用之前已经定义。

示例

由三个点计算出它们是否位于一个圆弧段。



程序代码	注释
N10 DEF REAL PKT[3,2]=(20,50,50,40,65,20)	; 变量用于给定圆弧点。
N20 DEF REAL ERG[3]	; 用于结果的变量。
N30 DEF BOOL STATUS	; 用于状态的变量。
N40 STATUS=CALCDAT(PKT,3,ERG)	; 调用测定的圆弧数据。
N50 IF STATUS == FALSE GOTOF ERROR	; 跳转到出错位置。

19.7 断开轮廓预处理（EXECUTE）

使用 EXECUTE 来断开轮廓预处理并且同时在正常处理模式中重新接通。

句法

```
EXECUTE(<FEHLER>)
```

含义

EXECUTE:	用于结束轮廓预处理的预定义程序	
<错误>:	故障反馈信息变量	
	类型：	INT
	变量值表示轮廓是否可以无故障加工：	
	0	错误
	1	无错误

示例

程序代码

```
...
N30 CONTPRON(...)
N40 G1 X... Z...
...
N100 EXECUTE(...)
...
```

外部循环编程 20

20.1 工艺循环

20.1.1 引言

目录

本章介绍了车削、铣削、磨削工艺循环。

结构

循环说明的结构如下：

- **句法**
  循环名称和传递参数的调用顺序
- **参数**
  参数说明表，指出单个参数的含义

参数说明

表格中列出了一个参数的以下数据：名称、说明、取值范围和与其它参数的关联性。

“参考参数”这一栏的用途在于，在回译外部生成的循环时，可以很方便地再次找到在系统上编程的值。

“仅供界面显示”参数

在表格中，有些参数标记了“仅供界面显示”，这些参数不影响循环的功能，只用于完整回译循环。如果没有编写这些参数，仍可以回译循环，但是这些参数栏就会突出显示，必须在对话框中输入数据。

“保留”参数

带有“保留”标识的参数必须写入 0 或空格，这样它后面的指令参数才能和内部循环参数一致。例外：字符串参数“”或空格。

按照位置模式重复执行循环

钻削循环和铣削循环可以按照位置模式重复执行，即模态式调用。必须在同一行、在循环前输入 MCALL，例如：MCALL CYCLE83（...）。

说明

如果某些传递参数，例如：<_VARI>, <_GMODE>, <_DMODE>, <_AMODE>被作为参数间接写入，在回译循环时会打开输入对话框，但是无法保存数据，因为有些下拉框的赋值不是唯一的。

20.1.2 工艺专用一览

下表中列出了所有可用外部可编程工艺循环并针对各个工艺进行了分配：

工艺	工艺循环
钻削	• CYCLE81 - 钻削，钻中心孔 (页 811) • CYCL82 - 钻削, 锪平面 (页 812) • CYCLE85 - 铰孔 (页 823) • CYCLE86 - 镗孔 (页 825) • CYCLE83 - 深孔钻削 (页 815) • CYCLE830 - 深孔钻削 2 (页 855) • CYCLE84 – 刚性攻丝 (页 819) • CYCLE840 - 带补偿夹具的攻丝 (页 865) • CYCLE78 - 螺纹铣削 (页 805) • CYCLE802 - 任意位置 (页 851) • HOLES1 – 成排孔 (页 760) • CYCLE801 - 方阵/框架 (页 849) • HOLES2 - 圆周孔 (页 760)
车削	• CYCLE951- 切削 (页 879) • CYCLE930 - 凹槽 (页 872) • CYCLE940 - 退刀槽 (页 875) • CYCLE99 - 螺纹车削 (页 836) • CYCLE98 - 螺纹链 (页 830) • CYCLE92 - 切断 (页 826)
轮廓车削	• CYCLE62 - 轮廓调用 (页 786) • CYCLE952 - 槽式车削 (页 882)

工艺	工艺循环
铣削	• CYCLE61- 平面铣削 (页 783) • POCKET3 - 铣削矩形腔 (页 763) • POCKET4 - 铣削圆形腔 (页 767) • CYCLE76 - 铣削矩形凸台 (页 799) • CYCLE77 - 铣削圆形凸台 (页 802) • CYCLE79 - 多边形 (页 808) • SLOT1- 纵向槽 (页 771) • SLOT2 - 圆弧槽 (页 774) • CYCLE899 - 铣削敞开槽 (页 869) • LONGHOLE - 长孔 (页 777) • CYCLE70 - 螺纹铣削 (页 792) • CYCLE60 - 雕刻循环 (页 780)
轮廓铣削	• CYCLE62 - 轮廓调用 (页 786) • CYCLE72 - 轨迹铣削 (页 795) • CYCLE63 - 轮廓腔铣削 (页 787) • CYCLE64 - 预钻轮廓腔 (页 791)
磨削	• CYCLE495 - 成型 (页 842) • CYCLE435 - 设置修整器坐标 (页 841) • CYCLE4071 - 反向点处带进给的纵向磨削 (页 889) • CYCLE4072 - 反向点处带进给的纵向磨削以及中断信号 (页 891) • CYCLE4073 - 带连续进给的纵向磨削 (页 894) • CYCLE4074 - 带连续进给的纵向磨削以及中断信号 (页 895) • CYCLE4075 - 反向点处带进给的平面磨削 (页 899) • CYCLE4077 - 反向点处带进给的平面磨削以及中断信号 (页 901) • CYCLE4078 - 带连续进给的平面磨削 (页 904) • CYCLE4079 - 带间歇进给的平面磨削 (页 906)
其它	• CYCLE800 – 回转 (页 844) • CYCLE832 - 快速设定 (页 861)
所有	• GROUP_BEGIN - 程序块开始 (页 909) • GROUP_END - 程序块结束 (页 910) • GROUP_ADDEND - 附加运行结束 (页 910)

20.1.3 HOLES1 – 成排孔

句法

```
HOLES1(<SPCA>, <SPCO>, <STA1>, <FDIS>, <DBH>, <NUM>, <_VARI>,
<_UMODE>, <_HIDE>, <_NSP>, <_DMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义		
1	X0	<SPCA>	REAL	成排孔参考点的第 1 轴位置（绝对）		
2	Y0	<SPCO>	REAL	成排孔参考点的第 2 轴位置（绝对）		
3	α0	<STA1>	REAL	初始旋转角（和第 1 轴所成夹角）		
4	L0	<FDIS>	REAL	第 1 个孔和参考点的间距		
5	L	<DBH>	REAL	孔间距		
6	N	<NUM>	INT	钻孔数量		
7		<_VARI>	INT	保留		
8		<_UMODE>	INT	保留		
9		<_HIDE>	字符串 [200]	被隐藏的位置 • 最多 198 个字符 • 连续位置号，例如“1,3”（跳过位置 1 和 3）		
10		<_NSP>	INT	保留		
11		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/18/19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）

20.1.4 HOLES2 - 圆周孔

句法

```
HOLES2(<CPA>, <CPO>, <RAD>, <STA1>, <INDA>, <NUM>, <_VARI>,
<_UMODE>, <_HIDE>, <_NSP>, <_DMODE>)
```

参数

编号	对话框参数	参数 内部	数据类型	含义	
1	X0	<CPA>	REAL	圆周孔中心点的第 1 轴位置（绝对） 第 1 轴的参考点	（XY 上） （XA、XB、ZC 上）
2	Y0	<CPO>	REAL	圆周孔中心点的第 2 轴位置（绝对） 第 2 轴的参考点	（XY 上） （XA、XB、ZC 上）
3	R	<RAD>	REAL	圆周孔的半径	（XY 上）
4	α0	<STA1>	REAL	起始角 或第 1 回转轴位置	（XY 上） （XA、XB、ZC 上）
5	α1	<INDA>	REAL	增量角（仅针对非整圆）	（XY、XA、YB、ZC 上）
				< 0 = 顺时针	
				> 0 = 逆时针	
6	N	<NUM>	INT	孔的数量	

编号	对话框参数	参数 内部	数据类型	含义			
7		<_VARI>	INT	加工方式			
				个位：	保留		
				十位：	定位方式		
					0 =	定位 - 直线	
					1 =	定位 - 圆弧	
				百位：	保留		
				千位：	圆模式		
					0 =	兼容模式，当 INDA = 0 时为整圆，当 INDA <> 0 时为非整圆	
					1 =	整圆	
					2 =	节距圆	
				万位：	带回转轴的位置模式		
					0 =	XY（不带回转轴）	（XY 上）
					1 =	XA（X 轴以及绕其旋转的回转轴）	（仅限 XA 上）
					2 =	YB（Y 轴及绕其旋转的回转轴）	（仅限 YB 上）
					3 =	ZC（Z 轴及绕其旋转的回转轴）	（仅限 ZC 上）
				百万位 + 十万位：	偏移（多个回转轴绕同一轴旋转时，下标太大，则为第 1 轴）		
					00 =	1. A 轴、B 轴或 C 轴	
					01 =	2. A 轴、B 轴或 C 轴	
					...		
					10 =	20. A 轴、B 轴或 C 轴	
8		<_UMODE>	INT	保留			
9		<_HIDE>	字符串 [200]	保留			
10		<_NSP>	INT	保留			

编号	对话框参数	参数 内部	数据类型	含义		
11		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/18/19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）

20.1.5 POCKET3 - 铣削矩形腔

句法

```
POCKET3(<_RTP>, <_RFP>, <_SDIS>, <_DP>, <_LENG>, <_WID>, <_CRAD>,
<_PA>, <_PO>, <_STA>, <_MID>, <_FAL>, <_FALD>, <_FFP1>, <_FFD>,
<_CDIR>, <_VARI>, <_MIDA>, <_AP1>, <_AP2>, <_AD>, <_RAD1>,
<_DP1>, <_UMODE>, <_FS>, <_ZFS>, <_GMODE>, <_DMODE>, <_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义
1	RP	<_RTP>	REAL	退回平面（绝对）
2	Z0	<_RFP>	REAL	刀具轴的参考点（绝对）
3	SC	<_SDIS>	REAL	安全距离（和参考点的间距，无符号）
4	Z1	<_DP>	REAL	腔深（绝对/增量），参见 <_AMODE>）
5	L	<_LENG>	REAL	腔长（增量，带符号）
6	W	<_WID>	REAL	腔宽（增量，带符号）
7	R	<_CRAD>	REAL	腔拐角的半径
8	X0	<_PA>	REAL	参考点，第 1 根轴（绝对）
9	YO	<_PO>	REAL	参考点，第 2 根轴（绝对）
10	α0	<_STA>	REAL	纵向轴 (L) 轴和第 1 轴的夹角
11	DZ	<_MID>	REAL	最大切深
12	UXY	<_FAL>	REAL	边沿精加工余量

编号	对话框参数	内部参数	数据类型	含义		
13	UZ	<_FALD>	REAL	底部精加工余量		
14	F	<_FFP1>	REAL	平面进给率		
15	FZ	<_FFD>	REAL	深度进给率		
16		<_CDIR>	INT	铣削方向：	0 =	顺铣
					1 =	逆铣
17		<_VARI>	INT	加工方式		
				个位：		
					1 =	粗加工
					2 =	精加工
					4 =	边沿精加工
					5 =	倒角
				十位：		
					0 =	预钻削，用 G0 进刀
					1 =	垂直，以 G1 进刀
					2 =	螺线
					3 =	沿着凹槽纵向轴摆动
				百位：	保留	
18	DXY	<_MIDA>	REAL	最大切宽（单位，参见<_AMODE>）		
19	L1	<_AP1>	REAL	预加工长度（增量）		
20	W1	<_AP2>	REAL	预加工宽度（增量）		
21	AZ	<_AD>	REAL	预加工深度（增量）		
22	ER	<_RAD1>	REAL	沿“螺线”插入时，螺线的半径		
	EW			“往复”进刀时的最大插入角度		
23	EP	<_DP1>	REAL	沿“螺线”插入时，螺线的螺距		
24		<_UMODE>	INT	保留		
25	FS	<_FS>	REAL	倒角宽度（增量）		
26	ZFS	<_ZFS>	REAL	加工倒角时的插入深度（刀尖），参见 <_AMODE>		

编号	对话框参数	内部参数	数据类型	含义		
27		<_GMODE>	INT	几何值计算模式（所编写的几何值）		
				个位：	保留	
				十位：	保留	
				百位：	加工/仅计算起点	
					0 =	兼容模式
					1 =	正常加工
				千位：	通过中心/拐点标注尺寸	
					0 =	兼容模式
					1 =	通过中心标注尺寸
					2 =	标注拐点，腔位置 +LENG/+WID
					3 =	标注拐点，腔位置 - LENG/+WID
					4 =	标注拐点，腔位置 +LENG/-WID
					5 =	标注拐点，腔位置 -LENG/-WID
				万位：	完整加工/二次加工	
					0 =	0 = 兼容模式（<_AP1>, <_AP2> 和 <_AD> 与之前的处理方式一样）
					1 =	完整加工
					2 =	二次加工

编号	对话框参数	内部参数	数据类型	含义		
28		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/G18/G19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
				十位：	进给率类型：G 功能组(G94/G95），用于平面进给率以及深度进给率	
					0 =	兼容模式
					1 =	G 代码保持调用循环前的状态。用于平面进给率以及深度进给率的 G94/G95 相同
				百位：	---	保留
				千位：	---	保留
				万位：	工艺对话框内的工艺调整 (页 911)	
					0 =	输入：完整
					1 =	输入：简单
29		<_AMODE>	INT	替代模式		
				个位：	腔深（Z1）	
					0 =	绝对（兼容模式）
					1 =	增量
				十位：	切宽（DXY）的单位	
					0 =	mm
					1 =	刀具半径的 %
				百位：	加工倒角时的插入深度（ZFS）	
					0 =	绝对
					1 =	增量

20.1.6 POCKET4 - 铣削圆形腔

句法

```
POCKET4(<_RTP>, <_RFP>, <_SDIS>, <_DP>, <_CDIAM>, <_PA>, <_PO>,
<_MID>, <_FAL>, <_FALD>, <_FFP1>, <_FFD>, <_CDIR>, <_VARI>,
<_MIDA>, <_AP1>, <_AD>, <_RAD1>, <_DP1>, <_UMODE>, <_FS>, <_ZFS>,
<_GMODE>, <_DMODE>, <_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义		
1	RP	<_RTP>	REAL	退回平面（绝对）		
2	Z0	<_RFP>	REAL	刀具轴的参考点（绝对）		
3	SC	<_SDIS>	REAL	安全距离（和参考点的间距，无符号）		
4	Z1	<_DP>	REAL	腔深（绝对/增量），参见 <_AMODE>）		
5	∅	<_CDIAM>	REAL	腔直径或半径，参见 <_DMODE>		
6	X0	<_PA>	REAL	参考点，第 1 根轴（绝对）		
7	Y0	<_PO>	REAL	参考点，第 2 根轴（绝对）		
8	DZ	<_MID>	REAL	最大切深，参见<_VARI> = 平面方式 最大螺线螺距，参见 <_VARI> = 螺线		
9	UXY	<_FAL>	REAL	平面精加工余量		
10	UZ	<_FALD>	REAL	深度精加工余量		
11	F	<_FFP1>	REAL	表面加工进给率		
12	FZ	<_FFD>	REAL	深度进给率		
13		<_CDIR>	INT	铣削方向	0 =	顺铣
					1 =	逆铣

编号	对话框参数	内部参数	数据类型	含义		
14		<_VARI>	INT	加工方式		
				个位：	加工	
					1 =	粗加工
					2 =	精加工
					4 =	边沿精加工
					5 =	倒角
				十位：	进刀方式（粗加工和精加工）	
					0 =	预钻削，用 G0 进刀（预加工腔体）
					1 =	垂直，以 G1 进刀
					2 =	螺线
				百位：	保留	
				千位：		
					0 =	平面方式
					1 =	螺线
15	DXY	<_MIDA>	REAL	最大切宽，参见 <_AMODE>, 0 = 0.8 x 刀具直径		
16	∅	<_AP1>	REAL	预加工直径/半径（增量）		
17	AZ	<_AD>	REAL	预加工深度（增量）		
18	ER	<_RAD1>	REAL	沿“螺线”插入时，螺线的半径		
19	EP	<_DP1>	REAL	沿“螺线”插入时，螺线的螺距		
20		<_UMODE>	INT	保留		
21	FS	<_FS>	REAL	倒角宽度（增量）		
22	ZFS	<_ZFS>	REAL	加工倒角时的插入深度（刀尖），参见 <_AMODE>		

编号	对话框参数	内部参数	数据类型	含义	
23		<_GMODE>	INT	几何值计算模式（所编写的几何值）	
				个位：	保留
				十位：	保留
				百位：	加工/起始点计算
					0 = 兼容模式
					1 = 正常加工
				千位：	保留
				万位：	完整加工/二次加工
					0 = 0 = 兼容模式（<_AP1> 和 <_AD> 与之前的处理方式一样）
					1 = 完整加工
					2 = 二次加工

编号	对话框参数	内部参数	数据类型	含义		
24		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/18/19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
				十位：	进给率类型：G 功能组(G94/G95），用于平面进给率以及深度进给率	
					0 =	兼容模式
					1 =	G 代码保持调用循环前的状态。用于平面进给率以及深度进给率的 G94/G95 相同
				百位：		
					0 =	兼容模式（<_CDIAM>/<_AP1>作为半径指定）
					1 =	<_CDIAM>/<_AP1> 作为直径指定
				千位：	---	保留
				万位：	工艺对话框内的工艺调整 (页 911)	
					0 =	输入：完整
					1 =	输入：简单

编号	对话框参数	内部参数	数据类型	含义	
25		<_AMODE>	INT	替代模式	
				个位：	腔深（Z1）
					0 = 绝对（兼容模式）
					1 = 增量
				十位：	切宽（DXY）的单位
					0 = mm
					1 = 刀具半径的 %
				百位：	加工倒角时的插入深度（ZFS）
					0 = 绝对
					1 = 增量

20.1.7 SLOT1- 纵向槽

句法

```
SLOT1 (<RTP>, <RFP>, <SDIS>, <_DP>, <_DPR>, <NUM>, <LENG>, <WID>,
<_CPA>, <_CPO>, <RAD>, <STA1>, <INDA>, <FFD>, <FFP1>, <_MID>,
<CDIR>, <_FAL>, <VARI>, <_MIDF>, <FFP2>, <SSF>, <_FALD>, <_STA2>,
<_DP1>, <_UMODE>, <_FS>, <_ZFS>, <_GMODE>, <_DMODE>, <_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义
1	RP	<RTP>	REAL	退回平面（绝对）
2	Z0	<RFP>	REAL	刀具轴的参考点（绝对）
3	SC	<SDIS>	REAL	安全距离（和参考点的间距，无符号）
4	Z1	<_DP>	REAL	槽深（绝对）
5		<_DPR>	REAL	相对于 Z0 的槽深（增量），无符号
6		<NUM>	INT	槽数量 = 1
7	L	<LENG>	REAL	槽长度
8	W	<WID>	REAL	槽宽度
9	X0	<_CPA>	REAL	参考点在平面第 1 轴的位置
10	Y0	<_CPO>	REAL	参考点在平面第 2 轴的位置

编号	对话框参数	内部参数	数据类型	含义		
11		<RAD>	REAL	保留		
12	α	<STA1>	REAL	旋转角度		
13		<INDA>	REAL	保留		
14	FZ	<FFD>	REAL	深度进给率		
15	F	<FFP1>	REAL	进给率		
16	DZ	<_MID>	REAL	最大切深		
17		<CDIR>	INT	铣削方向	0 =	顺铣
					1 =	逆铣
18	UXY	<_FAL>	REAL	端面或边缘精加工余量		
19		<VARI>	INT	加工方式		
				个位：		
					0 =	保留
					1 =	粗加工
					2 =	精加工
					4 =	边缘精加工（仅加工边缘）
					5 =	倒角
				十位：	进刀	
					0 =	预钻削，用 G0 进刀（预加工槽）
					1 =	垂直，以 G1 进刀
					2 =	螺线
					3 =	往复
				百位：	保留	
20	DZF	<_MIDF>	REAL	保留		
21	FF	<FFP2>	REAL	保留		
22	SF	<SSF>	REAL	保留		
23	UZ	<_FALD>	REAL	深度精加工余量		
24	ER	<_STA2>	REAL	沿“螺线”插入时，螺线的半径		
	EW			“往复”进刀时的最大插入角度		
25	EP	<_DP1>	REAL	沿“螺线”插入时，每转的插入深度		
26		<_UMODE>	INT	保留		

编号	对话框参数	内部参数	数据类型	含义		
27	FS	<_FS>	REAL	倒角宽度（增量）		
28	ZFS	<_ZFS>	REAL	加工倒角时的插入深度（刀尖），参见 <_AMODE>		
29		<_GMODE>	INT	几何值计算模式（所编写的几何值）		
				个位：	保留	
				十位：	保留	
				百位：	加工/仅计算起点	
					1 =	正常加工
				千位：	标注参考点、槽位置	
					0 =	中心
					1 =	内左 +L
					2 =	内右 -L
					3 =	左边 +L
					4 =	右边 -L
30		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/18/19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
				十位：	保留	
				百位：	保留	
				千位：	软件版本标识	
					1 =	功能扩展 SLOT1
				万位：	工艺对话框内的工艺调整 (页 911)	
					0 =	输入：完整
					1 =	输入： 简单

编号	对话框参数	内部参数	数据类型	含义		
31		<_AMODE>	INT	替代模式		
				个位：	最终深度 Z1（绝对/增量）	
					0 =	兼容性
					1 =	Z1（增量）
					2 =	Z1（绝对）
				十位：	保留	
				百位：	加工倒角时的插入深度 ZFS	
					0 =	ZFS （绝对）
					1 =	ZFS （增量）

说明

相对以往的软件版本，此循环融合了很多新功能。 因此，在输入对话框中不再显示某些参数（<NUM>, <RAD>, <INDA>）。 通过“MCALL”，并选择所需位置模式，例如“HOLES2”，可以编写更多样式的槽。

20.1.8 SLOT2 - 圆弧槽

句法

```
SLOT2(<RTP>, <RFP>, <SDIS>, <_DP>, <_DPR>, <NUM>, <AFSL>, <WID>,
<_CPA>, <_CPO>, <RAD>, <STA1>, <INDA>, <FFD>, <FFP1>, <_MID>,
<CDIR>, <_FAL>, <VARI>, <_MIDF>, <FFP2>, <SSF>, <_FFCP>,
<_UMODE>, <_FS>, <_ZFS>, <_GMODE>, <_DMODE>, <_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义
1	RP	<RTP>	REAL	退回平面（绝对）
2	Z0	<RFP>	REAL	刀具轴的参考点（绝对）
3	SC	<SDIS>	REAL	安全距离（和参考点的间距，无符号）
4	Z1	<_DP>	REAL	槽深（绝对）
5		<_DPR>	REAL	相对于 Z0 的槽深（增量），无符号
6	N	<NUM>	INT	槽数量
7	α1	<AFSL>	REAL	槽张角

编号	对话框参数	内部参数	数据类型	含义		
8	W	<WID>	REAL	槽宽度		
9	X0	<_CPA>	REAL	参考点（即圆心）在平面第 1 轴的位置		
10	Y0	<_CPO>	REAL	参考点（即圆心）在平面第 2 轴的位置		
11	R	<RAD>	REAL	圆半径		
12	α0	<STA1>	REAL	起始角		
13	α2	<INDA>	REAL	增量角		
14	FZ	<FFD>	REAL	深度进给率		
15	F	<FFP1>	REAL	进给率		
16	DZ	<_MID>	REAL	最大切深		
17		<CDIR>	INT	铣削方向	0 =	顺铣
					1 =	逆铣
18	UXY	<_FAL>	REAL	端面或边缘精加工余量		
19		<VARI>	INT	加工方式		
				个位：		
					0 =	完整加工
					1 =	粗加工
					2 =	精加工
					3 =	边沿精加工
					5 =	倒角
				十位：		
					0 =	沿 G0 直线定位
					1 =	沿圆弧轨迹
				百位：	保留	
				千位：		
					0 =	兼容模式，当 <INDA> = 0 时为整圆，当 <INDA> <> 0 时为非整圆
					1 =	整圆
					2 =	节距圆
20	DZF	<_MIDF>	REAL	保留		
21		<FFP2>	REAL	保留		
22		<SSF>	REAL	保留		

编号	对话框参数	内部参数	数据类型	含义		
23	FF	<_FFCP>	REAL	保留		
24		<_UMODE>	INT	保留		
25	FS	<_FS>	REAL	倒角宽度（增量）		
26	ZFS	<_ZFS>	REAL	加工倒角时的插入深度（刀尖），参见 <_AMODE>		
27		<_GMODE>	INT	几何值计算模式（所编写的几何值）		
				个位：	保留	
				十位：	保留	
				百位：	加工/仅计算起点	
					0 =	兼容模式
					1 =	正常加工
28		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/18/19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
				十位：	保留	
				百位：	保留	
				千位：	软件版本标识	
					1 =	SLOT2 功能， SW 2.5 以上版本
				万位：	工艺对话框内的工艺调整 (页 911)	
					0 =	输入：完整
					1 =	输入： 简单

编号	对话框参数	内部参数	数据类型	含义		
29		<_AMODE>	INT	替代模式		
				个位：	最终深度 Z1（绝对/增量）	
					0 =	兼容性
					1 =	Z1（增量）
					2 =	Z1（绝对）
				十位：	保留	
				百位：	加工倒角时的插入深度 ZFS	
					0 =	ZFS （绝对）
					1 =	ZFS （增量）

20.1.9 LONGHOLE - 长孔

句法

```
LONGHOLE(<RTP>, <RFP>, <SDIS>, <_DP>, <_DPR>, <NUM>, <LENG>,
<_CPA>, <_CPO>, <RAD>, <STA1>, <INDA>, <FFD>, <FFP1>, <MID>,
<_VARI>, <_UMODE>, <_GMODE>, <_DMODE>, <_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义
1	RP	<RTP>	REAL	退回平面（绝对）
2	Z0	<_RFP>	REAL	刀具轴的参考点（绝对）
3	SC	<SDIS>	REAL	安全距离（和参考点的间距，无符号）
4	Z1	<_DP>	REAL	长孔深度（绝对）
5		<_DPR>	REAL	相对于 Z0 的长孔深度（增量），无符号
6		<NUM>	INT	长孔数量 = 1
7	L	<LENG>	REAL	长孔长度
8	X0	<_CPA>	REAL	参考点，第 1 根上的间距
9	Y0	<_CPO>	REAL	参考点，第 2 根上的间距
10		<RAD>	REAL	保留
11	α0	<STA1>	REAL	旋转角度
12		<INDA>	REAL	保留

编号	对话框参数	内部参数	数据类型	含义		
13	FZ	<FFD>	REAL	深度进给率		
14	F	<FFP1>	REAL	进给率		
15	DZ	<MID>	REAL	最大切深		
16		<_VARI>	INT	加工方式		
				个位：	进刀方式	
					1 =	与 G1 垂直
					3 =	往复
				百位：	保留	
17		<_UMODE>	INT	保留		
18		<_GMODE>	INT	几何值计算模式（所编写的几何值）		
				个位：	保留	
				十位：	保留	
				百位：	加工/仅计算起点	
					0 =	兼容模式
					1 =	正常加工
				千位：	标注参考点、槽位置	
					0 =	中心
					1 =	内左 +L
					2 =	内右 -L
					3 =	左边 +L
					4 =	右边 -L

编号	对话框参数	内部参数	数据类型	含义		
19		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/18/19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
				十位：	进给方式：G 功能组(G94/G95），用于平面进给以及深度进给	
					0 =	兼容模式
					1 =	G 代码保持调用循环前的状态。用于平面进给以及深度进给的 G94/G95 相同
				百位：	保留	
				千位：	软件版本标识	
					1 =	功能扩展 LONGHOLE （标注参考点）
20		<_AMODE>	INT	替代模式		
				个位：	最终深度 Z1（绝对/增量）	
					0 =	兼容性
					1 =	Z1（增量）
					2 =	Z1（绝对）

说明

相对以往的软件版本，此循环融合了很多新功能。因此，在输入对话框中不再显示某些参数（<NUM>, <RAD>, <INDA>）。通过“MCALL”，并选择所需位置模式，例如“HOLES2”，可以编写更多样式的槽。

20.1.10 CYCLE60 - 雕刻循环

句法

```
CYCLE60(<_TEXT>, <_RTP>, <_RFP>, <_SDIS>, <_DP>, <_DPR>, <_PA>,
<_PO>, <_STA>, <_CP1>, <_CP2>, <_WID>, <_DF>, <_FFD>, <_FFP1>,
<_VARI>, <_CODEP>, <_UMODE>, <_GMODE>, <_DMODE>, <_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义
1		<_TEXT>	字符串 [200]	待雕刻的文本（最多 100 个字符）
2	RP	<_RTP>	REAL	退回平面（绝对）
3	Z0	<_RFP>	REAL	刀具轴的参考点（绝对）
4	SC	<_SDIS>	REAL	安全距离（和参考点的间距，无符号）
5	Z1	<_DP>	REAL	深度（绝对），参见 <_AMODE>
6	Z1	<_DPR>	REAL	深度（增量），参见 <_AMODE>
7	X0	<_PA>	REAL	参考点在平面第 1 轴的位置（绝对），直角坐标，参见 <_VARI>
	R			参考点长度（半径），极坐标， 参见 <_VARI>
8	Y0	<_PO>	REAL	参考点在平面第 2 轴的位置（绝对），直角坐标，参见 <_VARI>
	α0			参考点，和第 1 轴所成角度，极坐标，参见 <_VARI>
9	α1	<_STA>	REAL	直线排列文本：文本和第 1 轴所成角度，参见<_VARI>
10	XM	<_CP1>	REAL	圆弧排列文本：圆心在平面第 1 轴的位置（绝对），直角坐标，参见<_VARI>
	LM			圆弧排列文本：圆心相对于 WNP 的长度（半径），极坐标，参见 <_VARI>
11	YM	<_CP2>	REAL	圆弧排列文本：圆心在平面第 2 轴的位置（绝对），直角坐标，参见 <_VARI>
	αM			圆弧排列文本：圆心和第 1 轴所成角度，极坐标，参见 <_VARI>
12	W	<_WID>	REAL	字符高度（无符号）

编号	对话框参数	内部参数	数据类型	含义
13	DX1 DX2	<_DF>	REAL	字符间距/总宽度，参见 <_VARI>
	α2			开角，参见“ <_VARI>”
14	FZ	<_FFD>	REAL	深度进给，参见 _<_DMODE>
15	F	<_FFP1>	REAL	表面加工进给率

编号	对话框参数	内部参数	数据类型	含义		
16		<_VARI>	INT	加工（雕刻文本的对齐和参考点）		
				个位：	参考点	
					0 =	直角坐标
					1 =	极坐标
				十位：	文本对齐	
					0 =	文本直线排列
					1 =	文本沿上半圆排列
					2 =	文本沿下半圆排列
				百位：	保留	
				千位：	水平文本的参考点	
					0 =	左
					1 =	中心
					2 =	右
				万位：	竖直文本的参考点	
					0 =	下
					1 =	中心
					2 =	上
				十万位：	文本长度	
					0 =	字符间距
					1 =	文本总宽度（仅在直线排列文本时）
					2 =	张角 (仅在圆弧排列文本时)
				百万：	圆弧圆心	
					0 =	直角坐标
					1 =	极坐标
				千万位：	镜像文字	
					0 =	兼容性
					1 =	镜像文字开
					2 =	镜像文字关
17		<_CODEP>	INT	字体的代码页编号，目前只有 1252		
18		<_UMODE>	INT	保留		

编号	对话框参数	内部参数	数据类型	含义		
19		_GMODE>	INT	几何值计算模式（所编写的几何值）		
				个位：	保留	
				十位：	保留	
				百位：	加工/仅计算起点	
					0 =	兼容模式
					1 =	正常加工
20		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/18/19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
				十位：	进给方式：G 功能组(G94/G95），用于平面进给以及深度进给	
					0 =	兼容模式
					1 =	G 代码保持调用循环前的状态。用于平面进给以及深度进给的 G94/G95 相同
21		<_AMODE>	INT	替代模式		
				个位：	最终深度（<_DP>, <_DPR>）	
					0 =	兼容性
					1 =	增量（<_DPR>）
					2 =	绝对（<_DP>）

20.1.11 CYCLE61- 平面铣削

句法

```
CYCLE61(<_RTP>, <_RFP>, <_SDIS>, <_DP>, <_PA>, <_PO>, <_LENG>,
<_WID>, <_MID>, <_MIDA>, <_FALD>, <_FFP1>, <_VARI>, <_LIM>,
<_DMODE>, <_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义		
1	RP	<_RTP>	REAL	退回平面（绝对）		
2	Z0	<_RFP>	REAL	刀具轴的参考点，毛坯高度（绝对）		
3	SC	<_SDIS>	REAL	安全距离（和参考点的间距，无符号）		
4	Z1	<_DP>	REAL	成品高度（绝对/增量），参见 <_AMODE>		
5	X0	<_PA>	REAL	拐点 1 的第 1 轴位置（绝对）		
6	Y0	<_PO>	REAL	拐点 1 的第 2 轴位置（绝对）		
7	X1	<_LENG>	REAL	拐点 2 的第 1 轴位置（绝对/增量），参见 <_AMODE>		
8	Y1	<_WID>	REAL	拐点 2 的第 2 轴位置（绝对/增量），参见 <_AMODE>		
9	DZ	<_MID>	REAL	最大切深		
10	DXY	<_MIDA>	REAL	最大切宽（单位，参见<_AMODE>）		
11	UZ	<_FALD>	REAL	底部精加工余量		
12	F	<_FFP1>	REAL	加工进给率		
13		<_VARI>	INT	加工方式		
				个位：	加工	
					1 =	粗加工
					2 =	精加工
				十位：	加工方向	
					1 =	平行于第 1 轴，沿着一个方向
					2 =	平行于第 2 轴，沿着一个方向
					3 =	平行于第 1 轴，方向不断交替
					4 =	平行于第 2 轴，方向不断交替

编号	对话框参数	内部参数	数据类型	含义		
14		<_LIM>	INT	限位		
				个位：	第 1 轴的负限位	
					0 =	否
					1 =	是
				十位：	第 1 轴的正限位	
					0 =	否
					1 =	是
				百位：	第 2 轴的负限位	
					0 =	否
					1 =	是
				千位：	第 2 轴的正限位	
					0 =	否
					1 =	是
15		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/18/19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）

编号	对话框参数	内部参数	数据类型	含义		
16		<_AMODE>	INT	替代模式		
				个位：	最终深度（<_DP>）	
					0 =	绝对
					1 =	增量
				十位：	切宽的单位（<_MIDA>）	
					0 =	mm
					1 =	刀具半径的 %
				百位：	保留	
				千位：	平面长度	
					0 =	增量
					1 =	绝对
				万位：	平面宽度	
					0 =	增量
					1 =	绝对

20.1.12 CYCLE62 - 轮廓调用

句法

```
CYCLE62(<_KNAME>,<_TYPE>, <_LAB1>, <_LAB2>)
```

参数

编号	对话框参数	内部参数	数据类型	含义		
1	PRG/CON	<_KNAME>	字符串 [140]	在 _TYPE = 2 时，不能写入轮廓名称或子程序名称		
2		<_TYPE>	INT	确定轮廓输入方式		
					0 =	子程序
					1 =	轮廓名
					2 =	标签
					3 =	子程序中的标签

编号	对话框参数	内部参数	数据类型	含义
3	LAB1	<_LAB1>	STRING[32]	标签 1，轮廓起始
4	LAB2	<_LAB2>	STRING[32]	标签 2，轮廓结束

20.1.13 CYCLE63 - 轮廓腔铣削

句法

```
CYCLE63(<_PRG>, <_VARI>, <_RP>, <_Z0>, <_SC>, <_Z1>, <_F>, <_FZ>,
<_DXY>, <_DZ>, <_UXY>, <_UZ>, <_CDIR>, <_XS>, <_YS>, <_ER>,
<_EP>, <_EW>, <_FS>, <_ZFS>, <_TR>, <_DR>, <_UMODE>, <_GMODE>,
<_DMODE>, <_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义		
1	PRG	<_PRG>	字符串 [100]	清理程序名称		
2		<_VARI>	INT	加工方式		
				个位：	工艺加工	
					1 =	粗加工
					3 =	底部精加工
					4 =	边沿精加工
					5 =	倒角
				十位：	进刀方式	
					0 =	中心插入
					1 =	螺线插入
					2 =	往复插入
				百位：	保留	
				千位：	退刀返回模式	
					0 =	退刀回返回平面
					1 =	退刀回“参考点＋安全距离”
				万位：	底部粗加工和精加工的起点	
					0 =	自动方式
					1 =	手动
3	RP	<_RP>	REAL	退回平面（绝对）		
4	Z0	<_Z0>	REAL	刀具轴的参考点（绝对）		
5	SC	<_SC>	REAL	安全距离（和参考点的间距，无符号）		
6	Z1	<_Z1>	REAL	最终深度（参见 <_AMODE> 个位）		
7	F	<_F>	REAL	粗加工/精加工时的平面进给率		
8	FZ	<_FZ>	REAL	深度进给率		
9	DXY	<_DXY>	REAL	切宽的单位（参见 <_AMODE> 十位）		
10	DZ	<_DZ>	REAL	切深		
11	UXY	<_UXY>	REAL	边沿精加工余量		
12	UZ	<_UZ>	REAL	底部精加工余量		

编号	对话框参数	内部参数	数据类型	含义		
13		<_CDIR>	INT	铣削方向	0 =	顺铣
					1 =	逆铣
14	XS	<_XS>	REAL	起点 X，绝对		
15	YS	<_YS>	REAL	起点 Y，绝对		
16	ER	<_ER>	REAL	螺线式插入： 半径		
17	EP	<_EP>	REAL	螺线式插入： 螺距		
18	EW	<_EW>	REAL	往复式插入： 最大插入角		
19	FS	<_FS>	REAL	倒角宽度（增量）		
20	ZFS	<_ZFS>	REAL	加工倒角时的刀尖插入深度（参见 <_AMODE> 百位）		
21	TR	<_TR>	STRING[32]	余料加工时的参考刀具名称		
22	DR	<_DR>	INT	余料加工时的参考刀具 D 号		
23		<_UMODE>	INT	保留		
24		<_GMODE>	INT	几何值计算模式（所编写的几何值）		
				个位：	保留	
				十位：	保留	
				百位：	加工/仅计算起点	
					0 =	正常加工（不要求兼容模式）
					1 =	正常加工
					2 =	保留

编号	对话框参数	内部参数	数据类型	含义		
25		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/G18/G19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
				十位：	保留	
				百位：	工艺模式	
					1 =	型腔
					2 =	凸台
				千位：	加工剩余材料	
					0 =	否
					1 =	是
				万位：	工艺对话框内的工艺调整 (页 911)	
					0 =	输入：完整
					1 =	输入： 简单
26		<_AMODE>	INT	替代模式		
				个位：	最终深度（Z1）	
					0 =	绝对（兼容模式）
					1 =	增量
				十位：	切宽（DXY）的单位	
					0 =	mm
					1 =	刀具半径的 %
				百位：	加工倒角时的插入深度（ZFS）	
					0 =	绝对
					1 =	增量
				千位：	---	保留

20.1.14 CYCLE64 - 预钻轮廓腔

句法

```
CYCLE64(<_PRG>, <_VARI>, <_RP>, <_Z0>, <_SC>, <_Z1>, <_F>,
<_DXY>, <_UXY>, <_UZ>, <_CDIR>, <_TR>, <_DR>, <_UMODE>, <_GMODE>,
<_DMODE>, <_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义		
1	PRG	<_PRG>	字符串 [100]	钻削/钻中心孔程序的名称		
2		<_VARI>	INT	加工方式		
				个位：	保留	
				十位：	保留	
				百位：	保留	
				千位：	退刀返回模式	
					0 =	退刀回返回平面
					1 =	退刀回“参考点＋安全距离”
3	RP	<_RP>	REAL	退回平面（绝对）		
4	Z0	<_Z0>	REAL	参考点（绝对）		
5	SC	<_SC>	REAL	安全距离（和参考点的间距，无符号）		
6	Z1	<_Z1>	REAL	钻削/中心孔深度（参见 <_AMODE> 个位）		
7	F	<_F>	REAL	钻削/钻中心孔的进给率		
8	DXY	<_DXY>	REAL	切宽的单位（参见 <_AMODE> 十位）		
9	UXY	<_UXY>	REAL	边沿精加工余量		
10	UZ	<_UZ>	REAL	底部精加工余量		
11		<_CDIR>	INT	铣削方向	0 =	顺铣
					1 =	逆铣
12	TR	<_TR>	STRING[20]	参考刀具名称		
13	DR	<_DR>	INT	参考刀具 D 号		
14		<_UMODE>	INT	保留		

编号	对话框参数	内部参数	数据类型	含义		
15		<_GMODE>	INT	几何值计算模式（所编写的几何值）		
				个位：	保留	
				十位：	保留	
				百位：	加工/仅计算起点	
					0 =	正常加工（不要求兼容模式）
					1 =	正常加工
					2 =	保留
25		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/G18/G19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
				十位：	工艺模式）	
					1 =	预钻削
					2 =	钻中心孔
26		<_AMODE>	INT	替代模式		
				个位：	钻削/中心孔深度 Z1	
					0 =	绝对（兼容模式）
					1 =	增量
				十位：	切宽（DXY）的单位	
					0 =	mm
					1 =	刀具半径的 %

20.1.15 CYCLE70 - 螺纹铣削

句法

```
CYCLE70(<_RTP>, <_RFP>, <_SDIS>, <_DP>, <_DIATH>, <_H1>, <_FAL>,
<_PIT>, <_NT>, <_MID>, <_FFR>, <_TYPTH>, <_PA>, <_PO>, <_NSP>,
<_VARI>, <_PITA>, <_PITM>, <_PTAB>, <_PTABA>, <_GMODE>, <_DMODE>,
<_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义		
1	RP	<_RTP>	REAL	退回平面（绝对）		
2	Z0	<_RFP>	REAL	刀具轴的参考点（绝对）		
3	SC	<_SDIS>	REAL	安全距离（和参考点的间距，无符号）		
4	Z1	<_DP>	REAL	螺纹长度（绝对、增量），参见 <_AMODE> 注意在孔底上保留间距（最小半个螺距）		
5	∅	<_DIATH>	REAL	螺纹额定直径		
6	H1	<_H1>	REAL	螺纹深度		
7	U	<_FAL>	REAL	精加工余量		
8	P	<_PIT>	REAL	螺距（选择 <_PITA>： 毫米，英寸，MODUL，牙/英寸)		
9	NT	<_NT>	INT	切削刃的齿数 刀具长度始终相对于下面的切削齿！		
10	DXY	<_MID>	REAL	每刀切削量 <_MID> > <_H1>：一刀到底		
11	F	<_FFR>	REAL	铣削进给率		
12		<_TYPTH>	INT	螺纹类型	0 =	内螺纹
					1 =	外螺纹
13	X0	<_PA>	REAL	圆心第 1 轴的位置（绝对）		
14	Y0	<_PO>	REAL	圆心第 2 轴的位置（绝对）		
15	αS	<_NSP>	REAL	起始角度（多牙螺纹）		
16		<_VARI>	INT	加工方式		
				个位：		
					1 =	粗加工
					2 =	精加工
				十位：		
					1 =	从上往下加工
					2 =	从下往上加工
				百位：		
					0 =	右旋螺纹
					1 =	左旋螺纹

编号	对话框参数	内部参数	数据类型	含义		
17		<_PITA>	INT	螺距计算		
					0 =	兼容模式
					1 =	螺距,单位毫米
					2 =	螺距单位：牙/英寸(TPI)
					3 =	螺距,单位英寸
					4 =	螺距，单位 MODUL
18		<_PITM>	STRING[15]	表示螺距输入方式的字符串（“仅供界面显示”）		
19		<_PTAB>	STRING[20]	表示螺纹表的字符串 （“”，“ISO”，“BSW”，“BSP”，“UNC”）（“仅供界面显示”）		
20		<_PTABA>	STRING[20]	表示螺纹表中各个选项的字符串（例如：“M 10”，“M 12”， ...）（“仅供界面显示”）		
21		<_GMODE>	INT	几何值计算模式（所编写的几何值）		
				个位：	保留	
				十位：	保留	
				百位：	加工/起始点计算	
					0 =	兼容模式
					1 =	正常加工
22		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/G18/G19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
23		<_AMODE>	INT	替代模式		
				个位：	螺纹长度（<_DP>）	
					0 =	绝对
					1 =	增量

20.1.16 CYCLE72 - 轨迹铣削

句法

```
CYCLE72(<_KNAME>, <_RTP>, <_RFP>, <_SDIS>, <_DP>, <_MID>, <_FAL>,
<_FALD>, <_FFP1>, <_FFD>, <_VARI>, <_RL>, <_AS1>, <_LP1>, <_FF3>,
<_AS2>, <_LP2>,<_UMODE>, <_FS>, <_ZFS>, <_GMODE>, <_DMODE>,
<_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义
1		<_KNAME>	字符串 [141]	轮廓子程序名称
2	RP	<_RTP>	REAL	退回平面（绝对）
3	Z0	<_RFP>	REAL	刀具轴的参考点（绝对）
4	SC	<_SDIS>	REAL	安全距离（和参考点的间距，无符号）
5	Z1	<_DP>	REAL	终点，最终深度（绝对/增量），参见 <_AMODE>
6	DZ	<_MID>	REAL	最大切深（增量，无符号）
7	UXY	<_FAL>	REAL	轮廓边沿的精加工余量（增量）
8	UZ	<_FALD>	REAL	底部的精加工余量（增量），无符号
9	FX	<_FFP1>	REAL	轮廓进给率
10	FZ	<_FFD>	REAL	深度方向的进给率（或三维空间内的进给）

编号	对话框参数	内部参数	数据类型	含义		
11		<_VARI>	INT	加工方式		
				个位：	加工	
					1 =	粗加工
					2 =	精加工
					5 =	倒角
				十位：		
					0 =	以 G0 中间位移
					1 =	以 G1 中间位移
				百位：	在轮廓结束处退回	
					0 =	在轮廓结束处退刀到参考点
					1 =	在轮廓结束处退刀到参考点 + <_SDIS>
					2 =	在轮廓结束处退回 <_SDIS>
					3 =	在轮廓结束处不退刀，以轮廓加工进给率进刀到下一个起点
				千位：	保留	
				万位：	加工轮廓	
					0 =	向前加工轮廓
					1 =	向后加工轮廓 在“轮廓向后加工”时： • 最多 170 个轮廓元素，包含倒角和倒圆 • 只计算平面 (X/Y) 中的值和 F 值
12		<_RL>	INT	加工方向		
					40 =	沿轮廓中心（G40，走刀/退刀：直线或垂直）
					41 =	沿轮廓左侧（G41，走刀/退刀：直线或圆弧）
					42 =	沿轮廓右侧（G42，走刀/退刀：直线或圆弧）

编号	对话框参数	内部参数	数据类型	含义		
13		<_AS1>	INT	走刀（逼近轮廓）方式		
				个位：		
					1 =	直线
					2 =	四分圆
					3 =	半圆
					4 =	垂直逼近和回退
				十位：		
					0 =	平面内的最后一个动作
					1 =	三维空间内的最后一个动作
14	L1	<_LP1>	REAL	走刀距离，或走刀半径（增量），无符号		
15	FZ	<_FF3>	REAL	定位进给率（G94/G95 同轮廓上相同）		
16		<_AS2>	INT	退刀（离开轮廓）方式，不适用于垂直走刀和退刀		
				个位：		
					1 =	直线
					2 =	四分圆
					3 =	半圆
				十位：		
					0 =	平面内的最后一个动作
					1 =	三维空间内的最后一个动作
17	L2	<_LP2>	REAL	退刀距离或退刀半径（增量），无符号		
18		<_UMODE>	INT	保留		
19	FS	<_FS>	REAL	倒角宽度（增量）		
20	ZFS	<_ZFS>	REAL	加工倒角时的插入深度（刀尖），参见 <_AMODE>		
21		<_GMODE>	INT	几何值计算模式（所编写的几何值）		
				个位：	保留	
				十位：	保留	
				百位：	加工/仅计算起点	
					0 =	兼容模式
					1 =	正常加工

编号	对话框参数	内部参数	数据类型	含义		
22		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/18/19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
				十位：	进给方式：G 功能组(G94/G95），用于平面进给以及深度进给	
					0 =	兼容模式
					1 =	G 代码保持调用循环前的状态。用于平面进给以及深度进给的 G94/G95 相同
				千位：		
					0 =	兼容模式：轮廓名称在 <_KNAME> 中
					1 =	在 CYCLE62 中写入了轮廓名称，并将它传送到 _SC_CONT_NAME 中
23		<_AMODE>	INT	替代模式		
				个位：	终点 Z1 （<_DP>）	
					0 =	绝对（兼容模式）
					1 =	增量
				十位：	切宽的单位	
					0 =	毫米，英寸
					1 =	保留
				百位：	加工倒角时的插入深度（<_ZFS>）	
					0 =	绝对
					1 =	增量

说明

如果间接将以下传递参数作为“参数”写入程序，则无法反译输入对话框：

```
<_VARI>, <_RL>, <_AS1>, <_AS2>, <_UMODE>, <_GMODE>, <_DMODE>,
<_AMODE>
```

20.1.17 CYCLE76 - 铣削矩形凸台

句法

```
CYCLE76(<_RTP>, <_RFP>, <_SDIS>, <_DP>, <_DPR>, <_LENG>, <_WID>,
<_CRAD>, <_PA>, <_PO>, <_STA>, <_MID>, <_FAL>, <_FALD>, <_FFP1>,
<_FFD>, <_CDIR>, <_VARI>, <_AP1>, <_AP2>, <_FS>, <_ZFS>,
<_GMODE>, <_DMODE>, <_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义
1	RP	<_RTP>	REAL	退回平面（绝对）
2	Z0	<_RFP>	REAL	刀具轴的参考点（绝对）
3	SC	<_SDIS>	REAL	安全距离（和参考点的间距，无符号）
4	Z1	<_DP>	REAL	钻深（绝对）
5		<_DPR>	REAL	相对于 Z0 的钻深（增量），无符号
6	L	<_LENG>	REAL	凸台长度，参见 <_GMODE>（无符号）
7	W	<_WID>	REAL	凸台宽度，参见 <_GMODE>（无符号）
8	R	<_CRAD>	REAL	凸台拐角半径（无符号）
9	X0	<_PA>	REAL	凸台参考点第 1 轴的位置（绝对）
10	Y0	<_PO>	REAL	凸台参考点第 2 轴的位置（绝对）
11	α0	<_STA>	REAL	纵向轴(L)轴和第 1 轴的夹角
12	DZ	<_MID>	REAL	最大切深（增量，无符号）
13	UXY	<_FAL>	REAL	轮廓边沿的精加工余量（增量）
14	UZ	<_FALD>	REAL	底部的精加工余量（增量），无符号
15	FX	<_FFP1>	REAL	轮廓进给率
16	FZ	<_FFD>	REAL	深度进给率

编号	对话框参数	内部参数	数据类型	含义		
17		<_CDIR>	INT	铣削方向（无符号）		
				个位：		
					0 =	顺铣
					1 =	逆铣
18		<_VARI>	INT	加工		
				个位：		
					1 =	粗加工
					2 =	精加工
					5 =	倒角
19	L1	<_AP1>	REAL	凸台毛坯长度		
20	W1	<_AP2>	REAL	凸台毛坯宽度		
21	FS	<_FS>	REAL	倒角宽度（增量）		
22	ZFS	<_ZFS>	REAL	加工倒角时的插入深度（绝对，增量），参见 <_AMODE>		

编号	对话框参数	内部参数	数据类型	含义	
23		<_GMODE>	INT	几何值计算模式（所编写的几何值）	
				个位：	保留
				十位：	保留
				百位：	加工/仅计算起点
					0 = 兼容模式
					1 = 正常加工
				千位：	通过中心或拐点标注凸台尺寸
					0 = 兼容模式
					1 = 通过中心标注尺寸
					2 = 标注拐点， 凸台 +L +W
					3 = 标注拐点， 凸台 -L +W
					4 = 标注拐点， 凸台 +L -W
					5 = 标注拐点， 凸台 -L -W
				万位：	完整加工/二次加工
					0 = 兼容模式
					1 = 完整加工
					2 = 二次加工

编号	对话框参数	内部参数	数据类型	含义		
24		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/18/19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
				十位：	---	保留
				百位：	---	保留
				千位：	---	保留
				万位：	工艺对话框内的工艺调整 (页 911)	
					0 =	输入：完整
					1 =	输入： 简单
25		<_AMODE>	INT	替代模式		
				个位：	最终深度 Z1（DP）	
					0 =	兼容性
					1 =	增量
					2 =	绝对
				十位：	保留	
				百位：	加工倒角时的插入深度（ZFS）	
					0 =	绝对
					1 =	增量

20.1.18 CYCLE77 - 铣削圆形凸台

句法

```
CYCLE77(<_RTP>, <_RFP>, <_SDIS>, <_DP>, <_DPR>, <_CDIAM>, <_PA>,
<_PO>, <_MID>, <_FAL>, <_FALD>, <_FFP1>, <_FFD>, <_CDIR>,
<_VARI>, <_AP1>, <_FS>, <_ZFS>, <_GMODE>, <_DMODE>, <_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义		
1	RP	<_RTP>	REAL	退回平面（绝对）		
2	Z0	<_RFP>	REAL	刀具轴的参考点（绝对）		
3	SC	<_SDIS>	REAL	安全距离（和参考点的间距，无符号）		
4	Z1	<_DP>	REAL	钻深（绝对）		
5		<_DPR>	REAL	相对于 Z0 的钻深（增量），无符号		
6	∅	<_CDIAM>	REAL	凸台直径（无符号）		
7	X0	<_PA>	REAL	凸台参考点第 1 轴的位置（绝对）		
8	Y0	<_PO>	REAL	凸台参考点第 2 轴的位置（绝对）		
9	DZ	<_MID>	REAL	最大切深（增量，无符号）		
10	UXY	<_FAL>	REAL	轮廓边沿的精加工余量（增量）		
11	UZ	<_FALD>	REAL	底部的精加工余量（增量），无符号		
12	FX	<_FFP1>	REAL	轮廓进给率		
13	FZ	<_FFD>	REAL	深度进给率		
14		<_CDIR>	INT	铣削方向（无符号）		
				个位：		
					0 =	顺铣
					1 =	逆铣
15		<_VARI>	INT	加工方式		
				个位：	加工	
					1 =	粗加工，保留精加工余量
					2 =	精加工（余量 X/Y/Z=0）
					5 =	倒角
16	∅1	<_AP1>	REAL	凸台坯件的直径		
17	FS	<_FS>	REAL	倒角宽度（增量）		
18	ZFS	<_ZFS>	REAL	加工倒角时的插入深度（刀尖），绝对/增量，参见 <_AMODE>		

编号	对话框参数	内部参数	数据类型	含义		
19		<_GMODE>	INT	几何值计算模式（所编写的几何值）		
				个位：	保留	
				十位：	保留	
				百位：	加工/仅计算起点	
					0 =	兼容模式
					1 =	正常加工
				千位：	保留	
				万位：	完整加工/二次加工	
					0 =	0 = 兼容模式（<_AP1>与之前的处理方式一样）
					1 =	完整加工
					2 =	二次加工
20		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/18/19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
				十位：	---	保留
				百位：	---	保留
				千位：	---	保留
				万位：	工艺对话框内的工艺调整 (页 911)	
					0 =	输入：完整
					1 =	输入： 简单

编号	对话框参数	内部参数	数据类型	含义		
21		<_AMODE>	INT	替代模式		
				个位：	最终深度 Z1（DP）	
					0 =	绝对（兼容模式）
					1 =	增量
					2 =	绝对
				十位：	保留	
				百位：	加工倒角时的插入深度（ZFS）	
					0 =	绝对
					1 =	增量

20.1.19 CYCLE78 - 螺纹铣削

句法

```
CYCLE78(<_RTP>, <_RFP>, <_SDIS>, <_DP>, <_ADPR>, <_FDPR>,
<_LDPR>, <_DIAM>, <_PIT>, <_PITA>, <_DAM>, <_MDEP>, <_VARI>,
<_CDIR>, <_GE>, <_FFD>, <_FRDP>, <_FFR>, <_FFP2>, <_FFA>,
<_PITM>, <_PTAB>, <_PTABA>, <_GMODE>, <_DMODE>, <_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义
1	RP	<_RTP>	REAL	回退平面（绝对）
2	Z0	<_RFP>	REAL	刀具轴的参考点（绝对）
3	SC	<_SDIS>	REAL	安全距离（和参考点的间距，无符号）
4	Z1	<_DP>	REAL	最终钻深（绝对/增量），参见 <_AMODE>
5		<_ADPR>	REAL	慢速孔定位时的钻深（增量），和 <_VARI> 万位一起生效
6	D	<_FDPR>	REAL	最大切深（增量） D ≥ Z1 ⇒ 一刀到底 D < Z1 ⇒ 深度钻削循环，多次进刀，带排屑
7	ZR	<_LDPR>	REAL	钻穿时的剩余钻深（增量），采用进给率 FR
8	∅	<_DIAM>	REAL	螺纹额定直径
9	P	<_PIT>	REAL	螺距值

编号	对话框参数	内部参数	数据类型	含义		
10		<_PITA>	INT	计算螺距 P		
					1 =	螺距，单位毫米/转
					2 =	螺距，单位牙/英寸
					3 =	螺距，单位英寸/转
					4 =	螺距，单位 MODUL
11	DF	<_DAM>	REAL	后续每刀切削量或下调百分比（递减），参见 <_AMODE>		
12	V1	<_MDEP>	REAL	最小切削量（增量，只针对切削下调量为百分比时）		
13		<_VARI>	INT	加工方式		
				个位：	保留	
				十位：	在螺纹铣削前排屑	
					0 =	螺纹铣削前不排屑（仅限最终钻深上）
					1 =	螺纹铣削前排屑（仅限最终钻深上）
				百位：	右旋/左旋螺纹	
					0 =	右旋螺纹
					1 =	左旋螺纹
				千位：	采用钻削进给率的剩余钻深	
					0 =	无采用钻削进给率 FR 的剩余钻深
					1 =	采用钻削进给率 FR 的剩余钻深
				万位：	慢速孔定位	
					0 =	无慢速孔定位
					1 =	慢速孔定位 进给率 = 0.3 F1，当 F1< 0.15 毫米/转时 进给率 = 0.1 毫米/转，当 F1 ≥ 0.15 毫米/转时
14		<_CDIR>	INT	铣削方向	0 =	顺铣
					1 =	逆铣
					4 =	逆铣 + 顺铣（粗加工 + 精加工组合）

编号	对话框参数	内部参数	数据类型	含义		
15	Z2	<_GE>	REAL	螺纹铣削前的返回量（增量）		
16	F1	<_FFD>	REAL	钻削进给率（毫米/分钟、英寸/分钟或毫米/转）		
17	FR	<_FRDP>	REAL	剩余钻深的进给率（毫米/分钟或毫米/转）		
18	F2	<_FFR>	REAL	螺纹铣削进给率（毫米/分钟或毫米/齿）		
19	FS	<_FFP2>	REAL	<_CDIR> = 4 的精加工进给率（毫米/分钟或毫米/齿）		
20		<_FFA>	INT	计算进给率		
				个位：	钻削进给率 F1	
				十位：	剩余钻深的进给率 FR	
				百位：	螺纹铣削的进给率 F2	
				千位：	精加工进给率 FS	
21		<_PITM>	STRING[15]	表示螺距输入方式的字符串（“仅供界面显示”）[1]		
22		<_PTAB>	STRING[20]	表示螺纹表的字符串 （“”，“ISO”，“BSW”，“BSP”，“UNC”）（“仅供界面显示”）[1]		
23		<_PTABA>	STRING[20]	表示螺纹表中各个选项的字符串（例如：“M 10”，“M 12”， ...）（“仅供界面显示”）[1]		
24		<_GMODE>	INT	几何值计算模式（所编写的几何值），预留		
25		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/18/19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
26		<_AMODE>	INT	替代模式		
				个位：	钻深 = 最终钻深 Z1（绝对/增量）	
					0 =	绝对
					1 =	增量
				十位：	后续每刀切削量或下调百分比 DF	
					0 =	绝对值
					1 =	百分比（0.001 到 100 %)

说明

[1)] 参数 21、 22 和 23 只用于输入对话框螺纹表中的螺纹选择。 在处理循环期间，无法通过循环定义来访问螺纹表。

20.1.20 CYCLE79 - 多边形

句法

```
CYCLE79(<_RTP>, <_RFP>, <_SDIS>, <_DP>, <_NUM>, <_SWL>, <_PA>,
<_PO>, <_STA>, <_RC>, <_AP1>, <_MIDA>, <_MID>, <_FAL>, <_FALD>,
<_FFP1>, <_CDIR>, <_VARI>, <_FS>, <_ZFS>, <_GMODE>, <_DMODE>,
<_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义
1	RP	<_RTP>	REAL	退回平面（绝对）
2	Z0	<_RFP>	REAL	刀具轴的参考点（绝对）
3	SC	<_SDIS>	REAL	安全距离（和参考点的间距，无符号）
4	Z1	<_DP>	REAL	多边形深度（绝对/增量），参见 <_AMODE>
5	N	<_NUM>	INT	边数 (1...n)
6	SW / L	<_SWL>	REAL	对边宽度或边长（根据 <_VARI>） （对边宽度为“SW”，边长为“L”） 只有偶数边数才有对边宽度
7	X0	<_PA>	REAL	凸台参考点的第 1 轴位置（绝对）
8	Y0	<_PO>	REAL	凸台参考点的第 2 轴位置（绝对）
9	α0	<_STA>	REAL	边中点和第 1 轴（X 轴）所成旋转角
10	R1/FS1	<_RC>	REAL	<_NUM> > 2 时的拐角倒圆或倒角，参见 <_AMODE>， 增量，无符号 （R1 表示倒圆，FS1 表示倒角）
11	∅	<_AP1>	REAL	凸台的管直径
12	DXY	<_MIDA>	REAL	最大切宽（单位，参见<_AMODE>）
13	DZ	<_MID>	REAL	最大切深
14	UXY	<_FAL>	REAL	边沿精加工余量

编号	对话框参数	内部参数	数据类型	含义		
15	UZ	<_FALD>	REAL	底部精加工余量		
16	F	<_FFP1>	REAL	加工进给率		
17		<_CDIR>	INT	铣削方向	0 =	顺铣
					1 =	逆铣
18		<_VARI>	INT	加工方式		
				个位：	加工	
					1 =	粗加工
					2 =	精加工
					3 =	边沿精加工
					5 =	倒角
				十位：	对边宽度或边长	
					0 =	对边宽度
					1 =	边沿长度
19	FS	<_FS>	REAL	倒角宽度（增量）		
20	ZFS	<_ZFS>	REAL	加工倒角时的插入深度（刀尖），参见 <_AMODE>		
21		<_GMODE>	INT	几何值计算模式（所编写的几何值）		
				个位：	保留	
				十位：	保留	
				百位：	加工/仅计算起点	
					1 =	正常加工

编号	对话框参数	内部参数	数据类型	含义		
22		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/18/19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
				十位：	---	保留
				百位：	---	保留
				千位：	---	保留
				万位：	工艺对话框内的工艺调整 (页 911)	
					0 =	输入：完整
					1 =	输入： 简单
23		<_AMODE>	INT	替代模式		
				个位：	最终深度（<_DP>）	
					0 =	绝对
					1 =	增量
				十位：	切宽的单位（<_MIDA>）	
					0 =	mm
					1 =	刀具半径的 %
				百位：	加工倒角时的插入深度（<_ZFS>）	
					0 =	绝对
					1 =	增量
				千位：	角度倒圆（<_RC>）	
					0 =	半径
					1 =	倒角

20.1.21 CYCLE81 - 钻削，钻中心孔

句法

```
CYCLE81(<RTP>, <RFP>, <SDIS>, <DP>, <DPR>, <DTB>, <_GMODE>,
<_DMODE>, <_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义		
1	RP	<RTP>	REAL	回退平面（绝对）		
2	Z0	<RFP>	REAL	参考点（绝对）		
3	SC	<SDIS>	REAL	安全距离（和参考点的间距，无符号）		
4	Z1/∅	<DP>	REAL	钻深（绝对）/钻孔直径（绝对），参见 <_GMODE>		
5	Z1	<DPR>	REAL	钻深（增量）		
6	DT	<DTB>	REAL	在最终钻深处的停留时间，参见 <_AMODE>		
7		<_GMODE>	INT	几何值计算模式（所编写的几何值）		
				个位：	保留	
				十位：	钻中心孔，相对于深度或直径	
					0 =	兼容性，深度
					1 =	直径
8		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/G18/G19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）

编号	对话框参数	内部参数	数据类型	含义		
9		<_AMODE>	INT	替代模式		
				个位：	钻深 Z1（绝对/增量）	
					0 =	兼容性，取决于 DP/DPR
					1 =	增量
					2 =	绝对
				十位：	在最终钻深处的停留时间 DT，单位为秒或转	
					0 =	兼容性，取决于 DTB 的符号（> 0 则单位为秒； < 0 则单位为转）
					1 =	单位秒
					2 =	单位转

20.1.22 CYCL82 - 钻削, 锪平面

句法

```
CYCLE82(<RTP>, <RFP>, <SDIS>, <DP>, <DPR>, <DTB>, <_GMODE>,
<_DMODE>, <_AMODE>, <_VARI>, <S_ZA>, <S_FA>, <S_ZD>, <S_FD>)
```

参数

编号	对话框参数	内部参数	数据类型	含义
1	RP	<RTP>	REAL	回退平面（绝对）
2	Z0	<RFP>	REAL	参考点（绝对）
3	SC	<SDIS>	REAL	安全距离（和参考点的间距，无符号）
4	Z1	<DP>	REAL	钻深（绝对），参见 <_AMODE>
5	Z1	<DPR>	REAL	钻深（增量），参见 <_AMODE>
6	DT	<DTB>	REAL	在最终钻深处的停留时间，参见 <_AMODE>

编号	对话框参数	内部参数	数据类型	含义		
7		<_GMODE>	INT	几何值计算模式（所编写的几何值）		
				个位：	保留	
				十位：	钻削深度相对于刀尖或刀柄	
					0 =	兼容性，刀尖
					1 =	刀柄
8		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/G18/G19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
				十位：	保留	
				百位：	保留	
				千位：	保留	
				万位：	工艺对话框内的工艺调整 (页 911)	
					0 =	输入： 完整
					1 =	输入： 简单

编号	对话框参数	内部参数	数据类型	含义		
9		<_AMODE>	INT	替代模式		
				个位：	钻深 Z1（绝对/增量）	
					0 =	兼容性，取决于 DP/DPR
					1 =	增量
					2 =	绝对
				十位：	在最终钻深处的停留时间 DT，单位为秒或转	
					0 =	兼容性，取决于 DT 的符号（> 0 则单位为秒； < 0 则单位为转）
					1 =	单位秒
					2 =	单位转
				百位：	孔定位时的钻深 ZA，绝对/增量	
					0 =	增量
					1 =	绝对
				千位：	孔定位进给率	
					0 =	钻削进给率的 %
					1 =	F/min
					2 =	F/U
				万位：	剩余钻深 ZD，绝对/增量	
					0 =	增量
					1 =	绝对
				十万位：	剩余钻深进给率	
					0 =	钻削进给率的 %
					1 =	F/min
					2 =	F/U

编号	对话框参数	内部参数	数据类型	含义	
10		<_VARI>	INT	加工方式：孔定位/底部钻削	
				个位：	保留
				十位：	保留
				百位：	保留
				千位：	底部钻削
					0 = 底部钻削“否”
					1 = 底部钻削“是”
				万位：	孔定位
					0 = 孔定位“否”
					1 = 孔定位“是”
11	ZA	<S_ZA>	REAL	孔定位深度，绝对深度或为相对于参考点的深度（参见 <_AMODE> 百位）	
12	FA	<S_FA>	REAL	孔定位进给率，数值或 %（参见 <_AMODE> 千位）	
13	ZD	<S_ZD>	REAL	剩余深度，绝对深度或相对于最终钻深的深度（结合 <_AMODE> 万位）	
14	FD	<S_FD>	REAL	剩余钻深进给率，数值或 %（结合 <_AMODE> 十万位）	

20.1.23 CYCLE83 - 深孔钻削

句法

```
CYCLE83(<RTP>, <RFP>, <SDIS>, <DP>, <DPR>, <FDEP>, <FDPR>,
<_DAM>, <DTB>, <DTS>, <FRF>, <VARI>, <_AXN>, <_MDEP>, <_VRT>,
<_DTD>, <_DIS1>, <_GMODE>, <_DMODE>, <_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义
1	RP	<RTP>	REAL	回退平面（绝对）
2	Z0	<RFP>	REAL	参考点（绝对）
3	SC	<SDIS>	REAL	安全距离（和参考点的间距，无符号）
4	Z1	<DP>	REAL	最终钻深（绝对），参见 <_AMODE>
5	Z1	<DPR>	REAL	最终钻深（增量），参见 <_AMODE>

编号	对话框参数	内部参数	数据类型	含义		
6	D	<FDEP>	REAL	第 1 钻深（绝对），参见 <_AMODE>		
7	D	<FDPR>	REAL	第 1 钻深（增量），参见 <_AMODE>		
8	DF	<_DAM>	REAL	后续每刀切削量或下调百分比，参见 <_AMODE>		
9	DTB	<DTB>	REAL	在钻深处的停留时间，参见 <_AMODE>		
10	DTS	<DTS>	REAL	起点上的停留时间（仅在排屑时），参见 <_AMODE>		
11	FD1	<FRF>	REAL	首刀进给率百分比，参见 <_AMODE>		
12		<VARI>	INT	加工方式		
				个位：	断屑/排屑	
					0 =	断屑
					1 =	排屑
13		<_AXN>	INT	刀具轴		
					0 =	第 3 几何轴
					1 =	第 1 几何轴
					2 =	第 2 几何轴
					> 2	第 3 几何轴
14	V1	<_MDEP>	REAL	最小切削量（只针对切削下调量为百分比时）		
15	V2	<_VRT>	REAL	每次加工后的退刀量（仅在断屑时）		
					> 0	可变的退刀量
					0 =	默认值 1 mm
16	DT	<_DTD>	REAL	在最终钻深处的停留时间，参见 <_AMODE>		
17	V3	<_DIS1>	REAL	提前距离（仅在排屑时），参见 <_AMODE>		
18		<_GMODE>	INT	几何值计算模式（所编写的几何值）		
				个位：	保留	
				十位：	钻削深度相对于刀尖或刀柄	
					0 =	刀尖
					1 =	刀柄

编号	对话框参数	内部参数	数据类型	含义		
19		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/G18/G19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
				十位：	---	保留
				百位：	---	保留
				千位：	---	保留
				万位：	工艺对话框内的工艺调整 (页 911)	
					0 =	输入：完整
					1 =	输入：简单

编号	对话框参数	内部参数	数据类型	含义		
20		<_AMODE>	INT	替代模式		
				个位：	钻深 = 最终钻深 Z1（绝对/增量）	
					0 =	兼容性，取决于 <DP>/<DPR>
					1 =	增量
					2 =	绝对
				十位：	首个钻深处的停留时间 DTB，单位为秒或转	
					0 =	兼容性，取决于 DTB 的符号（> 0 则单位为秒； < 0 则单位为转）
					1 =	单位秒
					2 =	单位转
				百位：	在起点处的停留时间 DTS，单位为秒或转	
					0 =	兼容性，取决于 DTS 的符号（> 0 则单位为秒； < 0 则单位为转）
					1 =	单位秒
					2 =	单位转
				千位：	在最终钻深处的停留时间 DTD，单位为秒或转	
					0 =	兼容性，取决于 DTD 的符号（> 0 则单位为秒； < 0 则单位为转）
					1 =	单位秒
					2 =	单位转
				万位：	第 1 钻深 D（绝对/增量）	
					0 =	兼容性，取决于 <FDEPF>/<DPR>
					1 =	增量
					2 =	绝对

编号	对话框参数	内部参数	数据类型	含义		
				十万位：	后续每刀切削量或下调百分比 <_DAM>	
					0 =	兼容性，取决于 <_DAM> 的符号（> 0 则为进给量；< 0 则为 “0.001 到 1.0” 之间的某个系数）
					1 =	递减量
					2 =	百分比（0.001 到 100 %)
				百万位：	提前距离 V3，自动/手动	
					0 =	兼容性，取决于 <_DIS1> 的符号（= 0 则为自动； > 0 则为手动）
					1 =	自动（在循环中计算得出）
					2 =	手动（手动输入的数值）
				千万位：	首刀进给率系数/百分比 <FRF>	
					0 =	兼容性，系数（0.001 到 1.0， FRF = 0 表示 100 %）
					1 =	百分比（0.001 到 999.999 %)

20.1.24 CYCLE84 – 刚性攻丝

句法

```
CYCLE84(<RTP>, <RFP>, <SDIS>, <DP>, <DPR>, <DTB>, <SDAC>, <MPIT>,
<PIT>, <POSS>, <SST>, <SST1>, <_AXN>, <_PITA>, <_TECHNO>,
<_VARI>, <_DAM>, <_VRT>, <_PITM>, <_PTAB>, <_PTABA>, <_GMODE>,
<_DMODE>, <_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义
1	RP	<RTP>	REAL	回退平面（绝对）
2	Z0	<RFP>	REAL	参考点（绝对）

编号	对话框参数	内部参数	数据类型	含义			
3	SC	<SDIS>	REAL	安全距离（和参考点的间距，无符号）			
4	Z1	<DP>	REAL	钻深 = 最终钻深（绝对），参见 <_AMODE>			
5	Z1	<DPR>	REAL	钻深 = 最终钻深（增量），参见 <_AMODE>			
6	DT	<DTB>	REAL	在钻深上的停留时间，以秒为单位			
7	SDE	<SDAC>	INT	循环结束之后的旋转方向			
8		<MPIT>	REAL	螺纹尺寸只限“ISO 米制”（在运行时会内部换算螺距）			
9	P	<PIT>	REAL	螺距值，单位参见 <_PITA>			
10	αS[1)]	<POSS>	REAL	主轴定向停止时的位置			
11	S	<SST>	REAL	攻丝时的主轴转速			
12	SR	<SST1>	REAL	回退的主轴速度			
13		<_AXN>	INT	钻孔轴	0 =	3. 几何轴	
					1 =	1. 几何轴	
					2 =	2. 几何轴	
					≥ 3 =	3. 几何轴	
14		<_PITA>	INT	螺距的单位（计算 <PIT> 和 <MPIT>）			
					0 =	螺距,单位毫米	- 计算 <MPIT>/<PIT>
					1 =	螺距,单位毫米	- 计算 <PIT>
					2 =	螺距,单位 TPI	- 计算 <PIT>（螺纹牙/英寸）
					3 =	螺距,单位英寸	- 计算 <PIT>
					4 =	模数	- 计算 <PIT>

编号	对话框参数	内部参数	数据类型	含义		
15		<_TECHNO>	INT	工艺 1)		
				个位：	准停特性	
					0 =	调用循环前的准停特性保持有效
					1 =	准停 G601
					2 =	准停 G602
					3 =	准停 G603
				十位：	前馈	
					0 =	带/不带前馈控制，保持调用循环前的状态
					1 =	带前馈控制 FFWON
					2 =	不带前馈控制 FFWOF
				百位：	加速度	
					0 =	SOFT/BRISK/DRIVE 保持调用循环前的状态
					1 =	带加加速度限制 SOFT
					2 =	不带加加速度限制 BRISK
					3 =	降低的加速度 DRIVE
				千位：	MCALL 主轴模式	
					0 =	在 MCALL 时，再次激活主轴模式
					1 =	在 MCALL 时，保持位置控制
16		<_VARI>	INT	加工方式		
				个位：	0 =	1 刀到底
					1 =	断屑（深孔攻丝）
					2 =	排屑（深孔攻丝）
				千位：	ISO/SIEMENS 模式和输入对话框不相关	
					0 =	根据 ISO 兼容性调用
					1 =	根据西门子设置调用
17	D	<_DAM>	REAL	最大切深（仅限排屑/断屑）		
18	V2	<_VRT>	REAL	每次加工后的退刀量（仅在断屑时），参见 <_AMODE>		
19		<_PITM>	STRING[15]	表示螺距输入方式的字符串 2)		

编号	对话框参数	内部参数	数据类型	含义		
20		<_PTAB>	STRING[5]	表示螺纹表的字符串 （“”，“ISO”，“BSW”，“BSP”，“UNC”）[2)]		
21		<_PTABA>	STRING[20]	表示螺纹表中各个选项的字符串（例如：“M 10”，“M 12”， ...）[2)]		
22		<_GMODE>	INT	几何值计算模式（所编写的几何值）		
				个位：	保留	
				十位：	保留	
23		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/G18/G19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
				十位：	保留	
				百位：	保留	
				千位：	兼容模式（仅用于输入对话框的回译），当 MD 52216 位 0 = 1[1)]	
					0 =	显示工艺参数（兼容性）： 工艺参数有效
					1 =	不显示工艺参数： 工艺保持调用循环前的状态
				万位：	工艺对话框内的工艺调整 (页 911)	
					0 =	输入：完整
					1 =	输入： 简单

编号	对话框参数	内部参数	数据类型	含义		
24		<_AMODE>	INT	替代模式		
				个位：	钻深 = 最终钻深 Z1（绝对/增量）	
					0 =	兼容性，取决于 <DP>/<DPR>
					1 =	增量
					2 =	绝对
				十位：	保留	
				百位：	保留	
				千位：	螺纹方向：左旋/右旋	
					0 =	兼容性， 取决于 PIT/MPIT 的符号
					1 =	右
					2 =	左
				万位：	保留	
				十万位：	保留	
				百万位：	每次加工后的回退量 V2，手动/自动	
					0 =	兼容性，取决于 <_VRT> （> 0 则为可变值，≤ 0 则为缺省值 1 毫米/0.0394 英寸）
					1 =	自动（缺省值 1 毫米/0.0394 英寸）
					2 =	手动（和 V2 中输入的一样）

[1] 可以通过设定数据“SD52216 $MCS_FUNCTION_MASK_DRILL”隐藏工艺输入栏。

[2] 参数 19、 20 和 21 只用于输入对话框螺纹表中的螺纹选择。 在处理循环期间，无法通过循环定义来访问螺纹表。

20.1.25 CYCLE85 - 铰孔

句法

```
CYCLE85(<RTP>, <RFP>, <SDIS>, <DP>, <DPR>, <DTB>, <FFR>, <RFF>,
<_GMODE>, <_DMODE>, <_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义		
1	RP	<RTP>	REAL	回退平面（绝对）		
2	Z0	<RFP>	REAL	参考点（绝对）		
3	SC	<SDIS>	REAL	安全距离（和参考点的间距，无符号）		
4	Z1	<DP>	REAL	钻深（绝对），参见 <_AMODE>		
5	Z1	<DPR>	REAL	钻深（增量），参见 <_AMODE>		
6	DT	<DTB>	REAL	在最终钻深处的停留时间，参见 <_AMODE>		
7	F	<FFR>	REAL	进给率		
8	FR	<RFF>	REAL	退回进给率		
9		<_GMODE>	INT	保留		
10		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/G18/G19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
11		<_AMODE>	INT	替代模式（钻削）		
				个位：	钻深 Z1（绝对/增量）	
					0 =	兼容性，取决于 DP/DPR
					1 =	增量
					2 =	绝对
				十位：	在最终钻深处的停留时间 DT，单位为秒或转	
					0 =	兼容性，取决于 DT 的符号（> 0 则单位为秒； < 0 则单位为转）
					1 =	单位秒
					2 =	单位转

20.1.26 CYCLE86 - 镗孔

句法

```
CYCLE86(<RTP>, <RFP>, <SDIS>, <DP>, <DPR>, <DTB>, <SDIR>, <RPA>,
<RPO>, <RPAP>, <POSS>, <_GMODE>, <_DMODE>, <_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义		
1	RP	<RTP>	REAL	回退平面（绝对）		
2	Z0	<RFP>	REAL	参考点（绝对）		
3	SC	<SDIS>	REAL	安全距离（和参考点的间距，无符号）		
4	Z1	<DP>	REAL	钻深（绝对），参见 <_AMODE>		
5	Z1	<DPR>	REAL	钻深（增量），参见 <_AMODE>		
6	DT	<DTB>	REAL	在最终钻深处的停留时间，参见 <_AMODE>		
7	DIR	<SDIR>	INT	主轴旋转方向	3 =	M3
					4 =	M4
8	DX	<RPA>	REAL	X 轴的退刀返回量		
9	DY	<RPO>	REAL	Y 轴的退刀返回量		
10	DZ	<RPAP>	REAL	Z 轴的退刀返回量		
11	SPOS	<POSS>	REAL	退刀返回时的主轴位置（用于定向的主轴停止，单位：度）		
12		<_GMODE>	INT	几何值计算模式（所编写的几何值）		
				个位：	退刀返回模式	
					0 =	退刀返回，兼容性
					1 =	不退刀
13		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/G18/G19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）

编号	对话框参数	内部参数	数据类型	含义		
14		<_AMODE>	INT	替代模式		
				个位：	钻深 Z1（绝对/增量）	
					0 =	兼容性，取决于 <DP>/<DPR>
					1 =	增量
					2 =	绝对
				十位：	在最终钻深处的停留时间 DT，单位为秒或转	
					0 =	兼容性，取决于 DT 的符号（> 0 则单位为秒； < 0 则单位为转）
					1 =	单位秒
					2 =	单位转

20.1.27 CYCLE92 - 切断

句法

```
CYCLE92(<_SPD>, <_SPL>, <_DIAG1>, <_DIAG2>, <_RC>, <_SDIS>,
<_SV1>, <_SV2>, <_SDAC>, <_FF1>, <_FF2>, <_SS2>, <_DIAGM>,
<_VARI>, <_DN>, <_DMODE>, <_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义
1	X0	<_SPD>	REAL	参考点（绝对，始终为直径）
2	Y0	<_SPL>	REAL	参考点（绝对）
3	X1	<_DIAG1>	REAL	减速加工的深度，参见 <_AMODE>（个位）
4	X2	<_DIAG2>	REAL	最终深度，参见 <_AMODE>（十位）
5	R/FS	<_RC>	REAL	倒圆半径或倒角宽度，参见<_AMODE>（千位）
6	SC	<_SDIS>	REAL	安全距离（和参考点的间距，无符号）
7	S	<_SV1>	REAL	恒定主轴转速，参见 <_AMODE> （万位）
	V			恒定切削速度
8	SV	<_SV2>	REAL	恒定切削速度时的最大转速

编号	对话框参数	内部参数	数据类型	含义		
9	DIR	<_SDAC>	INT	主轴旋转方向	3 =	表示 M3
					4 =	表示 M4
10	F	<_FF1>	REAL	慢速前的进给率		
11	FR	<_FF2>	REAL	慢速进给率，一直加工到最终深度		
12	SR	<_SS2>	REAL	慢速，一直加工到最终深度		
13	XM	<_DIAGM>	REAL	驶出接料箱深度（绝对，始终是直径）		
14		<_VARI>	INT	加工方式		
				个位：	返回	
					0 =	返回到 <_SPD> + <_SDIS>
					1 =	在结束处没有退回
				十位：	工件接料箱	
					0 =	否，不执行 M 指令
					1 =	是，调取 CUST_TECHCYC（101）- 驶出接料箱，CUST_TECHCYC（102）- 关闭接料箱
15		<_DN>	INT	刀沿 2 的 D 号，如果没有写入 ⇒ D+1		
20		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/G18/G19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）

编号	对话框参数	内部参数	数据类型	含义		
21		<_AMODE>	INT	替代模式		
				个位：	慢速加工的深度（<_DIAG1>）	
					0 =	绝对， X 轴坐标，直径
					1 =	增量， X 轴坐标，半径
				十位：	最终深度（<_DIAG2>）	
					0 =	绝对， X 轴坐标，直径
					1 =	增量， X 轴坐标，半径
				百位：	保留	
				千位：	半径/倒角（<_RC>）	
					0 =	半径
					1 =	倒角
				万位：	主轴转数/切削速度（<_SV1>）	
					0 =	恒定主轴转速
					1 =	恒定切削速度

20.1.28 CYCLE95 - 切削轮廓

句法

```
CYCLE95(<NPP>, <MID>, <FALZ>, <FALX>, <FAL>, <FF1>, <FF2>, <FF3>,
<_VARI>, <DT>, <DAM>, <_VRT>, <_GMODE>, <_DMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义
1	CON	<NPP>	字符串 [140]	轮廓名
2	D	<MID>	REAL	粗加工时的最大切深，参见 <_GMODE>
3	UZ	<FALZ>	REAL	Z 轴的精加工余量
4	UX	<FALX>	REAL	X 轴的精加工余量
5	U	<FAL>	REAL	与轮廓平行的精加工余量（两个轴上均有效）
6	F	<FF1>	REAL	粗加工进给
7	FY	<FF2>	REAL	退刀槽插入进给率

编号	对话框参数	内部参数	数据类型	含义		
8	FS	<FF3>	REAL	精加工进给率		
9		<_VARI>	INT	加工方式		
				个位和十位：		
					1 =	粗加工，纵向，外侧
					2 =	粗加工，横向，外侧
					3 =	粗加工，纵向，内侧
					4 =	粗加工，横向，内侧
					5 =	精加工，纵向，外侧
					6 =	精加工，横向，外侧
					7 =	精加工，纵向，内侧
					8 =	精加工，横向，内侧
					9 =	完整加工，纵向，外侧
					10 =	完整加工，横向，外侧
					11 =	完整加工，纵向，内侧
					12 =	完整加工，横向，内侧
				百位：		
					0 =	沿轮廓返回，无余角
					1 =	在轮廓上不带拉削
					2 =	在前一个交点前沿轮廓返回，可能产生余角
10	DT	<DT>	REAL	进给中断点上的停留时间		
11	DI	<DAM>	REAL	进给中断的距离		
12	VRT	<_VRT>	REAL	从轮廓离开的退刀量		
					0 =	不管设置的单位制是公制还是英制，内部都会使用 1 毫米的退刀量。
					> 0 =	退刀量

编号	对话框参数	内部参数	数据类型	含义		
13		<_GMODE>	INT	几何值计算模式（所编写的几何值）		
				个位：	进刀深度	
					0 =	进刀深度是半径值还是直径值依据 G 组 DIAMON/DIAMOF
					1 =	进刀深度作为半径值，不受 DIAMON/DIAMOF 的影响
14		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/G18/G19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
				千位：		
					0 =	兼容模式：轮廓名称在 NPP 中
					1 =	在 CYCLE62 中写入了轮廓名称，并将它传送到 _SC_CONT_NAME 中

20.1.29 CYCLE98 - 螺纹链

句法

```
CYCLE98(<_PO1>, <_DM1>, <_PO2>, <_DM2>, <_PO3>, <_DM3>, <_PO4>,
<_DM4>, <APP>, <ROP>, <TDEP>, <FAL>, <_IANG>, <NSP>, <NRC>,
<NID>, <_PP1>, <_PP2>, <_PP3>, <_VARI>, <_NUMTH>, <_VRT>, <_MID>,
<_GDEP>, <_IFLANK>, <_PITA>, <_PITM1>, <_PITM2>, <_PITM3>,
<_DMODE>, <_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义
1	Z0	<_PO1>	REAL	参考点 Z 轴坐标（绝对）
2	X0	<_DM1>	REAL	参考点 X 轴坐标（绝对），始终是直径

编号	对话框参数	内部参数	数据类型	含义	
3	Z1	<_PO2>	REAL	中间点 1 的 Z 轴坐标（绝对/增量），参见 <_AMODE>（个位）	
4	X1	<_DM2>	REAL	中间点 1 的 X 轴坐标（绝对/增量），参见 <_AMODE>（十位）或	
	X1α			螺纹斜度 1 （-90°到 90°） 绝对值始终是直径值，增量值始终是半径值	
5	Z2	<_PO3>	REAL	中间点 2 的 Z 轴坐标（绝对/增量），参见 <_AMODE>（百位）	
6	X2	<_DM3>	REAL	中间点 2 的 X 轴坐标（绝对/增量），参见 <_AMODE>（千位）或	
	X2α			螺纹斜度 2 （-90°到 90°） 绝对值始终是直径值，增量值始终是半径值	
7	Z3	<_PO4>	REAL	终点 Z 轴坐标（绝对/增量），参见 <_AMODE>（万位）	
8	X3	<_DM4>	REAL	终点 X 轴坐标（绝对/增量），参见 <_AMODE>（十万位）或	
	X3α			螺纹斜度 3 （-90°到 90°） 绝对值始终是直径值，增量值始终是半径值	
9	LW	<APP>	REAL	螺纹导入量（增量，无符号）	
10	LR	<ROP>	REAL	螺纹导出量（增量，无符号）	
11	H1	<TDEP>	REAL	螺纹深度（增量，无符号）	
12	U	<FAL>	REAL	X 和 Z 轴的精加工余量	
13	DP	<_IANG>	REAL	进刀斜度，距离或角度，参见_<_AMODE> （百万位）	
	αP			进刀斜度根据参数 <_VARI>（百位）的设置生效。	
					<_VARI_HUNDERTER = 0 定义兼容模式：
				> 0 =	在一个齿面上进刀
				0 =	垂直于螺纹进刀
				< 0 =	在交替齿面上进刀
				_VARI_HUNDERTER 定义 <>0：	
				> 0 =	在正向齿面上进刀
				0 =	中心进刀
				< 0 =	在负向齿面上进刀
14	α0	<NSP>	REAL	第 1 个螺纹牙的起始角偏移	
15		<NRC>	INT	粗切次数，参见 <_VARI> （万位）	
16	NN	<NID>	INT	空切次数	

编号	对话框参数	内部参数	数据类型	含义		
17	P0	<_PP1>	REAL	第 1 个螺纹段的螺距，参见<_PITA>		
18	P1	<_PP2>	REAL	第 2 个螺纹段的螺距，参见<_PITA>		
19	P2	<_PP3>	REAL	第 3 个螺纹段的螺距，参见<_PITA>		
20		<_VARI>	INT	加工		
				个位：	工艺	
					1 =	外螺纹采用直线进给率
					2 =	内螺纹采用直线进给率
					3 =	外螺纹采用递减进给率，切削截面保持恒定
					4 =	内螺纹采用递减进给率，切削截面保持恒定
				十位：	保留	
				百位：	进刀方式	
					0 =	<_IANG> 的兼容模式
					1 =	单侧进刀
					2 =	交替进刀
				千位：	保留	
				万位：	切深选择	
					0 =	兼容性，粗切次数（<_NRC>）
					1 =	第 1 个进刀深度（<_MID>）
				十万位：	加工方式	
					0 =	兼容性（粗加工和精加工）
					1 =	粗加工
					2 =	精加工
					3 =	粗加工和精加工
				百万：	多头螺纹的加工顺序	
					0 =	螺纹升序加工
					1 =	螺纹降序加工
21	N	<_NUMTH>	INT	螺纹头数		

编号	对话框参数	内部参数	数据类型	含义		
22		<_VRT>	REAL	返回距离（增量）		
					0 =	不管设置的单位制是公制还是英制，内部都会使用 1 毫米的退刀量。
					> 0 =	退刀量
23	D1	<_MID>	REAL	首次进刀深度，参见 <_VARI>（万位）		
24	DA	<_GDEP>	REAL	螺纹变化深度（仅限“多头”）		
					0 =	不考虑螺纹变化深度
					> 0 =	考虑螺纹变化深度
25		<_IFLANK>	REAL	进刀宽度（仅供界面显示）		
26		<_PITA>	INT	螺距计算		
					0 =	螺距的兼容模式，计算 <_PP1> ~ <_PP3> 和之前一样，依据生效的单位制：公制或英制
					1 =	螺距,单位毫米
					2 =	螺距单位：TPI（每英寸的螺纹牙数）
					3 =	螺距,单位英寸
					4 =	模数
27		<_PITM1>	STRING[15]	表示螺距输入方式的字符串（“仅供界面显示”）		
28		<_PITM2>	STRING[15]	表示螺距输入方式的字符串（“仅供界面显示”）		
29		<_PITM3>	STRING[15]	表示螺距输入方式的字符串（“仅供界面显示”）		

编号	对话框参数	内部参数	数据类型	含义		
30		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/G18/G19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
				十位：	---	保留
				百位：	---	保留
				千位：	---	保留
				万位：	工艺对话框内的工艺调整 (页 911)	
					0 =	输入：完整
					1 =	输入： 简单

编号	对话框参数	内部参数	数据类型	含义		
31		<_AMODE>	INT	替代模式		
				个位：	中间点 1 的 Z 轴坐标(Z1)	
					0 =	绝对
					1 =	增量
				十位：	中间点 1 的 X 轴坐标(X1)	
					0 =	绝对
					1 =	增量
					2 =	α
				百位：	中间点 2 的 Z 轴坐标(Z2)	
					0 =	绝对
					1 =	增量
				千位：	中间点 2 的 X 轴坐标(X2)	
					0 =	绝对
					1 =	增量
					2 =	α
				万位：	终点 Z 轴坐标(Z3)	
					0 =	绝对
					1 =	增量
				十万位：	终点 X 轴坐标(X3)	
					0 =	绝对
					1 =	增量
					2 =	α
				百万：	选择进刀斜面：角度或宽度	
					0 =	进刀角度<_IANG>
					1 =	进刀斜面 <_IFLANK>
				千万位：	单头/多头	
					0 =	兼容模式（计算起始角 <_NSP>）
					1 =	单头（带起始角偏移 <_NSP>）
					2 =	多头

20.1.30 CYCLE99 - 螺纹车削

句法

```
CYCLE99(<_SPL>, <_SPD>, <_FPL>, <_FPD>, <_APP>, <_ROP>, <_TDEP>,
<_FAL>, <_IANG>, <_NSP>, <_NRC>, <_NID>, <_PIT>, <_VARI>,
<_NUMTH>, <_SDIS>, <_MID>, <_GDEP>, <_PIT1>, <_FDEP>, <_GST>,
<_GUD>, <_IFLANK>, <_PITA>, <_PITM>, <_PTAB>, <_PTABA>, <_DMODE>,
<_AMODE>, <_S_XRS>)
```

参数

编号	对话框参数	内部参数	数据类型	含义		
1	Z0	<_SPL>	REAL	参考点（绝对）		
2	X0	<_SPD>	REAL	参考点（绝对，始终为直径）		
3	Z1	<_FPL>	REAL	终点，结合<_AMODE>（个位）		
4	X1	<_FPD>	REAL	终点，结合<_AMODE>（十位）		
5	LW/LW2	<_APP>	REAL	螺纹前置量，结合<_AMODE>（百位）或 螺纹导入长度 = 螺纹导出长度，结合<_AMODE>（百位）		
6	LR	<_ROP>	REAL	螺纹导出长度		
7	H1	<_TDEP>	REAL	螺纹深度		
8	U	<_FAL>	REAL	X 和 Z 轴的精加工余量		
9	DP	<_IANG>	REAL	进刀斜度，距离或角度，结合<_AMODE>（千位）		
	αP				> 0 =	在正向齿面上进刀
					< 0 =	在负向齿面上进刀
					0 =	中心进刀
10	α0	<_NSP>	REAL	起始角设置（仅限“单头”）		
11	ND	<_NRC>	INT	粗切次数，参见<_VARI>（万位）		
12	NN	<_NID>	INT	空切次数		
13	P	<_PIT>	REAL	螺距值，参见<_PITA>		

编号	对话框参数	内部参数	数据类型	含义		
14		<_VARI>	INT	加工方式		
				个位：	工艺	
					1 =	外螺纹采用直线进给率
					2 =	内螺纹采用直线进给率
					3 =	外螺纹采用递减进给率，切削截面保持恒定
					4 =	内螺纹采用递减进给率，切削截面保持恒定
				十位：	保留	
				百位：	进刀方式	
					1 =	单侧进刀
					2 =	交替进刀
				千位：	保留	
				万位：	切深选择	
					0 =	粗切次数（<_NRC>）
					1 =	第 1 个进刀深度（<_MID>）
				十万位：	加工方式	
					1 =	粗加工
					2 =	精加工
					3 =	粗加工和精加工
				百万：	多头螺纹的加工顺序	
					0 =	螺纹升序加工
					1 =	螺纹降序加工
15	N	<_NUMTH>	INT	螺纹头数		
16	VR	<_SDIS>	REAL	返回距离，增量		
17	D1	<_MID>	REAL	首次进刀深度，参见 <_VARI>（万位）		
18	DA	<_GDEP>	REAL	螺纹变化深度（仅限“多头”）		
					0 =	不考虑螺纹变化深度
					> 0 =	考虑螺纹变化深度

编号	对话框参数	内部参数	数据类型	含义		
19	G	<_PIT1>	REAL	每转的螺距变化量		
					0 =	螺距恒定（G33）
					> 0 =	螺距逐渐变大（G34）
					< 0 =	螺距逐渐变小（G35）
20		<_FDEP>	REAL	插入深度（无符号）		
21	N1	<_GST>	INT	起始螺线 N1 = 1...N，参见_<_AMODE>（十万位）		
22		<_GUD>	INT	保留		
23		<_IFLANK>	REAL	进刀宽度（仅供界面显示）		
24		<_PITA>	INT	螺距的单位（计算 PIT 和/或 MPIT）		
					0 =	螺距单位：毫米 - 计算 MPIT/PIT
					1 =	螺距单位：毫米 - 计算 PIT
					2 =	螺距单位：TPI - 计算 PIT（每英寸的螺纹牙数）
					3 =	螺距单位：英寸 - 计算 PIT
					4 =	MODUL - 计算 PIT
25		<_PITM>	STRING[15]	表示螺距输入方式的字符串（“仅供界面显示”）[1)]		
26		<_PTAB>	STRING[20]	表示螺纹表的字符串（“仅供界面显示”）[1)]		
27		<_PTABA>	STRING[20]	表示螺纹表选择的字符串（“仅供界面显示”）[1)]		

编号	对话框参数	内部参数	数据类型	含义		
28		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/G18/G19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
				十位：	螺纹种类	
					0 =	纵向螺纹
					1 =	横向螺纹
					2 =	圆锥螺纹
				百位：	---	保留
				千位：	---	保留
				万位：	工艺对话框内的工艺调整 (页 911)	
					0 =	输入：完整
					1 =	输入： 简单

编号	对话框参数	内部参数	数据类型	含义		
29		<_AMODE>	INT	替代模式		
				个位：	Z 轴的螺纹长度	
					0 =	绝对
					1 =	增量
				十位：	X 轴的螺纹长度	
					0 =	绝对， X 轴坐标，直径
					1 =	增量， X 轴坐标，半径
					2 =	α
				百位：	导入量计算 <_APP>	
					0 =	螺纹前置量<_APP>
					1 =	螺纹导入量 = 螺纹导出量 <_APP> = -<_ROP>
					2 =	指定螺纹导入量 <_APP> = -<_APP>
				千位：	选择进刀斜面：角度或宽度	
					0 =	进刀角度<_IANG>
					1 =	进刀斜面 <_IFLANK>
				万位：	单头/多头	
					0 =	单头（带起始角偏移 <_NSP>）
					1 =	多头
				十万位：	起始螺线<_GST>	
					0 =	完整加工
					1 =	从该螺线开始加工
					2 =	只加工该螺线
				百万位：	纵向螺纹上的悬垂度补偿	
					0 =	球螺纹的高度 XS
					1 =	球螺纹的半径 RS
30	XS / RS	<_S_XRS>	REAL	纵向螺纹上的悬垂度补偿，和 <_AMODE> 一起使用： 百万位		

说明

[1] 参数 <_PITM>, <_PTAB>和<_PTABA>只用于输入对话框螺纹表中的螺纹选择。
在处理循环期间，无法通过循环定义来访问螺纹表。

20.1.31 CYCLE435 - 设置修整器坐标

句法

```
CYCLE435(<_T>, <_DD>, <S_TA>, <S_DA>, <S_AD>, <S_AL>, <S_PVD>,
<S_PVL>, <S_PD>, <S_PL>, <_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义		
1		<_T>	STRING[32]	砂轮的刀具名称		
2		<_DD>	INT	砂轮的刀沿名称		
3		<S_TA>	STRING[32]	修整器基准点 - 修整器名称		
4		<S_DA>	INT	修整器的刀沿编号		
5		<S_AD>	REAL	修整量直径		
6		<S_AL>	REAL	修整量平面		
7		<S_PVD>	REAL	成型磨削偏移直径		
8		<S_PVL>	REAL	成型磨削偏移平面		
9		<S_PD>	REAL	成型余量直径		
10		<S_PL>	REAL	成型余量平面		
11		<_AMODE>	INT	替代模式		
				个位：	循环末尾的有效刀具	
					0 =	修整器生效
					1 =	砂轮生效

20.1.32 CYCLE495 - 成型

句法

```
CYCLE495(<_T>, <_DD>, <_SC>, <_F>, <_VARI>, <_D>, <_DX>, <_DZ>,
<S_PA>, <S_N>, <_DMODE>, <_AMODE>, <S_FW>, <S_HW>)
```

参数

编号	对话框参数	内部参数	数据类型	含义
1		<_T>	STRING[20]	砂轮的刀具名称
2		<_DD>	INT	砂轮的刀沿名称
3		<_SC>	REAL	绕行障碍时的退刀量，增量
4		<_F>	REAL	成型进给率

编号	对话框参数	内部参数	数据类型	含义		
5		<_VARI>	INT	加工方式		
				个位：	成型方式	
					1 =	与轴平行
					2 =	与轮廓平行
				十位：	加工方向	
					0 =	拉 刀沿位置为 1 到 4
					1 =	推 刀沿位置为 1 到 4
					2 =	交替 刀沿位置为 1 到 8
					3 =	头 → 尾 刀沿位置为 1 到 8
					4 =	尾 → 头 刀沿位置为 1 到 8
				百位：	进刀方向	
					1 =	G18 上为 X- 或 G19 上为 Y-
					2 =	G18 上为 X+ 或 G19 上为 Y+
					3 =	G18 和 G19 上均为 Z-
					4 =	G18 和 G19 上均为 Z+
6		<_D>	REAL	“与轴平行”成型时的修整量		
7		<_DX>	REAL	“与轮廓平行”成型时 G18 上的修整量 X 或 G19 上的修整量 Y		
8		<_DZ>	REAL	“与轮廓平行”成型时 G18 和 G19 上的修整量 Z		
9		<S_PA>	REAL	成型余量		
10		<S_N>	INT	成型程序中的冲程次数		

编号	对话框参数	内部参数	数据类型	含义		
11		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/G18/G19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
12		<_AMODE>	INT	替代模式		
				个位：	选择重新成型/继续成型	
					1 =	新建
					2 =	继续
				十位：	选择成型余量	
					0 =	从毛坯到最深点
					1 =	从毛坯到最高点
13		<S_FW>	REAL	修整器的后角		
14		<S_HW>	REAL	修整器的夹持角		

20.1.33 CYCLE800 – 回转

句法

```
CYCLE800(<_FR>, <_TC>, <_ST>, <_MODE>, <_X0>, <_Y0>, <_Z0>, <_A>,
<_B>, <_C>, <_X1>, <_Y1>, <_Z1>, <_DIR>, <_FR_I>, <_DMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义		
1		<_FR>	INT	空转模式：	0 =	无空运行
					1 =	机床轴回退 Z
					2 =	机床轴 Z 轴回退，然后是 X 轴和 Y 轴回退
					3 =	保留
					4 =	刀具方向上回退，最大
					5 =	刀具方向上回退，增量
2		<_TC>	STRING[32]	回转数据组名称：	""	（无名称），当仅有 1 个回转数据组时
					"0"	撤销回转数据组（删除回转框架）

编号	对话框参数	内部参数	数据类型	含义		
3		<_ST>	INT	转换状态		
				个位：		
					0 =	新建，删除回转平面，并重新计算当前参数
					1 =	添加，把回转平面添加到有效的回转平面上
				十位：	是/否跟踪刀尖（只有在调试区设置了“回转”功能后，才有效）	
					0 =	刀尖不跟踪运行
					1 =	跟踪刀尖（TRAORI）
				百位：	调整/对齐刀具（该功能显示在输入对话框“回转”的“刀具”中）	
					0 =	不调整刀具
					1 =	调整刀具（半圆铣刀优先）
					2 =	对齐**车刀**（只有在调试回转时设置了车削工艺的 B 轴运动转换时，才有效）
					3 =	对齐**铣刀**（只有在调试回转时设置了车削工艺的 B 轴运动转换时，才有效）
				千位：	JOG 中回转的内部参数	
				万位：	参见参数 <_DIR>	
					0 =	回转“是”
					1 =	回转“否”，方向“负”[3]
					2 =	回转“否”，方向“正”[3]
				十万位：	参见参数 <_DIR>	
					0 =	兼容性
					1 =	优化“负”方向选择（仅用于操作界面）[4]
					2 =	优化“正”方向选择（仅用于操作界面）[4]

编号	对话框参数	内部参数	数据类型	含义		
4		<_MODE> 5)	INT	回转模式：计算回转角度和回转顺序（位编码！）		
				位：7 6	0 0:	轴的回转角 -> 参见参数 <_A>, <_B>, <_C>
					0 1:	立体空间角 -> 参见参数 <_A>, <_B> 1)
					1 0:	投影角 -> 参见参数 <_A>, <_B>, <_C> 1)
					1 1:	回转模式“回转轴，直接” -> 参见参数 <_A>, <_B> 1)
				位：5 4 3 2 1 0 （对于立体空间角没有作用！）	x x x x 0 1	_A 绕 X 轴第 1 次旋转
					x x x x 1 0	_A 绕 Y 轴第 1 次旋转
					x x x x 1 1	_A 绕 Z 轴第 1 次旋转
					x x 0 1 x x	_B 绕 X 轴第 2 次旋转
					x x 1 0 x x	_B 绕 Y 轴第 2 次旋转
					x x 1 1 x x	_B 绕 Z 轴第 2 次旋转
					0 1 x x x x	_C 绕 X 轴第 3 次旋转
					1 0 x x x x	_C 绕 Y 轴第 3 次旋转
					1 1 x x x x	_C 绕 Z 轴第 3 次旋转
5	X0	<_X0>	REAL	旋转之前参考点的 X 轴坐标		
6	Y0	<_Y0>	REAL	旋转之前参考点的 Y 轴坐标		
7	Z0	<_Z0>	REAL	旋转之前参考点的 Z 轴坐标		
8	X(A)	<_A>	REAL	按照参数 <_MODE> 中的设置进行第 1 次旋转		
9	Y(B)	<_B>	REAL	按照参数 <_MODE> 中的设置进行第 2 次旋转		
10	Z(C)	<_C>	REAL	按照参数 <_MODE> 中的设置进行第 3 次旋转		

编号	对话框参数	内部参数	数据类型	含义		
11	X1	<_X1>	REAL	旋转之后参考点的 X 轴坐标		
12	Y1	<_Y1>	REAL	旋转之后参考点的 Y 轴坐标		
13	Z1	<_Z1>	REAL	旋转之后参考点的 Z 轴坐标		
14	- 或者 +	<_DIR>	INT	触发回转轴的动作（默认 = -1！）		
					-1 =	回转轴 1 或 2 定位到更小位置值上 2）
					+1 =	回转轴 1 或 2 定位到更大位置值上 2）
					0 =	无回转（仅计算回转框架） 1） 3）
15	FR	<_FR_I>	REAL	刀具方向上的空转增量		
16		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/G18/G19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
				十位：	显示对齐刀具时的 β 值	
					0 =	值
					1 =	箭头

说明

如果间接将以下传递参数作为“参数”写入程序，则无法反译输入对话框： <_FR>, <_ST>, <_TC>, <_MODE>, <_DIR>

1) 只有调试回转（IBN SCHWENKEN）时设置了这些函数时，才允许选择。

2) 只有在调试回转时设置了回转轴 1 或 2 的方向参考时，才允许选择。

在方向参数为“否”时，没有下拉框

3) 回转选择“否”可能被 SD 55221 位 0 的设置隐藏

<_DIR> = 0 和 _ST 万位 = 1 时，相当于设置了：回转“否”，方向“负”

<_DIR> = 0 和 _ST 万位 = 2 时，相当于设置了：回转“否”，方向“正”

4) 当带方向参考的回转轴位于极点位置时，也就是位置值为零，也可以选择回转轴 1 或 2 的方向。

5) 编码示例：轴旋转，旋转顺序 ZYX

二进制码：00011011； 十进制码：27

轴标识“XYZ”对应 NC 通道的几何轴。围绕 XYZ 轴的旋转可以单独进行，例如，在执行 CYCLE800 时，不可以按照 ZXZ 的顺序旋转。

20.1.34 CYCLE801 - 方阵/框架

句法

```
CYCLE801(<_SPCA>, <_SPCO>, <_STA>, <_DIS1>, <_DIS2>, <_NUM1>,
<_NUM2>, <_VARI>, <_UMODE>, <_ANG1>, <_ANG2>, <_HIDE>, <_NSP>,
<_DMODE>)
```

参数

编号	对话框参数	参数 内部	数据类型	含义		
1	X0	<_SPCA>	REAL	位置模式（方阵/框架）参考点的第 1 轴位置（绝对）		
2	Y0	<_SPCO>	REAL	位置模式（方阵/框架）参考点的第 2 轴位置（绝对）		
3	α0	<_STA>	REAL	初始旋转角 （和第 1 轴所成夹角）	< 0 =	顺时针旋转
					> 0 =	逆时针旋转
4	L1	<_DIS1>	REAL	列间距（即和第 1 轴之间的距离，无符号）		
5	L2	<_DIS2>	REAL	行间距（即和第 2 轴之间的距离，无符号）		

编号	对话框参数	参数 内部	数据类型	含义		
6	N1	<_NUM1>	INT	列数		
7	N2	<_NUM2>	INT	行数		
8		<_VARI>	INT	加工方式		
				个位：	位置模式	
					0 =	栅格
					1 =	框架
				十位：	保留	
				百位：	保留	
9		<_UMODE>	INT	保留		
10	αX	<_ANG1>	REAL	行和第 1 轴所成的夹角（即相对于第 1 轴的倾斜度）		
					< 0 =	顺时针测量 （0 到 -90 度）
					> 0 =	逆时针测量 （0 到 90 度）
11	αY	<_ANG2>	REAL	列和第 2 轴所成的夹角（即相对于第 2 轴的倾斜度）		
					< 0 =	顺时针测量 （0 到 -90 度）
					> 0 =	逆时针测量 （0 到 90 度）
12		<_HIDE>	字符串 [200]	被隐藏的位置 • 最多 198 个字符 • 连续位置号，例如“1,3”（跳过位置 1 和 3）		
13		<_NSP>	INT	保留		
14		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/G18/G19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）

20.1.35 CYCLE802 - 任意位置

句法

```
CYCLE802(<_XA>, <_YA>, <_X0>, <_Y0>, <_X1>, <_Y1>, <_X2>, <_Y2>,
<_X3>, <_Y3>, <_X4>, <_Y4>, <_X5>, <_Y5>, <_X6>, <_Y6>, <_X7>,
<_Y7>, <_X8>, <_Y8>, <_VARI>, <_UMODE>, <_DMODE>, <S_ABA>,
<S_AB0>, <S_AB1>, <S_AB2>, <S_AB3>, <S_AB4>, <S_AB5>, <S_AB6>,
<S_AB7>, <S_AB8>)
```

参数

编号	对话框参数	参数 内部	数据类型	含义		
1		<_XA>	INT	所有 X 位置的备选（9 位十进制值） 数位: 876543210（这个位置对应钻削位置 Xn）		
				位值：	1 =	绝对（第 1 个输入的位置始终是绝对坐标）
					2 =	增量
2		<_YA>	INT	所有 Y 轴位置的备选（9 位十进制值） 数位: 876543210（这个位置对应钻削位置 Yn）		
				位值：	1 =	绝对（第 1 个输入的位置始终是绝对坐标）
					2 =	增量
3	X0	<_X0>	REAL	第 1 个位置 X 轴坐标		
4	Y0	<_Y0>	REAL	第 1 个位置 Y 轴坐标		
5	X1	<_X1>	REAL	第 2 个位置 X 轴坐标		
6	Y1	<_Y1>	REAL	第 2 个位置 Y 轴坐标		
7	X2	<_X2>	REAL	第 3 个位置 X 轴坐标		
8	Y2	<_Y2>	REAL	第 3 个位置 Y 轴坐标		
9	X3	<_X3>	REAL	第 4 个位置 X 轴坐标		
10	Y3	<_Y3>	REAL	第 4 个位置 Y 轴坐标		
11	X4	<_X4>	REAL	第 5 个位置 X 轴坐标		
12	Y4	<_Y4>	REAL	第 5 个位置 Y 轴坐标		
13	X5	<_X5>	REAL	第 6 个位置 X 轴坐标		
14	Y5	<_Y5>	REAL	第 6 个位置 Y 轴坐标		

编号	对话框参数	参数 内部	数据类型	含义
15	X6	<_X6>	REAL	第 7 个位置 X 轴坐标
16	Y6	<_Y6>	REAL	第 7 个位置 Y 轴坐标
17	X7	<_X7>	REAL	第 8 个位置 X 轴坐标
18	Y7	<_Y7>	REAL	第 8 个位置 Y 轴坐标
19	X8	<_X8>	REAL	第 9 个位置 X 轴坐标
20	Y8	<_Y8>	REAL	第 9 个位置 Y 轴坐标

编号	对话框参数	参数 内部	数据类型	含义		
21		<_VARI>	INT	加工		
				百位：	（仅限于在生产车间使用）（刚开始只使用 0 和 2）	
					0 =	不夹紧主轴
					1 =	仅在以 G00 或 G01 垂直式插入时夹紧主轴
					2 =	整个加工过程中夹紧主轴
				千位：	保留	
				万位：	带/不带回转轴的位置模式 - 轴组合（参见 <_VARI> 十万位）	
					0 =	XY（仅 X 轴和 Y 轴，不带回转轴，兼容性）
					1 =	X、Y 或 Z 和回转轴： XA, YB, ZC （1 个回转轴和几何轴，该回转轴围绕着几何轴旋转）
					2 =	XY 和回转轴： XYA, XYB, XYC （1 个回转轴和第 1 几何轴、第 2 几何轴，不带 TRACYL）
				十万位：	回转轴	
					0 =	不带回转轴 （仅 X 轴和 Y 轴，兼容性）
					1 =	A 轴（绕 X 轴旋转的回转轴）
					2 =	B 轴（绕 Y 轴旋转的回转轴）
					3 =	C 轴（绕 Z 轴旋转的回转轴）
				千万位 + 百万位：	带回转轴的位置模式 - 偏移 （多个回转轴绕同一个轴旋转时；下标太大时，则为第 1 个轴）	
					00 =	1. A 轴、B 轴 或 C 轴或兼容时
					01 =	2. A 轴、B 轴或 C 轴
					...	

编号	对话框参数	参数 内部	数据类型	含义		
					19 =	20. A 轴、B 轴或 C 轴
22		<_UMODE>	INT	选择需夹紧的主轴：（仅限于从生产车间调用）（调用用户循环 CUST_TECHCYC）		
					3 =	夹紧/松开主主轴
					23 =	夹紧/松开副主轴
23		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/G18/G19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
24		<S_ABA>	INT	所有 AB 位置的备选（9 位十进制值） 数位：876543210（位置对应于 AB 位置）		
				位值：	1 =	绝对（第 1 个输入的位置始终是绝对坐标）
					2 =	增量
25	A0	<S_AB0>	REAL	带回转轴位置模式上的第 1 个回转轴位置（参见<_VARI>）		
26	A1	<S_AB1>	REAL	带回转轴位置模式上的第 2 个回转轴位置		
27	A2	<S_AB2>	REAL	带回转轴位置模式上的第 3 个回转轴位置		
28	A3	<S_AB3>	REAL	带回转轴位置模式上的第 4 个回转轴位置		
29	A4	<S_AB4>	REAL	带回转轴位置模式上的第 5 个回转轴位置		
30	A5	<S_AB5>	REAL	带回转轴位置模式上的第 6 个回转轴位置		
31	A6	<S_AB6>	REAL	带回转轴位置模式上的第 7 个回转轴位置		
32	A7	<S_AB7>	REAL	带回转轴位置模式上的第 8 个回转轴位置		
33	A8	<S_AB8>	REAL	带回转轴位置模式上的第 9 个回转轴位置		

说明

可以忽略多余的参数 X1/Y1/A1 到 X8/Y8/A8。 但一定要完整地输入备选位置“<_XA>, <_YA>”和“<S_ABA>”，它们适用于所有的 9 个位置。

在位置模式 XA、YB 或 ZC（一个几何轴和回转轴）上，在调用循环之前需要定位不会运动到位置模式上方的轴（G17 和 XA 上为 Y 轴）。

20.1.36 CYCLE830 - 深孔钻削 2

句法

```
CYCLE830(<RTP>, <RFP>, <SDIS>, <_DP>, <FDEP>, <_DAM>, <DTB>,
<DTS>, <FRF>, <VARI>, <_MDEP>, <_VRT>, <_DTD>, <_DIS1>, <S_FP>,
<S_SDAC2>, <S_SV2>, <S_FB>, <_SDAC>, <_SV1>, <S_SPOS>, <S_ZA>,
<S_FA>, <S_ZP>, <S_FS>, <S_ZS1>, <S_ZS2>, <S_N>, <S_ZD>, <S_FD>,
<S_FR>, <S_SDAC3>, <S_SV3>, <S_CON>, <S_COFF>, <_GMODE>,
<_DMODE>, <_AMODE>, <S_AMODE2>, <S_AMODE3>, <S_ZPV>)
```

参数

编号	对话框参数	内部参数	数据类型	含义
1	RP	<RTP>	REAL	回退平面（绝对）
2	Z0	<RFP>	REAL	参考点（绝对）
3	SC	<SDIS>	REAL	安全距离（和参考点的间距，无符号）
4	Z1	<_DP>	REAL	最终钻深，绝对/增量（参见 <_AMODE> 个位）
5	D	<FDEP>	REAL	第 1 钻深，绝对深度或增量深度：孔定位中相对于参考点或没有孔定位时相对于试钻深度（参见 <_AMODE> 万位）
6	DF	<_DAM>	REAL	后续每刀切削量或下调百分比（参见 <_AMODE> 十万位）
7	DTB	<DTB>	REAL	每个钻深的停留时间（参见 <_AMODE> 十位）
8	DTS	<DTS>	REAL	在起点处的停留时间，以便排屑（参见 <_AMODE> 百位）
9	FD1	<FRF>	REAL	首刀进给率百分比（参见 <_AMODE> 千万位）

编号	对话框参数	内部参数	数据类型	含义		
10		<VARI>	INT	加工		
				个位：	断屑/排屑	
					0 =	一步到底
					1 =	断屑
					2 =	排屑
					3 =	断屑和排屑
				十位：	排屑时返回	
					0 =	到试钻深度
					1 =	到安全距离
				百位：	软首切	
					0 =	否
					1 =	支持
				千位：	底部钻削	
					0 =	否
					1 =	支持
				万位：	孔定位/试钻	
					0 =	无孔定位
					1 =	有孔定位
					2 =	有试钻孔
				十万位：	返回	
					0 =	到试钻深度
					1 =	返回到返回平面
11	V1	<_MDEP>	REAL	最小切削量（只针对切削下调量为百分比时）		
12	V2	<_VRT>	REAL	每次加工后的退刀量，增量值（仅在断屑时）		
					0 =	默认值 1 mm
					> 0 =	可变的退刀量
13	DT	<_DTD>	REAL	在最终钻深处的停留时间（参见 <_AMODE> 千位）		
14	V3	<_DIS1>	REAL	提前距离，增量值，仅在排屑时（参见 <_AMODE> 百万位）		
15	FP	<S_FP>	REAL	试钻进给率，为数值或 % 值（参见 <S_AMODE2> 百位）		

编号	对话框参数	内部参数	数据类型	含义		
16		<S_SDAC2>	INT	进刀时的主轴旋转方向		
					3 =	M3
					4 =	M4
					5 =	M5 (默认)
17	SP	<S_SV2>	REAL	进刀方式	恒定主轴转速（参见 <S_AMODE2> 千万位）	
	V4				恒定切削速度	
				主轴转速是钻削转速的 %		
18	F	<S_FB>	REAL	钻削进给率（参见 <S_AMODE2> 个位）		
19		<_SDAC>	REAL	钻削时的主轴旋转方向		
					3 =	M3
					4 =	M4
20	S	<_SV1>	REAL	钻削方式	恒定主轴转速（参见 <S_AMODE2> 百万位）	
	V5				恒定切削速度	
21	SPOS	<S_SPOS>	REAL	主轴位置，仅在使用 M5 进刀时		
22	ZA	<S_ZA>	REAL	孔定位深度，绝对深度或相对于参考点的深度（参见 <S_AMODE3> 个位）		
23	FA	<S_FA>	REAL	孔定位进给率，数值或 %（参见 <S_AMODE2> 十位）		
24	ZP	<S_ZP>	REAL	试钻深度，绝对深度、相对于参考点的深度或孔直径系数（参见 <S_AMODE3> 十位）		
25	FS	<S_FS>	REAL	首刀进给率，数值或 %（结合 <S_AMODE2> 千位）		
26	ZS1	<S_ZS1>	REAL	采用恒定进给率时的每钻深度（增量）		
27	ZS2	<S_ZS2>	REAL	进给率不保持恒定（上调）时的每钻深度（增量）		
28	N	<S_N>	INT	每次排屑前的断屑次数		
29	ZD	<S_ZD>	REAL	剩余深度，绝对深度或相对于最终钻深的深度（结合 <S_AMODE3> 百位）		
30	FD	<S_FD>	REAL	剩余钻深进给率，数值或 %（结合 <S_AMODE2> 万位）		
31	FR	<S_FR>	REAL	返回速率（参见 <S_AMODE2> 十万位）		

编号	对话框参数	内部参数	数据类型	含义		
32		<S_SDAC3>	INT	返回时的主轴旋转方向		
					3 =	M3
					4 =	M4
					5 =	M5
33	SR	<S_SV3>	REAL	返回方式	恒定主轴转速（参见 <S_AMODE2> 亿位）	
	V6				恒定切削速度	
				主轴转速是钻削转速的 %		
34	冷却液开	<S_CON>	STRING[10]	冷却液开，M 指令或子程序调用		
35	冷却液关	<S_COFF>	STRING[10]	冷却液关，M 指令或子程序调用		
36		<_GMODE>	INT	几何值计算模式（所编写的几何值）		
				个位：	保留	
				十位：	钻削深度相对于刀尖或刀柄	
					0 =	刀尖
					1 =	刀柄
37		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/G18/G19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
				十位：	保留	
				百位：	保留	
				千位：	保留	
				万位：	工艺对话框内的工艺调整 (页 911)	
					0 =	输入：完整
					1 =	输入：简单

编号	对话框参数	内部参数	数据类型	含义		
38		<_AMODE>	INT	替代模式 1		
				个位：	钻深 = 最终钻深 Z1（绝对/增量）	
					0 =	增量
					1 =	绝对
				十位：	每个钻深处的停留时间 DTB，单位为秒或转	
					0 =	单位秒
					1 =	单位转
				百位：	用于排屑的停留时间 DTS，单位为秒或转	
					0 =	单位秒
					1 =	单位转
				千位：	在最终钻深处的停留时间 DT，单位为秒或转	
					0 =	单位秒
					1 =	单位转
				万位：	第 1 钻深 D，绝对/增量	
					0 =	增量
					1 =	绝对
				十万位：	后续每刀切削量或下调百分比 DF	
					0 =	绝对值
					1 =	百分比（0.001 到 100 %）
				百万位：	提前距离 V3，自动/手动	
					0 =	自动（在循环中计算得出）
					1 =	手动（手动输入的数值）

编号	对话框参数	内部参数	数据类型	含义		
39		<S_AMODE2>	INT	替代模式 2		
				个位：	个位：钻削进给率 F	
					0 =	F/min
					1 =	F/U
				十位：	孔定位进给率 FA	
					0 =	钻削进给率的 %
					1 =	F/min
					2 =	F/U
				百位：	试钻进给率 FP	
					0 =	钻削进给率的 %
					1 =	F/min
					2 =	F/U
				千位：	首切进给率 FS	
					0 =	钻削进给率的 %
					1 =	F/min
					2 =	F/U
				万位：	底部钻削进给率 FD	
					0 =	钻削进给率的 %
					1 =	F/min
					2 =	F/U
				十万位：	返回速率 FR	
					0 =	F/min
					1 =	快速移动
				百万位：	钻削 - 主轴转速/切削速度（S/V5）	
					0 =	恒定主轴转速
					1 =	恒定切削速度
				千万位：	进刀时的主轴转速/切削速度（SP/V4）	
					0 =	恒定主轴转速
					1 =	恒定切削速度
					2 =	主轴转速是钻削转速的 %

编号	对话框参数	内部参数	数据类型	含义		
				亿位：	退刀时的主轴转速/切削速度（SR/V6）	
					0 =	恒定主轴转速
					1 =	恒定切削速度
					2 =	主轴转速是钻削转速的 %
40		<S_AMODE3>	INT	替代模式 3		
				个位：	孔定位时的钻深 ZA，绝对/增量	
					0 =	增量
					1 =	绝对
				十位：	试钻深度 ZP	
					0 =	增量
					1 =	绝对
					2 =	孔直径的系数
				百位：	剩余钻深 ZD，增量/绝对	
					0 =	增量
					1 =	绝对
41	ZPV	<S_ZPV>	REAL	相对于试钻深度的提前距离，增量		

20.1.37 CYCLE832 - 快速设定

句法

```
CYCLE832(<S_TOL>, <S_TOLM>, <S_OTOL>)
```

说明

CYCLE832 可以减少机床厂商调试机床时必需的优化工作，这些工作主要涉及到参与加工的轴的优化、NCU 的设置（前馈、急动限制等）。

参数

编号	对话框参数	内部参数	数据类型	含义		
1	公差	<S_TOL>	REAL	轮廓公差 轮廓公差等同于几何轴的轴公差。		
2		<S_TOLM>	INT	加工方式（工艺）		
				个位：		
					0 =	取消
					1 =	精加工(Finish)
					2 =	预精整（Semifinish）
					3 =	粗加工 (Rough)
				十位：		
					0 =	兼容 1) 或无定位公差
					1 =	参数中的定向公差 <S_OTOL>
				百位 ... 十万位	已占用 由于 兼容性	
				百万位：		
					0 =	兼容性自动使用提供的最好的模具制造功能： ● 选件“臻优曲面”未激活： ⇒ 精优曲面 ● 选件“臻优曲面”激活： ⇒ 臻优曲面，带平滑
					1 =	臻优曲面，不带平滑
					2 =	臻优曲面，带平滑
3	定位公差	<S_OTOL>	REAL	定位公差或版本标识 CYCLE832 用于刀具定位的公差参数。 在采用动态定位转换（例如 5 轴加工）的机床上执行高速加工程序时需要此参数。 **必须**编程参数 <S_OTOL>。其也适用于 3 轴机床上无刀具定位的程序应用（<S_OTOL> = 1）。		

1) 定位公差由轮廓公差乘以循环设定数据 SD55441 至 SD55443 中的系数推导得出。

文档：

调试手册 基本软件和操作软件；SINUMERIK Operate (IM9)，章节“配置高速设定功能（CYCLE832）”

纯文本输入

为使循环调用指令更简单易懂，也可以纯文本形式输入参数 <S_TOLM>（加工方式）参数。纯文本可为任意语言。允许采用以下输入：

_OFF	针对	0	取消
_FINISH	针对	1	精加工
_SEMIFIN	针对	2	初精整
_ROUGH	针对	3	粗加工
_ORI_FINISH	针对	11	输入了定向公差的精加工
_ORI_SEMIFIN	针对	12	输入了定向公差的初精整
_ORI_ROUGH	针对	13	输入了定向公差的粗加工
_TOP_SURFACE_SMOOTH_OFF	针对	1000000	臻优曲面，不带平滑
_TOP_SURFACE_SMOOTH_ON	针对	2000000	臻优曲面，带平滑

臻优曲面的纯文本输入是按以下示例中的方式组合纯文本的：

```
CYCLE832(0.1, _TOP_SURFACE_SMOOTH_OFF+_ORI_FINISH, 1)
```

说明

这些纯文本以 G 功能组 59（轨迹插补的动态模式）的功能名称为依据。通过这些纯文本可在应用中明确区分 3 轴机床和含多轴定位转换（TRAORI）的机床。

撤销选择 CYCLE832

撤销选择 CYCLE832 时，参数 <S_TOL> 必须为零。

示例：CYCLE832(0,0,1)

句法 CYCLE832() 同样用于撤销选择 CYCLE832。

示例

示例 1：在无定位转换的 3 轴机床上使用 CYCLE832

a) 通过输入纯文本调用循环

程序代码	注释
G710	；单位制为公制。
CYCLE832(0.004,_FINISH,1)	；调用 CYCLE832： 轮廓公差 = 0.004 mm，加工方式：精加工
...	；执行高速加工程序

b) 不使用纯文本进行循环调用

程序代码	注释
G710	; s. o.
CYCLE832(0.004,1,1)	; s. o.
...	; s. o.

示例 2：在含定位转换的 5 轴机床上使用 CYCLE832

a) 通过输入纯文本调用和取消循环

程序代码	注释
G710	；单位制为公制。
TRAORI	；激活定位转换。
CYCLE832(0.3,_ORI_ROUGH,0.8)	；调用 CYCLE832： 轮廓公差 = 0,3 mm，加工方式：输入了定位公差的粗加工；定位公差 = 0.8 度
...	；执行高速加工程序
CYCLE832(0,_OFF,1)	; 轮廓公差=0， 加工类型：取消选择 CYCLE832， 定向公差=0 度

b) 不使用纯文本调用和取消循环

程序代码	注释
G710	; s. o.
TRAORI	; s. o.
CYCLE832(0.3,13,0.8)	; s. o.
...	; s. o.
CYCLE832(0,0,1)	; s. o.

20.1.38 CYCLE840 - 带补偿夹具的攻丝

句法

```
CYCLE840(<RTP>, <RFP>, <SDIS>, <DP>, <DPR>, <DTB>, <SDR>, <SDAC>,
<ENC>, <MPIT>, <PIT>, <_AXN>, <_PITA>, <_TECHNO>, <_PITM>,
<_PTAB>, <_PTABA>, <_GMODE>, <_DMODE>, <_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义
1	RP	<RTP>	REAL	回退平面（绝对）
2	Z0	<RFP>	REAL	参考点（绝对）
3	SC	<SDIS>	REAL	安全距离（和参考点的间距，无符号）
4	Z1	<DP>	REAL	钻深（绝对），参见 <_AMODE>
5	Z1	<DPR>	REAL	钻深（增量），参见 <_AMODE>
6	DT	<DTB>	REAL	退回后，在钻深/安全距离处的停留时间，单位秒，参见 <ENC>
7		<SDR>	INT	退回时的旋转方向
8	SDE	<SDAC>	INT	循环结束之后的旋转方向

编号	对话框参数	内部参数	数据类型	含义			
9		<ENC>	INT	有主轴编码器的攻丝（G33）/无主轴编码器的攻丝（G63）			
					0 =	带主轴编码器	- 螺距取决于 <MPIT>/ <PIT> - 无停留时间 DT
					20 =	带主轴编码器	- 螺距取决于 <MPIT>/ <PIT> - 有退回后在安全距离上的停留时间 DT
					11 =	无主轴编码器	- 螺距取决于 <MPIT>/ <PIT> - 有在钻深处的停留时间 DT
					1 =	无主轴编码器	- 螺距取决于编程设置的进给率 - 有在钻深处的停留时间 DT（进给率 = 转速 · 螺距）
10		<MPIT>	REAL	螺纹尺寸只限“ISO 米制”（在运行时会内部换算螺距） 取值范围： 3 到 48（ M3 到 M48），<PIT> 的备选			
11		<PIT>	REAL	螺距值，单位参见 <_PITA> 取值范围： > 0， MPIT 的备选			
12		<_AXN>	INT	钻孔轴	0 =	3. 几何轴	
					1 =	1. 几何轴	
					2 =	2. 几何轴	
					≥ 3 =	3. 几何轴	

编号	对话框参数	内部参数	数据类型	含义			
13		<_PITA>	INT	螺距的单位（计算 <PIT> 和 <MPIT>）			
					0 =	螺距,单位毫米	- 计算 <MPIT>/<PIT>
					1 =	螺距,单位毫米	- 计算 <PIT>
					2 =	螺距,单位 TPI	- 计算 <PIT>（螺纹牙/英寸）
					3 =	螺距,单位英寸	- 计算 <PIT>
					4 =	模数	- 计算 <PIT>
14		<_TECHNO>	INT	工艺 [1]			
				个位：	准停特性		
					0 =	调用循环前的准停特性保持有效	
					1 =	准停 G601	
					2 =	准停 G602	
					3 =	准停 G603	
				十位：	前馈		
					0 =	带/不带前馈控制，保持调用循环前的状态	
					1 =	带前馈控制 FFWON	
					2 =	不带前馈控制 FFWOF	
15		<_PITM>	STRING[15]	表示螺距输入方式的字符串 [2]			
16		<_PTAB>	STRING[5]	表示螺纹表的字符串 （“”，“ISO”，“BSW”，“BSP”，“UNC”）[2]			
17		<_PTABA>	STRING[20]	表示螺纹表中各个选项的字符串（例如：“M 10”，“M 12”， ...）[2]			
18		<_GMODE>	INT	保留			

编号	对话框参数	内部参数	数据类型	含义		
19		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/G18/G19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
				十位：	保留	
				百位：	保留	
				千位：	兼容性模式（仅用于输入对话框的回译），当 MD 52216 位 0 = 1[1)]	
					0 =	显示工艺参数（兼容性）： 工艺参数有效
					1 =	不显示工艺参数： 工艺保持调用循环前的状态
				万位：	工艺对话框内的工艺调整 (页 911)	
					0 =	输入：完整
					1 =	输入： 简单
20		<_AMODE>	INT	替代模式		
				个位：	钻深 Z1（绝对/增量）	
					0 =	兼容性，取决于 <DP>/<DPR>
					1 =	增量
					2 =	绝对

[1)] 可以通过设定数据“SD52216 MCS_FUNCTION_MASK_DRILLL”隐藏工艺输入栏。

[2)] 参数 15、 16 和 17 只用于输入对话框螺纹表中的螺纹选择。 在处理循环期间，无法通过循环定义来访问螺纹表！

20.1.39 CYCLE899 - 铣削敞开槽

句法

```
CYCLE899(<_RTP>, <_RFP>, <_SDIS>, <_DP>, <_LENG>, <_WID>, <_PA>,
<_PO>, <_STA>, <_MID>, <_MIDA>, <_FAL>, <_FALD>, <_FFP1>,
<_CDIR>, <_VARI>, <_GMODE>, <_DMODE>, <_AMODE>, <_UMODE>, <_FS>,
<_ZFS>)
```

参数

编号	对话框参数	内部参数	数据类型	含义		
1	RP	<_RTP>	REAL	退回平面（绝对）		
2	Z0	<_RFP>	REAL	刀具轴的参考点（绝对）		
3	SC	<_SDIS>	REAL	安全距离（和参考点的间距，无符号）		
4	Z1	<_DP>	REAL	槽深（绝对/增量），参见 <_AMODE>		
5	L	<_LENG>	REAL	槽长度（增量）		
6	W	<_WID>	REAL	槽宽度（增量）		
7	X0	<_PA>	REAL	参考点/起点的第 1 轴的位置（绝对）		
8	Y0	<_PO>	REAL	参考点/起点的第 2 轴的位置（绝对）		
9	α0	<_STA>	REAL	和第 1 轴所成旋转角		
10	DZ	<_MID>	REAL	最大深度进给（增量），仅限旋风铣削		
11	DXY	<_MIDA>	REAL	最大切宽，参见 <_AMODE>		
12	UXY	<_FAL>	REAL	边沿精加工余量		
13	UZ	<_FALD>	REAL	底部精加工余量		
14	F	<_FFP1>	REAL	进给率		
15		<_CDIR>	INT	铣削方向		
				个位：		
					0 =	顺铣
					1 =	逆铣
					4 =	交替式

编号	对话框参数	内部参数	数据类型	含义		
16		<_VARI>	INT	加工		
				个位：		
					1 =	粗加工
					2 =	精加工
					3 =	底部精加工
					4 =	边沿精加工
					5 =	预精整
					6 =	倒角
				十位：	保留	
				百位：	保留	
				千位：		
					1 =	旋风铣
					2 =	插铣
17		<_GMODE>	INT	几何值计算模式（所编写的几何值）		
				个位：	保留	
				十位：	保留	
				百位：	加工/仅计算起点	
					1 =	正常加工
				千位：	通过中心/边标注尺寸	
					0 =	通过中心标注尺寸
					1 =	通过左侧边标注尺寸，左侧边即第 1 轴的负向
					2 =	通过右侧边标注尺寸，右侧边即第 1 轴的正向

编号	对话框参数	内部参数	数据类型	含义		
18		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/G18/G19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
				十位：	---	保留
				百位：	---	保留
				千位：	---	保留
				万位：	工艺对话框内的工艺调整 (页 911)	
					0 =	输入：完整
					1 =	输入： 简单
19		<_AMODE>	INT	替代模式		
				个位：	槽深度 Z1	
					0 =	绝对
					1 =	增量
				十位：	切宽（<_MIDA>）的单位	
					0 =	mm
					1 =	刀具半径的 %
				百位：	加工倒角时的插入深度 ZFS	
					0 =	绝对
					1 =	增量
20		<_UMODE>	INT	保留		
21	FS	<_FS>	REAL	倒角宽度（增量）		
22	ZFS	<_ZFS>	REAL	加工倒角时的插入深度（刀尖），参见 <_AMODE>)		

20.1.40 CYCLE930 - 凹槽

句法

```
CYCLE930(<_SPD>, <_SPL>, <_WIDG>, <_WIDG2>, <_DIAG>, <_DIAG2>,
<_STA>, <_ANG1>, <_ANG2>, <_RCO1>, <_RCI1>, <_RCI2>, <_RCO2>,
<_FAL>, <_IDEP1>, <_SDIS>, <_VARI>, <_DN>, <_NUM>, <_DBH>,
<_FF1>, <_NR>, <_FALX>, <_FALZ>, <_DMODE>, <_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义		
1	X0	<_SPD>	REAL	参考点 X 轴坐标（始终是直径）		
2	Z0	<_SPL>	REAL	参考点 Z 轴坐标		
3	B1	<_WIDG>	REAL	底部切槽宽度		
4	B2	<_WIDG2>	REAL	顶部切槽宽度（仅供界面显示）		
5	T1	<_DIAG>	REAL	参考点处的切槽深度， 在“绝对”和“纵向加工”时为直径，其它为增量		
6	T2	<_DIAG2>	REAL	相对于参考点的切槽深度（仅供界面显示） 在“绝对”和“纵向加工”时为直径，其它为增量		
7	α0	<_STA>	REAL	斜面角度（-180 ≤ _<_STA> ≤ 180）		
8	α1	<_ANG1>	REAL	参考点上切槽面上的侧面角度 1 （0 ≤ <_ANG1> < 90）		
9	α2	<_ANG2>	REAL	相对于参考点的侧面角度 2（0 ≤ <_ANG2> < 90）		
10	R1/FS1	<_RCO1>	REAL	倒圆半径或倒角宽度 1，外侧，在参考点上		
11	R2/FS2	<_RCI1>	REAL	倒圆半径或倒角宽度 2，内侧，在参考点上		
12	R3/FS3	<_RCI2>	REAL	倒圆半径或倒角宽度 3，内侧，参考点对面		
13	R4/FS4	<_RCO2>	REAL	倒圆半径或倒角宽度 4，外侧，参考点对面		
14	U	<_FAL>	REAL	X 轴和 Z 轴的精加工余量，参见 <_VARI> （万位），无符号		
15	D	<_IDEP1>	REAL	插入时的最大切深，无符号		
					0 =	1 刀到底
					> 0 =	第 1 刀<_IDEP1>；第 2 刀 <_IDEP1>，以此类推。
16	SC	<_SDIS>	REAL	安全距离（无符号）		

编号	对话框参数	内部参数	数据类型	含义		
17		<_VARI>	INT	加工方式		
				个位：	保留	
				十位：	工艺加工	
					1 =	粗加工
					2 =	精加工
					3 =	粗加工和精加工
				百位：	横向/纵向；内侧/外侧位置 +Z/+Z 或 +X/-X	
					1 =	纵向/外侧 +Z
					2 =	横向/内侧 -X
					3 =	纵向/内侧 +Z
					4 =	横向/内侧 +X
					5 =	纵向/外侧 -Z
					6 =	横向/外侧 -X
					7 =	纵向/内侧 -Z
					8 =	横向/外侧 +X
				千位：	参考点位置	
					0 =	顶部参考点
					1 =	底部参考点
				万位：	定义同精加工余量	
					0 =	与轮廓平行的精加工余量 U
					1 =	单独的精加工余量 UX 和 UZ
18		<_DN>	INT	刀具第 2 刀沿的 D 号		
					> 0 =	切槽刀第 2 刀沿的刀补 D 号
					0 =	不写入第 2 刀沿
19	N	<_NUM>	INT	切槽数量（0 = 1 个切槽）		
20	DP	<_DBH>	REAL	切槽距离（只有<_NUM> > 1 时，才需要用到该参数）		
21	F	<_FF1>	REAL	进给率		

编号	对话框参数	内部参数	数据类型	含义		
22		<_NR>	INT	切槽形状的标识（相当于用于选择形状的垂直软键）		
					0 =	90°侧边，无倒角/倒圆
					1 =	斜侧边，有倒角/倒圆（无 α0）
					2 =	同 1，但在锥面上（有 α0）
23	UX	<_FALX>	REAL	X 轴的精加工余量，参见 <_VARI>（万位），无符号		
24	UZ	<_FALZ>	REAL	Z 轴的精加工余量，参见 <_VARI>（万位），无符号		
25		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/G18/G19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）

编号	对话框参数	内部参数	数据类型	含义		
26		<_AMODE>	INT	替代模式		
				个位：	标注深度（仅供界面显示）	
					0 =	在参考点上
					1 =	相对于参考点
				十位：	深度	
					0 =	绝对
					1 =	增量
				百位：	标注宽度（仅供界面显示）	
					0 =	在外径上（顶部）
					1 =	在内径上（底部）
				千位：	倒圆/倒角 1（<_RCO1>）	
					0 =	半径
					1 =	倒角
				万位：	倒圆/倒角 2（<_RCI1>）	
					0 =	半径
					1 =	倒角
				十万位：	倒圆/倒角 3（<_RCI2>）	
					0 =	半径
					1 =	倒角
				百万：	倒圆/倒角 4（<_RCO2>）	
					0 =	半径
					1 =	倒角

20.1.41 CYCLE940 - 退刀槽

通过 CYCLE940 可以写入不同形状的退刀槽，但是，有些退刀槽的编程参数明显不同。

参数表中添加了一列，指出参数适用的退刀槽形状。退刀槽形状标识和循环对话框中的垂直选择软键一一对应：

- E:E 形退刀槽
- F:F 形退刀槽

- A-D:DIN 螺纹退刀槽（形状 A-D）
- T:螺纹退刀槽（形状自由定义）

句法

```
CYCLE940(<_SPD>, <_SPL>, <_FORM>, <_LAGE>, <_SDIS>, <_FFP>,
<_VARI>, <_EPD>, <_EPL>, <_R1>, <_R2>, <_STA>, <_VRT>, <_MID>,
<_FAL>, <_FALX>, <_FALZ>, <_PITI>, <_PTAB>, <_PTABA>, <_DMODE>,
<_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	适用的退刀槽形状				含义		
				E	F	A-D	T			
1	X0	<_SPD>	REAL	x	x	x	x	参考点 X 轴坐标（始终是直径）		
2	Z0	<_SPL>	REAL	x	x	x	x	参考点 Z 轴坐标（绝对）		
3	FORM	<_FORM>	CHAR	x	x	x	x	退刀槽形状（大写字母，例如：T） 选择采用哪个表格中的退刀槽数据		
									A =	外侧，参考 DIN76， A = 标准
									B =	外侧，参考 DIN76， B = 短
									C =	内侧，参考 DIN76， C = 标准
									D =	内侧，参考 DIN76， D = 短
									E =	参考 DIN509
									F =	参考 DIN509
									T=	任意形状
4	位置	<_LAGE>	INT	x	x	x	x	退刀槽的位置 （平行 Z 轴）	0 =	外部 + Z: ____|
									1 =	外部 - Z: ____|
									2 =	内部 +Z: /-----|
									3 =	内部 -Z: |-----\
5	SC	<_SDIS>		x	x	x	x	安全距离（增量）		

编号	对话框参数	内部参数	数据类型	适用的退刀槽形状				含义		
6	F	<_FFP>		x	x	x	x	加工进给率：毫米/转		
7		<_VARI>	INT	-	-	x	x	加工方式		
								个位：	加工	
									1 =	粗加工
									2 =	精加工
									3 =	粗加工 + 精加工
								十位：	加工方案	
									0 =	与轮廓平行
									1 =	纵向
				E 形和 F 形退刀槽的加工和精加工一样，一步完成。						
8	X1	<_EPD>		x	x	-	-	X 轴余量（绝对/增量），参见 <_AMODE>		
				-	-	-	x	退刀槽深度（绝对/增量），参见 <_AMODE>		
9	Z1	<_EPL>		-	x	-	-	Z 轴余量		
				-	-	-	x	退刀槽宽度（绝对/增量），参见 <_AMODE>		
10	R1	<_R1>		-	-	-	x	斜面上的倒圆半径		
11	R2	<_R2>		-	-	-	x	拐角上的倒圆半径		
12	α	<_STA>		-	-	x	x	插入角度		
13	VX	<_VRT>		x	x	-	-	X 轴的横进给（绝对/增量），参见<_AMODE>		
				-	-	x	x	精加工时 X 轴的横进给（绝对/增量），参见 <_AMODE>		
14	D	<_MID>		-	-	x	x	切深		
15	U	<_FAL>		-	-	x	x	平行与轮廓的精加工余量，参见 <_AMODE>		
16	UX	<_FALX>		-	-	x	x	X 轴精加工余量		
17	UZ	<_FALZ>		-	-	x	x	Z 轴精加工余量		

<table>
<tr><th>编号</th><th>对话框参数</th><th>内部参数</th><th>数据类型</th><th colspan="4">适用的退刀槽形状</th><th colspan="4">含义</th></tr>
<tr><td rowspan="5">18</td><td rowspan="5">P</td><td rowspan="5"><_PITI></td><td rowspan="5">INT</td><td>-</td><td>-</td><td>x</td><td>-</td><td colspan="4">螺距选择，A-D，对应 M1 ... M68</td></tr>
<tr><td colspan="4" rowspan="2"></td><td>0 = 0.20
1 = 0.25
2 = 0.30
3 = 0.35
4 = 0.40
5 = 0.45</td><td>6 = 0.50
7 = 0.60
8 = 0.70
9 = 0.75
10 = 0.80
11 = 1.00</td><td>12 = 1.25
13 = 1.50
14 = 1.75
15 = 2.00
16 = 2.50
17 = 3.00</td><td>18 = 3.50
19 = 4.00
20 = 4.50
21 = 5.00
22 = 5.50
23 = 6.00</td></tr>
<tr><td>x</td><td>x</td><td>-</td><td>-</td><td colspan="4">半径/深度的选择，E、F</td></tr>
<tr><td colspan="4"></td><td>0 = 0.6 · 0.3
1 = 1.0 · 0.4
2 = 1.0 · 0.2
3 = 1.6 ··0.3</td><td>4 = 2.5 · 0.4
5 = 4.0 · 0.5
6 = 0.4 · 0.2
7 = 0.6 · 0.2</td><td colspan="2">8 = 0.1 · 0.1
9 = 0.2 ··0.1</td></tr>
<tr><td>19</td><td></td><td><_PTAB></td><td>字符串
[5]</td><td colspan="4"></td><td colspan="4">表示螺纹表的字符串 （“”，“ISO”，“BSW”，“BSP”，“UNC”）
（“仅供界面显示”）</td></tr>
<tr><td>20</td><td></td><td><_PTABA></td><td>字符串
[20]</td><td colspan="4"></td><td colspan="4">表示螺纹表中各个选项的字符串
（例如：“M 10”，“M 12”， ...）
（“仅供界面显示”）</td></tr>
<tr><td rowspan="6">21</td><td rowspan="6"></td><td rowspan="6"><_DMODE></td><td rowspan="6">INT</td><td colspan="4"></td><td colspan="4">显示模式</td></tr>
<tr><td>x</td><td>x</td><td>x</td><td>x</td><td rowspan="5">个位：</td><td colspan="3">加工平面 G17/G18/G19</td></tr>
<tr><td colspan="4" rowspan="4"></td><td>0 =</td><td>兼容性，在调用循环前有效的平面保持有效</td></tr>
<tr><td>1 =</td><td>G17 （仅在循环中有效）</td></tr>
<tr><td>2 =</td><td>G18 （仅在循环中有效）</td></tr>
<tr><td>3 =</td><td>G19 （仅在循环中有效）</td></tr>
</table>

编号	对话框参数	内部参数	数据类型	适用的退刀槽形状				含义		
22		<_AMODE>	INT					替代模式		
				x	x	-	x	个位：	参数<_EPD>：X 轴余量或退刀槽深度	
									0 =	绝对（始终是直径）
									1 =	增量
				x	x	-	x	十位：	参数<_EPL> ： Z 轴余量或退刀槽宽度	
									0 =	绝对
									1 =	增量
				x	x	x	x	百位：	参数 <_VRT>： X 轴横进给	
									0 =	绝对（始终是直径）
									1 =	增量
				-	-	x	x	千位：	精加工余量	
									0 =	与轮廓平行的精加工余量（<_FAL>）
									1 =	单独的精加工余量（<_FALX>/<_FALZ>）

20.1.42 CYCLE951- 切削

句法

```
CYCLE951(<_SPD>, <_SPL>, <_EPD>, <_EPL>, <_ZPD>, <_ZPL>, <_LAGE>,
<_MID>, <_FALX>, <_FALZ>, <_VARI>, <_RF1>, <_RF2>, <_RF3>,
<_SDIS>, <_FF1>, <_NR>, <_DMODE>, <_AMODE>)
```

参数

编号	对话框参数	内部参数	数据类型	含义
1	X0	<_SPD>	REAL	参考点（绝对，始终为直径）
2	Z0	<_SPL>	REAL	参考点（绝对）
3	X1	<_EPD>	REAL	终点
4	Z1	<_EPL>	REAL	终点

编号	对话框参数	内部参数	数据类型	含义		
5	XM α1 α2	<_ZPD>	REAL	中间点，参见 <_DMODE>（十位）		
6	ZM α1 α2	<_ZPL>	REAL	中间点，参见 <_DMODE>（十位）		
7	位置	<_LAGE>	INT	切削角的位置	0 =	外侧/后侧
					1 =	外侧/前侧
					2 =	内侧/后侧
					3 =	内侧/前侧
8	D	<_MID>	REAL	插入时的最大切深		
9	UX	<_FALX>	REAL	X 轴的精加工余量		
10	UZ	<_FALZ>	REAL	Z 轴的精加工余量		
11		<_VARI>	INT	加工方式		
				个位：	坐标系中的切削方向（横向或纵向）	
					1 =	纵向
					2 =	横向
				十位：		
					1 =	粗加工，保留精加工余量
					2 =	精加工
				百位：	预留	
				千位：	预留	
				万位：	预留	
12	R1/FS1	<_RF1>	REAL	倒圆半径或倒角宽度 1，参见 <_AMODE>（万位）		
13	R2/FS2	<_RF2>	REAL	倒圆半径或倒角宽度 2，参见 <_AMODE>（十万位）		
14	R3/FS3	<_RF3>	REAL	倒圆半径或倒角宽度 3，参见 <_AMODE> （百万位）		
15	SC	<_SDIS>	REAL	安全距离		
16	F	<_FF1>	REAL	粗加工/精加工进给率		

编号	对话框参数	内部参数	数据类型	含义		
17		<_NR>	INT	切削方式标识（相当于用于选择形状的垂直软键）：		
					0 =	切削 1，90 度拐角，无倒圆/倒角
					1 =	切削 2，90 度拐角，带倒圆/倒角
					2 =	切削 3，任意拐角，带倒圆/倒角
18		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/G18/G19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
				十位：	输入格式<_ZPD>/<_ZPL>	
					0 =	Xm/Zm
					1 =	Xm/α1
					2 =	Xm/α2
					3 =	α1/Zm
					4 =	α2/Zm
					5 =	α1/α2

编号	对话框参数	内部参数	数据类型	含义		
21		<_AMODE>	INT	替代模式		
				个位：	中间点 X 轴坐标	
					0 =	绝对， X 轴坐标，直径
					1 =	增量， X 轴坐标，半径
				十位：	中间点 Z 轴坐标	
					0 =	绝对
					1 =	增量
				百位：	终点 X 轴坐标	
					0 =	绝对， X 轴坐标，直径
					1 =	增量， X 轴坐标，半径
				千位：	终点 Z 轴坐标	
					0 =	绝对
					1 =	增量
				万位：	倒圆/倒角 1	
					0 =	半径
					1 =	倒角
				十万位：	倒圆/倒角 2	
					0 =	半径
					1 =	倒角
				百万位：	倒圆/倒角 3	
					0 =	半径
					1 =	倒角

20.1.43 CYCLE952 - 槽式车削

句法

```
CYCLE952(<_PRG>, <_CON>, <_CONR>, <_VARI>, <_F>, <_FR>, <_RP>,
<_D>, <_DX>, <_DZ>, <_UX>, <_UZ>, <_U>, <_U1>, <_BL>, <_XD>,
<_ZD>, <_XA>, <_ZA>, <_XB>, <_ZB>, <_XDA>, <_XDB>, <_N>, <_DP>,
<_DI>, <_SC>, <_DN>, <_GMODE>, <_DMODE>, <_AMODE>, <_PK>, <_DCH>,
<_FS>)
```

参数

编号	对话框参数	内部参数	数据类型	含义
1	PRG	<_PRG>	STRING[100]	切削程序名称
2	CON	<_CON>	STRING[100]	用于读取更新过的毛坯轮廓的程序的名称（余料加工）
3	CONR	<_CONR>	STRING[100]	用于写入更新过的毛坯轮廓（参见 <_AMODE> 万位）的程序的名称

编号	对话框参数	内部参数	数据类型	含义		
4		<_VARI>	INT	加工方式		
				个位：	切削类型	
					1 =	纵向
					2 =	横向
					3 =	与轮廓平行
				十位：	工艺加工（参见 <_GMODE> 百位）	
					1 =	粗加工
					2 =	精加工
					3 =	保留
					4 =	双通道粗加工
					5 =	双通道精加工
				百位：	加工方向	
					1 =	加工方向 X-
					2 =	加工方向 X+
					3 =	加工方向 Z-
					4 =	加工方向 Z+
				千位：	进刀方向	
					1 =	外部 X-
					2 =	内部 X+
					3 =	端面 Z-
					4 =	后侧 Z+
				万位：	定义同精加工余量	
					0 =	单独的精加工余量 UX 和 UZ
					1 =	与轮廓平行的精加工余量 U
				十万位：	沿轮廓返回	
					0 =	兼容性，自动沿轮廓返回
					1 =	沿轮廓返回
					2 =	不沿轮廓返回
					3 =	自动沿轮廓返回
				百万位：	退刀槽	

编号	对话框参数	内部参数	数据类型	含义		
					0 =	在“槽式车削”、“槽式车削余料”、“往复车削”和“往复车削余料”中不计算该位
					1 =	加工凹轮廓
					2 =	不加工凹轮廓
				千万位：	旋转中心前/后	
					0 =	旋转中心前加工
					1 =	保留
5	F	<_F>	REAL	粗加工/精加工进给率		
	FZ			往复车削的横向进给		
6	FR	<_FR>	REAL	在粗切退刀槽时的插入进给率		
	FX			往复车削的纵向进给		
7	RP	<_RP>	REAL	内部加工时的退回平面（绝对，始终是直径）		
8	D	<_D>	REAL	粗加工切削量（参见 <_AMODE> 个位）		
9	DX	<_DX>	REAL	X 轴切削量（参见 <_AMODE> 个位）		
10	DZ	<_DZ>	REAL	粗加工切削量（参见 <_AMODE> 个位）		
11	UX	<_UX>	REAL	X 轴精加工余量（参见 <_VARI> 万位）		
12	UZ	<_UZ>	REAL	Z 轴精加工余量（参见 <_VARI> 万位）		
13	U	<_U>	REAL	与轮廓平行的精加工余量（参见 <_VARI> 万位）		
14	U1	<_U1>	REAL	其它精加工余量（参见 <_AMODE> 千位）		
15	BL	<_BL>	INT	毛坯定义	1 =	圆柱体，有余量
					2 =	成品轮廓余量
					3 =	指定了毛坯轮廓
16	XD	<_XD>	REAL	X 轴毛坯定义（参见 <_AMODE> 十万位）		
17	ZD	<_ZD>	REAL	Z 轴毛坯定义（参见 <_AMODE> 百万位）		
18	XA	<_XA>	REAL	限位 1 X 轴坐标（绝对，始终是直径）		
19	ZA	<_ZA>	REAL	限位 1 Z 轴坐标（绝对）		
20	XB	<_XB>	REAL	限位 2 X 轴坐标（参见 <_AMODE> 千万位）		
21	ZB	<_ZB>	REAL	限位 2 Z 轴坐标（参见 <_AMODE> 亿位）		
22	XDA	<_XDA>	REAL	端面上第 1 切槽位置的切槽限值 1（绝对，始终是直径）		
23	XDB	<_XDB>	REAL	端面上第 1 切槽位置的切槽限值 2（绝对，始终是直径）		
24	N	<_N>	INT	凹槽数量		

编号	对话框参数	内部参数	数据类型	含义		
25	DP	<_DP>	REAL	凹槽间距	纵向凹槽：平行 Z 轴	
					横向凹槽：平行 X 轴	
26	DI	<_DI>	REAL	进给停止的距离	0 =	无中断
					> 0 =	带中断
27	SC	<_SC>	REAL	绕行障碍时的安全距离，增量		
28	D2	<_DN>	INT	刀沿 2 的 D 号，如果没有写入 ⇒ D+1		
29		<_GMODE>	INT	几何值计算模式（所编写的几何值）		
				个位：	保留	
				十位：	保留	
				百位：	加工/仅计算起点	
					0 =	正常加工（不要求兼容模式）
					1 =	正常加工
					2 =	计算起点位置 - 不加工（仅用于从 ShopMill/ShopTurn 调用命令）
				千位：	限位	
					0 =	否
					1 =	是
				万位：	限位 1 的 X 轴坐标	
					0 =	否
					1 =	是
				十万位：	限位 2 的 X 轴坐标	
					0 =	否
					1 =	是
				百万位：	限位 1 的 Z 轴坐标	
					0 =	否
					1 =	是
				千万位：	限位 2 的 Z 轴坐标	
					0 =	否
					1 =	是

编号	对话框参数	内部参数	数据类型	含义		
30		<_DMODE>	INT	显示模式		
				个位：	加工平面 G17/G18/G19	
					0 =	兼容性，在调用循环前有效的平面保持有效
					1 =	G17 （仅在循环中有效）
					2 =	G18 （仅在循环中有效）
					3 =	G19 （仅在循环中有效）
				十位：	工艺模式	
					1 =	轮廓切削
					2 =	轮廓槽式车削
					3 =	往复车削
				百位：	加工剩余材料	
					0 =	否
					1 =	是
				千位：	---	保留
				万位：	工艺对话框内的工艺调整 (页 911)	
					0 =	输入：完整
					1 =	输入：简单

编号	对话框参数	内部参数	数据类型	含义		
31		<_AMODE>	INT	替代模式		
				个位：	选择切削量	
					0 =	切削类型“和轮廓平行”中的切削量 DX 和 DZ
					1 =	D 轴切削量
				十位：	进刀方案	
					0 =	可变切削深度（90 ... 100 %）
					1 =	恒定切削深度
				百位：	切削分段	
					0 =	均匀
					1 =	边沿对齐
				千位：	选择“轮廓余量 U1 ，双精加工”	
					0 =	否
					1 =	是
				万位：	选择“更新毛坯”	
					0 =	否
					1 =	是
				十万位：	选择“毛坯余量 XD”	
					0 =	绝对， X 轴坐标，直径
					1 =	增量， X 轴坐标，半径
				百万位：	选择“毛坯余量 ZD”	
					0 =	绝对
					1 =	增量
				千万位：	选择“限位 2 XB”	
					0 =	绝对， X 轴坐标，直径
					1 =	增量， X 轴坐标，半径
				亿位：	选择“限位 2 ZB”	
					0 =	绝对
					1 =	增量
				十亿位：		
					0 =	导向通道

编号	对话框参数	内部参数	数据类型	含义		
					1 =	随动通道
32		<_PK>	INT	机床配置有 2 个以上的通道时，配对通道的编号		
33	DCH	<_DCH>	REAL	通道偏移		
34	FS	<_FS>	REAL	完整加工中精加工时的进给率		

20.1.44 CYCLE4071 - 反向点处带进给的纵向磨削

句法

```
CYCLE4071(<S_A>, <S_B>, <S_W>, <S_U>, <S_I>, <S_K>, <S_H>,
<S_A1>, <S_A2>)
```

参数

编号	参数	数据类型	含义
1	<S_A>	REAL	起始进给深度
2	<S_B>	REAL	结束进给深度
3	<S_W>	REAL	磨削宽度
4	<S_U>	REAL	修光时间
5	<S_I>	REAL	进给率
6	<S_K>	REAL	横向进给率
7	<S_H>	INT	重复调用的次数
8	<S_A1>	AXIS	进给轴（可选）或第 1 几何轴
9	<S_A2>	AXIS	往复轴（可选）或第 2 几何轴

功能

该循环用于执行<S_H>次进给。其中的起始进给深度和结束进给深度可以不同。各次进给之间会执行切向运动。

过程

1. 在往复轴当前位置上启动循环。
2. 进给轴运行至起始进给深度 P1 <S_A>，进给率为 P5 <S_I>。
3. 以修光时间 P4 <S_U>进行光磨。

4. 往复轴运行，以磨削宽度 P3 <S_W>作为运行行程，横向进给率为 P6 <S_K>。
5. 进给轴运行至结束进给深度 P2 <S_B>，进给率为 P5 <S_I>。
6. 以修光时间 P4 <S_U>进行光磨。
7. 往复轴运行，以磨削宽度 P4 <S_A>作为运行行程运行至起始点，横向进给率为 P6 <S_K>。

该过程无法用一个程序段中断。

根据已编程的重复 P7（<S_H>）的数量重复该过程。

示例

以以下循环参数进行两种摆动运动：

- 起始进给深度：0.02 mm
- 结束进给深度：0.01 mm
- 冲程：100 mm
- 修光时间：1 s
- 进给率：1 mm/min
- 横向进给率：1000 mm/min
- 重复加工的次数：2
- 往复轴和进给轴：标准几何轴

程序代码

```
N10 T1 D1
N20 CYCLE4071(0.02,0.01,100,1,1,1000,2)
N30 M30
```

20.1.45 CYCLE4072 - 反向点处带进给的纵向磨削以及中断信号

句法

```
CYCLE4072(<S_GAUGE>, <S_A>, <S_B>, <S_W>, <S_U>, <S_I>, <S_K>,
<S_H>, <S_A1>, <S_A2>)
```

参数

编号	参数	数据类型	含义
1	<S_GAUGE>	STRING	进给中断条件： 1. 快速输入编号 2. 逻辑表达式
2	<S_A>	REAL	起始进给深度
3	<S_B>	REAL	结束进给深度
4	<S_W>	REAL	磨削宽度
5	<S_U>	REAL	修光时间
6	<S_I>	REAL	进给率
7	<S_K>	REAL	横向进给率
8	<S_H>	INT	重复调用的次数
9	<S_A1>	AXIS	进给轴（可选）或第 1 几何轴
10	<S_A2>	AXIS	往复轴（可选）或第 2 几何轴

功能

该循环可在参考外部中断信号的条件下执行<S_H>次进给。其中的起始进给深度和结束进给深度可以不同。各次进给之间会执行切向运动。当中断条件满足时，取消深度进给。深度进给取消后，始终会执行一个完整的冲程。

过程

1. 在往复轴当前位置上启动循环。
2. 进给轴运行至起始进给深度 P2 <S_A>，进给率为 P6 <S_I>。
3. 以修光时间 P5 <S_U>进行光磨。
4. 往复轴运行，以磨削宽度 P4 <S_W>作为运行行程，横向进给率为 P7 <S_K>。
5. 进给轴运行至结束进给深度 P3 <S_B>，进给率为 P6 <S_I>。
6. 以修光时间 P5 <S_U>进行光磨。

7. 往复轴运行，以磨削宽度 P4 <S_W>作为运行行程运行至起始点，横向进给率为 P7 <S_K>。
8. 无中断：重复上述过程，直到达到编程的重复数量 P7（<S_H>）。
   中断：达到下一个起始点时结束加工。

该过程无法用一个程序段中断。

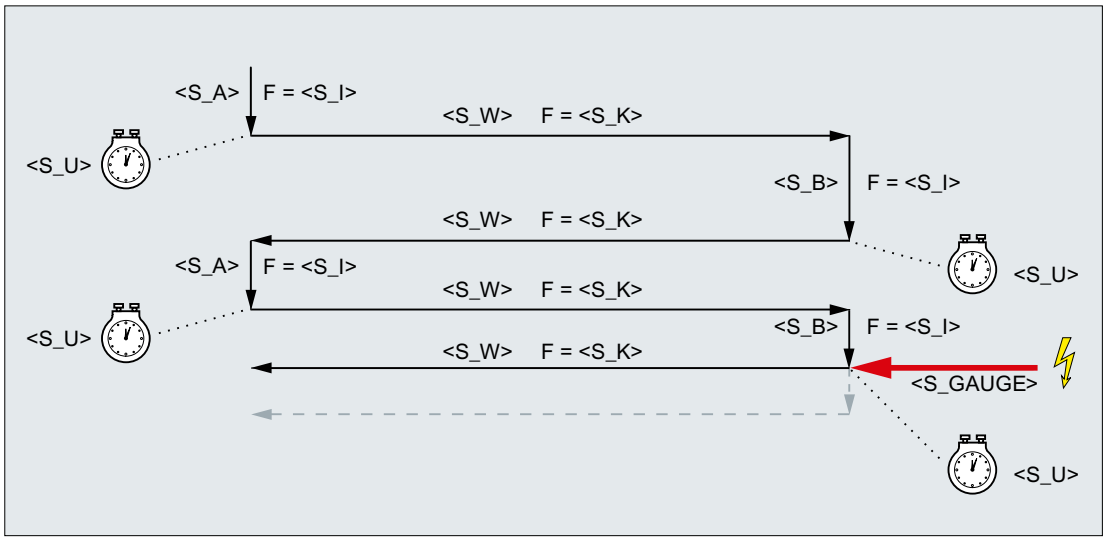


图 20-1　取消结束进给

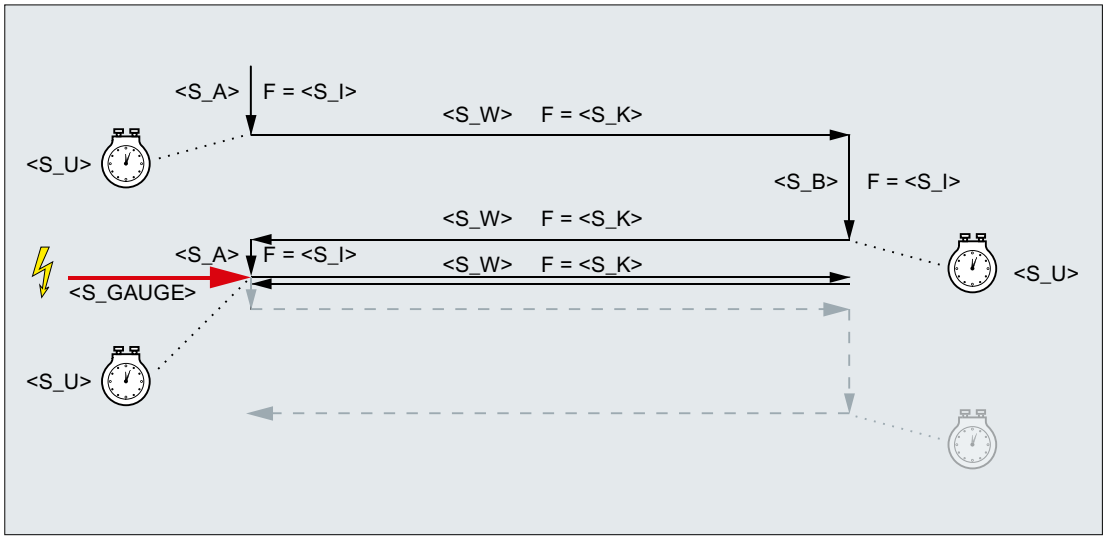


图 20-2　取消起始进给

资源

作为资源，循环使用了一个跨程序段的同步动作和一个同步动作变量。同步动作是动态地从同步动作带的空闲区域中测定的(CUS.DIR - 1 ..., CMA.DIR - 1000 ..., CST.DIR – 1199 ...)。同步动作变量使用的是 SYG_IS[1]。

示例

示例 1：具有两个冲程的往复运行

循环参数：

- 起始进给深度：0.02 mm
- 结束进给深度：0.01 mm
- 冲程：100 mm
- 修光时间：1 s
- 进给率：1 mm/min
- 横向进给率：1000 mm/min
- 重复加工的次数：2
- 往复轴和进给轴：标准几何轴

中断信号：快速输入 1 ($A_IN[1])

程序代码

```
N10 T1 D1
N20 CYCLE4072("1",0.02,0.01,100,1,1,1000,2)
N30 M30
```

示例 2： 具有两个冲程的往复运行

循环参数：

- 起始进给深度：0.02 mm
- 结束进给深度：0.01 mm
- 冲程：100 mm
- 修光时间：1 s
- 进给率：1 mm/min
- 横向进给率：1000 mm/min
- 重复加工的次数：2
- 往复轴和进给轴：标准几何轴

中断信号：变量 $A_DBR[20] < 0.01

程序代码

```
N10 T1 D1
```

程序代码

```
N20 CYCLE4072("($A_DBR[20]<0.01)",0.02,0.01,100,1,1,1000,2)
N30 M30
```

20.1.46 CYCLE4073 - 带连续进给的纵向磨削

句法

```
CYCLE4073(<S_A>, <S_B>, <S_W>, <S_U>, <S_K>, <S_H>, <S_A1>,
<S_A2>)
```

参数

编号	参数	数据类型	含义
1	<S_A>	REAL	起始进给深度
2	<S_B>	REAL	结束进给深度
3	<S_W>	REAL	磨削宽度
4	<S_U>	REAL	修光时间
5	<S_K>	REAL	横向进给率
6	<S_H>	INT	重复调用的次数
7	<S_A1>	AXIS	进给轴（可选）或第 1 几何轴
8	<S_A2>	AXIS	往复轴（可选）或第 2 几何轴

功能

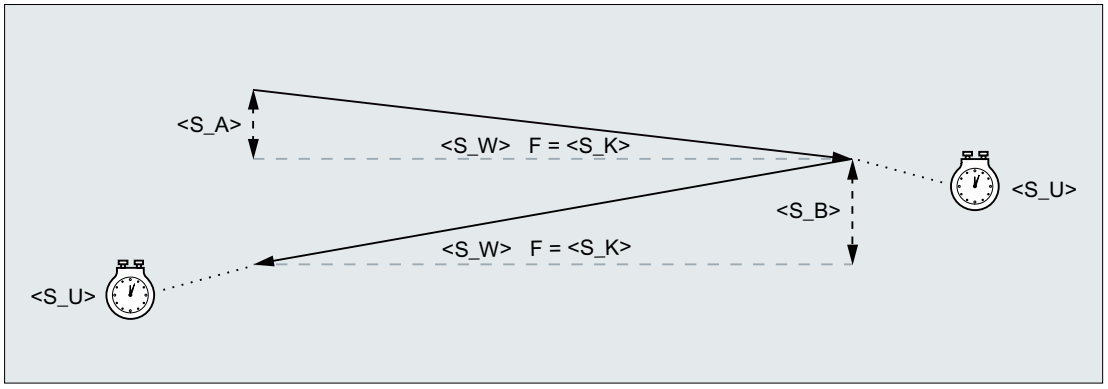
该循环用于执行<S_H>次进给。其中从起始点到结束点的进给可与从结束点到起始点的进给不同。

过程

1. 在往复轴当前位置上启动循环，进给深度为 0。
2. 往复轴运行，以磨削宽度 P3 <S_W>作为运行行程，横向进给率为 P5 <S_K>，进给深度连续增加，直到达到起始点 P1 <S_A>上的进给深度。
3. 以修光时间 P4 <S_U>进行光磨。
4. 往复轴运行，以磨削宽度 P3 <S_W>作为起始点上的运行行程，横向进给率为 P5 <S_K>，进给深度连续增加，直到达到结束点 P2 <S_B>上的进给深度。
5. 以修光时间 P4 <S_U>进行光磨。

该过程无法用一个程序段中断。

根据已编程的重复 P7（<S_H>）的数量重复该过程。



示例

具有两个冲程的往复运行

循环参数：

- 起始进给深度：0.02 mm
- 结束进给深度：0.01 mm
- 冲程：100 mm
- 修光时间：1 s
- 横向进给率：1000 mm/min
- 重复加工的次数：2
- 往复轴和进给轴：标准几何轴

程序代码

```
N10 T1 D1
N20 CYCLE4073(0.02,0.01,100,1,1000,2)
N30 M30
```

20.1.47 CYCLE4074 - 带连续进给的纵向磨削以及中断信号

句法

```
CYCLE4074(<S_GAUGE>, <S_A>, <S_B>, <S_W>, <S_U>, <S_K>, <S_H>,
<S_A1>, <S_A2>)
```

参数

编号	参数	数据类型	含义
1	<S_GAUGE>	STRING	进给中断条件： 1. 快速输入编号 2. 逻辑表达式
2	<S_A>	REAL	起始进给深度
3	<S_B>	REAL	结束进给深度
4	<S_W>	REAL	磨削宽度
5	<S_U>	REAL	修光时间
6	<S_K>	REAL	横向进给率
7	<S_H>	INT	重复调用的次数
8	<S_A1>	AXIS	进给轴（可选）或第 1 几何轴
9	<S_A2>	AXIS	往复轴（可选）或第 2 几何轴

功能

该循环可在参考外部中断信号的条件下执行<S_H>次进给。其中的起始进给深度和结束进给深度可以不同。当中断条件满足时，取消深度进给。深度进给取消后，始终会执行一个完整的冲程。

过程

1. 在往复轴当前位置上启动循环，进给深度为 0。
2. 往复轴运行，以磨削宽度 P4 <S_W>作为运行行程，横向进给率为 P6 <S_K>，进给深度连续增加，直到达到起始点 P2 <S_A>上的进给深度。
3. 以修光时间 P5 <S_U>进行光磨。
4. 往复轴运行，以磨削宽度 P4 <S_W>作为起始点上的运行行程，横向进给率为 P6 <S_K>，进给深度连续增加，直到达到结束点 P3 <S_B>上的进给深度。
5. 以修光时间 P5 <S_U>进行光磨。
6. 无中断：重复上述过程，直到达到编程的重复数量 P7（<S_H>）。
   中断：取消深度进给。达到下一个起始点时结束加工。

该过程无法用一个程序段中断。

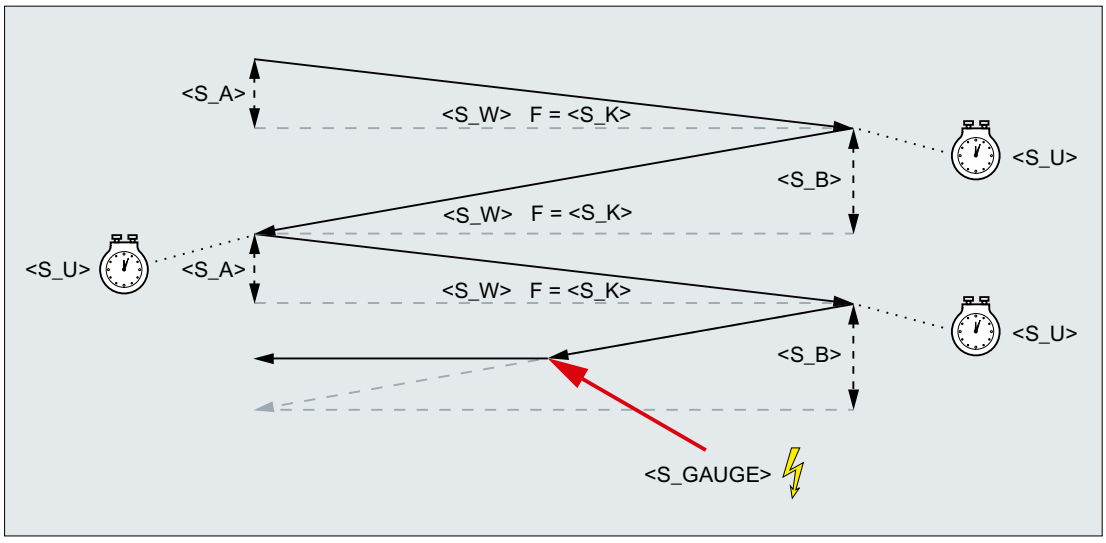


图 20-3　取消从结束点到起始点的进给

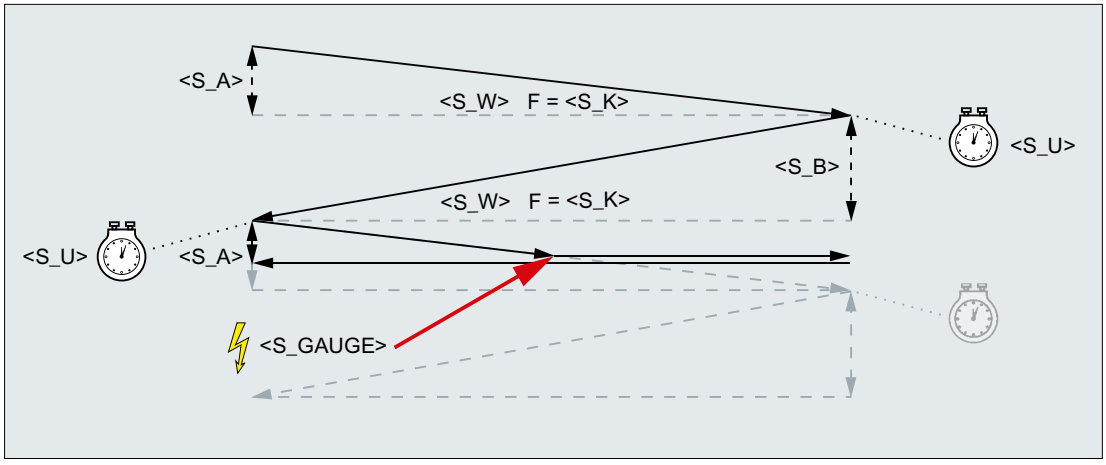


图 20-4　取消从起始点到结束点的进给

资源

作为资源，循环使用了一个跨程序段的同步动作和一个同步动作变量。同步动作是动态地从同步动作带的空闲区域中测定的(CUS.DIR - 1 ..., CMA.DIR - 1000 ..., CST.DIR – 1199 ...)。同步动作变量使用的是 SYG_IS[1]。

示例

示例 1：具有两个冲程的往复运行

循环参数：

- 起始进给深度：0.02 mm
- 结束进给深度：0.01 mm
- 冲程：100 mm
- 修光时间：1 s
- 横向进给率：1000 mm/min
- 重复加工的次数：2
- 往复轴和进给轴：标准几何轴

中断信号：快速输入 1 ($A_IN[1])

程序代码

```
N10 T1 D1
N20 CYCLE4074("1",0.02,0.01,100,1,1000,2)
N30 M30
```

示例 2： 具有两个冲程的往复运行

循环参数：

- 起始进给深度：0.02 mm
- 结束进给深度：0.01 mm
- 冲程：100 mm
- 修光时间：1 s
- 横向进给率：1000 mm/min
- 重复加工的次数：2
- 往复轴和进给轴：标准几何轴

中断信号：变量 $A_DBR[20] < 0.01

程序代码

```
N10 T1 D1
N20 CYCLE4074("($A_DBR[20]<0.01)",0.02,0.01,100,1,1000,2)
N30 M30
```

20.1.48 CYCLE4075 - 反向点处带进给的平面磨削

句法

```
CYCLE4075(<S_I>, <S_J>, <S_K>, <S_A>, <S_R>, <S_F>, <S_P>,
<S_A1>, <S_A2>)
```

参数

编号	参数	数据类型	含义
1	<S_I>	REAL	起始进给深度
2	<S_J>	REAL	结束进给深度
3	<S_K>	REAL	总进给深度
4	<S_A>	REAL	磨削宽度
5	<S_R>	REAL	进给率
6	<S_F>	REAL	横向进给率
7	<S_P>	REAL	修光时间
8	<S_A1>	AXIS	进给轴（可选）
9	<S_A2>	AXIS	往复轴（可选）

功能

该循环用于在进给工步 P3 <S_K>中执行总进给。其中的起始进给深度 P1 <S_I>和结束进给深度 P2 <S_J>可以不同。各次进给之间会执行切向运动。

位移说明 P1 到 P4 可以为负或正。

进给轴 P8 <S_A1> 和/或往复轴 P9 <S_A2>的说明是可选的。如果其中一个或两个参数都未说明，则循环会使用通道的前两个几何轴。

如果起始进给深度 P1 <S_I>和结束进给深度 P2 <S_J>的总和为 0 或总进给深度 P3 <S_K>为 0，则只执行一个光磨冲程。

过程

1. 在往复轴当前位置上启动循环。
2. 进给轴运行至起始进给深度 P1 <S_I>，进给率为 P5 <S_R>。
3. 以修光时间 P7 <S_P>进行光磨。
4. 往复轴运行，以磨削宽度 P4 <S_A>作为运行行程，横向进给率为 P6 <S_F>。

5. 进给轴运行至结束进给深度 P2 <S_J>，进给率为 P5 <S_R>。
6. 以修光时间 P7 <S_P>进行光磨。
7. 往复轴运行，以磨削宽度 P4 <S_A>作为运行行程运行至起始点，横向进给率为 P6 <S_F>。

该过程无法用一个程序段中断。

重复上述过程，直到达到总进给深度 P3 <S_K>。最后一个冲程因此不会均匀分布。

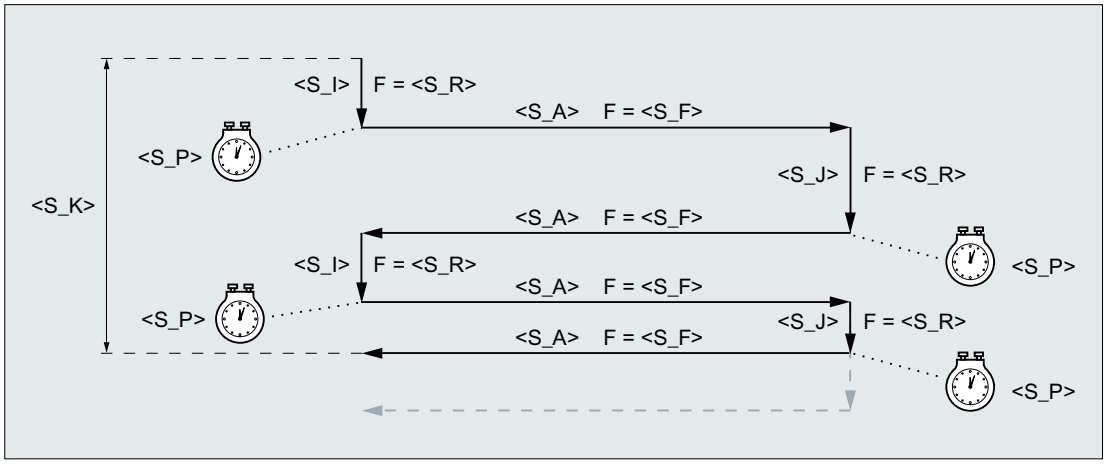


图 20-5　在第二个反向点进给时达到总进给深度

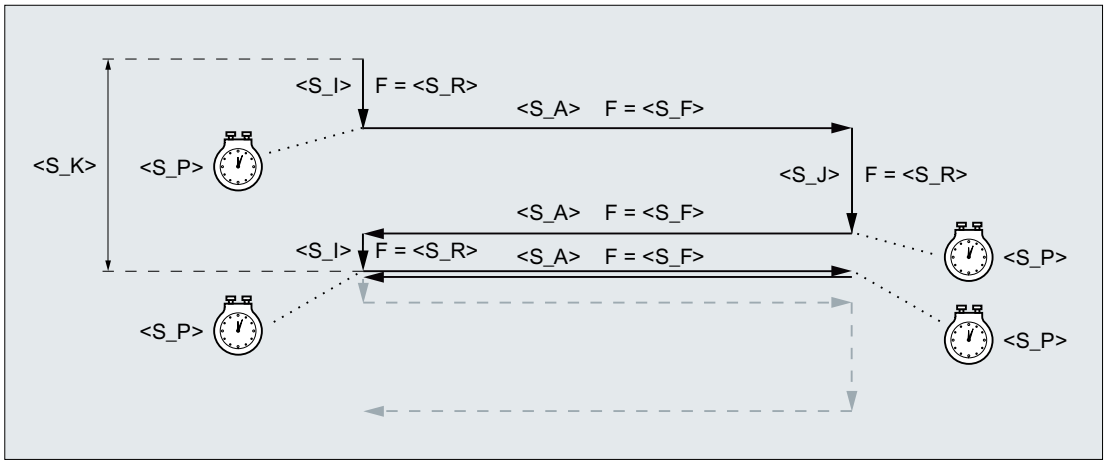


图 20-6　在第一个反向点进给时达到总进给深度

示例

往复运行：

- 0.02 mm 起始进给深度
- 0.01 mm 结束进给深度
- 总进给深度 1 mm

- 冲程 100 mm
- 进给率 1 mm/min
- 横向进给率 1000 mm/min
- 修光时间 1 s
- 标准几何轴

程序代码

```
N10 T1 D1
N20 CYCLE4075(0.02,0.01,1,100,1,1000,1)
N30 M30
```

20.1.49 CYCLE4077 - 反向点处带进给的平面磨削以及中断信号

句法

```
CYCLE4077(<S_GAUGE>, <S_I>, <S_J>, <S_K>, <S_A>, <S_R>, <S_F>,
<S_P>, <S_A1>, <S_A2>)
```

参数

编号	参数	数据类型	含义
1	<S_GAUGE>	STRING	进给中断条件： ● 快速输入编号 ● 逻辑表达式
2	<S_I>	REAL	起始进给深度
3	<S_J>	REAL	结束进给深度
4	<S_K>	REAL	总进给深度
5	<S_A>	REAL	磨削宽度
6	<S_R>	REAL	进给率
7	<S_F>	REAL	横向进给率
8	<S_P>	REAL	修光时间
9	<S_A1>	AXIS	进给轴（可选）
10	<S_A2>	AXIS	往复轴（可选）

功能

该循环用于在进给工步 P4 <S_K>中执行总进给。其中的起始进给深度 P2 <S_I>和结束进给深度 P3 <S_J>可以不同。各次进给之间会执行切向运动。当快速输入的中断信号为 1 或满足中断条件时，取消深度进给。深度进给取消后，会执行一个完整的冲程。

位移说明 P2 到 P5 可以为负或正。

进给轴 P9 <S_A1> 和/或往复轴 P10 <S_A2>的说明是可选的。如果其中一个或两个参数都未说明，则循环会使用通道的前两个几何轴。

如果起始进给深度 P2 <S_I>和结束进给深度 P3 <S_J>的总和为 0 或总进给深度 P4 <S_K>为 0，则只执行一个光磨冲程。

过程

1. 在往复轴当前位置上启动循环。
2. 进给轴运行至起始进给深度 P2 <S_I>，进给率为 P6 <S_R>。
3. 以修光时间 P8 <S_P>进行光磨。
4. 往复轴运行，以磨削宽度 P5 <S_A>作为运行行程，横向进给率为 P7 <S_F>。
5. 进给轴运行至结束进给深度 P3 <S_J>，进给率为 P6 <S_R>。
6. 以修光时间 P8 <S_P>进行光磨。
7. 往复轴运行，以磨削宽度 P5 <S_A>作为运行行程运行至起始点，横向进给率为 P7 <S_F>。
8. 无中断：重复上述过程，直到达到总进给深度 P4 <S_K>。最后一个冲程因此不会均匀分布。
   中断：在起始点上结束加工。

该过程无法用一个程序段中断。

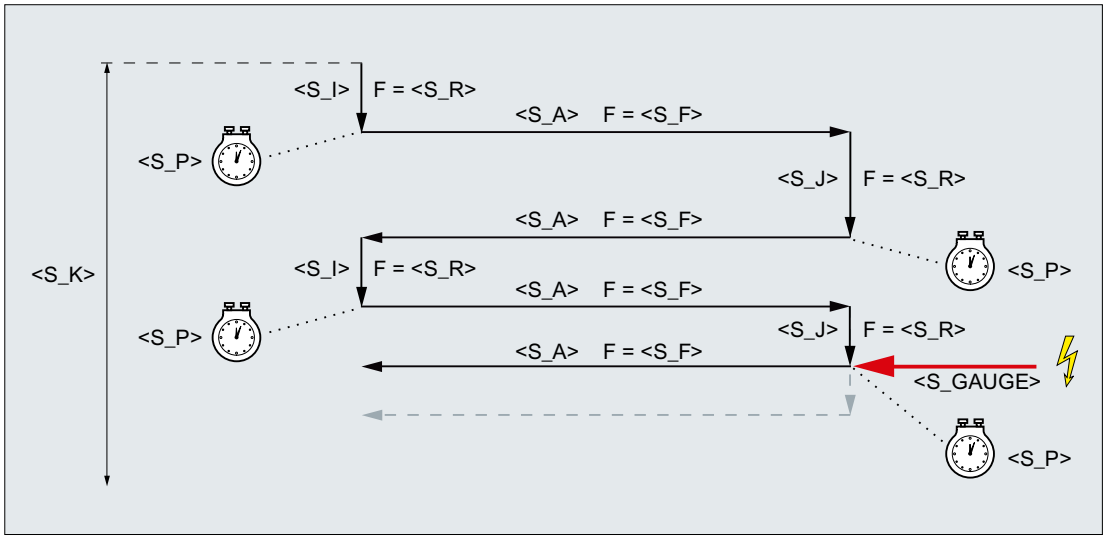


图 20-7　取消结束进给

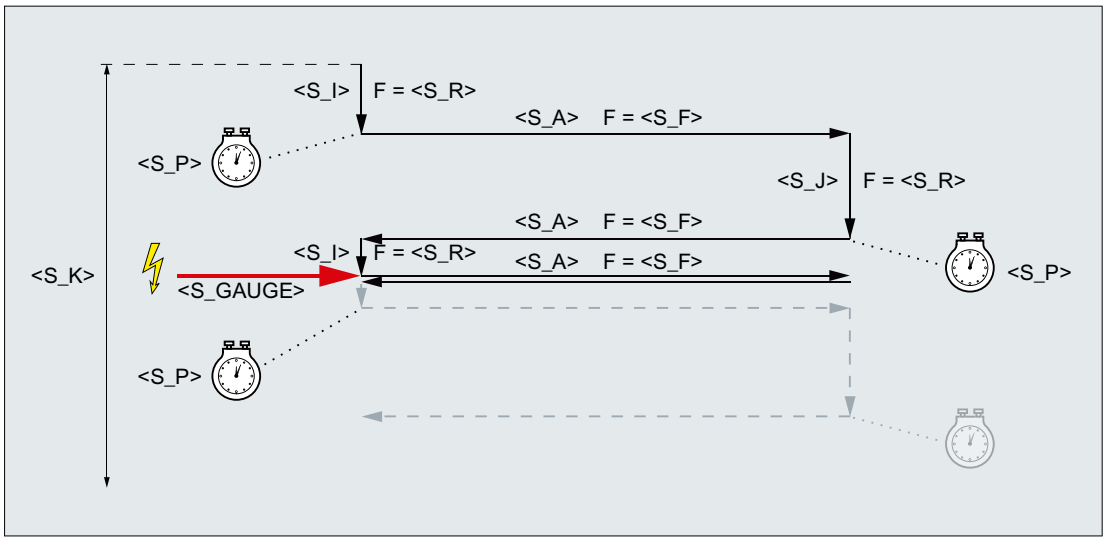


图 20-8　取消起始进给

资源

作为资源，循环使用了一个跨程序段的同步动作和一个同步动作变量。同步动作是动态地从同步动作带的空闲区域中测定的(CUS.DIR - 1 ..., CMA.DIR - 1000 ..., CST.DIR – 1199 ...)。同步动作变量使用的是 SYG_IS[1]。

示例

示例 1

往复运行：

- 0.02 mm 起始进给深度
- 0.01 mm 结束进给深度
- 总进给深度 1 mm
- 冲程 100 mm
- 进给率 1 mm/min
- 横向进给率 1000 mm/min
- 修光时间 1 s
- 标准几何轴

中断信号：快速输入 1 ($A_IN[1])

程序代码

```
N10 T1 D1
N20 CYCLE4077("1",0.02,0.01,1,100,1,1000,1)
N30 M30
```

示例 2

往复运行：

- 0.02 mm 起始进给深度
- 0.01 mm 结束进给深度
- 总进给深度 1 mm
- 冲程 100 mm
- 进给率 1 mm/min
- 横向进给率 1000 mm/min
- 修光时间 1 s
- 标准几何轴

中断信号：双端口 RAM 变量 20 小于 0.01 ($A_DBR[20] < 0.01)

程序代码

```
N10 T1 D1
N20 CYCLE4077("($A_DBR[20]<0.01)",0.02,0.01,1,100,1,1000,1)
N30 M30
```

20.1.50 CYCLE4078 - 带连续进给的平面磨削

句法

```
CYCLE4078(<S_I>, <S_J>, <S_K>, <S_A>, <S_F>, <S_P>, <S_A1>,
<S_A2>)
```

参数

编号	参数	数据类型	含义
1	<S_I>	REAL	从起始点到结束点的进给
2	<S_J>	REAL	从结束点到起始点的进给

编号	参数	数据类型	含义
3	<S_K>	REAL	总进给深度
4	<S_A>	REAL	磨削宽度
5	<S_F>	REAL	进给率
6	<S_P>	REAL	修光时间
7	<S_A1>	AXIS	进给轴（可选）
8	<S_A2>	AXIS	往复轴（可选）

功能

该循环用于通过连续进给执行总进给 P3 <S_K>。其中的起始到结束的进给深度 P1 <S_I> 和结束到起始的进给深度 P2 <S_J>可以不同。

位移说明 P1 到 P4 可以为负或正。

进给轴 P8 <S_A1> 和/或往复轴 P9 <S_A2>的说明是可选的。如果其中一个或两个参数都未说明，则循环会使用通道的前两个几何轴。

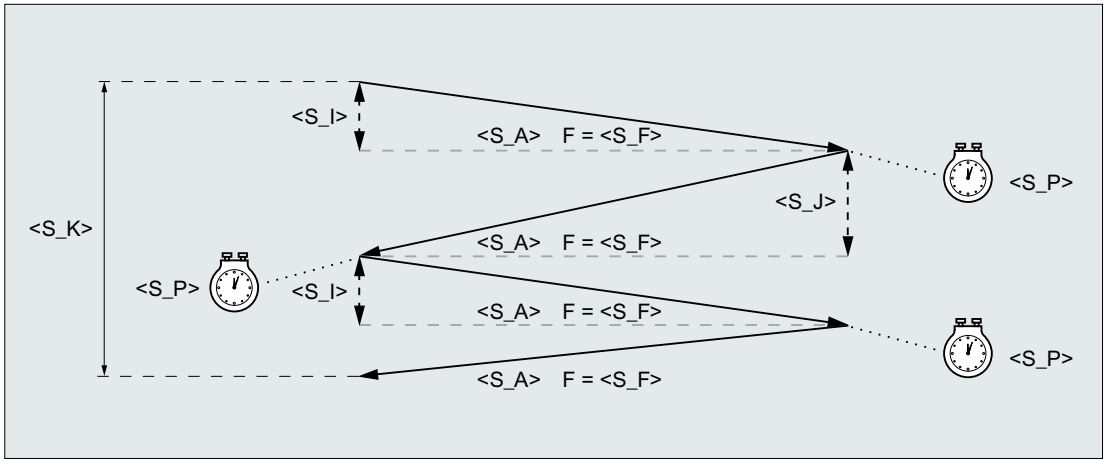
如果进给深度 P1 <S_I>和 P2 <S_J>的总和为 0 或总进给深度 P3 <S_K>为 0，则只执行一个光磨冲程。

过程

1. 在往复轴当前位置上启动循环，进给深度为 0。
2. 往复轴运行，以磨削宽度 P4 <S_A>作为运行行程，进给率为 P5 <S_F>，进给深度连续增加，直到达到起始点 P1 <S_I>上的进给深度。
3. 以修光时间 P7 <S_P>进行光磨。
4. 往复轴运行，以磨削宽度 P4 <S_A>作为起始点上的运行行程，进给率为 P5 <S_F>，进给深度连续增加，直到达到结束点 P2 <S_J>上的进给深度。
5. 以修光时间 P7 <S_P>进行光磨。
6. 往复轴运行，以磨削宽度 P4 <S_A>作为运行行程运行至起始点，进给率为 P5 <S_F>。

该过程无法用一个程序段中断。

重复上述过程，直到达到总进给深度 P3 <S_K>。最后一个冲程因此不会均匀分布。



示例

往复运行：

- 20 mm 起始进给深度
- 10 mm 结束进给深度
- 总进给深度 100 mm
- 冲程 100 mm
- 进给率 1000 mm/min
- 修光时间 1 s
- 标准几何轴

程序代码

```
N10 T1 D1
N20 CYCLE4078(20,10,100,100,1000,1)
N30 M30
```

20.1.51 CYCLE4079 - 带间歇进给的平面磨削

句法

```
CYCLE4079(<S_I>, <S_J>, <S_K>, <S_A>, <S_R>, <S_F>, <S_P>,
<S_A1>, <S_A2>)
```

参数

编号	参数	数据类型	含义
1	<S_I>	REAL	起始进给深度
2	<S_J>	REAL	结束进给深度
3	<S_K>	REAL	总进给深度
4	<S_A>	REAL	磨削宽度
5	<S_R>	REAL	进给率
6	<S_F>	REAL	横向进给率
7	<S_P>	REAL	修光时间
8	<S_A1>	AXIS	进给轴（可选）
9	<S_A2>	AXIS	往复轴（可选）

功能

该循环用于在进给工步 P3 <S_K>中执行总进给。其中的起始进给深度 P1 <S_I>和结束进给深度 P2 <S_J>可以不同。各次进给之间会执行切向运动。

位移说明 P1 到 P4 可以为负或正。

进给轴 P8 <S_A1> 和/或往复轴 P9 <S_A2>的说明是可选的。如果其中一个或两个参数都未说明，则循环会使用通道的前两个几何轴。

如果起始进给深度 P1 <S_I>和结束进给深度 P2 <S_J>的总和为 0 或总进给深度 P3 <S_K>为 0，则只执行一个光磨冲程。

过程

1. 在往复轴当前位置上启动循环。
2. 进给轴运行至起始进给深度 P1 <S_I>，进给率为 P5 <S_R>。
3. 以修光时间 P7 <S_P>进行光磨。
4. 往复轴运行，以磨削宽度 P4 <S_A>作为运行行程，横向进给率为 P6 <S_F>。
5. 进给轴运行至结束进给深度 P2 <S_J>，进给率为 P5 <S_R>。
6. 以修光时间 P7 <S_P>进行光磨。
7. 往复轴运行，以磨削宽度 P4 <S_A>作为运行行程运行至起始点，横向进给率为 P6 <S_F>。

该过程无法用一个程序段中断。

重复上述过程，直到达到总进给深度 P3 <S_K>。最后一个冲程因此不会均匀分布。

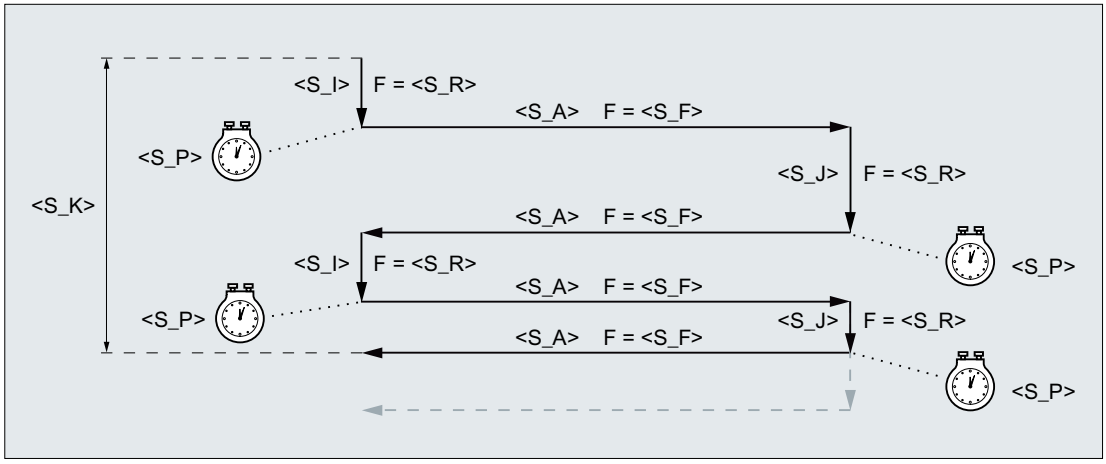


图 20-9　在第二个反向点进给时达到总进给深度

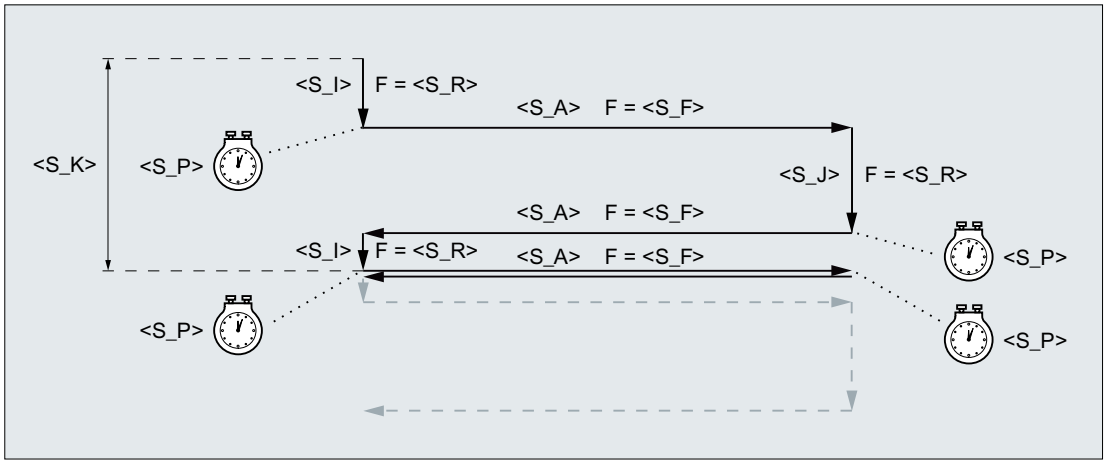


图 20-10　在第一个反向点进给时达到总进给深度

示例

往复运行：

- 0.02 mm 起始进给深度
- 0.01 mm 结束进给深度
- 总进给深度 1 mm
- 冲程 100 mm
- 进给率 1 mm/min
- 横向进给率 1000 mm/min

- 修光时间 1 s
- 标准几何轴

程序代码

```
N10 T1 D1
N20 CYCLE4079(0.02,0.01,1,100,1,1000,1)
N30 M30
```

20.1.52 GROUP_BEGIN - 程序块开始

句法

```
GROUP_BEGIN(<_LEVEL>, <_NAME>, <_SP>, <_MODE>, <S_ICON>)
```

参数

编号	对话框参数	内部参数	数据类型	含义		
1		<_LEVEL>	INT	平面		
				0 =	主级面	
				1 =	第 1 子级	
2		<_NAME>	STRING[128]	块名称		
3		<_SP>	INT	主轴		
				0 =	无主轴	
				1 =	主主轴	
				2 =	副主轴	
4		<_MODE>	INT	模式		
				位 0	= 1	GROUP_ADDEND 已存在
				位 1	= 1	ShopTurn:自动回退（至换刀点）
				位 12	预留	
				位 13	预留	
5		<S_ICON>	STRING[32]	图标的名称（仅用于操作界面）		

20.1.53 GROUP_END - 程序块结束

句法

```
GROUP_END(<_LEVEL>, <_SP>)
```

参数

编号	对话框参数	内部参数	数据类型	含义	
1		<_LEVEL>	INT	平面	
				0 =	主级面
				1 =	第 1 子级
2		<_SP>	INT	主轴	
				0 =	无主轴
				1 =	主主轴
				2 =	副主轴

20.1.54 GROUP_ADDEND - 附加运行结束

句法

```
GROUP_ADDEND(<_LEVEL>, <_SP>)
```

参数

编号	对话框参数	内部参数	数据类型	含义	
1		<_LEVEL>	INT	平面	
				0 =	主级面
				1 =	第 1 子级
2		<_SP>	INT	主轴	
				0 =	无主轴
				1 =	主主轴
				2 =	副主轴

20.1.55 前提条件

20.1.55.1 工艺对话框内的工艺调整

在激活“工艺调整”后，便可以在不同的循环对话框内选择简化输入，此时对话框内只显示最重要的循环参数。

在以下循环对话框中可以选择简化输入：

工艺	循环对话框
钻削	深孔钻削
	攻丝
铣削	矩形腔
	轮廓铣削：型腔
车削	车削螺纹：纵向
	轮廓车削：切削
	轮廓车削：切槽
	轮廓车削：往复车削

在上述循环对话框中，操作界面上会提供选项“输入：**简单**”和“输入：**完整**”。

没有显示的循环参数

在简单输入中没有显示的循环参数要么预设为工艺适用、无法修改的固定值，要么采用由通道专用循环设定数据设置的值。参见下文“调试”>“通道专用循环设定数据”

“输入：完整” > “输入：简单”

如果在完整输入设置下填满了一个循环对话框，但接着切换到简单输入，那么在生成循环调用时那些不再显示的参数会使用默认值或设定数据值。

调试

通道专用配置机床数据

使用以下机床数据可以激活循环对话框内的工艺调整功能：

MD52210 $MCS_FUNCTION_MASK_DISP，位 9 = 1（显示选项“简单输入”）

通道专用循环设定数据

如果激活了循环对话框内的工艺调整功能，则可以通过以下设定数据预设一些循环参数的值。

序号	名称	含义
SD55300	$SCS_EASY_SAFETY_CLEARANCE	安全距离
SD55301	$SCS_EASY_DWELL_TIME	暂停时间
SD55305	$SCS_EASY_DRILL_DEEP_FD1	深孔钻削：百分比值：第 1 进给率
SD55306	$SCS_EASY_DRILL_DEEP_DF	深孔钻削：百分比值：进刀量
SD55307	$SCS_EASY_DRILL_DEEP_V1	深孔钻削：最小切深
SD55308	$SCS_EASY_DRILL_DEEP_V2	深孔钻削：回退量
SD55309	$SCS_EASY_THREAD_RETURN_DIST	车削螺纹：回程距离

20.2 测量循环

测量循环是西门子提供的用于解决特定测量任务的特殊子程序。 正如一般的循环，测量循环也可以通过参数根据具体问题加以调整。

测量循环适用的工艺和范围包括：

- 车床/铣床的刀具测量
- 车床/铣床的工件测量

文献

对测量循环的详细说明请见：

测量循环编程手册

表 21

21.1 指令

说明

循环

指令列表包含了 NC 程序（G 代码）中可能出现的所有循环，即可通过屏幕在程序编辑器中编写的循环或者无需编程支持即可编写的循环。不考虑因兼容性因素仍保留在控制系统中，但已不能通过 SINUMERIK Operate 程序编辑器进行编辑的循环（“兼容性循环”）。

指令 A - C

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
:	O	NC 主程序段号、跳转标记结束、连接运算符		+		PGAsl
*	O	乘法运算符		+		PGAsl
+	O	加法运算符		+		PGAsl
-	O	减法运算符		+		PGAsl
<	O	关系运算符，小于号		+		PGAsl
<<	O	字符串的连接运算符		+		PGAsl
<=	O	关系运算符，小于等于		+		PGAsl
=	O	赋值运算符		+		PGAsl
>=	O	关系运算符，大于等于		+		PGAsl
/	O	除法运算符		+		PGAsl
/0 … … /7		程序段跳转（第 1 跳转级）。 ... ... 程序段跳转（第 8 跳转级）		+		PGsl
A	A	轴名称	m/s	+		PGAsl
A2	A	刀具定向：RPY 角或欧拉角	s	+		PGAsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
A3	A	刀具定向：第 1 个方向矢量的分量	s	+		PGAsl
A4	A	刀具定向：第 1 个程序段开始处的面正交矢量分量	s	+		PGAsl
A5	A	刀具定向：第 1 个程序段结束处的面正交矢量分量	s	+		PGAsl
A6	A	刀具定向：第 1 个圆锥旋转轴方向矢量的分量	s	+		PGAsl
A7	A	刀具定向：第 1 个圆锥表面上中间定向的矢量分量	s	+		PGAsl
ABS	F	绝对值		+	+	PGAsl
AC	K	坐标/位置的绝对尺寸值	s	+		PGsl
ACC	K	当前轴向加速度的控制	m	+	+	PGsl
ACCLIMA	K	当前最大轴向加速度的控制	m	+	+	PGAsl
ACN	K	绝对尺寸值，用于回转轴运行到负向上的某个位置	s	+		PGsl
ACOS	F	反余弦 （三角函数）		+	+	PGAsl
ACP	K	绝对尺寸值，用于回转轴运行到正向上的某个位置	s	+		PGsl
ACTBLOCNO	P	输出报警程序段的当前编号，即使当前程序段显示被抑制(DISPLOF)！		+		PGAsl
ADDFRAME	F	计算并激活测得的框架		+	-	PGAsl, FB1sl (K2)
ADIS	A	用于路径功能 G1, G2, G3, ... 的平滑距离	m	+		PGsl
ADISPOS	A	用于快速移动 G0 的平滑距离	m	+		PGsl
ADISPOSA	P	用于 IPOBRKA 的公差窗口尺寸	m	+	+	PGAsl
ALF	A	快速退刀角度	m	+		PGAsl
AMIRROR	G	可编程镜像	s	+		PGsl
AND	K	逻辑与		+		PGAsl
ANG	A	轮廓线角度	s	+		PGsl
AP	A	极角	m/s	+		PGsl

指令	类型 1)	含义	W 2)	TP 3)	SA 4)	说明参见 5)
1) 2) 3) 4) 5) 详细说明参见图例 (页 959)。						
APR	K	读取/显示的存取保护		+		PGAsl
APRB	K	读取 BTSS 的权限		+		PGAsl
APRP	K	读取零件程序的权限		+		PGAsl
APW	K	写存取保护		+		PGAsl
APWB	K	写入 BTSS 的权限		+		PGAsl
APWP	K	写入零件程序的权限		+		PGAsl
APX	K	存取保护定义，用于执行指定的语言单元		+		PGAsl
AR	A	张角	m/s	+		PGsl
AROT	G	可编程旋转	s	+		PGsl
AROTS	G	可编程的框架旋转，带立体角	s	+		PGsl
AS	K	宏指令定义		+		PGAsl
ASCALE	G	可编程缩放	s	+		PGsl
ASIN	F	反正弦算术函数		+	+	PGAsl
ASPLINE	G	Akima 样条	m	+		PGAsl
ATAN2	F	反正切 2		+	+	PGAsl
ATOL	A	用于压缩器功能、方向平滑以及平滑方式的轴专有公差	m	+		PGAsl
ATRANS	G	叠加的可编程零点偏移	s	+		PGsl
AUXFUDEL	P	将通道专用辅助功能从全局列表删除		+	-	FB1sl (H2)
AUXFUDELG	P	将一个通道专用辅助功能组的所有辅助功能从全局列表删除		+	-	FB1sl (H2)
AUXFUMSEQ	P	获取 M 辅助功能的输出顺序		+	-	FB1sl (H2)
AUXFUSYNC	P	通过辅助功能全局列表为通道专用 SERUPRO-Ende-ASUP 创建一个类型为 String 的完整零件程序段		+	-	FB1sl (H2)
AX	K	可变轴标识符	m/s	+		PGAsl
AXCTSWE	P	轴容器旋转		+	-	PGAsl
AXCTSWEC	P	取消轴容器旋转使能		+	+	PGAsl
AXCTSWED	P	旋转轴容器（用于调试的指令类型！）		+	-	PGAsl
AXIS	K	轴标识符、轴地址		+		PGAsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
AXNAME	F	把输入字符串转换为一个轴标识符		+	-	PGAsl
AXSTRING	F	把字符串转换为主轴号		+	-	PGAsl
AXTOCHAN	P	指定轴为某一特定通道。由 NC 程序和从同步动作都可以。		+	+	PGAsl
AXTOSPI	F	将轴标识符转换为一个主轴下标		+	-	PGAsl
B	A	轴名称	m/s	+		PGAsl
B2	A	刀具定向：RPY 角或欧拉角	s	+		PGAsl
B3	A	刀具定向：第 2 个方向矢量的分量	s	+		PGAsl
B4	A	刀具定向：第 2 个程序段开始处的面正交矢量分量	s	+		PGAsl
B5	A	刀具定向：第 2 个程序段结束处的面正交矢量分量	s	+		PGAsl
B6	A	刀具定向：第 2 个圆锥旋转轴方向矢量的分量	s	+		PGAsl
B7	A	刀具定向：第 2 个圆锥表面上中间定向的矢量分量	s	+		PGAsl
B_AND	O	位方式“与”		+		PGAsl
B_OR	O	位方式“或”		+		PGAsl
B_NOT	O	位方式“非”		+		PGAsl
B_XOR	O	位方式“异－或”		+		PGAsl
BAUTO	G	通过后面的 3 个点定义样条段	m	+		PGAsl
BLOCK	K	和关键字 TO 一起，在一个间接的子程序调用中定义一个需要处理的程序部分		+		PGAsl
BLSYNC	K	在下一个程序段转换时才开始处理中断程序		+		PGAsl
BNAT [6]	G	自然过渡到第一个样条程序段	m	+		PGAsl
BOOL	K	数据类型：真值 TRUE/FALSE 或者 1/0		+		PGAsl
BOUND	F	检查，值是否在已定义的值域中。同时返回检验值。		+	+	PGAsl
BRISK [6]	G	跃变式的轨迹加速度	m	+		PGAsl
BRISKA	P	激活编程轴跃变式的轨迹加速度		+	-	PGAsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
BSPLINE	G	B 样条	m	+		PGAsl
BTAN	G	切线过渡到第一个样条程序段	m	+		PGAsl
C	A	轴名称	m/s	+		PGAsl
C2	A	刀具定向：RPY 角或欧拉角	s	+		PGAsl
C3	A	刀具定向：第 3 个方向矢量的分量	s	+		PGAsl
C4	A	刀具定向：第 3 个程序段开始处的面正交矢量分量	s	+		PGAsl
C5	A	刀具定向：第 3 个程序段结束处的面正交矢量分量	s	+		PGAsl
C6	A	刀具定向：第 3 个圆锥旋转轴方向矢量的分量	s	+		PGAsl
C7	A	刀具定向：第 3 个圆锥表面上中间定向的矢量分量	s	+		PGAsl
CAC	K	运行至某个绝对位置		+		PGAsl
CACN	K	从负方向运行至表中某数值所在位置（绝对值）		+		PGAsl
CACP	K	从正方向运行至表中某数值所在位置（绝对值）		+		PGAsl
CALCDAT	F	从 3 个或者 4 个点中计算某个圆的半径和中点。		+	-	PGAsl
CALCPOSI	F	检查是否超出保护区、工作范围限制和软件限位开关		+	-	PGAsl
CALL	K	间接子程序调用		+		PGAsl
CALLPATH	P	子程序调用时可编程的查找路径		+	-	PGAsl
CANCEL	P	中止模态同步动作		+	-	FBSYsl
CASE	K	有条件程序跳转		+		PGAsl
CDC	K	直接运行至某位置		+		PGAsl
CDOF [6]	G	防撞监控关闭	m	+		PGsl
CDOF2	G	防撞监控关闭，3D 圆周铣削时	m	+		PGsl
CDON	G	防撞监控启用	m	+		PGsl
CFC [6]	G	轮廓处的恒定进给	m	+		PGsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
CFIN	G	仅内曲面上的恒定进给，而不是外曲面上	m	+		PGsl
CFINE	F	为 FRAME 变量赋值一个精偏		+	-	PGAsl
CFTCP	G	刀尖基准点、中心轨迹上的恒定进给	m	+		PGsl
CHAN	K	规定数据有效区。		+		PGAsl
CHANDATA	P	设定通道号，用于通道数据存取		+	-	PGAsl
CHAR	K	数据类型：ASCII 字符		+		PGAsl
CHF	A	倒角； 值＝ 倒角长度	s	+		PGsl
CHKDM	F	刀库的单一性检查		+	-	FBWsl
CHKDNO	F	D 号的单一性检查		+	-	PGAsl
CHR	A	倒角； 值＝倒角长度，在运动方向		+		PGsl
CIC	K	运行至表中某数值位置（增量值）		+		PGAsl
CIP	G	通过中间点进行圆弧插补	m	+		PGsl
CLEARM	P	复位一个/多个标号，用于通道协调		+	+	PGAsl
CLRINT	P	撤销选择中断		+	-	PGAsl
CMIRROR	F	对一个坐标轴的镜像		+	-	PGAsl
COARSEA	K	在到达“粗准停”时运行结束	m	+		PGAsl
COLLPAIR	F	检测保护区能否形成一个防撞对		+		PGAsl
COMPCAD	G	激活压缩器功能 COMPCAD	m	+		PGAsl
COMPCURV	G	激活压缩器功能 COMPCURV	m	+		PGAsl
COMPLETE		读入和读出数据的控制指令		+		PGAsl
COMPOF [6]	G	关闭数控程序段压缩器	m	+		PGAsl
COMPON	G	激活压缩器功能 COMPON	m	+		PGAsl
COMPSURF	G	激活压缩器功能 COMPSURF	m	+		PGAsl
CONTDCON	P	打开以表格形式译码的轮廓		+	-	PGAsl
CONTPRON	P	激活参考点处理		+	-	PGAsl
CORROF	P	取消所有有效的叠加运动。		+	-	PGsl

指令	类型 1)	含义	W 2)	TP 3)	SA 4)	说明参见 5)
1) 2) 3) 4) 5) 详细说明参见图例 (页 959)。						
CORRTRAFO	F	更改机床运动模型中的偏移矢量或定向轴的方向矢量		+	-	PGAsl
COS	F	余弦 （三角函数）		+	+	PGAsl
COUPDEF	P	定义 ELG 组合/同步主轴组合		+	-	PGAsl
COUPDEL	P	删除 ELG 组合		+	-	PGAsl
COUPOF	P	撤消 ELG 组/同步主轴对		+	-	PGAsl
COUPOFS	P	关闭 ELG 组合/同步主轴对，停止跟随主轴		+	-	PGAsl
COUPON	P	激活 ELG 组/同步主轴对		+	-	PGAsl
COUPONC	P	激活 ELG 组合/同步主轴对，并采用上一次编程		+	-	PGAsl
COUPRES	P	复位 ELG 组合		+	-	PGAsl
CP 6)	G	轨迹运行	m	+		PGAsl
CPBC	K	同类耦合：程序段切换标准		+	+	FB3sl (M3)
CPDEF	K	同类耦合：创建耦合模块		+	+	FB3sl (M3)
CPDEL	K	同类耦合：删除耦合模块		+	+	FB3sl (M3)
CPFMOF	K	同类耦合：完全关闭时跟随轴的响应方式		+	+	FB3sl (M3)
CPFMON	K	同类耦合：激活时跟随轴的响应方式		+	+	FB3sl (M3)
CPFMSON	K	同类耦合：同步模式		+	+	FB3sl (M3)
CPFPOS	K	同类耦合：跟随轴同步		+	+	FB3sl (M3)
CPFRS	K	同类耦合：参考坐标系		+	+	FB3sl (M3)
CPLA	K	同类耦合：定义一个引导轴		+	-	FB3sl (M3)
CPLCTID	K	同类耦合：曲线图表的编号		+	+	FB3sl (M3)
CPLDEF	K	同类耦合：定义引导轴并创建耦合模块		+	+	FB3sl (M3)
CPLDEL	K	同类耦合：删除耦合模块中的引导轴		+	+	FB3sl (M3)
CPLDEN	K	同类耦合：耦合系数分母		+	+	FB3sl (M3)
CPLINSC	K	同类耦合：引导轴输入值的缩放系数		+	+	FB3sl (M3)
CPLINTR	K	同类耦合：引导轴输入值的偏移值		+	+	FB3sl (M3)
CPLNUM	K	同类耦合：耦合系数分子		+	+	FB3sl (M3)

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
CPLOF	K	同类耦合：关闭耦合模块中的引导轴		+	+	FB3sl (M3)
CPLON	K	同类耦合：激活耦合模块中的引导轴		+	+	FB3sl (M3)
CPLOUTSC	K	同类耦合：耦合输出值的缩放系数		+	+	FB3sl (M3)
CPLOUTTR	K	同类耦合：耦合输出值的偏移值		+	+	FB3sl (M3)
CPLPOS	K	同类耦合：引导轴的同步位置		+	+	FB3sl (M3)
CPLSETVAL	K	同类耦合：耦合基准		+	+	FB3sl (M3)
CPMALARM	K	同类耦合：抑制耦合专用报警输出		+	+	FB3sl (M3)
CPMBRAKE	K	同类耦合：特定停止信号和指令下跟随轴的响应方式		+	-	FB3sl (M3)
CPMPRT	K	同类耦合：通过程序测试进行程序段查找的情况下零件程序启动时的耦合响应方式		+	+	FB3sl (M3)
CPMRESET	K	同类耦合：复位时耦合的响应方式		+	+	FB3sl (M3)
CPMSTART	K	同类耦合：零件程序启动时耦合的响应方式		+	+	FB3sl (M3)
CPMVDI	K	同类耦合：对特定 NC/PLC 接口信号的跟随轴响应方式		+	+	FB3sl (M3)
CPOF	K	同类耦合：关闭耦合模块		+	+	FB3sl (M3)
CPON	K	同类耦合：激活耦合模块		+	+	FB3sl (M3)
CPRECOF [6]	G	取消可编程的轮廓精度	m	+		PGAsl
CPRECON	G	激活可编程的轮廓精度	m	+		PGAsl
CPRES	K	同类耦合：激活配置的同步主轴耦合数据		+	-	FB3sl (M3)
CPROT	P	激活/关闭通道专用的保护区		+	-	PGAsl
CPROTDEF	P	定义通道专用的保护区		+	-	PGAsl
CPSETTYPE	K	同类耦合：耦合类型		+	+	FB3sl (M3)
CPSYNCOP	K	同类耦合：“粗略”位置同步运行的阈值		+	+	FB3sl (M3)
CPSYNCOP2	K	同类耦合：“粗”位置同步运行的阈值 2		+	+	FB3sl (M3)

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
CPSYNCOV	K	同类耦合：“粗略”速度同步运行的阈值		+	+	FB3sl (M3)
CPSYNFIP	K	同类耦合：“精细”位置同步运行的阈值		+	+	FB3sl (M3)
CPSYNFIP2	K	同类耦合：“精”位置同步运行的阈值 2		+	+	FB3sl (M3)
CPSYNFIV	K	同类耦合：“精细”速度同步运行的阈值		+	+	FB3sl (M3)
CR	A	圆弧半径	s	+		PGsl
CROT	F	旋转当前坐标系		+	-	PGAsl
CROTS	F	以立体角进行可编程旋转（在指定轴旋转）	s	+	-	PGsl
CRPL	F	在任意平面内旋转框架		+	-	FB1sl (K2)
CSCALE	F	比例系数，用于多个轴		+	-	PGAsl
CSPLINE	F	立方样条	m	+		PGAsl
CT	G	切线过渡的圆弧	m	+		PGsl
CTAB	F	依据曲线表中的引导轴位置计算跟轴位置		+	+	PGAsl
CTABDEF	P	激活表格定义		+	-	PGAsl
CTABDEL	P	删除曲线表		+	-	PGAsl
CTABEND	P	关闭表格定义		+	-	PGAsl
CTABEXISTS	F	检查带号 n 的曲线表		+	+	PGAsl
CTABFNO	F	存储器中尚可使用的曲线表个数		+	+	PGAsl
CTABFPOL	F	存储器中尚可使用的多项式个数		+	+	PGAsl
CTABFSEG	F	存储器中尚可使用的曲线段个数		+	+	PGAsl
CTABID	F	提供曲线表的表编号		+	+	PGAsl
CTABINV	F	依据曲线表中的跟随轴位置计算引导轴位置		+	+	PGAsl
CTABISLOCK	F	返回 n 号的曲线表的禁止状态		+	+	PGAsl
CTABLOCK	P	禁止删除和改写		+	+	PGAsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
CTABMEMTYP	F	返回存储器，在该存储器中编制了 n 号的曲线表。		+	+	PGAsl
CTABMPOL	F	存储器中最大可用的多项式个数		+	+	PGAsl
CTABMSEG	F	存储器中最大可用的曲线段个数		+	+	PGAsl
CTABNO	F	在 SRAM 或者 DRAM 中定义的曲线表的个数		+	+	FB3sl (M3)
CTABNOMEM	F	在 SRAM 或者 DRAM 中定义的曲线表的个数		+	+	PGAsl
CTABPERIOD	F	返回编号为 n 的曲线表的周期性数据		+	+	PGAsl
CTABPOL	F	存储器中已经使用的多项式个数		+	+	PGAsl
CTABPOLID	F	n 号曲线表所使用的曲线多项式个数		+	+	PGAsl
CTABSEG	F	存储器中已经使用的曲线段的个数		+	+	PGAsl
CTABSEGID	F	n 号曲线表所使用的曲线段个数		+	+	PGAsl
CTABSEV	F	提供曲线表一个分段的跟随轴终值		+	+	PGAsl
CTABSSV	F	提供曲线表一个分段的跟随轴起始值		+	+	PGAsl
CTABTEP	F	提供曲线表结束处提供引导轴的值		+	+	PGAsl
CTABTEV	F	提供曲线表结束处跟随轴的值		+	+	PGAsl
CTABTMAX	F	提供曲线表跟随轴的最大值		+	+	PGAsl
CTABTMIN	F	提供曲线表跟随轴的最小值		+	+	PGAsl
CTABTSP	F	提供曲线表开始处引导轴的值		+	+	PGAsl
CTABTSV	F	提供曲线表开始处跟随轴的值		+	+	PGAsl
CTABUNLOCK	P	取消禁止删除和改写		+	+	PGAsl
CTOL	A	用于压缩器功能、定向平滑和平滑方式的轮廓公差	m	+		PGAsl
CTRANS	F	零点偏移，用于多个轴		+	-	PGAsl
CUT2D [6]	G	2D 刀具半径补偿	m	+		PGsl
CUT2DD	G	基于差动刀具的 2½ D 刀具半径补偿	m	+		PGsl
CUT2DF	G	2D 刀具半径补偿，相对于当前框架（倾斜平面）	m	+		PGsl
CUT2DFD	G	基于差动刀具的 2½ D 刀具半径补偿，相对于当前框架（倾斜平面）	m	+		PGsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
CUT3DC	G	圆周铣削的 3D 刀具半径补偿	m	+		PGAsl
CUT3DCC	G	圆周铣削的 3D 刀具半径补偿，考虑了具有 3D 半径补偿的限制面：加工面上的轮廓	m	+		PGAsl
CUT3DCCD	G	圆周铣削的 3D 刀具半径补偿，考虑了带有刀具中心点轨迹上的差分刀具的限制面：向限制面进给	m	+		PGAsl
CUT3DCD	G	以差分刀具为基准的圆周铣削 3D 刀具半径补偿	m	+		PGAsl
CUT3DF	G	带有定向变化的端面铣削的 3D 刀具半径补偿	m	+		PGAsl
CUT3DFD	G	以差分刀具为基准的带有定向变化的端面铣削的 3D 刀具半径补偿	m	+		PGAsl
CUT3DFF	G	带有恒定定向的端面铣削的 3D 刀具半径补偿。刀具定向通过 G17 - G19 来确定并且如果有可能是通过框架旋转的方向。	m	+		PGAsl
CUT3DFS	G	带有恒定定向的端面铣削的 3D 刀具半径补偿。刀具定向已通过 G17 - G19 确定且不受框架影响。	m	+		PGAsl
CUTCONOF [6]	G	取消恒定刀具半径补偿	m	+		PGsl
CUTCONON	G	激活恒定刀具半径补偿	m	+		PGsl
CUTMOD	A	激活可旋转刀具的补偿数据修改功能（与可定向刀架组合使用）	m	+		PGAsl
CUTMODK	A	激活可旋转刀具的补偿数据修改功能（与通过运动链定义的定向转换组合使用）	m	+		PGAsl
CYCLE60	C (T)	雕刻循环		+		PGAsl
CYCLE61	C (T)	平面铣削		+		PGAsl
CYCLE62	C (T)	轮廓调用		+		PGAsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
CYCLE63	C (T)	轮廓腔铣削		+		PGAsl
CYCLE64	C (T)	轮廓腔预钻削		+		PGAsl
CYCLE70	C (T)	螺纹铣削		+		PGAsl
CYCLE72	C (T)	轨迹铣削		+		PGAsl
CYCLE76	C (T)	铣削矩形凸台		+		PGAsl
CYCLE77	C (T)	铣削圆形凸台		+		PGAsl
CYCLE78	C (T)	钻削和螺纹铣削		+		PGAsl
CYCLE79	C (T)	多边形		+		PGAsl
CYCLE81	C (T)	钻削，钻中心孔		+		PGAsl
CYCLE82	C (T)	钻削，锪平面		+		PGAsl
CYCLE83	C (T)	深孔钻削		+		PGAsl
CYCLE84	C (T)	刚性攻丝		+		PGAsl
CYCLE85	C (T)	铰孔		+		PGAsl
CYCLE86	C (T)	镗孔		+		PGAsl
CYCLE92	C (T)	切断		+		PGAsl
CYCLE95	C (T)	轮廓切削		+		PGAsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
CYCLE98	C (T)	螺纹链		+		PGAsl
CYCLE99	C (T)	车削螺纹		+		PGAsl
CYCLE150	C (M)	显示/记录测量结果		+		BNMsl
CYCLE435	C (T)	计算修整器位置		+		PGAsl
CYCLE495	C (T)	成型磨削		+		PGAsl
CYCLE750	C (A)	CYCLE751 ... CYCLE759 的内部工作循环（包括原始功能调用的 MMC 指令）		-		FB3sl (T4)
CYCLE751	C (A)	打开/执行/关闭优化会议		M		FB3sl (T4)
CYCLE752	C (A)	将轴添加到优化会议中		M		FB3sl(T4)
CYCLE753	C (A)	选择优化模式		M		FB3sl (T4)
CYCLE754	C (A)	添加/删除数据组		M		FB3sl (T4)
CYCLE755	C (A)	备份/恢复数据组		M		FB3sl (T4)
CYCLE756	C (A)	激活优化结果		M		FB3sl (T4)
CYCLE757	C (A)	保存优化数据		M		FB3sl (T4)
CYCLE758	C (A)	修改参数值		M		FB3sl (T4)
CYCLE759	C (A)	读取参数值		M		FB3sl (T4)
CYCLE800	C (T)	回转		+		PGAsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
CYCLE801	C (T)	方阵/框架		+		PGAsl
CYCLE802	C (T)	任意位置		+		PGAsl
CYCLE830	C (T)	深孔钻削 2		+		PGAsl
CYCLE832	C (T)	快速设定		+		PGAsl
CYCLE840	C (T)	带弹性卡头的攻丝		+		PGAsl
CYCLE899	C (T)	铣削开口槽		+		PGAsl
CYCLE930	C (T)	切槽		+		PGAsl
CYCLE940	C (T)	退刀槽		+		PGAsl
CYCLE951	C (T)	轮廓车削		+		PGAsl
CYCLE952	C (T)	轮廓槽式车削		+		PGAsl
CYCLE961	C (M)	确定工件内侧或外侧拐角的位置并用作零点偏移		+		BNMsl
CYCLE971	C (M)	标定刀具测头，测量刀长和/或刀具半径（仅用于铣削工艺）		+		BNMsl
CYCLE973	C (M)	在工件的标准面上或在一个标准槽中标定工件测头（仅用于车削工艺）		+		BNMsl
CYCLE974	C (M)	确定工件零点在选中测量轴上的位置，或者采用 1 点测量法确定刀具补偿（仅用于车削工艺）		+		BNMsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
CYCLE976	C (M)	在一个标准环内或通过一个完全处于加工平面内的标准球来标定工件测头，或者在某个平面某个轴方向上标定工件测头		+		BNMsl
CYCLE977	C (M)	确定平面的工件零点位置，或者确定工件宽度或直径		+		BNMsl
CYCLE978	C (M)	测定工件某个边沿在工件坐标系中的位置		+		BNMsl
CYCLE979	C (M)	确定圆弧圆心在平面内的位置和圆弧半径		+		BNMsl
CYCLE982	C (M)	标定刀具测头，测量车刀、钻头和铣刀（仅用于车削工艺）		+		BNMsl
CYCLE994	C (M)	采用 2 点测量法确定工件零点在选中测量轴上的位置（仅用于车削工艺）		+		BNMsl
CYCLE995	C (M)	测定机床主轴的斜度		+		BNMsl
CYCLE996	C (M)	确定与回转轴运动转换相关的转换数据		+		BNMsl
CYCLE997	C (M)	确定球体的球心和直径，测定三个均匀分布的球体的球心		+		BNMsl
CYCLE998	C (M)	确定某个平面和加工平面之间的夹角、平面和工件坐标系中某个边沿的交角		+		BNMsl
CYCLE4071	C (T)	反向点处带进给的纵向磨削		+		PGAsl
CYCLE4072	C (T)	反向点处带进给以及中断信号的纵向磨削		+		PGAsl
CYCLE4073	C (T)	带连续进给的纵向磨削		+		PGAsl
CYCLE4074	C (T)	带连续进给以及中断信号的纵向磨削		+		PGAsl
CYCLE4075	C (T)	反向点处带进给的平面磨削		+		PGAsl

指令	类型 1)	含义	W 2)	TP 3)	SA 4)	说明参见 5)
1) 2) 3) 4) 5) 详细说明参见图例 (页 959)。						
CYCLE4077	C (T)	反向点处带进给以及中断信号的平面磨削		+		PGAsl
CYCLE4078	C (T)	带连续进给的平面磨削		+		PGAsl
CYCLE4079	C (T)	带间歇进给的平面磨削		+		PGAsl

指令 D - F

指令	类型 1)	含义	W 2)	TP 3)	SA 4)	说明参见 5)
1)2)3)4)5) 详细说明参见图例 (页 959)。						
D	A	刀具补偿号		+		PGsl
D0	A	如果编程 D0，则刀具的补偿无效。		+		PGsl
DAC	K	绝对、非模态有效、轴专用的直径编程	s	+		PGsl
DC	K	绝对尺寸参数，用于回转轴直接运行到某个位置	s	+		PGsl
DCI	K	分配数据级 I （= Individual，个人），仅 SINUMERIK 828D!		+		PGAsl
DCM	K	分配数据级 M （= Manufacturer，制造商），仅 SINUMERIK 828D!		+		PGAsl
DCU	K	分配数据级 U （= User，用户），仅 SINUMERIK 828D!		+		PGAsl
DEF	K	变量定义		+		PGAsl
DEFAULT	K	跳转到 CASE 回路		+		PGAsl
DEFINE	K	用于宏指令定义的关键字		+		PGAsl
DELAYFSTOF	P	定义一个停止延迟区的结尾	m	+	-	PGAsl
DELAYFSTON	P	定义一个停止延迟区的开始	m	+	-	PGAsl
DELDL	F	清除附加补偿		+	-	PGAsl
DELDTG	P	剩余行程删除		-	+	FBSYsl
DELETE	P	删除指定的文件。文件名可以用路径和文件标识给出。		+	-	PGAsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1][2][3][4][5] 详细说明参见图例 (页 959)。						
DELMLOWNER	F	删除刀具的所有人刀库刀位		+	-	FBWsl
DELMLRES	F	删除刀库刀位预留		+	-	FBWsl
DELMT	P	删除多刀		+	-	FBWsl
DELOBJ	F	运动链元素、保护区元素、碰撞对元素和转换数据元素的删除		+		PGAsl
DELT	P	删除刀具		+	-	FBWsl
DELTC	P	删除刀架数据组		+	-	FBWsl
DELTOOLENV	F	删除用于说明刀具环境的数据段		+	-	PGAsl
DIACYCOFA	K	轴专用、模态的直径编程：循环中的“关”	m	+		FB1sl (P1)
DIAM90	G	G90：直径编程；G91：半径编程	m	+		PGAsl
DIAM90A	K	G90 和 AC：轴专用、模态的直径编程；G91 和 IC：半径编程	m	+		PGsl
DIAMCHAN	K	MD 轴功能中的所有轴将接收直径编程的通道状态		+		PGsl
DIAMCHANA	K	接收直径编程的通道状态		+		PGsl
DIAMCYCOF	G	通道专用的直径编程：循环中的“关”	m	+		FB1sl (P1)
DIAMOF [6]	G	直径编程：关 参见机床制造商的初始设置	m	+		PGsl
DIAMOFA	K	轴专用的模态直径编程：关 参见机床制造商的初始设置	m	+		PGsl
DIAMON	G	直径编程：ON	m	+		PGsl
DIAMONA	K	轴专用的模态直径编程：开启 参见机床制造商的定义	m	+		PGsl
DIC	K	相对、非模态有效、轴专用的直径编程	s	+		PGsl
DILF	A	返回行程（长度）	m	+		PGsl
DISABLE	P	中断“关”		+	-	PGAsl
DISC	A	过渡圆弧刀具半径补偿加强	m	+		PGsl
DISCL	A	快速进刀终点和加工平面的间距		+		PGsl
DISPLOF	PA	抑制当前的程序段显示		+		PGAsl
DISPLON	PA	恢复当前程序段显示		+		PGAsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1][2][3][4][5] 详细说明参见图例 (页 959)。						
DISPR	A	Repos(再定位)-轨迹差值	s	+		PGAsl
DISR	A	Repos（再定位)距离	s	+		PGAsl
DISRP	A	采用平滑逼近和退回时，退回平面与加工平面之间的距离		+		PGsl
DITE	A	螺纹导出行程	m	+		PGsl
DITS	A	螺纹导入行程	m	+		PGsl
DIV	K	整除		+		PGAsl
DL	A	选择和地点无关的附加刀具补偿（DL、总调整补偿）	m	+		PGAsl
DO	A	同步动作的关键字，在满足条件时动作触发		-	+	FBSYsl
DRFOF	P	取消手轮偏移（DRF）	m	+	-	PGsl
DRIVE	G	与速度相关的轨迹加速度	m	+		PGAsl
DRIVEA	P	激活编程轴的弯曲形加速度特征曲线		+	-	PGAsl
DYNFINISH	G	精加工动态响应	m	+		PGAsl
DYNNORM [6]	G	常规动态响应	m	+		PGAsl
DYNPOS	G	定位模式、攻丝的动态响应	m	+		PGAsl
DYNROUGH	G	粗加工动态响应	m	+		PGAsl
DYNSEMIFIN	G	精加工动态响应	m	+		PGAsl
DZERO	P	将 TO 单元的所有 D 编号标识为无效		+	-	PGAsl
EAUTO	G	通过前面的 3 个点定义前一个样条段	m	+		PGAsl
EGDEF	P	电子齿轮定义		+	-	PGAsl
EGDEL	P	删除跟随轴的耦合定义		+	-	PGAsl
EGOFC	P	持续取消电子齿轮		+	-	PGAsl
EGOFS	P	选择性取消电子齿轮		+	-	PGAsl
EGON	P	启用电子齿轮		+	-	PGAsl
EGONSYN	P	启用电子齿轮		+	-	PGAsl
EGONSYNE	P	启用电子齿轮，按照预定的起动模式		+	-	PGAsl
ELSE	K	当 IF 条件不满足时，程序跳转		+		PGAsl
ENABLE	P	中断“开”		+	-	PGAsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1][2][3][4][5] 详细说明参见图例 (页 959)。						
ENAT [6]	G	自然过渡到下一个运行程序段	m	+		PGAsl
ENDFOR	K	FOR 计数循环的结束行		+		PGAsl
ENDIF	K	IF 跳转的结束行		+		PGAsl
ENDLABEL	K	零件程序通过 REPEAT 重复的结束标记		+		PGAsl, FB1sl (K1)
ENDLOOP	K	无限程序循环 LOOP 结束行		+		PGAsl
ENDPROC	K	带起始行 PROC 的一个程序的结束行		+		
ENDWHILE	K	WHILE-循环的结束行		+		PGAsl
ESRR	P	在驱动中设置驱动集成的 ESR 退回		+		PGAsl
ESRS	P	在驱动中设置驱动集成的 ESR 停止		+		PGAsl
ETAN	G	在样条开始以切线过渡到下一个运行程序段	m	+		PGAsl
EVERY	K	执行同步动作，当条件从 FALSE 过渡到 TRUE 时		-	+	FBSYsl
EX	K	用于冥数运算法则中赋值的关键字		+		PGAsl
EXECSTRING	P	传递一个字符串变量，包含待执行的零件程序行		+	-	PGAsl
EXECTAB	P	执行来自运动表中的元素		+	-	PGAsl
EXECUTE	P	激活程序执行		+	-	PGAsl
EXP	F	指数函数 ex		+	+	PGAsl
EXTCALL	A	执行外部子程序		+	+	PGAsl
EXTCLOSE	P	关闭已打开的用于写的外部设备/文件		+	-	PGAsl
EXTERN	K	申明一个子程序，带参数传递		+		PGAsl
EXTOPEN	P	打开通道的用于写的外部设备/文件		+	-	PGAsl
F	A	进给值 （和 G4 一起，同时在 F 中编程暂停时间）		+	+	PGsl
FA	K	轴向进给率	m	+	+	PGsl
FAD	A	横向进给，用于平滑逼近和离开		+		PGsl
FALSE	K	逻辑常量：假		+	+	PGAsl
FB	A	非模态有效进给率		+		PGsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1][2][3][4][5] 详细说明参见图例 (页 959)。						
FCTDEF	P	定义多项式函数		+	-	PGAsl
FCUB	G	按照立方样条改变进给率	m	+		PGAsl
FD	A	用于手轮叠加的轨迹进给率	s	+		PGsl
FDA	K	用于手轮叠加的轴向进给率	s	+		PGsl
FENDNORM [6]	G	取消拐角减速	m	+		PGAsl
FFWOF [6]	G	取消前馈控制	m	+		PGAsl
FFWON	G	激活前馈控制	m	+		PGAsl
FGREF	K	回转轴时为参考半径；定向轴时为轨迹参考系数（矢量插补）	m	+		PGsl
FGROUP	P	确定轴和轨迹进给率		+	-	PGsl
FI	K	用于存取框架数据的参数：精偏		+		PGAsl
FIFOCTRL	G	缓存控制	m	+		PGAsl
FILEDATE	P	提供最后一次写入文件的日期		+	-	PGAsl
FILEINFO	P	提供 FILEDATE、FILESIZE、FILESTAT 和 FILETIME 的总和信息		+	-	PGAsl
FILESIZE	P	提供当前文件大小		+	-	PGAsl
FILESTAT	P	提供的文件状态，如读取、写入、执行、显示、删除（rwxsd）的权限		+	-	PGAsl
FILETIME	P	提供最后一次写入文件的时间		+	-	PGAsl
FINEA	K	在到达“精准停”时运行结束	m	+		PGAsl
FL	K	同步轴的极限速度	m	+		PGsl
FLIN	G	线性可变进给率	m	+		PGAsl
FMA	K	轴向多个进给率	m	+		PGsl
FNORM [6]	G	标准进给率符合 DIN66025	m	+		PGAsl
FOC	K	非模态生效的转矩/力限制	s	-	+	FBSYsl
FOCOF	K	取消模态扭矩/力限制	m	-	+	FBSYsl
FOCON	K	激活模态扭矩/力限制	m	-	+	FBSYsl
FOR	K	带固定运行次数的计数循环		+		PGAsl
FP	A	固定点：将运行到的固定点的编号	s	+		PGsl
FPO	K	通过一个多项式编程的进给曲线		+		PGAsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1][2][3][4][5] 详细说明参见图例 (页 959)。						
FPR	P	回转轴标记		+	-	PGsl
FPRAOF	P	取消旋转进给率		+	-	PGsl
FPRAON	P	激活旋转进给率		+	-	PGsl
FRAME	K	用于确定坐标系的数据类型		+		PGAsl
FRC	A	用于倒角和倒圆的进给率	s	+		PGsl
FRCM	A	用于倒角和倒圆的模态进给率	m	+		PGsl
FROM	K	一旦满足条件，并且同步动作激活，则执行动作		-	+	FBSYsl
FTOC	P	修改刀具精补		-	+	FBSYsl
FTOCOF [6]	G	取消在线刀具精补	m	+		PGAsl
FTOCON	G	激活在线刀具精补	m	+		PGAsl
FXS	K	激活“运动到固定点停止”	m	+	+	PGsl
FXST	K	“运动到固定点停止”的力矩极限	m	+	+	PGsl
FXSW	K	“运动到固定点停止”的监控窗口		+	+	PGsl
FZ	K	每齿进给量	m	+		PGsl

指令 G - L

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
G0	G	线性插补，带快速移动（快速运行）	m	+		PGsl
G1 [6]	G	线性插补，带进给（直线插补）	m	+		PGsl
G2	G	顺时针圆弧插补	m	+		PGsl
G3	G	逆时针圆弧插补	m	+		PGsl
G4	G	暂停时间，给定时间	s	+		PGsl
G5	G	斜向切入式磨削	s	+		PGAsl
G7	G	斜向切入式磨削时的补偿运动	s	+		PGAsl
G9	G	准停 - 速度减少	s	+		PGsl
G17 [6]	G	选择工作平面 X/Y	m	+		PGsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
G18	G	选择工作平面 Z/X	m	+		PGsl
G19	G	选择工作平面 Y/Z	m	+		PGsl
G25	G	工作范围下限	s	+		PGsl
G26	G	工作范围上限	s	+		PGsl
G33	G	螺纹切削，等螺距	m	+		PGsl
G34	G	螺纹切削,增螺距	m	+		PGsl
G35	G	螺纹切削,减螺距	m	+		PGsl
G40 [6]	G	取消刀具半径补偿	m	+		PGsl
G41	G	刀具半径补偿，轮廓左边	m	+		PGsl
G42	G	刀具半径补偿，轮廓右边	m	+		PGsl
G53	G	抑制当前零点偏移（非模态有效）	s	+		PGsl
G54	G	第 1 个可设定的零点偏移	m	+		PGsl
G55	G	第 2 个可设定零点偏移	m	+		PGsl
G56	G	第 3 个可设定零点偏移	m	+		PGsl
G57	G	第 4 个可设定零点偏移	m	+		PGsl
G58 (840D sl)	G	可编程零点偏移绝对值（粗偏移）	s	+		PGsl
G58 (828D)	G	第 5 个可设定零点偏移	m	+		PGsl
G59 (840D sl)	G	叠加的可编程零点偏移（精偏移）	s	+		PGsl
G59 (828D)	G	第 6 个可设定零点偏移	m	+		PGsl
G60 [6]	G	准停 - 速度减少	m	+		PGsl
G62	G	激活刀具半径补偿（G41、G42）时，内角上的减速度	m	+		PGAsl
G63	G	带弹性卡头的攻丝	s	+		PGsl
G64	G	连续路径运行	m	+		PGsl
G70	G	英制尺寸，用于几何数据（长度）	m	+	+	PGsl
G71 [6]	G	公制尺寸，用于几何数据（长度）	m	+	+	PGsl
G74	G	回参考点运行	s	+		PGsl
G75	G	逼近固定点	s	+		PGsl
G90 [6]	G	绝对尺寸	m/s	+		PGsl
G91	G	增量尺寸	m/s	+		PGsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
G93	G	时间倒数进给率 rpm	m	+		PGsl
G94 [6]	G	直线进给率 F，单位：毫米/分钟、英寸/分钟、度/分钟	m	+		PGsl
G95	G	旋转进给率 F，单位毫米/转、英寸/转	m	+		PGsl
G96	G	旋转进给（同 G95 时）及恒定切削速度	m	+		PGsl
G97	G	旋转进给和恒定主轴转速（恒定切削速度“关”）	m	+		PGsl
G110	G	极点编程，相对于最后编程的给定位置	s	+		PGsl
G111	G	极点编程，相对于当前工件坐标系的零点	s	+		PGsl
G112	G	极点编程，相对于最后有效的极点	s	+		PGsl
G140 [6]	G	由 G41/G42 确定的逼近方向 WAB	m	+		PGsl
G141	G	逼近方向 WAB，轮廓左边	m	+		PGsl
G142	G	逼近方向 WAB，轮廓右边	m	+		PGsl
G143	G	逼近方向 WAB，切线相关	m	+		PGsl
G147	G	以直线平滑逼近	s	+		PGsl
G148	G	以直线平滑返回	s	+		PGsl
G153	G	取消当前框架，包括基准框架	s	+		PGsl
G247	G	沿四分圆平滑逼近	s	+		PGsl
G248	G	沿四分圆平滑返回	s	+		PGsl
G290 [6]	G	转换到 SINUMERIK 模式 ON	m	+		FBWsl
G291	G	转换到 ISO2/3 模式 ON	m	+		FBWsl
G331	G	不带弹性卡头的螺纹切削，正向螺距，右旋螺纹	m	+		PGsl
G332	G	不带弹性卡头的螺纹切削，负向螺距，左旋螺纹	m	+		PGsl
G335	G	顺时针球螺纹车削	m	+		PGsl
G336	G	逆时针球螺纹车削	m	+		PGsl
G340 [6]	G	空间逼近程序段（深度和平面上相等（螺旋线））	m	+		PGsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
G341	G	首先在垂直轴上进给(z)，然后在平面中运动	m	+		PGsl
G347	G	以半圆平滑逼近	s	+		PGsl
G348	G	以半圆平滑返回	s	+		PGsl
G450 [6]	G	过渡圆弧	m	+		PGsl
G451	G	等距离交点	m	+		PGsl
G460 [6]	G	启用轮廓碰撞监控，用于逼近程序段和退回程序段	m	+		PGsl
G461	G	在 TRC 程序段中插入一个圆弧	m	+		PGsl
G462	G	在 TRC 程序段中插入一条直线	m	+		PGsl
G500 [6]	G	取消所有可设定的框架，基本框架激活	m	+		PGsl
G505 ... G599	G	5. ... 99.可设定的零点偏移	m	+		PGsl
G601 [6]	G	在精准停时切换程序段	m	+		PGsl
G602	G	在粗准停时切换程序段	m	+		PGsl
G603	G	在 IPO 程序段结束处切换程序段	m	+		PGsl
G621	G	所有拐角处都减速	m	+		PGAsl
G641	G	连续路径运行，根据位移标准进行平滑（= 可编程的平滑间距）	m	+		PGsl
G642	G	连续路径运行，按照定义的公差开展平滑	m	+		PGsl
G643	G	连续路径运行，按照定义的公差开展平滑（程序段内部）	m	+		PGsl
G644	G	连续路径运行，采用允许的最大动态响应开展平滑	m	+		PGsl
G645	G	连续路径运行，按照定义的公差对拐角和程序段切线过渡开展平滑	m	+		PGsl
G700	G	英制尺寸，用于几何数据和工艺数据（长度、进给率）	m	+	+	PGsl
G710 [6]	G	公制尺寸，用于几何数据和工艺数据（长度、进给率）	m	+	+	PGsl
G810 [6], ..., G819	G	给 OEM 用户保留的 G 代码组		+		PGAsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
G820 [6], ..., G829	G	给 OEM 用户保留的 G 代码组		+		PGAsl
G931	G	运行时间规定的进给率，取消恒定轨迹速度	m	+		
G942	G	取消线性进给、恒定切削速度或者主轴转速	m	+		
G952	G	取消旋转进给、恒定切削速度或者主轴转速	m	+		
G961	G	线性进给（同 G94 时）及恒定切削速度	m	+		PGsl
G962	G	线性进给、旋转进给和恒定切削速度	m	+		PGsl
G971	G	线性进给和恒定主轴转速（恒定切削速度“关”）	m	+		PGsl
G972	G	线性进给或恒定主轴转速（恒定切削速度“关”）	m	+		PGsl
G973	G	旋转进给，无主轴转速限制以及恒定主轴转速（ISO 模式下无 LIMS 的 G97）	m	+		PGsl
GEOAX	P	给几何轴 1－3 分配新的通道轴		+	-	PGAsl
GET	P	更换通道间已经使能的轴		+	+	PGAsl
GETACTT	F	从具有相同名称的刀具组中获取有效的刀具。		+	-	FBWsl
GETACTTD	F	确定绝对 D 号所属的 T 号		+	-	PGAsl
GETD	P	直接更换通道间的轴		+	-	PGAsl
GETDNO	F	提供某个刀具(T)某个刀沿(CE)的 D 号		+	-	PGAsl
GETEXET	P	读取换入的 T 号		+	-	FBWsl
GETFREELOC	P	为指定的刀具查找刀库中的空位		+	-	FBWsl
GETSELT	P	提供一个预选的 T 号		+	-	FBWsl
GETT	F	给刀具名确定 T 号		+	-	FBWsl
GETTCOR	F	读取刀具长度或刀具长度分量		+	-	PGAsl
GETTENV	F	读取 T 号、D 号和 DL 号		+	-	PGAsl
GETVARAP	F	读取对系统变量/用户变量的存取权限		+	-	PGAsl
GETVARDFT	F	读取系统变量/用户变量的缺省值		+	-	PGAsl
GETVARLIM	F	读取系统变量/用户变量的限值		+	-	PGAsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
GETVARPHU	F	读取系统变量/用户变量的物理单位		+	-	PGAsl
GETVARTYP	F	读取系统变量/用户变量的数据类型		+	-	PGAsl
GFRAME0 ... GFRAME100	G	激活通道中数据管理的磨削框架 <n>	m	+		PGsl
GOTO	K	跳转指令首先向前，然后向后（方向首先向程序结束处，然后向程序开始）		+		PGAsl
GOTOB	K	跳转指令，向后（程序起始方向）		+		PGAsl
GOTOC	K	和 GOTO 一样，报警 14080 “没有找到跳转目标”会被抑制		+		PGAsl
GOTOF	K	跳转指令，向前（程序结束方向）		+		PGAsl
GOTOS	K	跳回至程序开始处		+		PGAsl
GP	K	位置属性间接编程的关键字		+		PGAsl
GROUP_ ADDEND	C (T)	附加运行结束		+		PGAsl
GROUP_BEGIN	C (T)	程序块开始		+		PGAsl
GROUP_END	C (T)	程序块结束		+		PGAsl
GWPSOF	P	撤销选择恒定砂轮圆周速度(GWPS)	s	+	-	PGsl
GWPSON	P	选择恒定砂轮圆周速度(GWPS)	s	+	-	PGsl
H...	A	输出至 PLC 的辅助功能		+	+	PGsl/FB1sl (H2)
HOLES1	C (T)	成排孔		+		PGAsl
HOLES2	C (T)	圆弧孔		+		PGAsl
I	A	插补参数	s	+		PGsl
I1	A	中间点坐标	s	+		PGsl
IC	K	增量尺寸	s	+		PGsl
ICYCOF	P	根据 ICYCOF，一个工艺循环的所有程序段会在一个插补周期中执行		+	+	FBSYsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
ICYCON	P	根据 ICYCON，一个工艺循环的每个程序段都会在一个单独的插补周期中执行		+	+	FBSYsl
ID	K	表示模态同步动作	m	-	+	FBSYsl
IDS	K	表示模态静态同步动作		-	+	FBSYsl
IF	K	在零件程序/工艺循环中引入一个有条件的跳转		+	+	PGAsl
INDEX	F	确定输入字符串中一个字符的索引		+	-	PGAsl
INICF	K	NEWCONF 时初始化变量		+		PGAsl
INIPO	K	上电时初始化变量		+		PGAsl
INIRE	K	复位时初始化变量		+		PGAsl
INIT	P	选择某个 NC 程序，然后在某个通道中执行该程序		+	-	PGAsl
INITIAL		生成所有区域的 INI 文件		+		PGAsl
INT	K	数据类型：带正负号的整数值		+		PGAsl
INTERSEC	F	计算两个轮廓单元之间的交点		+	-	PGAsl
INVCCW	G	逆时针方向渐开线运行	m	+		PGsl
INVCW	G	顺时针方向渐开线运行	m	+		PGsl
INVFRAME	F	从一个框架计算出逆转框架		+	-	FB1sl (K2)
IP	K	可变的插补参数		+		PGAsl
IPOBRKA	P	运动条件，自制动斜坡开始点	m	+	+	
IPOENDA	K	在到达“插补停止”时运行结束	m	+		PGAsl
IPTRLOCK	P	不支持搜索功能的程序部分开始，中断指针位于下一个机床功能程序段上。	m	+	-	PGAsl
IPTRUNLOCK	P	不支持搜索功能的程序部分结束，中断指针位于中断时处理的当前程序段上。	m	+	-	PGAsl
IR	A	球螺纹车削时的圆心坐标（X 方向）		+		PGsl
ISAXIS	F	检查被设为参数的几何轴是否为 1		+	-	PGAsl
ISD	A	插入深度	m	+		PGAsl
ISFILE	F	检查在 NC 用户存储器中是否有一个文件		+	-	PGAsl
ISNUMBER	F	检查是否可以把输入字符串转换成数字		+	-	PGAsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
ISOCALL	K	间接调用某个以 ISO 语言编程的程序		+		PGAsl
ISVAR	F	检查传送参数是否包含一个 NC 知晓的变量		+	-	PGAsl
J	A	插补参数	s	+		PGsl
J1	A	中间点坐标	s	+		PGsl
JERKA	P	激活借助机床数据设定的、编程轴的加速度属性		+	-	
JERKLIM	K	最大轴向急动的递增或递减	m	+		PGAsl
JERKLIMA	K	最大轴向急动的递增或递减	m	+	+	PGAsl
JR	A	球螺纹车削时的圆心坐标（Y 方向）		+		PGsl
K	A	插补参数	s	+		PGsl
K1	A	中间点坐标	s	+		PGsl
KONT	G	在刀具补偿时绕行轮廓	m	+		PGsl
KONTC	G	以曲率恒定的多项式逼近/后退	m	+		PGsl
KONTT	G	以切线恒定的多项式逼近/后退	m	+		PGsl
KR	A	球螺纹车削时的圆心坐标（Z 方向）		+		PGsl
L	A	子程序号	s	+	+	PGAsl
LEAD	A	导角 1. 刀具定向 2. 定向多项式	m	+		PGAsl
LEADOF	P	取消轴引导值耦合		+	+	PGAsl
LEADON	P	激活轴引导值耦合		+	+	PGAsl
LENTOAX	F	提供生效刀具 L1、L2 和 L3 长度赋值给纵坐标、横坐标和垂直坐标的信息。		+	-	PGAsl
LFOF [6]	G	取消“螺纹切削时快速退刀”	m	+		PGsl
LFON	G	激活“螺纹切削时快速退刀”	m	+		PGsl
LFPOS	G	使由 POLFMASK 或 POLFMLIN 指明的轴退回到由 POLF 编写的绝对位置上	m	+		PGsl
LFTXT [6]	G	快速退刀时的退回平面由轨迹切线和当前的刀具方向确定	m	+		PGsl

指令	类型 1)	含义	W 2)	TP 3)	SA 4)	说明参见 5)
1) 2) 3) 4) 5) 详细说明参见图例 (页 959)。						
LFWP	G	快速退刀时的退回平面由当前的加工平面确定(G17/G18/G19)	m	+		PGsl
LIFTFAST	K	快速退刀		+		PGsl
LIMS	K	转速限制 用于 G96/G961 和 G97	m	+		PGsl
LLI	K	变量的下限值		+		PGAsl
LN	F	自然对数		+	+	PGAsl
LOCK	P	使用 ID 锁止同步动作 (停止工艺循环)		-	+	FBSYsl
LONGHOLE	C (T)	长孔形		+		PGAsl
LOOP	K	引入一个无限循环		+		PGAsl

指令 M - R

指令	类型 1)	含义	W 2)	TP 3)	SA 4)	说明参见 5)
1) 2) 3) 4) 5) 详细说明参见图例 (页 959)。						
M0		程序停止		+	+	PGsl
M1		可选停止		+	+	PGsl
M2		主程序结束（同 M30）		+	+	PGsl
M3		主轴顺时针旋转		+	+	PGsl
M4		主轴逆时针旋转		+	+	PGsl
M5		主轴停止		+	+	PGsl
M6		换刀		+	+	PGsl
M17		子程序程序结束		+	+	PGsl
M19		主轴运动到 SD43240 指定的位置		+	+	PGsl
M30		主程序结束（同 M2）		+	+	PGsl
M40		自动齿轮换档		+	+	PGsl
M41 ... M45		齿轮级 1 ... 5		+	+	PGsl
M70		过渡到进给轴运行		+	+	PGsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
MASLDEF	P	定义主/从轴连接		+	+	PGAsl
MASLDEL	P	分离主/从轴连接，删除连接定义		+	+	PGAsl
MASLOF	P	关闭一个临时耦合		+	+	PGAsl
MASLOFS	P	取消临时的耦合，自动停止从动轴		+	+	PGAsl
MASLON	P	激活一个临时耦合		+	+	PGAsl
MATCH	F	在字符串中查找一个字符串		+	-	PGAsl
MAXVAL	F	两个变量中的较大值（算术函数）		+	+	PGAsl
MCALL	K	模态子程序调用		+		PGAsl
MEAC	K	轴持续测量，没有剩余行程删除	s	+	+	PGAsl
MEAFRAME	F	从测量点中计算框架		+	-	PGAsl
MEAS	A	带剩余行程删除的测量	s	+		PGAsl
MEASA	K	轴测量，带剩余行程删除	s	+	+	PGAsl
MEASURE	F	工件和刀具测量的计算方法		+	-	FB1sl (M5)
MEAW	A	测量，不带剩余行程删除	s	+		PGAsl
MEAWA	K	轴测量，没有剩余行程删除	s	+	+	PGAsl
MI	K	存取框架数据：镜像		+		PGAsl
MINDEX	F	确定输入字符串中一个字符的索引		+	-	PGAsl
MINVAL	F	两个变量中的较小值（算数函数）		+	+	PGAsl
MIRROR	G	可编程镜像	s	+		PGAsl
MMC	P	在 HMI 上从零件程序中调用互动式对话窗口		+	-	PGAsl
MOD	K	取模除法		+		PGAsl
MODAXVAL	F	得到模数回转轴的取模位置		+	-	PGAsl
MOV	K	启动定位轴		-	+	FBSYsl
MOVT	A	设定刀具方向上的运行终点				FB1(K2)
MSG	P	可编程的信息	m	+	-	PGsl
MVTOOL	P	用于移动刀具的语言指令		+	-	FBWsl
N	A	NC 分程序段号		+		PGsl
NAMETOINT	F	确定系统变量索引		+		PGAsl
NC	K	规定数据有效区。		+		PGAsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
NEWCONF	P	采用已经修改的机床数据，相当于激活机床数据		+	-	PGAsl
NEWMT	F	创建新多刀		+	-	FBWsl
NEWT	F	创建新的刀具		+	-	FBWsl
NORM [6]	G	在刀具补偿时，在起始点和终点处的标准设置	m	+		PGsl
NOT	K	逻辑“非”		+		PGAsl
NPROT	P	机床专用的保护区“激活/取消”		+	-	PGAsl
NPROTDEF	P	定义机床专用的保护区		+	-	PGAsl
NUMBER	F	转换输入字符串为数字		+	-	PGAsl
OEMIPO1	G	OEM 插补 1	m	+		PGAsl
OEMIPO2	G	OEM 插补 2	m	+		PGAsl
OF	K	CASE 回路中的关键字		+		PGAsl
OFFN	A	编程轮廓的加工余量	m	+		PGsl
OMA1	A	OEM 地址 1	m	+		PGAsl
OMA2	A	OEM 地址 2	m	+		PGAsl
OMA3	A	OEM 地址 3	m	+		PGAsl
OMA4	A	OEM 地址 4	m	+		PGAsl
OMA5	A	OEM 地址 5	m	+		PGAsl
OR	K	逻辑运算符，“或”连接		+		PGAsl
ORIAXES	G	线性插补加工轴或者方向轴	m	+		PGAsl
ORIAXPOS	G	虚拟的方向轴与回转轴位置的方向角	m	+		PGAsl
ORIC [6]	G	外拐角定向变化叠加在将要插入的圆弧程序段上	m	+		PGAsl
ORICONCCW	G	逆时针方向圆弧表面插补	m	+		PGAsl/FB3sl (F3)
ORICONCW	G	顺时针方向圆弧表面插补	m	+		PGAsl/FB3sl (F4)
ORICONIO	G	圆弧表面插补，指定了一个中间方向	m	+		PGAsl/FB3sl (F4)
ORICONTO	G	在切向过渡中的某个圆侧面上的插补 (最终定向说明)	m	+		PGAsl/FB3sl (F5)
ORICURVE	G	定向插补，其中指定了刀具的两个接触点的运动	m	+		PGAsl/FB3sl (F6)

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
ORID	G	在圆弧程序段之前执行定向变化	m	+		PGAsl
ORIEULER [6]	G	欧拉角方向角	m	+		PGAsl
ORIMKS	G	在机床坐标系中的刀具定向	m	+		PGAsl
ORIPATH	G	刀具定向参照于轨迹	m	+		PGAsl
ORIPATHS	G	刀具定向参照这个轨迹，拐点在方向变化中被平滑	m	+		PGAsl
ORIPLANE	G	平面插补 （相应于 ORIVECT） 大半径圆插补	m	+		PGAsl
ORIRESET	P	刀具定向的基本设置，最多带 3 个定向轴		+	-	PGAsl
ORIROTA [6]	G	规定的绝对旋转方向的旋转角度	m	+		PGAsl
ORIROTC	G	轨迹切线的切向旋转矢量	m	+		PGAsl
ORIROTR	G	相对于平面在起始方向和结束方向之间的旋转角度	m	+		PGAsl
ORIROTT	G	相对于方向矢量改变的旋转角度	m	+		PGAsl
ORIRPY	G	通过 RPY 角的定向角（XYZ）	m	+		PGAsl
ORIRPY2	G	通过 RPY 角的定向角（ZYX）	m	+		PGAsl
ORIS	A	定向改变	m	+		PGAsl
ORISOF [6]	G	取消定向曲线的平滑	m	+		PGAsl
ORISOLH	F	计算定向		+		PGAsl
ORISON	G	启用定向曲线的平滑	m	+		PGAsl
ORIVECT [6]	G	大圆弧插补（和 ORIPLANE 一致）	m	+		PGAsl
ORIVIRT1	G	通过虚拟定向轴的定向角（定义 1）	m	+		PGAsl
ORIVIRT2	G	通过虚拟定向轴的定向角（定义 1）	m	+		PGAsl
ORIWKS [6]	G	在工件坐标系中的刀具定向	m	+		PGAsl
OS	K	激活/取消摆动		+		PGAsl
OSB	K	摆动：起点	m	+		FB1sl (P5)
OSC	G	恒定平滑刀具定向	m	+		PGAsl
OSCILL	K	Axis:1-3 进给轴	m	+		PGAsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
OSCTRL	K	选件摆动	m	+		PGAsl
OSD	G	通过设定数据指定平滑长度来平滑刀具定向	m	+		PGAsl
OSE	K	摆动结束位置	m	+		PGAsl
OSNSC	K	摆动：无火花磨削次数	m	+		PGAsl
OSOF [6]	G	取消刀具定向平滑	m	+		PGAsl
OSP1	K	摆动：左侧换向点	m	+		PGAsl
OSP2	K	右侧换向点的摆动	m	+		PGAsl
OSS	G	在程序段结束处平滑刀具方向	m	+		PGAsl
OSSE	G	程序段开始和结束的刀具平滑定向	m	+		PGAsl
OST	G	通过用 SD（编程最大差）预设角度公差（单位：度）来精磨刀具定向定向运行）	m	+		PGAsl
OST1	K	摆动：在右换向点停止	m	+		PGAsl
OST2	K	摆动：在右换向点停止	m	+		PGAsl
OTOL	A	定向公差，用于压缩器功能、定向平滑和精磨方式	m	+		PGAsl
OVR	K	转速补偿	m	+		PGAsl
OVRA	K	轴的转速补偿	m	+	+	PGAsl
OVRRAP	K	快进补偿	m	+		PGAsl
P	A	零件程序运行次数		+		PGAsl
PAROT	G	工件坐标系和工件对准	m	+		PGsl
PAROTOF [6]	G	取消和工件相关的框架旋转	m	+		PGsl
PCALL	K	调用子程序，带绝对的路径参数和参数传递		+		PGAsl
PDELAYOF	G	取消冲压延迟	m	+		PGAsl
PDELAYON [6]	G	激活冲压延迟	m	+		PGAsl
PHI	K	绕圆锥方向轴定向的旋转角		+		PGAsl
PHU	K	变量的物理单位		+		PGAsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
PL	A	1. B 样条：节点间距 2. 多项式插补：多项式插补中参数间隔的长度	s	+		PGAsl
PM	K	每分钟		+		PGsl
PO	K	多项式插补的多项式系数	s	+		PGAsl
POCKET3	C (T)	铣削矩形腔		+		PGAsl
POCKET4	C (T)	铣削圆形腔		+		PGAsl
POLF	K	返回位置 LIFTFAST	m	+		PGsl/PGAsl
POLFA	P	用 $AA_ESR_TRIGGER 启动单个轴的退回位置	m	+	+	PGsl
POLFMASK	P	激活轴间无关联的退回运动	m	+	-	PGsl
POLFMLIN	P	激活轴间有线性关联的退回运动	m	+	-	PGsl
POLY	G	多项式插补	m	+		PGAsl
POLYPATH	P	多项式插补可选择，用于轴组 AXIS 或者 VECT	m	+	-	PGAsl
PON	G	激活冲压	m	+		PGAsl
PONS	G	在插补周期中激活冲压	m	+		PGAsl
POS	K	轴定位		+	+	PGsl
POSA	K	轴定位，超出程序段界限		+	+	PGsl
POSM	P	刀库定位		+	-	FBWsl
POSMT	P	在刀架上将多刀定位至刀位号		+	-	FBWsl
POSP	K	轴分段定位（摆动）		+		PGsl
POSRANGE	F	确定，某个轴当前插补的给定位置是否在规定的参考位置的窗口中。		+	+	FBSYsl
POT	F	平方 （算术函数）		+	+	PGAsl
PR	K	每转		+		PGsl
PREPRO	PA	表示经过预处理的子程序		+		PGAsl
PRESETON	P	实际值设定，有回参考点状态损失		+	+	PGAsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
PRESETONS	P	实际值设定，无回参考点状态损失		+	+	PGAsl
PRIO	K	在处理中断时设置优先级的关键字		+		PGAsl
PRLOC	K	复位时变量的初始化，仅在本地修改后		+		PGAsl
PROC	K	一个程序的第一个指令		+		PGAsl
PROTA	P	要求重新计算碰撞模型		+		PGAsl
PROTD	F	计算两个保护区之间的间距		+		PGAsl
PROTS	P	保护区状态设置		+		PGAsl
PSI	K	圆锥张角		+		PGAsl
PTP	G	点对点运行（PTP 运行）	m	+		PGAsl
PTPG0	G	在 G0 时为点对点运动，其余为 CP 轨迹运行	m	+		PGAsl
PTPWOC	G	无补偿运行的点对点运行，由定向变化导致	m	+		PGAsl
PUNCHACC	P	步冲时的位移控制式加速度		+	-	PGAsl
PUTFTOC	P	用于并行修整的刀具精补		+	-	PGAsl
PUTFTOCF	P	根据 FCTDEF 定义的功能、用于并行修整的刀具精补		+	-	PGAsl
PW	A	B 样条，点加权	s	+		PGAsl
QU	K	快速 辅助功能输出		+		PGsl
R...	A	计算参数也作为可设定的地址标识符，并带有数字扩展		+		PGAsl
RAC	K	绝对、非模态有效、轴专用的半径编程	s	+		PGsl
RDISABLE	P	读取禁止		-	+	FBSYsl
READ	P	在所说明的文件中读入一个或者多个行，并且在数组中存放所读入的信息。		+	-	PGAsl
REAL	K	数据类型：带有正负号的浮点变量（实数）		+		PGAsl
REDEF	K	系统变量，用户变量和 NC 语言指令的重新定义		+		PGAsl
RELEASE	P	使能机床轴，用于轴交换		+	+	PGAsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
REP	K	关键字，用同一个值初始化一个数组的所有元素		+		PGAsl
REPEAT	K	重复一个程序循环		+		PGAsl
REPEATB	K	重复一个程序行		+		PGAsl
REPOSA	G	所有轴再次逼近轮廓	s	+		PGAsl
REPOSH	G	以半圆再次逼近轮廓	s	+		PGAsl
REPOSHA	G	所有轴再次逼近轮廓，几何轴以半圆逼近	s	+		PGAsl
REPOSL	G	沿直线再次逼近轮廓	s	+		PGAsl
REPOSQ	G	以四分之一圆弧再次逼近轮廓	s	+		PGAsl
REPOSQA	G	所有轴再次沿直线逼近轮廓，几何轴沿四分圆逼近	s	+		PGAsl
RESETMON	P	用于激活设定值的语言指令		+	-	FBWsl
RET	P	子程序结束		+	+	PGAsl
RETB	P	子程序结束		+	+	PGAsl
RIC	K	相对、非模态有效、轴专用的半径编程	s	+		PGsl
RINDEX	F	确定输入字符串中一个字符的索引		+	-	PGAsl
RMB	G	再定位到程序段起点	m	+		PGAsl
RMBBL	G	再定位到程序段起点	s	+		PGAsl
RME	G	再定位到程序段终点	m	+		PGAsl
RMEBL	G	再定位到程序段终点	s	+		PGAsl
RMI [6]	G	再定位到中断点	m	+		PGAsl
RMIBL [6]	G	再定位到中断点	s	+		PGAsl
RMN	G	再定位到下一个轨迹点	m	+		PGAsl
RMNBL	G	再定位到下一个轨迹点	s	+		PGAsl
RND	A	轮廓角倒圆	s	+		PGsl
RNDM	A	模态倒圆	m	+		PGsl
ROT	G	可编程旋转	s	+		PGsl
ROTS	G	可编程的框架旋转，带立体角	s	+		PGsl
ROUND	F	小数位四舍五入		+	+	PGAsl

指令	类型 1)	含义	W 2)	TP 3)	SA 4)	说明参见 5)
1) 2) 3) 4) 5) 详细说明参见图例 (页 959)。						
ROUNDUP	F	向上取整输入值		+	+	PGAsl
RP	A	极半径	m/s	+		PGsl
RPL	A	平面中旋转	s	+		PGsl
RT	K	用于存取框架数据的参数：旋转		+		PGAsl
RTLIOF	G	G0，不带直线插补（单轴插补）	m	+		PGsl
RTLION 6)	G	带直线插补的 G0	m	+		PGsl

指令 S - Z

指令	类型 1)	含义	W 2)	TP 3)	SA 4)	说明参见 5)
1) 2) 3) 4) 5) 详细说明参见图例 (页 959)。						
S	A	主轴转速或 （G4，G96/G961 中含义不同）	m/s	+	+	PGsl
SAVE	PA	在子程序调用时保护信息		+		PGAsl
SBLOF	P	抑制单程序段		+	-	PGAsl
SBLON	P	取消单程序段抑制		+	-	PGAsl
SC	K	用于存取框架数据的参数：比例		+		PGAsl
SCALE	G	可编程缩放	s	+		PGsl
SCC	K	选择端面轴进行 G96/G961/G962 设置轴名称可以为几何轴、通道轴或者加工轴。		+		PGsl
SCPARA	K	编程伺服参数段		+	+	PGAsl
SD	A	样条度数	s	+		PGAsl
SET	K	关键字，用列表值初始化一个数组的所有元素		+		PGAsl
SETAL	P	设置报警		+	+	PGAsl
SETDNO	F	指定某个刀具(T)某个刀沿(CE)的 D 号		+	-	PGAsl
SETINT	K	确定在出现一个 NC 输入时应该激活哪一个中断程序		+		PGAsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
SETM	P	设置自有通道中的标记位		+	+	PGAsl
SETMS	P	机床数据中的主主轴复位		+	-	PGsl
SETMS (n)	P	主轴 n 应该作为主主轴		+		PGsl
SETMTH	P	设置主刀架编号		+	-	FBWsl
SETPIECE	P	考虑所有刀具的数量，它们将分配到主轴		+	-	FBWsl
SETTA	P	激活磨损组中的刀具		+	-	FBWsl
SETTCOR	F	考虑到所有标准条件，修改刀具分量		+	-	PGAsl
SETTIA	P	取消磨损组中的刀具		+	-	FBWsl
SF	A	用于螺纹切削的起始点偏移	m	+		PGsl
SIN	F	正弦（三角函数）		+	+	PGAsl
SIRELAY	F	激活由 SIRELIN、SIRELOUT 和 SIRELTIM 设定的安全功能		-	+	FBSIsl
SIRELIN	P	初始化功能块的输入值		+	-	FBSIsl
SIRELOUT	P	初始化功能块的输出值		+	-	FBSIsl
SIRELTIME	P	初始化功能块的计时器		+	-	FBSIsl
SLOT1	C (T)	纵向槽		+		PGAsl
SLOT2	C (T)	圆弧槽		+		PGAsl
SOFT	G	限制急动的轨迹加速度	m	+		PGAsl
SOFTA	P	激活编程的轴上、限制急动的轴加速度		+	-	PGAsl
SON	G	激活步冲	m	+		PGAsl
SONS	G	在插补周期内激活步冲	m	+		PGAsl
SPATH [6]	G	FGROUP 轴的轨迹基准为弧长。	m	+		PGAsl
SPCOF	P	主主轴或者主轴(n)从位置控制转换到转速控制	m	+	-	PGsl
SPCON	P	主主轴或者主轴从转速控制转换到位置控制	m	+	-	PGAsl
SPI	F	把主轴编号转换为一个轴名称		+	-	PGAsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
SPIF1 [6]	G	用于冲压/步冲的高速 NC 输入/输出字节 1	m	+		FB2sl (N4)
SPIF2	G	用于冲压/步冲的高速 NC 输入/输出字节 2	m	+		FB2sl (N4)
SPLINEPATH	P	确定样条连接		+	-	PGAsl
SPN	A	每个程序段中分段行程的数量	s	+		PGAsl
SPOF [6]	G	关闭分段行程， 关闭冲压、步冲	m	+		PGAsl
SPOS	K	主轴位置	m	+	+	PGsl
SPOSA	K	主轴位置超过程序段界限	m	+		PGsl
SPP	A	分段行程长度	m	+		PGAsl
SPRINT	F	返回有格式的输入字符串		+		PGAsl
SQRT	F	平方根 (算术函数) (square root)		+	+	PGAsl
SR	A	用于同步动作的摆动退回行程	s	+		PGsl
SRA	K	外部输入上，用于同步动作的轴摆动退回行程	m	+		PGsl
ST	A	用于同步动作的摆动无火花磨削时间	s	+		PGsl
STA	K	用于同步动作的、轴向摆动无火花磨削时间	m	+		PGsl
START	P	从运行的程序中，在几个通道中同时启动所选择的程序		+	-	PGAsl
STARTFIFO [6]	G	执行加工；并同时载满缓存	m	+		PGAsl
STAT		铰接位置	s	+		PGAsl
STOLF	K	G0 公差系数	m	+		PGAsl
STOPFIFO	G	停止执行，载满缓存，直至识别出 STARTFIFO、缓存已满或者程序结束	m	+		PGAsl
STOPRE	P	预处理停止，直到所有预处理的程序段完成主运行		+	-	PGAsl
STOPREOF	P	取消预处理停止		-	+	FBSYsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
STRING	K	数据类型：字符串		+		PGAsl
STRINGIS	F	检查现有的 NC 语言范围，检查专用于该命令所属的 NC 循环名称、用户变量、宏和标签名称是否存在、有效、已定义或激活。		+	-	PGAsl
STRLEN	F	确定一个字符串的长度		+	-	PGAsl
SUBSTR	F	确定输入字符串中一个字符的索引		+	-	PGAsl
SUPA	G	取消当前零点偏移，包括编程的偏移，系统框架，手轮偏移（DRF），外部零点偏移和叠加运动	s	+		PGsl
SVC	K	刀具切削速度	m	+		PGsl
SYNFCT	P	计算一个多项式，取决于运动同步动作中的一个条件		-	+	FBSYsl
SYNR	K	在执行时同步读取变量		+		PGAsl
SYNRW	K	在执行时同步读取和写入变量		+		PGAsl
SYNW	K	在执行时同步写入变量		+		PGAsl
T	A	调用刀具 （只有通过机床数据才能加以改变；否则需要使用 M6 指令）		+		PGsl
TAN	F	正切（三角函数）		+	+	PGAsl
TANG	P	切向控制：定义耦合		+	-	PGAsl
TANGDEL	P	切向控制：删除同步		+	-	PGAsl
TANGOF	P	切向控制：撤销同步		+	-	PGAsl
TANGON	P	切向控制：激活同步		+	-	PGAsl
TCA (828D: _TCA)	P	和刀具状态无关的刀具选择/刀具切换		+	-	FBWsl
TCARR	A	指定刀架，编号“m”		+		PGAsl
TCI	P	将刀具从周转箱换入刀库		+	-	FBWsl
TCOABS[6]	G	从当前刀具定向中确定刀具长度分量	m	+		PGAsl
TCOFR	G	从当前框架的方向确定刀具长度分量	m	+		PGAsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
TCOFRX	G	选择 X 方向的刀具、刀具点，以确定有效框架的刀具定向	m	+		PGAsl
TCOFRY	G	选择 Y 方向的刀具、刀具点，以确定有效框架的刀具定向	m	+		PGAsl
TCOFRZ	G	选择 Z 方向的刀具、刀具点，以确定有效框架的刀具定向	m	+		PGAsl
THETA	A	旋转角度	s	+		PGAsl
TILT	A	侧向角	m	+		PGAsl
TLIFT	P	切向控制：激活中间程序段的生成		+	-	PGAsl
TML	P	通过刀库刀位号选择刀具		+	-	FBWsl
TMOF	P	撤销选择刀具监控		+	-	PGAsl
TMON	P	选择刀具监控		+	-	PGAsl
TO	K	表示 FOR 计数循环中的终点值		+		PGAsl
TOFF	K	刀具长度分量方向上的刀具长度偏移，它和索引中指定的几何轴同时生效	m	+		PGsl
TOFFL	K	刀具长度分量 L1、L2 或 L3 方向上的刀具长度偏移	m	+		PGsl
TOFFOF	P	复位在线刀具长度补偿		+	-	PGAsl
TOFFON	P	激活在线刀具长度补偿		+	-	PGAsl
TOFFR	A	刀具半径偏移	m	+		PGsl
TOFRAME	G	WCS 的 Z 轴通过框架旋转和刀具方向平行	m	+		PGsl
TOFRAMEX	G	WCS 的 X 轴通过框架旋转和刀具方向平行	m	+		PGsl
TOFRAMEY	G	WCS 的 Y 轴通过框架旋转和刀具方向平行	m	+		PGsl
TOFRAMEZ	G	同 TOFRAME	m	+		PGsl
TOLOWER	F	将一个字符串的字母转换成小写字母		+	-	PGAsl
TOOLENV	F	保存所有当前状态，这些状态对于分析存储器中保存的刀具数据非常重要		+	-	PGAsl
TOOLGNT	F	获取一个刀具组的刀具数量		+	-	FBWsl

指令	类型 1)	含义	W 2)	TP 3)	SA 4)	说明参见 5)
1) 2) 3) 4) 5) 详细说明参见图例 (页 959)。						
TOOLGT	F	从刀具组获取刀具 T 号		+	-	FBWsl
TOROT	G	WCS 的 Z 轴通过框架旋转和刀具方向平行	m	+		PGsl
TOROTOF 6)	G	取消刀具方向框架旋转	m	+		PGsl
TOROTX	G	WCS 的 X 轴通过框架旋转和刀具方向平行	m	+		PGsl
TOROTY	G	WCS 的 Y 轴通过框架旋转和刀具方向平行	m	+		PGsl
TOROTZ	G	同 TOROT	m	+		PGsl
TOUPPER	F	将一个字符串的字母转换成大写字母		+	-	PGAsl
TOWBCS	G	基本坐标系中的磨损值（BCS）	m	+		PGAsl
TOWKCS	G	用于运动转换的刀头坐标系中的磨损值（与刀具旋转 MCS 不同）	m	+		PGAsl
TOWMCS	G	机床坐标系(MCS)中的磨损值	m	+		PGAsl
TOWSTD 6)	G	刀具长度中偏移的初始设定值	m	+		PGAsl
TOWTCS	G	刀具坐标系中的磨损值（刀架基准点 T 位于刀具夹持装置中）	m	+		PGAsl
TOWWCS	G	工件坐标系（WCS）中的磨损值	m	+		PGAsl
TR	K	框架变量的偏移分量		+		PGAsl
TRAANG	P	倾斜轴转换		+	-	PGAsl
TRACON	P	级联转换		+	-	PGAsl
TRACYL	P	圆柱：柱面转换		+	-	PGAsl
TRAFOOF	P	取消通道中激活的转换		+	-	PGAsl
TRAFOON	P	激活以运动链定义的转换		+	-	PGAsl
TRAILOF	P	取消异步耦合运动		+	+	PGAsl
TRAILON	P	激活异步耦合运动		+	+	PGAsl
TRANS	G	可编程零点偏移绝对值	s	+		PGsl
TRANSMIT	P	极坐标转换（端面加工）		+	-	PGAsl
TRAORI	P	4 轴转换、5 轴转换，同类转换		+	-	PGAsl
TRUE	K	逻辑常量：真		+		PGAsl
TRUNC	F	舍去小数点后位数		+	+	PGAsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
TU		轴交角	s	+		PGAsl
TURN	A	螺旋线圈数	s	+		PGsl
ULI	K	变量的上限值		+		PGAsl
UNLOCK	P	使能带 ID 的同步动作（继续工艺循环）		-	+	FBSYsl
UNTIL	K	结束一个 REPEAT 循环的条件		+		PGAsl
UPATH	G	FGROUP 轴的轨迹基准为曲线参数。	m	+		PGAsl
VAR	K	关键字：参数传递方式		+		PGAsl
VELOLIM	K	降低最大轴速度	m	+		PGAsl
VELOLIMA	K	降低或提高跟随轴的最大轴速度	m	+	+	PGAsl
WAITC	P	等待，直到主轴对程序段变化条件已经能满足轴/主轴的要求。		+	-	PGAsl
WAITE	P	等待另一个通道上的程序结束。		+	-	PGAsl
WAITENC	P	等待轴位置经过同步或补偿		+	-	PGAsl
WAITM	P	等待指定通道中的标记；以准停结束前一个程序段。		+	-	PGAsl
WAITMC	P	等待设定通道中的标记;仅当其它通道尚未到达标记时精确停止		+	-	PGAsl
WAITP	P	等待定位轴运动结束		+	-	PGsl
WAITS	P	等待到达主轴位置		+	-	PGsl
WALCS0 [6]	G	取消选择工件坐标系工作区界限	m	+	-	PGsl
WALCS1	G	WCS 工作区域限制组 1 生效	m	+	-	PGsl
WALCS2	G	WCS 工作区域限制组 2 生效	m	+	-	PGsl
WALCS3	G	WCS 工作区域限制组 3 生效	m	+	-	PGsl
WALCS4	G	WCS 工作区域限制组 4 生效	m	+	-	PGsl
WALCS5	G	WCS 工作区域限制组 5 生效	m	+	-	PGsl
WALCS6	G	WCS 工作区域限制组 6 生效	m	+	-	PGsl
WALCS7	G	WCS 工作区域限制组 7 生效	m	+	-	PGsl
WALCS8	G	WCS 工作区域限制组 8 生效	m	+	-	PGsl
WALCS9	G	WCS 工作区域限制组 9 生效	m	+	-	PGsl
WALCS10	G	WCS 工作区域限制组 10 生效	m	+	-	PGsl

指令	类型 [1]	含义	W [2]	TP [3]	SA [4]	说明参见 [5]
[1] [2] [3] [4] [5] 详细说明参见图例 (页 959)。						
WALIMOF	G	取消 BCS 工作区域限制	m	+	-	PGsl
WALIMON [6]	G	激活 BCS 工作区域限制	m	+	-	PGsl
WHEN	K	当条件满足后，执行该动作一次。		-	+	FBSYsl
WHENEVER	K	当条件满足后，在每个插补周期中循环执行该动作。		-	+	FBSYsl
WHILE	K	WHILE 程序循环开始		+		PGAsl
WRITE	P	文本写入到文件系统。 在指定文件的结束处插入一个程序段。		+	-	PGAsl
WRTPR	P	在 BTSS 变量中写入字符串		+	-	PGsl
X	A	轴名称	m/s	+	+	PGsl
XOR	O	逻辑“异－或”		+		PGAsl
Y	A	轴名称	m/s	+	+	PGsl
Z	A	轴名称	m/s	+	+	PGsl

图例说明

1) 指令类型：

A 地址

标识符，表示向其赋值（如 OVR=10）。还有一些地址，无需赋值也能激活或取消功能（例如 CPLON 和 CPLOF）。

C (A) AST 循环

预定义 NC 程序，用于通过 AST（= 自动伺服优化）进行自动再优化。根据具体优化情况可使用参数对这些循环进行调整设置，参数在调用时传输至循环。

C 测量循环

(M) 预定义 NC 程序，可普遍适用的测量过程编程，例如测定圆柱体工件的内直径。根据具体测量情况可使用参数对这些循环进行调整设置，参数在调用时传输至循环。

C (T) 工艺循环

预定义 NC 程序，可普遍适用的加工过程编程，例如螺纹钻削或铣削型腔。根据具体工况可使用参数对这些循环进行调整设置，参数在调用时传输至循环。

F 预定义功能（提供返回值）

预定义功能可调用用作表达式中的操作数。

G G 指令

G 指令划分成组。同一个 G 指令组中的 G 指令在一个程序段中只能出现一个。G 指令可模态有效（直到被同组中其他指令替代），或者是非模态有效（只在写入的程序段中有效）。

K 关键字

标识符，确定程序段的句法。如不向关键字赋值，则使用该关键字无法激活/取消 NC 功能。

示例：控制结构(IF, ELSE, ENDIF, WHEN, ...)，程序过程(GOTOB, GOTO, RET …)

O 运算符

算术、比较或逻辑运算的运算符

P 预定义程序（不提供返回值）

PA 程序属性

程序属性位于子程序定义行的末端：

```
PROC <程序名称>(...) <程序属性>
```

其确定了子程序运行时的特性。

[2] 指令的有效性：

m 模态

s 非模态

[3] 是否可在零件程序中编程：

+ 可编程

- 不可编程

M 只能由机床制造商编程

[4] 是否可在同步动作中编程：

+ 可编程

- 不可编程

T 仅可在工艺循环中编程

[5] 资料参考，即包含指令详细说明的资料：

PGsl 编程手册 基本原理

PGAsl 编程手册 工作准备

BNMsl 编程手册 测量循环

BHDsl 操作手册 车床版

BHFsl 操作手册 铣床版

FB1sl ( ) 功能手册 基本功能（括号中是相应功能的字母数字缩写）

FB2sl ( ) 功能手册 扩展功能（括号中是相应功能的字母数字缩写）

FB3sl ( ) 功能手册 特殊功能（括号中是相应功能的字母数字缩写）

FBSIsl 功能手册 Safety Integrated

FBSYsl 功能手册 同步动作

FBWsl 功能手册 刀具管理

[6] 程序初始的默认设置（若没有另行编程，即为控制系统的出厂设置）。

图 21-1 指令表中脚注的含义

21.2 指令：在 SINUMERIK 828D 上的可用性

说明

循环

当涉及以下需授权的选件时，循环会被标记为“可选”：

- 扩展工艺功能（产品编号：6FC5800-0AP58-0YB0）
- 测量循环（产品编号：6FC5800-0AP28-0YB0）
- 运动测量（产品编号：6FC5800-0AP18-0YB0）
- SINUMERIK 高级磨削（产品编号：6FC5800-0AS35-0YB0）

如果循环已通过选件“扩展工艺功能”成为系统的部分功能，则不再进行标记。

21.2.1 控制系统类型 铣削/车削

指令 A - C

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口 版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口 版本 (te822)
:	●	●	●	●	●	●	●	●
*	●	●	●	●	●	●	●	●
+	●	●	●	●	●	●	●	●
-	●	●	●	●	●	●	●	●
<	●	●	●	●	●	●	●	●
<<	●	●	●	●	●	●	●	●
<=	●	●	●	●	●	●	●	●
=	●	●	●	●	●	●	●	●
>=	●	●	●	●	●	●	●	●
/	●	●	●	●	●	●	●	●
/0 … /7	●	●	●	●	●	●	●	●
A	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
A2	-	-	-	-	-	-	-	-
A3	-	-	-	-	-	-	-	-
A4	-	-	-	-	-	-	-	-
A5	-	-	-	-	-	-	-	-
A6	-	-	-	-	-	-	-	-
A7	-	-	-	-	-	-	-	-
ABS	●	●	●	●	●	●	●	●
AC	●	●	●	●	●	●	●	●
ACC	●	●	●	●	●	●	●	●
ACCLIMA	●	●	●	●	●	●	●	●
ACN	●	●	●	●	●	●	●	●
ACOS	●	●	●	●	●	●	●	●
ACP	●	●	●	●	●	●	●	●
ACTBLOCNO	●	●	●	●	●	●	●	●
ADDFRAME	●	●	●	●	●	●	●	●
ADIS	●	●	●	●	●	●	●	●
ADISPOS	●	●	●	●	●	●	●	●
ADISPOSA	●	●	●	●	●	●	●	●
ALF	●	●	●	●	●	●	●	●
AMIRROR	●	●	●	●	●	●	●	●
AND	●	●	●	●	●	●	●	●
ANG	●	●	●	●	●	●	●	●
AP	●	●	●	●	●	●	●	●
APR	●	●	●	●	●	●	●	●
APRB	●	●	●	●	●	●	●	●
APRP	●	●	●	●	●	●	●	●
APW	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
APWB	●	●	●	●	●	●	●	●
APWP	●	●	●	●	●	●	●	●
APX	●	●	●	●	●	●	●	●
AR	●	●	●	●	●	●	●	●
AROT	●	●	●	●	●	●	●	●
AROTS	●	●	●	●	●	●	●	●
AS	●	●	●	●	●	●	●	●
ASCALE	●	●	●	●	●	●	●	●
ASIN	●	●	●	●	●	●	●	●
ASPLINE	○	○	○	○	○	○	○	○
ATAN2	●	●	●	●	●	●	●	●
ATOL	●	●	●	●	●	●	●	●
ATRANS	●	●	●	●	●	●	●	●
AUXFUDEL	●	●	●	●	●	●	●	●
AUXFUDELG	●	●	●	●	●	●	●	●
AUXFUMSEQ	●	●	●	●	●	●	●	●
AUXFUSYNC	●	●	●	●	●	●	●	●
AX	●	●	●	●	●	●	●	●
AXCTSWE	-	-	-	-	-	-	-	-
AXCTSWEC	-	-	-	-	-	-	-	-
AXCTSWED	-	-	-	-	-	-	-	-
AXIS	●	●	●	●	●	●	●	●
AXNAME	●	●	●	●	●	●	●	●
AXSTRING	●	●	●	●	●	●	●	●
AXTOCHAN	●	●	●	●	●	●	●	●
AXTOSPI	●	●	●	●	●	●	●	●
B	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
B2	-	-	-	-	-	-	-	-
B3	-	-	-	-	-	-	-	-
B4	-	-	-	-	-	-	-	-
B5	-	-	-	-	-	-	-	-
B6	-	-	-	-	-	-	-	-
B7	-	-	-	-	-	-	-	-
B_AND	●	●	●	●	●	●	●	●
B_OR	●	●	●	●	●	●	●	●
B_NOT	●	●	●	●	●	●	●	●
B_XOR	●	●	●	●	●	●	●	●
BAUTO	○	○	○	○	○	○	○	○
BLOCK	●	●	●	●	●	●	●	●
BLSYNC	●	●	●	●	●	●	●	●
BNAT	○	○	○	○	○	○	○	○
BOOL	●	●	●	●	●	●	●	●
BOUND	●	●	●	●	●	●	●	●
BRISK	●	●	●	●	●	●	●	●
BRISKA	●	●	●	●	●	●	●	●
BSPLINE	○	○	○	○	○	○	○	○
BTAN	○	○	○	○	○	○	○	○
C	●	●	●	●	●	●	●	●
C2	-	-	-	-	-	-	通道轴名称	通道轴名称
C3	-	-	-	-	-	-	-	-
C4	-	-	-	-	-	-	-	-
C5	-	-	-	-	-	-	-	-
C6	-	-	-	-	-	-	-	-

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
C7	-	-	-	-	-	-	-	-
CAC	●	●	●	●	●	●	●	●
CACN	●	●	●	●	●	●	●	●
CACP	●	●	●	●	●	●	●	●
CALCDAT	●	●	●	●	●	●	●	●
CALCPOSI	●	●	●	●	●	●	●	●
CALL	●	●	●	●	●	●	●	●
CALLPATH	●	●	●	●	●	●	●	●
CANCEL	●	●	●	●	●	●	●	●
CASE	●	●	●	●	●	●	●	●
CDC	●	●	●	●	●	●	●	●
CDOF	-	-	-	-	-	-	-	-
CDOF2	-	-	-	-	-	-	-	-
CDON	-	-	-	-	-	-	-	-
CFC	●	●	●	●	●	●	●	●
CFIN	●	●	●	●	●	●	●	●
CFINE	●	●	●	●	●	●	●	●
CFTCP	●	●	●	●	●	●	●	●
CHAN	●	●	●	●	●	●	●	●
CHANDATA	●	●	●	●	●	●	●	●
CHAR	●	●	●	●	●	●	●	●
CHF	●	●	●	●	●	●	●	●
CHKDM	●	●	●	●	●	●	●	●
CHKDNO	●	●	●	●	●	●	●	●
CHR	●	●	●	●	●	●	●	●
CIC	●	●	●	●	●	●	●	●
CIP	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
CLEARM	-	-	-	-	-	○	-	●
CLRINT	●	●	●	●	●	●	●	●
CMIRROR	●	●	●	●	●	●	●	●
COARSEA	●	●	●	●	●	●	●	●
COLLPAIR	-	-	-	-	-	-	-	-
COMPCAD	●	-	●	-	●	●	-	-
COMPCURV	●	-	●	-	●	●	-	-
COMPLETE	●	●	●	●	●	●	●	●
COMPOF	●	-	●	-	●	●	-	-
COMPON	●	-	●	-	●	●	-	-
COMPSURF	-	-	○	-	○	○	-	-
CONTDCON	●	●	●	●	●	●	●	●
CONTPRON	●	●	●	●	●	●	●	●
CORROF	●	●	●	●	●	●	●	●
CORRTRAFO	-	-	-	-	-	-	-	-
COS	●	●	●	●	●	●	●	●
COUPDEF	-	○	-	○	-	-	○	○
COUPDEL	-	○	-	○	-	-	○	○
COUPOF	-	○	-	○	-	-	○	○
COUPOFS	-	○	-	○	-	-	○	○
COUPON	-	○	-	○	-	-	○	○
COUPONC	-	○	-	○	-	-	○	○
COUPRES	-	○	-	○	-	-	○	○
CP	●	●	●	●	●	●	●	●
CPBC	○	○	○	○	○	○	○	○
CPDEF	○	○	○	○	○	○	○	○
CPDEL	○	○	○	○	○	○	○	○

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
CPFMOF	○	○	○	○	○	○	○	○
CPFMON	○	○	○	○	○	○	○	○
CPFMSON	○	○	○	○	○	○	○	○
CPFPOS	○	○	○	○	○	○	○	○
CPFRS	○	○	○	○	○	○	○	○
CPLA	○	○	○	○	○	○	○	○
CPLCTID	○	○	○	○	○	○	○	○
CPLDEF	○	○	○	○	○	○	○	○
CPLDEL	○	○	○	○	○	○	○	○
CPLDEN	○	○	○	○	○	○	○	○
CPLINSC	○	○	○	○	○	○	○	○
CPLINTR	○	○	○	○	○	○	○	○
CPLNUM	○	○	○	○	○	○	○	○
CPLOF	○	○	○	○	○	○	○	○
CPLON	○	○	○	○	○	○	○	○
CPLOUTSC	○	○	○	○	○	○	○	○
CPLOUTTR	○	○	○	○	○	○	○	○
CPLPOS	○	○	○	○	○	○	○	○
CPLSETVAL	○	○	○	○	○	○	○	○
CPMALARM	○	○	○	○	○	○	○	○
CPMBRAKE	○	○	○	○	○	○	○	○
CPMPRT	○	○	○	○	○	○	○	○
CPMRESET	○	○	○	○	○	○	○	○
CPMSTART	○	○	○	○	○	○	○	○
CPMVDI	○	○	○	○	○	○	○	○
CPOF	○	○	○	○	○	○	○	○
CPON	○	○	○	○	○	○	○	○

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
CPRECOF	●	●	●	●	●	●	●	●
CPRECON	●	●	●	●	●	●	●	●
CPRES	○	○	○	○	○	○	○	○
CPROT	●	●	●	●	●	●	●	●
CPROTDEF	●	●	●	●	●	●	●	●
CPSETTYPE	○	○	○	○	○	○	○	○
CPSYNCOP	○	○	○	○	○	○	○	○
CPSYNCOP2	○	○	○	○	○	○	○	○
CPSYNCOV	○	○	○	○	○	○	○	○
CPSYNFIP	○	○	○	○	○	○	○	○
CPSYNFIP2	○	○	○	○	○	○	○	○
CPSYNFIV	○	○	○	○	○	○	○	○
CR	●	●	●	●	●	●	●	●
CROT	●	●	●	●	●	●	●	●
CROTS	●	●	●	●	●	●	●	●
CRPL	●	●	●	●	●	●	●	●
CSCALE	●	●	●	●	●	●	●	●
CSPLINE	○	○	○	○	○	○	○	○
CT	●	●	●	●	●	●	●	●
CTAB	-	-	-	-	-	-	-	-
CTABDEF	-	-	-	-	-	-	-	-
CTABDEL	-	-	-	-	-	-	-	-
CTABEND	-	-	-	-	-	-	-	-
CTABEXISTS	-	-	-	-	-	-	-	-
CTABFNO	-	-	-	-	-	-	-	-
CTABFPOL	-	-	-	-	-	-	-	-
CTABFSEG	-	-	-	-	-	-	-	-

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
CTABID	-	-	-	-	-	-	-	-
CTABINV	-	-	-	-	-	-	-	-
CTABISLOCK	-	-	-	-	-	-	-	-
CTABLOCK	-	-	-	-	-	-	-	-
CTABMEMTYP	-	-	-	-	-	-	-	-
CTABMPOL	-	-	-	-	-	-	-	-
CTABMSEG	-	-	-	-	-	-	-	-
CTABNO	-	-	-	-	-	-	-	-
CTABNOMEM	-	-	-	-	-	-	-	-
CTABPERIOD	-	-	-	-	-	-	-	-
CTABPOL	-	-	-	-	-	-	-	-
CTABPOLID	-	-	-	-	-	-	-	-
CTABSEG	-	-	-	-	-	-	-	-
CTABSEGID	-	-	-	-	-	-	-	-
CTABSEV	-	-	-	-	-	-	-	-
CTABSSV	-	-	-	-	-	-	-	-
CTABTEP	-	-	-	-	-	-	-	-
CTABTEV	-	-	-	-	-	-	-	-
CTABTMAX	-	-	-	-	-	-	-	-
CTABTMIN	-	-	-	-	-	-	-	-
CTABTSP	-	-	-	-	-	-	-	-
CTABTSV	-	-	-	-	-	-	-	-
CTABUNLOCK	-	-	-	-	-	-	-	-
CTOL	●	●	●	●	●	●	●	●
CTRANS	●	●	●	●	●	●	●	●
CUT2D	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
CUT2DD	●	●	●	●	●	●	●	●
CUT2DF	●	●	●	●	●	●	●	●
CUT2DFD	●	●	●	●	●	●	●	●
CUT3DC	-	-	-	-	-	-	-	-
CUT3DCC	-	-	-	-	-	-	-	-
CUT3DCCD	-	-	-	-	-	-	-	-
CUT3DCD	-	-	-	-	-	-	-	-
CUT3DF	-	-	-	-	-	-	-	-
CUT3DFD	-	-	-	-	-	-	-	-
CUT3DFF	-	-	-	-	-	-	-	-
CUT3DFS	-	-	-	-	-	-	-	-
CUTCONOF	●	●	●	●	●	●	●	●
CUTCONON	●	●	●	●	●	●	●	●
CUTMOD	●	●	●	●	●	●	●	●
CUTMODK	-	-	-	-	-	-	-	-
CYCLE60	○	○	●	●	●	●	●	●
CYCLE61	●	●	●	●	●	●	●	●
CYCLE62	●	●	●	●	●	●	●	●
CYCLE63	○	○	●	●	●	●	●	●
CYCLE64	○	○	●	●	●	●	●	●
CYCLE70	○	○	●	●	●	●	●	●
CYCLE72	●	●	●	●	●	●	●	●
CYCLE76	●	●	●	●	●	●	●	●
CYCLE77	●	●	●	●	●	●	●	●
CYCLE78	○	○	●	●	●	●	●	●
CYCLE79	○	○	●	●	●	●	●	●
CYCLE81	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
CYCLE82	●	●	●	●	●	●	●	●
CYCLE83	●	●	●	●	●	●	●	●
CYCLE84	●	●	●	●	●	●	●	●
CYCLE85	●	●	●	●	●	●	●	●
CYCLE86	●	●	●	●	●	●	●	●
CYCLE92	●	●	●	●	●	●	●	●
CYCLE95	●	●	●	●	●	●	●	●
CYCLE98	●	●	●	●	●	●	●	●
CYCLE99	●	●	●	●	●	●	●	●
CYCLE150	○	○	○	○	○	○	○	○
CYCLE435	-	-	-	-	-	-	-	-
CYCLE495	-	-	-	-	-	-	-	-
CYCLE750	●	●	●	●	●	●	●	●
CYCLE751	●	●	●	●	●	●	●	●
CYCLE752	●	●	●	●	●	●	●	●
CYCLE753	●	●	●	●	●	●	●	●
CYCLE754	●	●	●	●	●	●	●	●
CYCLE755	●	●	●	●	●	●	●	●
CYCLE756	●	●	●	●	●	●	●	●
CYCLE757	●	●	●	●	●	●	●	●
CYCLE758	●	●	●	●	●	●	●	●
CYCLE759	●	●	●	●	●	●	●	●
CYCLE800	-	-	●	●	●	●	●	●
CYCLE801	●	●	●	●	●	●	●	●
CYCLE802	●	●	●	●	●	●	●	●
CYCLE830	○	○	●	●	●	●	●	●
CYCLE832	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
CYCLE840	●	●	●	●	●	●	●	●
CYCLE899	○	○	●	●	●	●	●	●
CYCLE930	●	●	●	●	●	●	●	●
CYCLE940	●	●	●	●	●	●	●	●
CYCLE951	●	●	●	●	●	●	●	●
CYCLE952	○	○	●	●	●	●	●	●
CYCLE961	○	○	○	○	○	○	○	○
CYCLE971	○	○	○	○	○	○	○	○
CYCLE973	○	○	○	○	○	○	○	○
CYCLE974	○	○	○	○	○	○	○	○
CYCLE976	○	○	○	○	○	○	○	○
CYCLE977	○	○	○	○	○	○	○	○
CYCLE978	○	○	○	○	○	○	○	○
CYCLE979	○	○	○	○	○	○	○	○
CYCLE982	○	○	○	○	○	○	○	○
CYCLE994	○	○	○	○	○	○	○	○
CYCLE995	○	○	○	○	○	○	○	○
CYCLE996	○	-	○	-	○	○	-	-
CYCLE997	○	○	○	○	○	○	○	○
CYCLE998	○	○	○	○	○	○	○	○
CYCLE4071	-	-	-	-	-	-	-	-
CYCLE4072	-	-	-	-	-	-	-	-
CYCLE4073	-	-	-	-	-	-	-	-
CYCLE4074	-	-	-	-	-	-	-	-
CYCLE4075	-	-	-	-	-	-	-	-
CYCLE4077	-	-	-	-	-	-	-	-

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
CYCLE4078	-	-	-	-	-	-	-	-
CYCLE4079	-	-	-	-	-	-	-	-

指令 D - F

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
D	●	●	●	●	●	●	●	●
D0	●	●	●	●	●	●	●	●
DAC	●	●	●	●	●	●	●	●
DC	●	●	●	●	●	●	●	●
DCI	●	●	●	●	●	●	●	●
DCM	●	●	●	●	●	●	●	●
DCU	●	●	●	●	●	●	●	●
DEF	●	●	●	●	●	●	●	●
DEFINE	●	●	●	●	●	●	●	●
DEFAULT	●	●	●	●	●	●	●	●
DELAYFSTON	●	●	●	●	●	●	●	●
DELAYFSTOF	●	●	●	●	●	●	●	●
DELDL	●	●	●	●	●	●	●	●
DELDTG	●	●	●	●	●	●	●	●
DELETE	●	●	●	●	●	●	●	●
DELMLOWNER	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口 版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口 版本 (te822)
DELMLRES	●	●	●	●	●	●	●	●
DELMT	●	●	●	●	●	●	●	●
DELOBJ	-	-	-	-	-	-	-	-
DELT	●	●	●	●	●	●	●	●
DELTC	●	●	●	●	●	●	●	●
DELTOOLENV	●	●	●	●	●	●	●	●
DIACYCOFA	●	●	●	●	●	●	●	●
DIAM90	●	●	●	●	●	●	●	●
DIAM90A	●	●	●	●	●	●	●	●
DIAMCHAN	●	●	●	●	●	●	●	●
DIAMCHANA	●	●	●	●	●	●	●	●
DIAMCYCOF	●	●	●	●	●	●	●	●
DIAMOF	●	●	●	●	●	●	●	●
DIAMOFA	●	●	●	●	●	●	●	●
DIAMON	●	●	●	●	●	●	●	●
DIAMONA	●	●	●	●	●	●	●	●
DIC	●	●	●	●	●	●	●	●
DILF	●	●	●	●	●	●	●	●
DISABLE	●	●	●	●	●	●	●	●
DISC	●	●	●	●	●	●	●	●
DISCL	●	●	●	●	●	●	●	●
DISPLOF	●	●	●	●	●	●	●	●
DISPLON	●	●	●	●	●	●	●	●
DISPR	●	●	●	●	●	●	●	●
DISR	●	●	●	●	●	●	●	●
DISRP	●	●	●	●	●	●	●	●
DITE	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
DITS	●	●	●	●	●	●	●	●
DIV	●	●	●	●	●	●	●	●
DL	-	-	-	-	-	-	-	-
DO	●	●	●	●	●	●	●	●
DRFOF	●	●	●	●	●	●	●	●
DRIVE	●	●	●	●	●	●	●	●
DRIVEA	●	●	●	●	●	●	●	●
DYNFINISH	●	●	●	●	●	●	●	●
DYNNORM	●	●	●	●	●	●	●	●
DYNPOS	●	●	●	●	●	●	●	●
DYNROUGH	●	●	●	●	●	●	●	●
DYNSEMIFIN	●	●	●	●	●	●	●	●
DZERO	●	●	●	●	●	●	●	●
EAUTO	○	○	○	○	○	○	○	○
EGDEF	-	○	-	○	-	-	○	○
EGDEL	-	○	-	○	-	-	○	○
EGOFC	-	○	-	○	-	-	○	○
EGOFS	-	○	-	○	-	-	○	○
EGON	-	○	-	○	-	-	○	○
EGONSYN	-	○	-	○	-	-	○	○
EGONSYNE	-	○	-	○	-	-	○	○
ELSE	●	●	●	●	●	●	●	●
ENABLE	●	●	●	●	●	●	●	●
ENAT	○	○	○	○	○	○	○	○
ENDFOR	●	●	●	●	●	●	●	●
ENDIF	●	●	●	●	●	●	●	●
ENDLABEL	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
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	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
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	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口 版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口 版本 (te822)
FTOC	●	●	●	●	●	●	●	●
FTOCOF	●	●	●	●	●	●	●	●
FTOCON	●	●	●	●	●	●	●	●
FXS	●	●	●	●	●	●	●	●
FXST	●	●	●	●	●	●	●	●
FXSW	●	●	●	●	●	●	●	●
FZ	●	●	●	●	●	●	●	●

指令 G - L

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口 版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口 版本 (te822)
G0	●	●	●	●	●	●	●	●
G1	●	●	●	●	●	●	●	●
G2	●	●	●	●	●	●	●	●
G3	●	●	●	●	●	●	●	●
G4	●	●	●	●	●	●	●	●
G5	●	●	●	●	●	●	●	●
G7	●	●	●	●	●	●	●	●
G9	●	●	●	●	●	●	●	●
G17	●	●	●	●	●	●	●	●
G18	●	●	●	●	●	●	●	●
G19	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
G25	●	●	●	●	●	●	●	●
G26	●	●	●	●	●	●	●	●
G33	●	●	●	●	●	●	●	●
G34	●	●	●	●	●	●	●	●
G35	●	●	●	●	●	●	●	●
G40	●	●	●	●	●	●	●	●
G41	●	●	●	●	●	●	●	●
G42	●	●	●	●	●	●	●	●
G53	●	●	●	●	●	●	●	●
G54	●	●	●	●	●	●	●	●
G55	●	●	●	●	●	●	●	●
G56	●	●	●	●	●	●	●	●
G57	●	●	●	●	●	●	●	●
G58	→ G505							
G59	→ G506							
G60	●	●	●	●	●	●	●	●
G62	●	●	●	●	●	●	●	●
G63	●	●	●	●	●	●	●	●
G64	●	●	●	●	●	●	●	●
G70	●	●	●	●	●	●	●	●
G71	●	●	●	●	●	●	●	●
G74	●	●	●	●	●	●	●	●
G75	●	●	●	●	●	●	●	●
G90	●	●	●	●	●	●	●	●
G91	●	●	●	●	●	●	●	●
G93	●	●	●	●	●	●	●	●
G94	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
G95	●	●	●	●	●	●	●	●
G96	●	●	●	●	●	●	●	●
G97	●	●	●	●	●	●	●	●
G110	●	●	●	●	●	●	●	●
G111	●	●	●	●	●	●	●	●
G112	●	●	●	●	●	●	●	●
G140	●	●	●	●	●	●	●	●
G141	●	●	●	●	●	●	●	●
G142	●	●	●	●	●	●	●	●
G143	●	●	●	●	●	●	●	●
G147	●	●	●	●	●	●	●	●
G148	●	●	●	●	●	●	●	●
G153	●	●	●	●	●	●	●	●
G247	●	●	●	●	●	●	●	●
G248	●	●	●	●	●	●	●	●
G290	●	●	●	●	●	●	●	●
G291	●	●	●	●	●	●	●	●
G331	●	●	●	●	●	●	●	●
G332	●	●	●	●	●	●	●	●
G335	●	●	●	●	●	●	●	●
G336	●	●	●	●	●	●	●	●
G340	●	●	●	●	●	●	●	●
G341	●	●	●	●	●	●	●	●
G347	●	●	●	●	●	●	●	●
G348	●	●	●	●	●	●	●	●
G450	●	●	●	●	●	●	●	●
G451	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
G460	●	●	●	●	●	●	●	●
G461	●	●	●	●	●	●	●	●
G462	●	●	●	●	●	●	●	●
G500	●	●	●	●	●	●	●	●
G505...G599	●	●	●	●	●	●	●	●
G601	●	●	●	●	●	●	●	●
G602	●	●	●	●	●	●	●	●
G603	●	●	●	●	●	●	●	●
G621	●	●	●	●	●	●	●	●
G641	●	●	●	●	●	●	●	●
G642	●	●	●	●	●	●	●	●
G643	●	●	●	●	●	●	●	●
G644	●	●	●	●	●	●	●	●
G645	●	●	●	●	●	●	●	●
G700	●	●	●	●	●	●	●	●
G710	●	●	●	●	●	●	●	●
G810...G819	-	-	-	-	-	-	-	-
G820...G829	-	-	-	-	-	-	-	-
G931	●	●	●	●	●	●	●	●
G942	●	●	●	●	●	●	●	●
G952	●	●	●	●	●	●	●	●
G961	●	●	●	●	●	●	●	●
G962	●	●	●	●	●	●	●	●
G971	●	●	●	●	●	●	●	●
G972	●	●	●	●	●	●	●	●
G973	●	●	●	●	●	●	●	●
GEOAX	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
GET	●	●	●	●	●	●	●	●
GETACTT	●	●	●	●	●	●	●	●
GETACTTD	●	●	●	●	●	●	●	●
GETD	-	-	-	-	-	○	-	●
GETDNO	●	●	●	●	●	●	●	●
GETEXET	●	●	●	●	●	●	●	●
GETFREELOC	●	●	●	●	●	●	●	●
GETSELT	●	●	●	●	●	●	●	●
GETT	●	●	●	●	●	●	●	●
GETTCOR	●	●	●	●	●	●	●	●
GETTENV	●	●	●	●	●	●	●	●
GETVARAP	●	●	●	●	●	●	●	●
GETVARDFT	●	●	●	●	●	●	●	●
GETVARLIM	●	●	●	●	●	●	●	●
GETVARPHU	●	●	●	●	●	●	●	●
GETVARTYP	●	●	●	●	●	●	●	●
GFRAME0 ... GFRAME100	-	-	-	-	-	-	-	-
GOTO	●	●	●	●	●	●	●	●
GOTOB	●	●	●	●	●	●	●	●
GOTOC	●	●	●	●	●	●	●	●
GOTOF	●	●	●	●	●	●	●	●
GOTOS	●	●	●	●	●	●	●	●
GP	●	●	●	●	●	●	●	●
GWPSOF	●	●	●	●	●	●	●	●
GROUP_ ADDEND	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
GROUP_BEGIN	●	●	●	●	●	●	●	●
GROUP_END	●	●	●	●	●	●	●	●
GWPSON	●	●	●	●	●	●	●	●
H...	●	●	●	●	●	●	●	●
HOLES1	●	●	●	●	●	●	●	●
HOLES2	●	●	●	●	●	●	●	●
I	●	●	●	●	●	●	●	●
I1	●	●	●	●	●	●	●	●
IC	●	●	●	●	●	●	●	●
ICYCOF	●	●	●	●	●	●	●	●
ICYCON	●	●	●	●	●	●	●	●
ID	●	●	●	●	●	●	●	●
IDS	●	●	●	●	●	●	●	●
IF	●	●	●	●	●	●	●	●
INDEX	●	●	●	●	●	●	●	●
INIPO	●	●	●	●	●	●	●	●
INIRE	●	●	●	●	●	●	●	●
INICF	●	●	●	●	●	●	●	●
INIT	-	-	-	-	-	○	-	●
INITIAL								
整数	●	●	●	●	●	●	●	●
INTERSEC	●	●	●	●	●	●	●	●
INVCCW	-	-	-	-	-	-	-	-
INVCW	-	-	-	-	-	-	-	-
INVFRAME	●	●	●	●	●	●	●	●
IP	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
IPOBRKA	●	●	●	●	●	●	●	●
IPOENDA	●	●	●	●	●	●	●	●
IPTRLOCK	●	●	●	●	●	●	●	●
IPTRUNLOCK	●	●	●	●	●	●	●	●
IR	●	●	●	●	●	●	●	●
ISAXIS	●	●	●	●	●	●	●	●
ISD	-	-	-	-	-	-	-	-
ISFILE	●	●	●	●	●	●	●	●
ISNUMBER	●	●	●	●	●	●	●	●
ISOCALL	●	●	●	●	●	●	●	●
ISVAR	●	●	●	●	●	●	●	●
J	●	●	●	●	●	●	●	●
J1	●	●	●	●	●	●	●	●
JERKA	●	●	●	●	●	●	●	●
JERKLIM	●	●	●	●	●	●	●	●
JERKLIMA	●	●	●	●	●	●	●	●
JR	●	●	●	●	●	●	●	●
K	●	●	●	●	●	●	●	●
K1	●	●	●	●	●	●	●	●
KONT	●	●	●	●	●	●	●	●
KONTC	●	●	●	●	●	●	●	●
KONTT	●	●	●	●	●	●	●	●
KR	●	●	●	●	●	●	●	●
L	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
LEAD 刀具定向 定向多项式	- -	- -	- -	- -	- -	- -	- -	- -
LEADOF	-	-	-	●	-	-	●	●
LEADON	-	-	-	●	-	-	●	●
LENTOAX	●	●	●	●	●	●	●	●
LFOF	●	●	●	●	●	●	●	●
LFON	●	●	●	●	●	●	●	●
LFPOS	●	●	●	●	●	●	●	●
LFTXT	●	●	●	●	●	●	●	●
LFWP	●	●	●	●	●	●	●	●
LIFTFAST	●	●	●	●	●	●	●	●
LIMS	●	●	●	●	●	●	●	●
LLI	●	●	●	●	●	●	●	●
LN	●	●	●	●	●	●	●	●
LOCK	●	●	●	●	●	●	●	●
LONGHOLE	●	●	●	●	●	●	●	●
LOOP	●	●	●	●	●	●	●	●

指令 M - R

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口 版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口 版本 (te822)
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	-	-	-	-	-	-	-	-

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口 版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口 版本 (te822)
MEASURE	●	●	●	●	●	●	●	●
MEAW	●	●	●	●	●	●	●	●
MEAWA	-	-	-	-	-	-	-	-
MI	●	●	●	●	●	●	●	●
MINDEX	●	●	●	●	●	●	●	●
MINVAL	●	●	●	●	●	●	●	●
MIRROR	●	●	●	●	●	●	●	●
MMC	●	●	●	●	●	●	●	●
MOD	●	●	●	●	●	●	●	●
MODAXVAL	●	●	●	●	●	●	●	●
MOV	●	●	●	●	●	●	●	●
MOVT	●	●	●	●	●	●	●	●
MSG	●	●	●	●	●	●	●	●
MVTOOL	●	●	●	●	●	●	●	●
N	●	●	●	●	●	●	●	●
NAMETOINT	●	●	●	●	●	●	●	●
NC	●	●	●	●	●	●	●	●
NEWCONF	●	●	●	●	●	●	●	●
NEWMT	●	●	●	●	●	●	●	●
NEWT	●	●	●	●	●	●	●	●
NORM	●	●	●	●	●	●	●	●
NOT	●	●	●	●	●	●	●	●
NPROT	●	●	●	●	●	●	●	●
NPROTDEF	●	●	●	●	●	●	●	●
NUMBER	●	●	●	●	●	●	●	●
OEMIPO1	-	-	-	-	-	-	-	-
OEMIPO2	-	-	-	-	-	-	-	-

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
OF	●	●	●	●	●	●	●	●
OFFN	●	●	●	●	●	●	●	●
OMA1	-	-	-	-	-	-	-	-
OMA2	-	-	-	-	-	-	-	-
OMA3	-	-	-	-	-	-	-	-
OMA4	-	-	-	-	-	-	-	-
OMA5	-	-	-	-	-	-	-	-
OR	●	●	●	●	●	●	●	●
ORIAXES	-	-	-	-	-	-	-	-
ORIAXPOS	-	-	-	-	-	-	-	-
ORIC	-	-	-	-	-	-	-	-
ORICONCCW	-	-	-	-	-	-	-	-
ORICONCW	-	-	-	-	-	-	-	-
ORICONIO	-	-	-	-	-	-	-	-
ORICONTO	-	-	-	-	-	-	-	-
ORICURVE	-	-	-	-	-	-	-	-
ORID	-	-	-	-	-	-	-	-
ORIEULER	-	-	-	-	-	-	-	-
ORIMKS	-	-	-	-	-	-	-	-
ORIPATH	-	-	-	-	-	-	-	-
ORIPATHS	-	-	-	-	-	-	-	-
ORIPLANE	-	-	-	-	-	-	-	-
ORIRESET	-	-	-	-	-	-	-	-
ORIROTA	-	-	-	-	-	-	-	-
ORIROTC	-	-	-	-	-	-	-	-
ORIROTR	-	-	-	-	-	-	-	-
ORIROTT	-	-	-	-	-	-	-	-

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
ORIRPY	-	-	-	-	-	-	-	-
ORIRPY2	-	-	-	-	-	-	-	-
ORIS	-	-	-	-	-	-	-	-
ORISOF	-	-	-	-	-	-	-	-
ORISOLH	-	-	-	-	-	-	-	-
ORISON	-	-	-	-	-	-	-	-
ORIVECT	-	-	-	-	-	-	-	-
ORIVIRT1	-	-	-	-	-	-	-	-
ORIVIRT2	-	-	-	-	-	-	-	-
ORIWKS	-	-	-	-	-	-	-	-
OS	-	-	-	-	-	-	-	-
OSB	-	-	-	-	-	-	-	-
OSC	-	-	-	-	-	-	-	-
OSCILL	-	-	-	-	-	-	-	-
OSCTRL	-	-	-	-	-	-	-	-
OSD	-	-	-	-	-	-	-	-
OSE	-	-	-	-	-	-	-	-
OSNSC	●	●	●	●	●	●	●	●
OSOF	-	-	-	-	-	-	-	-
OSP1	●	●	●	●	●	●	●	●
OSP2	●	●	●	●	●	●	●	●
OSS	-	-	-	-	-	-	-	-
OSSE	-	-	-	-	-	-	-	-
OST	-	-	-	-	-	-	-	-
OST1	●	●	●	●	●	●	●	●
OST2	●	●	●	●	●	●	●	●
OTOL	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
OVR	●	●	●	●	●	●	●	●
OVRA	●	●	●	●	●	●	●	●
OVRRAP	●	●	●	●	●	●	●	●
P	●	●	●	●	●	●	●	●
PAROT	●	●	●	●	●	●	●	●
PAROTOF	●	●	●	●	●	●	●	●
PCALL	●	●	●	●	●	●	●	●
PDELAYOF	-	-	-	-	-	-	-	-
PDELAYON	-	-	-	-	-	-	-	-
PHI	-	-	-	-	-	-	-	-
PHU	●	●	●	●	●	●	●	●
PL	-	-	-	-	-	-	-	-
PM	●	●	●	●	●	●	●	●
PO	-	-	-	-	-	-	-	-
POCKET3	●	●	●	●	●	●	●	●
POCKET4	●	●	●	●	●	●	●	●
POLF	●	●	●	●	●	●	●	●
POLFA	●	●	●	●	●	●	●	●
POLFMASK	●	●	●	●	●	●	●	●
POLFMLIN	●	●	●	●	●	●	●	●
POLY	-	-	-	-	-	-	-	-
POLYPATH	-	-	-	-	-	-	-	-
PON	-	-	-	-	-	-	-	-
PONS	-	-	-	-	-	-	-	-
POS	●	●	●	●	●	●	●	●
POSA	●	●	●	●	●	●	●	●
POSM	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
POSMT	●	●	●	●	●	●	●	●
POSP	●	●	●	●	●	●	●	●
POSRANGE	●	●	●	●	●	●	●	●
POT	●	●	●	●	●	●	●	●
PR	●	●	●	●	●	●	●	●
PREPRO	●	●	●	●	●	●	●	●
PRESETON	●	●	●	●	●	●	●	●
PRESETONS	●	●	●	●	●	●	●	●
PRIO	●	●	●	●	●	●	●	●
PRLOC	●	●	●	●	●	●	●	●
PROC	●	●	●	●	●	●	●	●
PROTA	●	●	●	●	●	●	●	●
PROTD	●	●	●	●	●	●	●	●
PROTS	●	●	●	●	●	●	●	●
PSI	-	-	-	-	-	-	-	-
PTP	●	●	●	●	●	●	●	●
PTPG0	●	●	●	●	●	●	●	●
点对点连接	●	●	●	●	●	●	●	●
PUNCHACC	-	-	-	-	-	-	-	-
PUTFTOC	●	●	●	●	●	●	●	●
PUTFTOCF	●	●	●	●	●	●	●	●
PW	○	○	○	○	○	○	○	○
QU	●	●	●	●	●	●	●	●
R...	●	●	●	●	●	●	●	●
RAC	●	●	●	●	●	●	●	●
RDISABLE	●	●	●	●	●	●	●	●
READ	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口 版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口 版本 (te822)
REAL	●	●	●	●	●	●	●	●
RELEASE	●	●	●	●	●	●	●	●
REP	●	●	●	●	●	●	●	●
REPEAT	●	●	●	●	●	●	●	●
REPEATB	●	●	●	●	●	●	●	●
REPOSA	●	●	●	●	●	●	●	●
REPOSH	●	●	●	●	●	●	●	●
REPOSHA	●	●	●	●	●	●	●	●
REPOSL	●	●	●	●	●	●	●	●
REPOSQ	●	●	●	●	●	●	●	●
REPOSQA	●	●	●	●	●	●	●	●
RESETMON	●	●	●	●	●	●	●	●
RET	●	●	●	●	●	●	●	●
RETB	●	●	●	●	●	●	●	●
RIC	●	●	●	●	●	●	●	●
RINDEX	●	●	●	●	●	●	●	●
RMB	●	●	●	●	●	●	●	●
RME	●	●	●	●	●	●	●	●
RMI	●	●	●	●	●	●	●	●
RMN	●	●	●	●	●	●	●	●
RND	●	●	●	●	●	●	●	●
RNDM	●	●	●	●	●	●	●	●
ROT	●	●	●	●	●	●	●	●
ROTS	●	●	●	●	●	●	●	●
ROUND	●	●	●	●	●	●	●	●
ROUNDUP	●	●	●	●	●	●	●	●
RP	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
RPL	●	●	●	●	●	●	●	●
RT	●	●	●	●	●	●	●	●
RTLIOF	●	●	●	●	●	●	●	●
RTLION	●	●	●	●	●	●	●	●

指令 S - Z

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
S	●	●	●	●	●	●	●	●
SAVE	●	●	●	●	●	●	●	●
SBLOF	●	●	●	●	●	●	●	●
SBLON	●	●	●	●	●	●	●	●
SC	●	●	●	●	●	●	●	●
SCALE	●	●	●	●	●	●	●	●
SCC	●	●	●	●	●	●	●	●
SCPARA	●	●	●	●	●	●	●	●
SD	○	○	○	○	○	○	○	○
SET	●	●	●	●	●	●	●	●
SETAL	●	●	●	●	●	●	●	●
SETDNO	●	●	●	●	●	●	●	●
SETINT	●	●	●	●	●	●	●	●
SETM	-	-	-	-	-	○	-	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
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	-	-	-	-	-	-	-	-

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
SPOF	-	-	-	-	-	-	-	-
SPOS	●	●	●	●	●	●	●	●
SPOSA	●	●	●	●	●	●	●	●
SPP	-	-	-	-	-	-	-	-
SPRINT	●	●	●	●	●	●	●	●
SQRT	●	●	●	●	●	●	●	●
SR	●	●	●	●	●	●	●	●
SRA	●	●	●	●	●	●	●	●
ST	●	●	●	●	●	●	●	●
STA	●	●	●	●	●	●	●	●
START	-	-	-	-	-	○	-	●
STARTFIFO	●	●	●	●	●	●	●	●
STAT	●	●	●	●	●	●	●	●
STOLF	●	●	●	●	●	●	●	●
STOPFIFO	●	●	●	●	●	●	●	●
STOPRE	●	●	●	●	●	●	●	●
STOPREOF	●	●	●	●	●	●	●	●
STRING	●	●	●	●	●	●	●	●
STRINGFELD	●	●	●	●	●	●	●	●
STRINGIS	●	●	●	●	●	●	●	●
STRLEN	●	●	●	●	●	●	●	●
SUBSTR	●	●	●	●	●	●	●	●
SUPA	●	●	●	●	●	●	●	●
SVC	●	●	●	●	●	●	●	●
SYNFCT	●	●	●	●	●	●	●	●
SYNR	●	●	●	●	●	●	●	●
SYNRW	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
SYNW	●	●	●	●	●	●	●	●
T	●	●	●	●	●	●	●	●
TAN	●	●	●	●	●	●	●	●
TANG	-	-	-	-	-	-	-	-
TANGDEL	-	-	-	-	-	-	-	-
TANGOF	-	-	-	-	-	-	-	-
TANGON	-	-	-	-	-	-	-	-
TCA (828D: _TCA)	●	●	●	●	●	●	●	●
TCARR	●	-	●	-	●	●	-	-
TCI	●	●	●	●	●	●	●	●
TCOABS	●	-	●	-	●	●	-	-
TCOFR	●	-	●	-	●	●	-	-
TCOFRX	●	-	●	-	●	●	-	-
TCOFRY	●	-	●	-	●	●	-	-
TCOFRZ	●	-	●	-	●	●	-	-
THETA	-	-	-	-	-	-	-	-
TILT	-	-	-	-	-	-	-	-
TLIFT	-	-	-	-	-	-	-	-
TML	●	●	●	●	●	●	●	●
TMOF	●	●	●	●	●	●	●	●
TMON	●	●	●	●	●	●	●	●
TO	●	●	●	●	●	●	●	●
TOFF	●	●	●	●	●	●	●	●
TOFFL	●	●	●	●	●	●	●	●
TOFFOF	●	●	●	●	●	●	●	●
TOFFON	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
TOFFR	●	●	●	●	●	●	●	●
TOFRAME	●	●	●	●	●	●	●	●
TOFRAMEX	●	●	●	●	●	●	●	●
TOFRAMEY	●	●	●	●	●	●	●	●
TOFRAMEZ	●	●	●	●	●	●	●	●
TOLOWER	●	●	●	●	●	●	●	●
TOOLENV	●	●	●	●	●	●	●	●
TOOLGNT	●	●	●	●	●	●	●	●
TOOLGT	●	●	●	●	●	●	●	●
TOROT	●	●	●	●	●	●	●	●
TOROTOF	●	●	●	●	●	●	●	●
TOROTX	●	●	●	●	●	●	●	●
TOROTY	●	●	●	●	●	●	●	●
TOROTZ	●	●	●	●	●	●	●	●
TOUPPER	●	●	●	●	●	●	●	●
TOWBCS	●	-	●	-	●	●	-	-
TOWKCS	●	-	●	-	●	●	-	-
TOWMCS	●	-	●	-	●	●	-	-
TOWSTD	●	-	●	-	●	●	-	-
TOWTCS	●	-	●	-	●	●	-	-
TOWWCS	●	-	●	-	●	●	-	-
TR	●	●	●	●	●	●	●	●
TRAANG	-	-	-	-	-	-	○	○
TRACON	-	-	-	-	-	-	○	○
TRACYL	○	○	○	○	○	○	○	○
TRAFOOF	●	●	●	●	●	●	●	●
TRAFOON	-	-	-	-	-	-	-	-

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口 版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口 版本 (te822)
TRAILOF	●	●	●	●	●	●	●	●
TRAILON	●	●	●	●	●	●	●	●
TRANS	●	●	●	●	●	●	●	●
TRANSMIT	○	○	○	○	○	○	○	○
TRAORI	-	-	-	-	-	-	-	-
TRUE	●	●	●	●	●	●	●	●
TRUNC	●	●	●	●	●	●	●	●
TU	●	●	●	●	●	●	●	●
TURN	●	●	●	●	●	●	●	●
ULI	●	●	●	●	●	●	●	●
UNLOCK	●	●	●	●	●	●	●	●
UNTIL	●	●	●	●	●	●	●	●
UPATH	●	●	●	●	●	●	●	●
VAR	●	●	●	●	●	●	●	●
VELOLIM	●	●	●	●	●	●	●	●
VELOLIMA	●	●	●	●	●	●	●	●
WAITC	●	●	●	●	●	●	●	●
WAITE	-	-	-	-	-	○	-	●
WAITENC	●	●	●	●	●	●	●	●
WAITM	-	-	-	-	-	○	-	●
WAITMC	-	-	-	-	-	○	-	●
WAITP	●	●	●	●	●	●	●	●
WAITS	●	●	●	●	●	●	●	●
WALCS0	●	●	●	●	●	●	●	●
WALCS1	●	●	●	●	●	●	●	●
WALCS2	●	●	●	●	●	●	●	●
WALCS3	●	●	●	●	●	●	●	●

指令	SINUMERIK 828D							
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 铣削 出口版本 (me42)	SW24x(5) CNC-SW 车削 出口版本 (te42)	SW26x(3) CNC-SW 铣削 出口版本 (me62)	SW26x(3) CNC-SW 车削 出口版本 (te62)	SW28x(2) CNC-SW 铣削 出口版本 (me821)	SW28x(1) CNC-SW 铣削 高级出口版本 (me822)	SW28x(2) CNC-SW 车削 出口版本 (te821)	SW28x(1) CNC-SW 车削 高级出口版本 (te822)
WALCS4	●	●	●	●	●	●	●	●
WALCS5	●	●	●	●	●	●	●	●
WALCS6	●	●	●	●	●	●	●	●
WALCS7	●	●	●	●	●	●	●	●
WALCS8	●	●	●	●	●	●	●	●
WALCS9	●	●	●	●	●	●	●	●
WALCS10	●	●	●	●	●	●	●	●
WALIMOF	●	●	●	●	●	●	●	●
WALIMON	●	●	●	●	●	●	●	●
WHEN	●	●	●	●	●	●	●	●
WHENEVER	●	●	●	●	●	●	●	●
WHILE	●	●	●	●	●	●	●	●
WORKPIECE	●	●	●	●	●	●	●	●
WRITE	●	●	●	●	●	●	●	●
WRTPR	●	●	●	●	●	●	●	●
X	●	●	●	●	●	●	●	●
XOR	●	●	●	●	●	●	●	●
Y	●	●	●	●	●	●	●	●
Z	●	●	●	●	●	●	●	●

21.2.2 控制系统类型 磨削

指令 A - C

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
:	●	●	●	●	●	●
*	●	●	●	●	●	●
+	●	●	●	●	●	●
-	●	●	●	●	●	●
<	●	●	●	●	●	●
<<	●	●	●	●	●	●
<=	●	●	●	●	●	●
=	●	●	●	●	●	●
>=	●	●	●	●	●	●
/	●	●	●	●	●	●
/0 … /7	●	●	●	●	●	●
A	●	●	●	●	●	●
A2	-	-	-	-	-	-
A3	-	-	-	-	-	-
A4	-	-	-	-	-	-
A5	-	-	-	-	-	-
A6	-	-	-	-	-	-
A7	-	-	-	-	-	-
ABS	●	●	●	●	●	●
AC	●	●	●	●	●	●
ACC	●	●	●	●	●	●
ACCLIMA	●	●	●	●	●	●
ACN	●	●	●	●	●	●

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
ACOS	●	●	●	●	●	●
ACP	●	●	●	●	●	●
ACTBLOCNO	●	●	●	●	●	●
ADDFRAME	●	●	●	●	●	●
ADIS	●	●	●	●	●	●
ADISPOS	●	●	●	●	●	●
ADISPOSA	●	●	●	●	●	●
ALF	●	●	●	●	●	●
AMIRROR	●	●	●	●	●	●
AND	●	●	●	●	●	●
ANG	●	●	●	●	●	●
AP	●	●	●	●	●	●
APR	●	●	●	●	●	●
APRB	●	●	●	●	●	●
APRP	●	●	●	●	●	●
APW	●	●	●	●	●	●
APWB	●	●	●	●	●	●
APWP	●	●	●	●	●	●
APX	●	●	●	●	●	●
AR	●	●	●	●	●	●
AROT	●	●	●	●	●	●
AROTS	●	●	●	●	●	●
AS	●	●	●	●	●	●
ASCALE	●	●	●	●	●	●
ASIN	●	●	●	●	●	●
ASPLINE	○	○	○	○	○	○
ATAN2	●	●	●	●	●	●

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
ATOL	●	●	●	●	●	●
ATRANS	●	●	●	●	●	●
AUXFUDEL	●	●	●	●	●	●
AUXFUDELG	●	●	●	●	●	●
AUXFUMSEQ	●	●	●	●	●	●
AUXFUSYNC	●	●	●	●	●	●
AX	●	●	●	●	●	●
AXCTSWE	-	-	-	-	-	-
AXCTSWEC	-	-	-	-	-	-
AXCTSWED	-	-	-	-	-	-
AXIS	●	●	●	●	●	●
AXNAME	●	●	●	●	●	●
AXSTRING	●	●	●	●	●	●
AXTOCHAN	●	●	●	●	●	●
AXTOSPI	●	●	●	●	●	●
B	●	●	●	●	●	●
B2	-	-	-	-	-	-
B3	-	-	-	-	-	-
B4	-	-	-	-	-	-
B5	-	-	-	-	-	-
B6	-	-	-	-	-	-
B7	-	-	-	-	-	-
B_AND	●	●	●	●	●	●
B_OR	●	●	●	●	●	●
B_NOT	●	●	●	●	●	●
B_XOR	●	●	●	●	●	●
BAUTO	○	○	○	○	○	○

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
BLOCK	●	●	●	●	●	●
BLSYNC	●	●	●	●	●	●
BNAT	○	○	○	○	○	○
BOOL	●	●	●	●	●	●
BOUND	●	●	●	●	●	●
BRISK	●	●	●	●	●	●
BRISKA	●	●	●	●	●	●
BSPLINE	○	○	○	○	○	○
BTAN	○	○	○	○	○	○
C	●	●	●	●	●	●
C2	-	-	通道轴名称	-	-	-
C3	-	-	-	-	-	-
C4	-	-	-	-	-	-
C5	-	-	-	-	-	-
C6	-	-	-	-	-	-
C7	-	-	-	-	-	-
CAC	●	●	●	●	●	●
CACN	●	●	●	●	●	●
CACP	●	●	●	●	●	●
CALCDAT	●	●	●	●	●	●
CALCPOSI	●	●	●	●	●	●
CALL	●	●	●	●	●	●
CALLPATH	●	●	●	●	●	●
CANCEL	●	●	●	●	●	●
CASE	●	●	●	●	●	●
CDC	●	●	●	●	●	●
CDOF	-	-	-	-	-	-

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
CDOF2	-	-	-	-	-	-
CDON	-	-	-	-	-	-
CFC	●	●	●	●	●	●
CFIN	●	●	●	●	●	●
CFINE	●	●	●	●	●	●
CFTCP	●	●	●	●	●	●
CHAN	●	●	●	●	●	●
CHANDATA	●	●	●	●	●	●
CHAR	●	●	●	●	●	●
CHF	●	●	●	●	●	●
CHKDM	●	●	●	●	●	●
CHKDNO	●	●	●	●	●	●
CHR	●	●	●	●	●	●
CIC	●	●	●	●	●	●
CIP	●	●	●	●	●	●
CLEARM	-	-	●	-	-	●
CLRINT	●	●	●	●	●	●
CMIRROR	●	●	●	●	●	●
COARSEA	●	●	●	●	●	●
COLLPAIR	-	-	-	-	-	-
COMPCAD	●	●	●	●	●	●
COMPCURV	●	●	●	●	●	●
COMPLETE	●	●	●	●	●	●
COMPOF	●	●	●	●	●	●
COMPON	●	●	●	●	●	●
COMPSURF	-	-	-	-	-	-
CONTDCON	●	●	●	●	●	●

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
CONTPRON	●	●	●	●	●	●
CORROF	●	●	●	●	●	●
CORRTRAFO	-	-	-	-	-	-
COS	●	●	●	●	●	●
COUPDEF	○	○	○	○	○	○
COUPDEL	○	○	○	○	○	○
COUPOF	○	○	○	○	○	○
COUPOFS	○	○	○	○	○	○
COUPON	○	○	○	○	○	○
COUPONC	○	○	○	○	○	○
COUPRES	○	○	○	○	○	○
CP	●	●	●	●	●	●
CPBC	○	○	○	○	○	○
CPDEF	○	○	○	○	○	○
CPDEL	○	○	○	○	○	○
CPFMOF	○	○	○	○	○	○
CPFMON	○	○	○	○	○	○
CPFMSON	○	○	○	○	○	○
CPFPOS	○	○	○	○	○	○
CPFRS	○	○	○	○	○	○
CPLA	○	○	○	○	○	○
CPLCTID	○	○	○	○	○	○
CPLDEF	○	○	○	○	○	○
CPLDEL	○	○	○	○	○	○
CPLDEN	○	○	○	○	○	○
CPLINSC	○	○	○	○	○	○
CPLINTR	○	○	○	○	○	○

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
CPLNUM	○	○	○	○	○	○
CPLOF	○	○	○	○	○	○
CPLON	○	○	○	○	○	○
CPLOUTSC	○	○	○	○	○	○
CPLOUTTR	○	○	○	○	○	○
CPLPOS	○	○	○	○	○	○
CPLSETVAL	○	○	○	○	○	○
CPMALARM	○	○	○	○	○	○
CPMBRAKE	○	○	○	○	○	○
CPMPRT	○	○	○	○	○	○
CPMRESET	○	○	○	○	○	○
CPMSTART	○	○	○	○	○	○
CPMVDI	○	○	○	○	○	○
CPOF	○	○	○	○	○	○
CPON	○	○	○	○	○	○
CPRECOF	●	●	●	●	●	●
CPRECON	●	●	●	●	●	●
CPRES	○	○	○	○	○	○
CPROT	●	●	●	●	●	●
CPROTDEF	●	●	●	●	●	●
CPSETTYPE	○	○	○	○	○	○
CPSYNCOP	○	○	○	○	○	○
CPSYNCOP2	○	○	○	○	○	○
CPSYNCOV	○	○	○	○	○	○
CPSYNFIP	○	○	○	○	○	○
CPSYNFIP2	○	○	○	○	○	○
CPSYNFIV	○	○	○	○	○	○

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
CR	●	●	●	●	●	●
CROT	●	●	●	●	●	●
CROTS	●	●	●	●	●	●
CRPL	●	●	●	●	●	●
CSCALE	●	●	●	●	●	●
CSPLINE	○	○	○	○	○	○
CT	●	●	●	●	●	●
CTAB	-	-	-	-	-	-
CTABDEF	-	-	-	-	-	-
CTABDEL	-	-	-	-	-	-
CTABEND	-	-	-	-	-	-
CTABEXISTS	-	-	-	-	-	-
CTABFNO	-	-	-	-	-	-
CTABFPOL	-	-	-	-	-	-
CTABFSEG	-	-	-	-	-	-
CTABID	-	-	-	-	-	-
CTABINV	-	-	-	-	-	-
CTABISLOCK	-	-	-	-	-	-
CTABLOCK	-	-	-	-	-	-
CTABMEMTYP	-	-	-	-	-	-
CTABMPOL	-	-	-	-	-	-
CTABMSEG	-	-	-	-	-	-
CTABNO	-	-	-	-	-	-
CTABNOMEM	-	-	-	-	-	-
CTABPERIOD	-	-	-	-	-	-
CTABPOL	-	-	-	-	-	-
CTABPOLID	-	-	-	-	-	-

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
CTABSEG	-	-	-	-	-	-
CTABSEGID	-	-	-	-	-	-
CTABSEV	-	-	-	-	-	-
CTABSSV	-	-	-	-	-	-
CTABTEP	-	-	-	-	-	-
CTABTEV	-	-	-	-	-	-
CTABTMAX	-	-	-	-	-	-
CTABTMIN	-	-	-	-	-	-
CTABTSP	-	-	-	-	-	-
CTABTSV	-	-	-	-	-	-
CTABUNLOCK	-	-	-	-	-	-
CTOL	●	●	●	●	●	●
CTRANS	●	●	●	●	●	●
CUT2D	●	●	●	●	●	●
CUT2DD	●	●	●	●	●	●
CUT2DF	●	●	●	●	●	●
CUT2DFD	●	●	●	●	●	●
CUT3DC	-	-	-	-	-	-
CUT3DCC	-	-	-	-	-	-
CUT3DCCD	-	-	-	-	-	-
CUT3DCD	-	-	-	-	-	-
CUT3DF	-	-	-	-	-	-
CUT3DFD	-	-	-	-	-	-
CUT3DFF	-	-	-	-	-	-
CUT3DFS	-	-	-	-	-	-
CUTCONOF	●	●	●	●	●	●
CUTCONON	●	●	●	●	●	●

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
CUTMOD	●	●	●	●	●	●
CUTMODK	-	-	-	-	-	-
CYCLE60	-	-	-	-	-	-
CYCLE61	-	-	-	-	-	-
CYCLE62	●	●	●	●	●	●
CYCLE63	-	-	-	-	-	-
CYCLE64	-	-	-	-	-	-
CYCLE70	-	-	-	-	-	-
CYCLE72	-	-	-	-	-	-
CYCLE76	-	-	-	-	-	-
CYCLE77	-	-	-	-	-	-
CYCLE78	-	-	-	-	-	-
CYCLE79	-	-	-	-	-	-
CYCLE81	-	-	-	-	-	-
CYCLE82	-	-	-	-	-	-
CYCLE83	-	-	-	-	-	-
CYCLE84	-	-	-	-	-	-
CYCLE85	-	-	-	-	-	-
CYCLE86	-	-	-	-	-	-
CYCLE92	-	-	-	-	-	-
CYCLE95	-	-	-	-	-	-
CYCLE98	-	-	-	-	-	-
CYCLE99	-	-	-	-	-	-
CYCLE150	-	-	-	-	-	-
CYCLE435	○	○	○	○	○	○
CYCLE495	○	○	○	○	○	○
CYCLE750	●	●	●	●	●	●

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
CYCLE751	●	●	●	●	●	●
CYCLE752	●	●	●	●	●	●
CYCLE753	●	●	●	●	●	●
CYCLE754	●	●	●	●	●	●
CYCLE755	●	●	●	●	●	●
CYCLE756	●	●	●	●	●	●
CYCLE757	●	●	●	●	●	●
CYCLE758	●	●	●	●	●	●
CYCLE759	●	●	●	●	●	●
CYCLE800	○	○	○	○	○	○
CYCLE801	-	-	-	-	-	-
CYCLE802	-	-	-	-	-	-
CYCLE830	-	-	-	-	-	-
CYCLE832	●	●	●	●	●	●
CYCLE840	-	-	-	-	-	-
CYCLE899	-	-	-	-	-	-
CYCLE930	-	-	-	-	-	-
CYCLE940	-	-	-	-	-	-
CYCLE951	-	-	-	-	-	-
CYCLE952	-	-	-	-	-	-
CYCLE961	-	-	-	-	-	-
CYCLE971	-	-	-	-	-	-
CYCLE973	-	-	-	-	-	-
CYCLE974	-	-	-	-	-	-
CYCLE976	-	-	-	-	-	-
CYCLE977	-	-	-	-	-	-
CYCLE978	-	-	-	-	-	-

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
CYCLE979	-	-	-	-	-	-
CYCLE982	-	-	-	-	-	-
CYCLE994	-	-	-	-	-	-
CYCLE995	-	-	-	-	-	-
CYCLE996	-	-	-	-	-	-
CYCLE997	-	-	-	-	-	-
CYCLE998	-	-	-	-	-	-
CYCLE4071	●	●	●	●	●	●
CYCLE4072	●	●	●	●	●	●
CYCLE4073	●	●	●	●	●	●
CYCLE4074	●	●	●	●	●	●
CYCLE4075	●	●	●	●	●	●
CYCLE4077	●	●	●	●	●	●
CYCLE4078	●	●	●	●	●	●
CYCLE4079	●	●	●	●	●	●

指令 D - F

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
D	●	●	●	●	●	●
D0	●	●	●	●	●	●
DAC	●	●	●	●	●	●

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
DC	●	●	●	●	●	●
DCI	●	●	●	●	●	●
DCM	●	●	●	●	●	●
DCU	●	●	●	●	●	●
DEF	●	●	●	●	●	●
DEFINE	●	●	●	●	●	●
DEFAULT	●	●	●	●	●	●
DELAYFSTON	●	●	●	●	●	●
DELAYFSTOF	●	●	●	●	●	●
DELDL	●	●	●	●	●	●
DELDTG	●	●	●	●	●	●
DELETE	●	●	●	●	●	●
DELMLOWNER	●	●	●	●	●	●
DELMLRES	●	●	●	●	●	●
DELMT	-	-	-	-	-	-
DELOBJ	-	-	-	-	-	-
DELT	●	●	●	●	●	●
DELTC	●	●	●	●	●	●
DELTOOLENV	●	●	●	●	●	●
DIACYCOFA	●	●	●	●	●	●
DIAM90	●	●	●	●	●	●
DIAM90A	●	●	●	●	●	●
DIAMCHAN	●	●	●	●	●	●
DIAMCHANA	●	●	●	●	●	●
DIAMCYCOF	●	●	●	●	●	●
DIAMOF	●	●	●	●	●	●
DIAMOFA	●	●	●	●	●	●

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
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	○	○	○	○	○	○

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
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	●	●	●	●	●	●

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
EXTERN	●	●	●	●	●	●
EXTOPEN	●	●	●	●	●	●
F	●	●	●	●	●	●
FA	●	●	●	●	●	●
FAD	●	●	●	●	●	●
FALSE	●	●	●	●	●	●
FB	●	●	●	●	●	●
FCTDEF	●	●	●	●	●	●
FCUB	●	●	●	●	●	●
FD	●	●	●	●	●	●
FDA	●	●	●	●	●	●
FENDNORM	●	●	●	●	●	●
FFWOF	●	●	●	●	●	●
FFWON	●	●	●	●	●	●
FGREF	●	●	●	●	●	●
FGROUP	●	●	●	●	●	●
FI	●	●	●	●	●	●
FIFOCTRL	●	●	●	●	●	●
FILEDATE	●	●	●	●	●	●
FILEINFO	●	●	●	●	●	●
FILESIZE	●	●	●	●	●	●
FILESTAT	●	●	●	●	●	●
FILETIME	●	●	●	●	●	●
FINEA	●	●	●	●	●	●
FL	●	●	●	●	●	●
FLIN	●	●	●	●	●	●
FMA	●	●	●	●	●	●

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
FNORM	●	●	●	●	●	●
FOCOF	○	○	○	○	○	○
FOCON	○	○	○	○	○	○
FOR	●	●	●	●	●	●
FP	●	●	●	●	●	●
FPO	-	-	-	-	-	-
FPR	●	●	●	●	●	●
FPRAOF	●	●	●	●	●	●
FPRAON	●	●	●	●	●	●
FRAME	●	●	●	●	●	●
FRC	●	●	●	●	●	●
FRCM	●	●	●	●	●	●
FROM	●	●	●	●	●	●
FTOC	●	●	●	●	●	●
FTOCOF	●	●	●	●	●	●
FTOCON	●	●	●	●	●	●
FXS	●	●	●	●	●	●
FXST	●	●	●	●	●	●
FXSW	●	●	●	●	●	●
FZ	●	●	●	●	●	●

指令 G - L

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
G0	●	●	●	●	●	●
G1	●	●	●	●	●	●
G2	●	●	●	●	●	●
G3	●	●	●	●	●	●
G4	●	●	●	●	●	●
G5	●	●	●	●	●	●
G7	●	●	●	●	●	●
G9	●	●	●	●	●	●
G17	●	●	●	●	●	●
G18	●	●	●	●	●	●
G19	●	●	●	●	●	●
G25	●	●	●	●	●	●
G26	●	●	●	●	●	●
G33	●	●	●	●	●	●
G34	●	●	●	●	●	●
G35	●	●	●	●	●	●
G40	●	●	●	●	●	●
G41	●	●	●	●	●	●
G42	●	●	●	●	●	●
G53	●	●	●	●	●	●
G54	●	●	●	●	●	●
G55	●	●	●	●	●	●
G56	●	●	●	●	●	●
G57	●	●	●	●	●	●
G58	→ G505					

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
G59	→ G506					
G60	●	●	●	●	●	●
G62	●	●	●	●	●	●
G63	●	●	●	●	●	●
G64	●	●	●	●	●	●
G70	●	●	●	●	●	●
G71	●	●	●	●	●	●
G74	●	●	●	●	●	●
G75	●	●	●	●	●	●
G90	●	●	●	●	●	●
G91	●	●	●	●	●	●
G93	●	●	●	●	●	●
G94	●	●	●	●	●	●
G95	●	●	●	●	●	●
G96	●	●	●	●	●	●
G97	●	●	●	●	●	●
G110	●	●	●	●	●	●
G111	●	●	●	●	●	●
G112	●	●	●	●	●	●
G140	●	●	●	●	●	●
G141	●	●	●	●	●	●
G142	●	●	●	●	●	●
G143	●	●	●	●	●	●
G147	●	●	●	●	●	●
G148	●	●	●	●	●	●
G153	●	●	●	●	●	●
G247	●	●	●	●	●	●

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
G248	●	●	●	●	●	●
G290	●	●	●	●	●	●
G291	-	-	-	-	-	-
G331	●	●	●	●	●	●
G332	●	●	●	●	●	●
G335	●	●	●	●	●	●
G336	●	●	●	●	●	●
G340	●	●	●	●	●	●
G341	●	●	●	●	●	●
G347	●	●	●	●	●	●
G348	●	●	●	●	●	●
G450	●	●	●	●	●	●
G451	●	●	●	●	●	●
G460	●	●	●	●	●	●
G461	●	●	●	●	●	●
G462	●	●	●	●	●	●
G500	●	●	●	●	●	●
G505...G599	●	●	●	●	●	●
G601	●	●	●	●	●	●
G602	●	●	●	●	●	●
G603	●	●	●	●	●	●
G621	●	●	●	●	●	●
G641	●	●	●	●	●	●
G642	●	●	●	●	●	●
G643	●	●	●	●	●	●
G644	●	●	●	●	●	●
G645	●	●	●	●	●	●

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
G700	●	●	●	●	●	●
G710	●	●	●	●	●	●
G810...G819	-	-	-	-	-	-
G820...G829	-	-	-	-	-	-
G931	●	●	●	●	●	●
G942	●	●	●	●	●	●
G952	●	●	●	●	●	●
G961	●	●	●	●	●	●
G962	●	●	●	●	●	●
G971	●	●	●	●	●	●
G972	●	●	●	●	●	●
G973	●	●	●	●	●	●
GEOAX	●	●	●	●	●	●
GET	●	●	●	●	●	●
GETACTT	●	●	●	●	●	●
GETACTTD	●	●	●	●	●	●
GETD	-	-	●	-	-	●
GETDNO	●	●	●	●	●	●
GETEXET	●	●	●	●	●	●
GETFREELOC	●	●	●	●	●	●
GETSELT	●	●	●	●	●	●
GETT	●	●	●	●	●	●
GETTCOR	●	●	●	●	●	●
GETTENV	●	●	●	●	●	●
GETVARAP	●	●	●	●	●	●
GETVARDFT	●	●	●	●	●	●
GETVARLIM	●	●	●	●	●	●

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
GETVARPHU	●	●	●	●	●	●
GETVARTYP	●	●	●	●	●	●
GFRAME0 ... GFRAME100	< 50	< 100	< 100	< 50	< 100	< 100
GOTO	●	●	●	●	●	●
GOTOB	●	●	●	●	●	●
GOTOC	●	●	●	●	●	●
GOTOF	●	●	●	●	●	●
GOTOS	●	●	●	●	●	●
GP	●	●	●	●	●	●
GWPSOF	●	●	●	●	●	●
GROUP_ADDEND	●	●	●	●	●	●
GROUP_BEGIN	●	●	●	●	●	●
GROUP_END	●	●	●	●	●	●
GWPSON	●	●	●	●	●	●
H...	●	●	●	●	●	●
HOLES1	-	-	-	-	-	-
HOLES2	-	-	-	-	-	-
I	●	●	●	●	●	●
I1	●	●	●	●	●	●
IC	●	●	●	●	●	●
ICYCOF	●	●	●	●	●	●
ICYCON	●	●	●	●	●	●
ID	●	●	●	●	●	●
IDS	●	●	●	●	●	●
IF	●	●	●	●	●	●
INDEX	●	●	●	●	●	●

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
INIPO	●	●	●	●	●	●
INIRE	●	●	●	●	●	●
INICF	●	●	●	●	●	●
INIT	-	-	●	-	-	●
INITIAL						
INT	●	●	●	●	●	●
INTERSEC	●	●	●	●	●	●
INVCCW	-	-	-	-	-	-
INVCW	-	-	-	-	-	-
INVFRAME	●	●	●	●	●	●
IP	●	●	●	●	●	●
IPOBRKA	●	●	●	●	●	●
IPOENDA	●	●	●	●	●	●
IPTRLOCK	●	●	●	●	●	●
IPTRUNLOCK	●	●	●	●	●	●
IR	●	●	●	●	●	●
ISAXIS	●	●	●	●	●	●
ISD	-	-	-	-	-	-
ISFILE	●	●	●	●	●	●
ISNUMBER	●	●	●	●	●	●
ISOCALL	-	-	-	-	-	-
ISVAR	●	●	●	●	●	●
J	●	●	●	●	●	●
J1	●	●	●	●	●	●
JERKA	●	●	●	●	●	●
JERKLIM	●	●	●	●	●	●
JERKLIMA	●	●	●	●	●	●

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
JR	●	●	●	●	●	●
K	●	●	●	●	●	●
K1	●	●	●	●	●	●
KONT	●	●	●	●	●	●
KONTC	●	●	●	●	●	●
KONTT	●	●	●	●	●	●
KR	●	●	●	●	●	●
L	●	●	●	●	●	●
LEAD 刀具定向 定向多项式	 - -	 - -	 - -	 - -	 - -	 - -
LEADOF	-	-	-	-	-	-
LEADON	-	-	-	-	-	-
LENTOAX	●	●	●	●	●	●
LFOF	●	●	●	●	●	●
LFON	●	●	●	●	●	●
LFPOS	●	●	●	●	●	●
LFTXT	●	●	●	●	●	●
LFWP	●	●	●	●	●	●
LIFTFAST	●	●	●	●	●	●
LIMS	●	●	●	●	●	●
LLI	●	●	●	●	●	●
LN	●	●	●	●	●	●
LOCK	●	●	●	●	●	●
LONGHOLE	-	-	-	-	-	-
LOOP	●	●	●	●	●	●

指令 M - R

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
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	-	○	○	-	○	○

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
MEASURE	●	●	●	●	●	●
MEAW	●	●	●	●	●	●
MEAWA	-	○	○	-	○	○
MI	●	●	●	●	●	●
MINDEX	●	●	●	●	●	●
MINVAL	●	●	●	●	●	●
MIRROR	●	●	●	●	●	●
MMC	●	●	●	●	●	●
MOD	●	●	●	●	●	●
MODAXVAL	●	●	●	●	●	●
MOV	●	●	●	●	●	●
MOVT	●	●	●	●	●	●
MSG	●	●	●	●	●	●
MVTOOL	●	●	●	●	●	●
N	●	●	●	●	●	●
NAMETOINT	●	●	●	●	●	●
NCK	●	●	●	●	●	●
NEWCONF	●	●	●	●	●	●
NEWMT	●	●	●	●	●	●
NEWT	-	-	-	-	-	-
NORM	●	●	●	●	●	●
NOT	●	●	●	●	●	●
NPROT	●	●	●	●	●	●
NPROTDEF	●	●	●	●	●	●
NUMBER	●	●	●	●	●	●
OEMIPO1	-	-	-	-	-	-
OEMIPO2	-	-	-	-	-	-

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
OF	●	●	●	●	●	●
OFFN	●	●	●	●	●	●
OMA1	-	-	-	-	-	-
OMA2	-	-	-	-	-	-
OMA3	-	-	-	-	-	-
OMA4	-	-	-	-	-	-
OMA5	-	-	-	-	-	-
OR	●	●	●	●	●	●
ORIAXES	-	-	-	-	-	-
ORIAXPOS	-	-	-	-	-	-
ORIC	-	-	-	-	-	-
ORICONCCW	-	-	-	-	-	-
ORICONCW	-	-	-	-	-	-
ORICONIO	-	-	-	-	-	-
ORICONTO	-	-	-	-	-	-
ORICURVE	-	-	-	-	-	-
ORID	-	-	-	-	-	-
ORIEULER	-	-	-	-	-	-
ORIMKS	-	-	-	-	-	-
ORIPATH	-	-	-	-	-	-
ORIPATHS	-	-	-	-	-	-
ORIPLANE	-	-	-	-	-	-
ORIRESET	-	-	-	-	-	-
ORIROTA	-	-	-	-	-	-
ORIROTC	-	-	-	-	-	-
ORIROTR	-	-	-	-	-	-
ORIROTT	-	-	-	-	-	-

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
ORIRPY	-	-	-	-	-	-
ORIRPY2	-	-	-	-	-	-
ORIS	-	-	-	-	-	-
ORISOF	-	-	-	-	-	-
ORISOLH	-	-	-	-	-	-
ORISON	-	-	-	-	-	-
ORIVECT	-	-	-	-	-	-
ORIVIRT1	-	-	-	-	-	-
ORIVIRT2	-	-	-	-	-	-
ORIWKS	-	-	-	-	-	-
OS	○	○	○	○	○	○
OSB	○	○	○	○	○	○
OSC	-	-	-	-	-	-
OSCILL	○	○	○	○	○	○
OSCTRL	○	○	○	○	○	○
OSD	-	-	-	-	-	-
OSE	○	○	○	○	○	○
OSNSC	●	●	●	●	●	●
OSOF	-	-	-	-	-	-
OSP1	●	●	●	●	●	●
OSP2	●	●	●	●	●	●
OSS	-	-	-	-	-	-
OSSE	-	-	-	-	-	-
OST	-	-	-	-	-	-
OST1	●	●	●	●	●	●
OST2	●	●	●	●	●	●
OTOL	●	●	●	●	●	●

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
OVR	●	●	●	●	●	●
OVRA	●	●	●	●	●	●
OVRRAP	●	●	●	●	●	●
P	●	●	●	●	●	●
PAROT	●	●	●	●	●	●
PAROTOF	●	●	●	●	●	●
PCALL	●	●	●	●	●	●
PDELAYOF	-	-	-	-	-	-
PDELAYON	-	-	-	-	-	-
PHI	-	-	-	-	-	-
PHU	●	●	●	●	●	●
PL	-	-	-	-	-	-
PM	●	●	●	●	●	●
PO	-	-	-	-	-	-
POCKET3	-	-	-	-	-	-
POCKET4	-	-	-	-	-	-
POLF	●	●	●	●	●	●
POLFA	●	●	●	●	●	●
POLFMASK	●	●	●	●	●	●
POLFMLIN	●	●	●	●	●	●
POLY	-	-	-	-	-	-
POLYPATH	-	-	-	-	-	-
PON	-	-	-	-	-	-
PONS	-	-	-	-	-	-
POS	●	●	●	●	●	●
POSA	●	●	●	●	●	●
POSM	●	●	●	●	●	●

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
POSMT	-	-	-	-	-	-
POSP	●	●	●	●	●	●
POSRANGE	●	●	●	●	●	●
POT	●	●	●	●	●	●
PR	●	●	●	●	●	●
PREPRO	●	●	●	●	●	●
PRESETON	●	●	●	●	●	●
PRESETONS	●	●	●	●	●	●
PRIO	●	●	●	●	●	●
PRLOC	●	●	●	●	●	●
PROC	●	●	●	●	●	●
PROTA	●	●	●	●	●	●
PROTD	●	●	●	●	●	●
PROTS	●	●	●	●	●	●
PSI	-	-	-	-	-	-
PTP	●	●	●	●	●	●
PTPG0	●	●	●	●	●	●
点对点连接	●	●	●	●	●	●
PUNCHACC	-	-	-	-	-	-
PUTFTOC	●	●	●	●	●	●
PUTFTOCF	●	●	●	●	●	●
PW	○	○	○	○	○	○
QU	●	●	●	●	●	●
R...	●	●	●	●	●	●
RAC	●	●	●	●	●	●
RDISABLE	●	●	●	●	●	●
READ	●	●	●	●	●	●

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
REAL	●	●	●	●	●	●
RELEASE	●	●	●	●	●	●
REP	●	●	●	●	●	●
REPEAT	●	●	●	●	●	●
REPEATB	●	●	●	●	●	●
REPOSA	●	●	●	●	●	●
REPOSH	●	●	●	●	●	●
REPOSHA	●	●	●	●	●	●
REPOSL	●	●	●	●	●	●
REPOSQ	●	●	●	●	●	●
REPOSQA	●	●	●	●	●	●
RESETMON	●	●	●	●	●	●
RET	●	●	●	●	●	●
RETB	●	●	●	●	●	●
RIC	●	●	●	●	●	●
RINDEX	●	●	●	●	●	●
RMB	●	●	●	●	●	●
RME	●	●	●	●	●	●
RMI	●	●	●	●	●	●
RMN	●	●	●	●	●	●
RND	●	●	●	●	●	●
RNDM	●	●	●	●	●	●
ROT	●	●	●	●	●	●
ROTS	●	●	●	●	●	●
ROUND	●	●	●	●	●	●
ROUNDUP	●	●	●	●	●	●
RP	●	●	●	●	●	●

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
RPL	●	●	●	●	●	●
RT	●	●	●	●	●	●
RTLIOF	●	●	●	●	●	●
RTLION	●	●	●	●	●	●

指令 S - Z

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
S	●	●	●	●	●	●
SAVE	●	●	●	●	●	●
SBLOF	●	●	●	●	●	●
SBLON	●	●	●	●	●	●
SC	●	●	●	●	●	●
SCALE	●	●	●	●	●	●
SCC	●	●	●	●	●	●
SCPARA	●	●	●	●	●	●
SD	○	○	○	○	○	○
SET	●	●	●	●	●	●
SETAL	●	●	●	●	●	●
SETDNO	●	●	●	●	●	●
SETINT	●	●	●	●	●	●
SETM	-	-	●	-	-	●

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
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	-	-	-	-	-	-

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
SPOF	-	-	-	-	-	-
SPOS	●	●	●	●	●	●
SPOSA	●	●	●	●	●	●
SPP	-	-	-	-	-	-
SPRINT	●	●	●	●	●	●
SQRT	●	●	●	●	●	●
SR	●	●	●	●	●	●
SRA	●	●	●	●	●	●
ST	●	●	●	●	●	●
STA	●	●	●	●	●	●
START	-	-	●	-	-	●
STARTFIFO	●	●	●	●	●	●
STAT	●	●	●	●	●	●
STOLF	●	●	●	●	●	●
STOPFIFO	●	●	●	●	●	●
STOPRE	●	●	●	●	●	●
STOPREOF	●	●	●	●	●	●
STRING	●	●	●	●	●	●
STRINGFELD	●	●	●	●	●	●
STRINGIS	●	●	●	●	●	●
STRLEN	●	●	●	●	●	●
SUBSTR	●	●	●	●	●	●
SUPA	●	●	●	●	●	●
SVC	●	●	●	●	●	●
SYNFCT	●	●	●	●	●	●
SYNR	●	●	●	●	●	●
SYNRW	●	●	●	●	●	●

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
SYNW	●	●	●	●	●	●
T	●	●	●	●	●	●
TAN	●	●	●	●	●	●
TANG	○	○	○	○	○	○
TANGDEL	○	○	○	○	○	○
TANGOF	○	○	○	○	○	○
TANGON	○	○	○	○	○	○
TCA (828D: _TCA)	●	●	●	●	●	●
TCARR	●	●	●	●	●	●
TCI	●	●	●	●	●	●
TCOABS	●	●	●	●	●	●
TCOFR	●	●	●	●	●	●
TCOFRX	●	●	●	●	●	●
TCOFRY	●	●	●	●	●	●
TCOFRZ	●	●	●	●	●	●
THETA	-	-	-	-	-	-
TILT	-	-	-	-	-	-
TLIFT	○	○	○	○	○	○
TML	●	●	●	●	●	●
TMOF	●	●	●	●	●	●
TMON	●	●	●	●	●	●
TO	●	●	●	●	●	●
TOFF	●	●	●	●	●	●
TOFFL	●	●	●	●	●	●
TOFFOF	●	●	●	●	●	●
TOFFON	●	●	●	●	●	●

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
TOFFR	●	●	●	●	●	●
TOFRAME	●	●	●	●	●	●
TOFRAMEX	●	●	●	●	●	●
TOFRAMEY	●	●	●	●	●	●
TOFRAMEZ	●	●	●	●	●	●
TOLOWER	●	●	●	●	●	●
TOOLENV	●	●	●	●	●	●
TOOLGNT	●	●	●	●	●	●
TOOLGT	●	●	●	●	●	●
TOROT	●	●	●	●	●	●
TOROTOF	●	●	●	●	●	●
TOROTX	●	●	●	●	●	●
TOROTY	●	●	●	●	●	●
TOROTZ	●	●	●	●	●	●
TOUPPER	●	●	●	●	●	●
TOWBCS	●	●	●	●	●	●
TOWKCS	●	●	●	●	●	●
TOWMCS	●	●	●	●	●	●
TOWSTD	●	●	●	●	●	●
TOWTCS	●	●	●	●	●	●
TOWWCS	●	●	●	●	●	●
TR	●	●	●	●	●	●
TRAANG	○	○	○	-	-	-
TRACON	○	○	○	-	-	-
TRACYL	○	○	○	○	○	○
TRAFOOF	●	●	●	●	●	●
TRAFOON	-	-	-	-	-	-

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
TRAILOF	●	●	●	●	●	●
TRAILON	●	●	●	●	●	●
TRANS	●	●	●	●	●	●
TRANSMIT	○	○	○	○	○	○
TRAORI	-	-	-	-	-	-
TRUE	●	●	●	●	●	●
TRUNC	●	●	●	●	●	●
TU	●	●	●	●	●	●
TURN	●	●	●	●	●	●
ULI	●	●	●	●	●	●
UNLOCK	●	●	●	●	●	●
UNTIL	●	●	●	●	●	●
UPATH	●	●	●	●	●	●
VAR	●	●	●	●	●	●
VELOLIM	●	●	●	●	●	●
VELOLIMA	●	●	●	●	●	●
WAITC	●	●	●	●	●	●
WAITE	-	-	●	-	-	●
WAITENC	●	●	●	●	●	●
WAITM	-	-	●	-	-	●
WAITMC	-	-	●	-	-	●
WAITP	●	●	●	●	●	●
WAITS	●	●	●	●	●	●
WALCS0	●	●	●	●	●	●
WALCS1	●	●	●	●	●	●
WALCS2	●	●	●	●	●	●
WALCS3	●	●	●	●	●	●

指令	SINUMERIK 828D					
● 标配件 ○ 选件 - 不可用	SW24x(5) CNC-SW 磨削 出口版本 (gce42)	SW26x(3) CNC-SW 磨削 出口版本 (gce62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gce82)	SW24x(5) CNC-SW 磨削 出口版本 (gse42)	SW26x(3) CNC-SW 磨削 出口版本 (gse62)	SW28x(1) CNC-SW 磨削 高级出口版本 (gse82)
WALCS4	●	●	●	●	●	●
WALCS5	●	●	●	●	●	●
WALCS6	●	●	●	●	●	●
WALCS7	●	●	●	●	●	●
WALCS8	●	●	●	●	●	●
WALCS9	●	●	●	●	●	●
WALCS10	●	●	●	●	●	●
WALIMOF	●	●	●	●	●	●
WALIMON	●	●	●	●	●	●
WHEN	●	●	●	●	●	●
WHENEVER	●	●	●	●	●	●
WHILE	●	●	●	●	●	●
WORKPIECE	●	●	●	●	●	●
WRITE	●	●	●	●	●	●
WRTPR	●	●	●	●	●	●
X	●	●	●	●	●	●
XOR	●	●	●	●	●	●
Y	●	●	●	●	●	●
Z	●	●	●	●	●	●

21.3 HMI 上的当前语言

下表列出操作界面上提供的语言。

在零件程序和同步动作中，可以通过以下系统变量查询当前设置的语言：

$AN_LANGUAGE_ON_HMI = <值>

<值>	语言	语言缩写
1	德语（德国）	CHI
2	法语	FRA
3	英语（英国）	ENG
4	西班牙语	ESP
6	意大利语	ITA
7	荷兰语	NLD
8	中文（简体）	CHS
9	瑞典语	SVE
18	匈牙利语	HUN
19	芬兰语	FIN
28	捷克语	CSY
50	葡萄牙语（巴西）	PTB
53	波兰语	PLK
55	丹麦语	DAN
57	俄语	RUS
68	斯洛伐克语	SKY
72	罗马尼亚语	ROM
80	中文（繁体）	CHT
85	韩语	KOR
87	日语	JPN
89	土耳其语	TRK

说明

在以下情况下，$AN_LANGUAGE_ON_HMI 会更新：

- 系统启动后。
- NC 和/或 PLC 复位后。
- 在 M2N 框架内转换到另一个 NC 后。
- 在 HMI 上切换语言后。

附录 A

A.1 缩略语列表

A	
A	输出
ADI4	Analog Drive Interface for 4 Axes：4 轴的模拟驱动接口
AC	Adaptive Control：自适应控制
ALM	Active Line Module：调节型电源模块
ARM	异步旋转电机
AS	自动化系统
ASCII	American Standard Code for Information Interchange：美国信息互换标准码
ASIC	Application Specific Integrated Circuit：用户自行开发的专用集成电路
ASUP	异步子程序
AUXFU	Auxiliary Function：辅助功能
AWL	指令列表
AWP	用户程序

B	
BA	运行方式
BAG	运行方式组
BCD	Binary Coded Decimals：用二进制代码编码的十进制数
BERO	无接触接近开关
BI	Binector Input：二进制互联输入
BICO	Binector Connector
BIN	Binary Files：二进制文件
BIOS	Basic Input Output System：基本输入输出系统
BCS	基准坐标系

B	
BO	Binector Output：二进制互联输出
BTSS	操作面板接口

C	
CAD	Computer-Aided Design：计算机辅助设计
CAM	Computer-Aided Manufacturing：计算机辅助制造
CC	Compile Cycle：编译循环
CEC	交叉误差补偿
CI	Connector Input：模拟量互联输入
CF 卡	Compact Flash-Card：CF 卡
CNC	Computerized Numerical Control：计算机数字控制
CO	Connector Output：模拟量互联输出
CoL	Certificate of License：许可证书
COM	通讯
CPA	Compiler Projecting Data:编译器的项目数据
CRT	Cathode Ray Tube：阴极射线管
CSB	Central Service Board:PLC 模块
CU	Control Unit：控制单元
CP	Communication Processor：通讯处理器
CPU	Central Processing Unit：中央处理器
CR	Communication Processor（回车键）
CTS	Clear To Send：串行接口发送就绪状态
CUTCOM	Cutter Radius Compensation：刀具半径补偿

D	
DAU	数模转换器
DB	PLC 数据模块
DBB	数据块-字节（PLC）
DBD	数据块-双字（PLC）
DBW	数据块-字（PLC）

D	
DBX	数据块-位（PLC）
DDE	Dynamic Data Exchange：动态数据交换
DDS	Drive Data Set:驱动数据组
DIN	德国工业标准
DIO	Data Input/Output:数据传送显示
DIR	Directory：目录
DLL	Dynamic Link Library：动态链接库
DO	Drive Object：驱动对象
DPM	Dual Port Memory：双端口存储器
DPR	Dual Port RAM：双端口存储器
DRAM	动态存储器（未缓冲）
DRF	Differential Resolver Function：微分旋转变压器功能（手轮）
DRIVE-CLiQ	带 IQ 的驱动组件连接
DRY	Dry Run：空运行进给
DSB	Decoding Single Block：解码单程序段
DSC	Dynamic Servo Control / Dynamic Stiffness Control：动态伺服控制
DW	数据字
双字	双字（当前 32 位）

E	
E	输入
EES	从外部存储器执行
I/O	输入/输出
ENC	Encoder:实际值编码器
EFP	简易外设模块（PLC I/O 模块）
EGB	静电敏感元器件
EMC	电磁兼容性
EN	欧洲标准
ENC	Encoder:实际值编码器
EnDat	编码器接口

E	
EPROM	Erasable Programmable Read Only Memory：可删除、可编程的只读存储器
ePS 网络服务	以网络为基础的机床远程维护服务
EQN	2048 正弦信号/转绝对值编码器的类型名称
ES	Engineering System：工程系统
ESR	扩展的停止和退回
ETC	ETC 键“>”；相同菜单中的扩展软键栏

F	
FB	功能块（PLC）
FC	Function Call:功能块（PLC）
FEPROM	Flash-EPROM:可读可写存储器
FIFO	First In First Out：存储器，工作无需地址说明，数据按存储的顺序读入
FIPO	精插补器
FPU	Floating Point Unit：浮点单元
FRK	铣削半径补偿
FST	Feed Stop：进给停止
FUP	功能图（一种 PLC 编程方法）
FW	固件

G	
GC	全局控制（PROFIBUS：广播报文）
GDIR	全局零件程序存储器
GEO	几何属性，例如几何值
GIA	Gear Interpolation Data:齿轮插补数据
GND	Signal Ground：信号地
GP	基本程序 (PLC)
GS	齿轮级
GSD	设备主数据文件，用于说明 PROFIBUS 从站。+

G	
GSDML	Generic Station Description Markup Language：基于 XML 的描述语言，用于创建 GSD 文件
GUD	Global User Data:全局用户数据

H	
HEX	十六进制数代号
HiFu	辅助功能
HLA	液压直线驱动
HMI	Human Machine Interface：SINUMERIK 操作介面
HSA	主轴驱动
HW	硬件

I	
IBN	调试
IKA	可插补补偿
IM	Interface-Modul：接口模块
IMR	Interface-Modul Receive:接收方接口模块
IMS	Interface-Modul Send:发送方接口模块
INC	Increment：增量尺寸
INI	Initializing Data：初始化数据
IPO	插补器
ISA	国际标准体系
ISO	International Standard Organization：国际标准组织

J	
JOG	Jogging:点动模式

K	
K_V	控制环的增益系数
K_P	比例增益

K	
$K_Ü$	传动比
LAD	梯形图（一种 PLC 编程方法）

L	
LAI	Logic Machine Axis Image：逻辑加工轴映射
LAN	Local Area Network：本地局域网
LCD	Liquid-Crystal Display：液晶显示器
LED	Light Emitting Diode：发光二极管
LF	Line Feed：进线电源
LMS	位置测量系统
LR	位置控制器
LSB	Least Significant Bit:最低(有效)位
LUD	Local User Data：用户数据（局部）

M	
MAC	Media Access Control：媒体访问控制
MAIN	Main program：主程序 (OB1，PLC)
MB	兆字节
MCI	Motion Control Interface：运动控制接口
MCIS	运动控制信息系统
MCP	Machine Control Panel：机床控制面板
MD	机床数据
MDA	Manual Data Automatic：手动数据输入
MDS	Motor Data Set:电机数据组
MELDW	信息字
MCS	机床坐标系
MM	电机模块
MPF	Main Program File：主程序（NC）
MSTT	机床控制面板

N	
NC	Numerical Control：带有程序段处理、运行范围等等的数控系统
NCU	Numerical Control Unit：NC 硬件单元
NRK	NC 操作系统名称
NST	接口信号
NURBS	非一致性数理 B 样条
NV	零点偏移
NX	Numerical Extension:轴扩展模块

O	
OB	PLC 中组织块
OEM	原始设备制造商
OP	Operation Panel：操作面板
OPI	Operation Panel Interface：操作面板接口
OPT	Options:选件
OLP	Optical Link Plug：光导线总线插头
OSI	Open Systems Interconnection：计算机通讯标准

P	
PAA	输出端过程图
PAE	输入端过程图
PC	个人计算机
PCIN	与控制系统进行数据更换的软件名称
PCMCIA	个人计算机存储卡国际协会： 存储器插卡标准
PCU	PC Unit：PC 主机（计算机元件）
PG	编程器
PKE	参数标识：PKW 的一部分
PKW	参数标识：值（PPO 的参数部分）
PLC	Programmable Logic Control：可编程逻辑控制器
PN	PROFINET

P	
PNO	PROFIBUS 用户组织
PO	上电
POE	程序组织单元
POS	位置/定位
POSMO A	Positioning Motor Actuator：定位电机
POSMO CA	Positioning Motor Compact AC:集成了功率模块、控制模块、定为单元以及程序存储器的完整驱动单元；使用交流电源
POSMO CD	Positioning Motor Compact DC：同 CA，但采用直流电源
POSMO SI	Positioning Motor Servo Integrated：定位电机；采用直流电源
PPO	Parameter Prozessdaten Objekt：通过 PROFIBUS–DP 和“转速可变驱动”传输时的循环数据报文
PPU	Panel Processing Unit：基于面板的 CNC 控制系统（如 SINUMERIK 828D）的核心硬件
PROFIBUS	Process Field Bus：串行数据总线
PRT	程序测试
PSW	程序控制字
PTP	Point to Point：点到点
PUD	Program Global User Data：程序全局用户变量
PZD	过程数据：PPO 的过程数据部分

Q	
QFK	象限误差补偿

R	
RAM	Random Access Memory：读写存储器
REF	返回参考点功能
REPOS	再定位功能
RISC	Reduced Instruction Set Computer：精简指令集计算机：带有小命令集和快速命令处理的处理器类型
ROV	Rapid Override：快速倍率
RP	R 参数，计算参数，预定义用户变量

R	
RPA	R-Parameter Active：NC 中用于 R 参数号的存储范围
RPY	Roll Pitch Yaw：一种坐标系旋转方式
RTLI	Rapid Traverse Linear Interpolation：快速运行时的直线插补
RTS	Request To Send：开启发送方，控制信号自串行数据接口
RTCP	Real Time Control Protocol：实时控制协议

S	
SA	同步动作
SBC	Safe Break Control:安全制动控制
SBL	Single Block：单程序段
SBR	Subroutine：子程序（PLC）
SD	设定数据
SDB	系统数据块
SEA	Setting Data Active：设定数据符号（文件类型）
SERUPRO	Search-Run by Program Test:通过程序测试的程序段查找
SFB	系统功能块
SFC	系统功能调用
SGE	安全输入
SGA	安全输出
SH	安全停止
SIM	单直列模块
SK	软键
SKP	Skip：跳至零件程序段末尾
SLM	同步直线电机
SM	步进电机
SMC	Sensor Module Cabinet Mounted：机柜安装式编码器模块
SME	Sensor Module Externally Mounted：外部安装的编码器模块
SMI	Sensor Module Integrated：集成编码器模块
SPF	Sub Program File：子程序（NC）
SPS	Speicherprogrammierbare Steuerung（可编程逻辑控制）= PLC
SRAM	静态存储器（缓存）

S	
SRK	刀沿半径补偿
SRM	同步旋转电机
SSFK	主轴丝杆螺距误差补偿
SSI	Serial Synchron Interface：串行同步接口
SSL	程序段搜索
STW	控制字
GWPS	砂轮外缘速度
SW	软件
SYF	System Files：系统文件
SYNACT	Synchronized Action：同步动作

T	
TB	Terminal Board：端子板（SINAMICS）
TCP	Tool Center Point：刀尖
TCP/IP	Transport Control Protocol / Internet Protocol：传输控制协议/因特网互联协议
TCU	Thin Client Unit： 薄型客户单元
TEA	Testing Data Active：机床数据标识
TIA	全集成自动化
TM	Terminal Module：端子模块（SINAMICS）
TO	Tool Offset：刀具补偿
TOA	Tool Offset Active：刀具补偿标识（文件类型）
TRANSMIT	Transform Milling Into Turning：车床上铣削加工的坐标转换
TTL	逻辑门电路（接口类型）
TZ	工艺循环

U	
UFR	User Frame：零点偏移
UP	子程序
USB	Universal Serial Bus： 通用串行总线
USV	不间断电源

V	
VDI	NC 和 PLC 间的内部通讯接口
VDI	德国工程师协会
VDE	德国电工技术人员联合会
VI	电压输入
VO	电压输出
VSA	进给驱动

W	
SAR	平滑逼近和退回功能
WCS	工件坐标系
WKZ	刀具
WLK	刀具长度补偿
WOP	现场编程
WPD	Work Piece Directory：工件目录
WRK	刀具半径补偿
WZ	刀具
WZK	刀具补偿
WZV	刀具管理
WZW	换刀

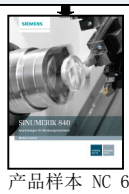
X	
XML	Extensible Markup Language：可扩展标记语言

Z	
ZOA	Zero Offset Active：零点偏移标识
ZSW	（驱动）状态字

A.2 资料概览

通用文档

广告印刷品
- SINUMERIK 840D sl
- SINUMERIK 828D
- SINUMERIK 828D BASIC

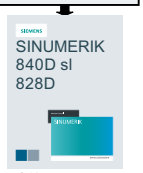


产品样本 NC 62
SINUMERIK 840D sl

产品样本 NC 82
SINUMERIK 828D

产品样本 PM 21
SIMOTION,
SINAMICS S120

系统手册
智能机床
操作



系统手册
Ctrl-Energy

用户文档

操作手册
- 通用
- 车削
- 铣削
- 磨削

编程手册
- 基础
- 工作准备
- 测量循环

编程手册
- ISO 车削
- ISO 铣削

诊断手册
报警

诊断手册
报警

制造商/服务文档

设备手册
- NCU
- 操作组件和联网
- ADI4

设备手册
调试手册
服务手册

调试手册
- CNC:NCK, PLC, 驱动
- 基础软件和操作软件

参数手册
- 机床数据
- 接口信号
- 变量

参数手册
- 机床数据
- 接口信号
- 参数
- 变量

系统手册
向导
机床
配置

制造商/服务文档

功能手册
- 基本功能
- 扩展功能
- 特殊功能
- 同步动作
- ISO 方言

功能手册
刀具管理

功能手册
Safety Integrated

功能手册
Safety Integrated

选型手册
- EMC 指令
- 工业安全

信息/培训

培训资料
- 简单铣削
  使用 ShopMill
- 简单车削
  使用 ShopTurn

手册
刀具和
模具制造

电子文档

DOConCD

工业在线
支持（SIOS）

网上商城

词汇表

C 样条

C 样条是最为人所知也是一种最常用的样条。支点之间的过渡为相切过渡，保持恒定曲率。使用 3 级多项式。

C 轴

一根作受控的旋转运动以定位工件主轴的轴。

CNC

参见 → NC

Computericed Numerical Control（计算机数控）：涵盖以下组件 → NC、→ PLC、HMI、→ COM。

CNC

参见 → NC

Computericed Numerical Control（计算机数控）：涵盖以下组件 → NC、→ PLC、HMI、→ COM。

CNC 标准语言

标准语言用于编写 NC 程序、→ 同步动作和 → 循环。其提供：控制结构、→ 用户自定义变量、→ 系统变量、→ 宏技术。

COM

控制系统部件，用于执行和和协调通讯。

CPU

中央处理单元，参见 → 可编程控制器

DRF

Differential Resolver Function:NC 功能，在自动方式下利用电子手轮产生增量式零点偏移。

HIGHSTEP

编程方法汇编，用于。→ 系统 AS300/AS400 中的 PLC。

HW-Config

SIMATIC S7 工具，用于配置和设置 S7 项目中的硬件。

JOG

控制系统的一种运行方式（调试方式）：在 JOG 运行方式下，可以调试机床。各个进给轴和主轴可以通过方向键断续运行。在 JOG 手动运行方式中还有其它的一些功能，如 → 回参考点运行，→ 重新定位以及预设定（设定实际值）。

Kp

传动比

KV

回路增益系数，控制理论中控制环的一个参数

MDA

控制系统的一种运行方式：手动输入，自动运行在 MDA 方式下，可以输入单个程序段或者几个程序段，按下 CYCLE START 键可以立即执行。

NC

→ CNC 的 Numerical Control（数控）组件，用于执行 → 零件程序，并协调机床的运动进程。

NRK

Numeric Robotic Kernel（NC →的操作系统）

NURBS

控制系统内部的运动控制和轨迹插补是基于 NURBS（**N**on **U**niform **R**ational **B**-**S**plines 非均匀有理 B 样条）进行的。因此，在系统内部所有插补均采用相同的方法。

OEM

为机床制造商提供了各种不同的解决方案（OEM 应用），制造商可以自己设计操作界面或者在系统中开发专用的工艺功能。

PLC

Programmable **L**ogic **C**ontrol: → 可编程逻辑控制器。 **->** 数控系统的组件： 用于调整机床控制逻辑。

PLC 编程

PLC 用软件 **STEP 7** 编程。 编程软件 STEP 7 基于 **WINDOWS** 标准操作系统开发，并包含创新的 STEP 5 编程功能。

PLC 可编程控制器

SINUMERIK 840D sl： 在 PLC 用户存储器中保存了 PLC 用户程序、用户数据与 PLC 基本程序。

R 参数

计算参数， → 零件程序编程人员可在程序中读写这些参数。

TOA 单元

每个 → TOA 区域可以包含多个 TOA 单元。 TOA 单元的数量以最大有效的 →通道数量为极限。 一个 TOA 单元包括一个刀具数据模块和一个刀库数据模块。 另外，也可包含一个刀架数据模块（选件）。

TOA 区域

TOA 区域包含所有刀具和刀库数据。 默认设置下，TOA 区域即 → 通道。 但通过机床数据可以确定，多个通道组成一个 → TOA 单元，以使这些通道也可使用通用的刀具管理数据。

V.24

用于数据输入/输出的串行接口。 通过该接口可以装载和备份加工程序以及制造商和用户数据。

WinSCP

WinSCP 是一个可自由使用的开源程序，用于 Windows 的文件传输。

安全功能

系统中始终处于激活的监控功能，可以及早识别出 → CNC、→ PLC 和机床中的故障，从而排除一切对工件、刀具或者机床可能造成的损害。 在故障发生时，加工过程会中断，驱动停止，故障原因被保存并作为报警显示。 同时通知 PLC 有一报警。

保护区

→ 加工区内、不允许刀尖进入的一个三维空间.

报警

所有→ 提示信息和报警都包含纯文本说明、日期时间以及指出清除条件的符号，显示在机床操作面板上。报警和信息单独显示。

1. 零件程序中的报警和信息
   报警和信息可以直接以纯文本的形式从零件程序中输出。
2. PLC 报警和信息
   机床报警和信息可以直接以纯文本的形式从 PLC 程序中输出。而无需另外的功能块软件包。

备份电池

备份电池可实现 → CPU 中的 → 用户程序、特定数据区、标志位、定时器和计数器的掉电保持。

倍率

一种手动干预方法或编程方法，允许操作人员覆盖已编程的进给率或者转速，使加工速度与具体的工件和材料相匹配。

比例

是 → 框架的一个分量，可以改变某个轴的比例尺。

编程码

编程码是 → 零件程序编程语言中具有特定含义的字符和字符串。

编辑器

利用编辑器可以进行程序/文本/程序段的创建、修改、补充、合并和插入。

变量定义

变量定义包括确定数据类型和变量名。 使用变量名可以调用该变量的值。

标准循环

标准循环适用于最常见的加工任务：

- 钻削/铣削
- 车削

在“程序”操作区“循环支持”菜单下列出了所有可使用的循环。 选择了所需加工循环后，屏幕上会以纯文本的形式显示需要赋值的参数。

波特率

数据传输速度（位/秒）。

补偿表

控制点组成的一张表格。补偿表指出和参照轴的位置对应的补偿轴的补偿值。

补偿存储器

控制系统中的一个数据区，刀具补偿数据存储在其中。

补偿值

测量编码器实测出的位置与程序目标位置之间的差值/

补偿轴

设定值或者实际值可以通过补偿值进行修改的轴。

参考点

机床上的一点，→ 机床轴的测量系统以该点为基准。

参照轴

在计算补偿值需要考虑其设定值或实际值的轴。

操作界面

操作界面（BOF）是控制系统的显示屏幕。它带有水平软键和垂直软键。

插补补偿

通过插补补偿，如 → 丝杠螺距误差补偿、垂度补偿、角度补偿和温度补偿，对机床的机械误差进行补偿。

插补器

→ NC 的逻辑单元，根据零件程序中目标位置确定轴的中间值。

成品轮廓

成品工件的轮廓。参见 →毛坯件。

程序段搜索

在进行零件程序测试时或者在中断一个加工后，可以通过“程序段搜索”功能找到程序中的任意位置，在此位置加工可以启动或者继续。

程序级

在通道中启动的零件程序作为程序级 0（主程序级）上的 → 主程序运行。 每个在主程序中调用的零件程序作为单独程序级 1 ... n 上的子程序运行。

程序块

程序块包含 → 零件程序的主程序和子程序。

从轮廓快速退刀

出现中断事件时，可以通过 CNC 加工程序触发一个运动，使刀具迅速离开当前正在加工的工件轮廓。 此外还可以设定退刀角度和退刀行程。 在快速退刀以后可以另外执行一个中断程序。

存档

读出文件和/或目录，存储到**外部**存储器设备中。

单位制：公制和英制

在加工程序中，位置值和螺距值可以用英制编程。不管程序中的尺寸指令(G70/G71)如何，系统始终采用基本单位制。

刀具

机床中进行加工的工具（诸如车刀、铣刀、钻头、激光…）

刀具半径补偿

为了可以直接编程所需的 → 工件轮廓，控制系统必须考虑所用刀具的半径，使刀具等距绕行编程的轮廓。（G41/G42）。

刀具补偿

计算轨迹时考虑到刀具尺寸。

刀沿半径补偿

在编程一个轮廓时，是从刀尖计算轮廓的。 但是，这在实际加工过程中无法实现，因为使用的刀具会有一个弯曲半径，控制系统必须考虑该半径。 此时，系统会使弯曲中心点偏移弯曲半径，从而等距绕行轮廓。

地址

地址用于标出特定操作数或操作数区域，例如：输入、输出等。

定位轴

在机床中执行辅助运动的轴。（例如刀库，托盘运输）。 定位轴不与 → 轨迹轴进行插补。

定向主轴准停

使工件主轴停止在一指定角度位置上，以便在某一确定位置进行辅助加工。

动态前馈控制

和加速度相关的动态前馈控制几乎可完全消除跟随误差导致的 → 轮廓误差。 因此可获得高加工精度，即使是在 → 轨迹速度很高的情况下。 前馈控制可以通过→ 零件程序逐轴选择或者撤销。

多项式插补

用多项式插补功能可以产生不同的曲线，如**线性函数、抛物线函数和幂函数**（SINUMERIK 840D sl）。

反比时间进给率

也可以编程一个程序段的轨迹行程所需要的时间（G93），而不用编程轴的进给速度。

分度轴

分度轴使工件或者刀具旋转到一个和分度头对应的角度位置。 到达分度位置后，回转轴“就位”。

辅助程序段

由“N”开头的程序段，包含一个加工步骤的信息，比如一个位置数据。

辅助功能

在 → 零件程序中，使用辅助功能可以把机床制造商定义的 → 参数传送到 → PLC 中，并释放其所定义功能。

刚性攻丝

用此功能可以不带补偿衬套攻螺纹。通过插补，主轴作为回转轴和钻削轴将螺纹精确钻至最终钻深，比如盲孔螺纹（前提条件：主轴作为进给轴运行）。

高速数字量输入/输出

通过该数字量输入可以启动高速程序（中断程序）。 通过该数字量输出端可以触发程序控制的高速开关功能。

工件

需由机床制造/加工的零件。

工件零点

工件零点构成了→ 工件坐标系的原点。 它由与→ 机床零点的距离定义。

工件轮廓

待加工→ 工件的目标轮廓。

工件坐标系

以→ 工件零点为原点的坐标系。 在工件坐标系中编程时，坐标和方向以工件坐标系为基准。

工作存储器

→ CPU 中的 RAM 存储器，在程序执行期间，处理器访问其上的用户程序。

工作区域限制

除限位开关之外，还可以使用工作区域限制功能对轴的行程进行限制。每根轴都可以使用一对值设定保护区。

公制单位

单位均为公制：比如：长度采用 mm（毫米），m（米）。

关键字

在 → 零件程序编程语言中具有一定含义、具有固定句法的指令字。

轨迹进给率

轨迹进给率作用于 → 轨迹轴。它是参与轨迹运动的 → 几何轴的合成进给率。

轨迹速度

最大可编程轨迹速度取决于输入精度。比如精度为 0.1 毫米，则可编程的最大轨迹速度为 1000 米/分钟。

轨迹轴

轨迹轴指 → 通道的所有加工轴，由 → 插补器控制，它们可以同时启动、加速、停止并同时达终点。

宏指令技术

包含有一系列指令，但名称只有一个。在程序中，一个名称就代表了这一系列指令。

回转轴

回转轴使工件或者刀具旋转到一个指定的角度位置。

机床固定点

由机床明确定义的点，比如：机床参考点。

机床控制面板

机床的控制面板有各个操作按键、旋钮开关等，以及各个显示单元如 LED。它们直接通过 PLC 对机床进行控制。

机床零点

机床固定点，所有测量系统均以此点为原点。

机床轴

在机床上实际存在的轴。

机床坐标系

以机床轴为基准的坐标系。

基本坐标系

一个直角坐标系，可转换为机床坐标系。

在 → 零件程序中使用基本坐标系中的轴名称。如果没有 → 有效的坐标转换，则它平行于 → 机床坐标系。不同点在于 → 轴名称。

极限转速

最大/最小（主轴）转速：在机床数据、→ PLC 数据或者 → 设定数据中可以限制主轴的最大转速。

极坐标

极坐标系指在一个平面中通过点到零点的距离、半径和一个固定轴之间的夹角来确定点位置的坐标系。

几何尺寸

→ 工件 → 在工件坐标系中的尺寸。

几何轴

几何轴构成一个 2 维或 3 维 → 工件坐标系，零件程序在该坐标系中编写工件的几何尺寸。

加工区

加工区是机床中刀尖可以进入的三维区域。参见 → 保护区

加工通道

采用通道结构可以进行并行运动，缩短非加工时间，比如在装料的同时可以进行加工。一条通道可以看作为一个独立的数控系统，可以译码、程序段预处理并进行插补。

加速度，带加加速度限制

为了在机床上获得最佳的加速方式，同时又要保护机械，在加工程序中可以在跃变式加速度和平缓式加速度之间进行转换。

间隙补偿

对机床上的机械间隙进行补偿，比如滚珠丝杠的反向间隙。每根轴可以分别输入间隙补偿。

接地

接地指设备上所有相互连接的不带电部件构成的整体，即使在出现故障时也不会出现危险的接触电压。

进给倍率

机床控制面板或者→ PLC 设定的、叠加在编程速度上的倍率值，以修改当前速度（0－200%） 另外，进给速度也可以在加工程序中通过一个百分比（1－200%）进行修改。

镜像

镜像使轮廓轴坐标值的符号相反。 可以同时对多个轴进行镜像。

绝对坐标

指出轴在某一个方向上相对于当前有效的坐标系零点的运动终点。参见 → 增量坐标。

可编程的工作区域限制

将刀具的运动空间限制在一个指定范围内。

可编程的框架

使用可编程的 → 框架可以在零件程序执行过程中动态地定义新的坐标系原点。 框架可分为全新框架和附加在原有框架上的附加框架。

可编程逻辑控制器

可编程逻辑控制器(PLC)是电子控制器，其功能以程序的形式存储。 因此，PLC 的结构和布线与控制系统的功能无关。 PLC 具有计算机的结构，它由带存储器的 CPU（中央模块）、输入/输出模块和内部总线系统构成。 输入/输出和编程语言以控制系统的需求为准。

快速移动

轴最快的运行速度。比如，可用于使刀具从静止状态运行到 → 工件轮廓或者从工件轮廓返回。快速移动速度可以根据不同机床在机床数据中设置。

框架

框架是一种运算规范，指出如何把一种直角坐标系转换到另一种直角坐标系。框架中包含几个部分 → 零点偏移、 → 旋转、 → 比例、 → 镜像。

连续路径运行

该方式可以在零件程序段分界处避免 → 轨迹轴急速停止，并尽可能保持相同的轨迹速度转到下一个程序段。

零点偏移

通过 → 框架相对于坐标系零点指定一个新的参考点。

1. 可设定
   可为每个 CNC 轴设定不同数量的零点偏移。可通过 G 代码选择的偏移可以选择性地使用。
2. 外部
   除了用于确定工件零点的所有偏移外，还可以通过手轮（DRF 偏移）或者由 PLC 叠加一个外部零点偏移。
3. 可编程
   TRANS 指令可以为所有的轨迹轴和定位轴确定零点偏移。

零件程序

数控系统中由一系列指令构成、用于实现特定 → 工件的加工的程序。 也就是说，在一个指定的 → 毛坯上进行指定的加工。

零件程序段

一行 → 零件程序，换行后结束。 分为→ 主程序段和→ 辅助程序段。

零件程序管理

零件程序可以按照→ 工件管理。 用户存储器的容量确定了需要管理的程序和数据的数量。每个文件（程序和数据）的名称可以最多为 24 个字符（字母和数字）。

轮廓

→ 工件的外边沿

轮廓监控

为确保轮廓精度，系统会监控跟随误差是否在定义的公差带内。比如，当驱动负载过高时，就可能导致跟随误差过大。在这种情况下会产生一个报警，轴停止运行。

螺旋线插补

螺旋线插补特别适用于利用成形铣刀简单地加工内螺纹和外螺纹以及铣削润滑槽。

螺旋线由两个运动组成：

- 平面中的回转运动
- 与此平面垂直的直线运动

毛坯

未经加工的工件。

名称

符合 DIN 66025 标准的“字”可补充变量标识/变量名（计算变量、系统变量和用户变量）、子程序名、关键字名和带多个地址字母的字。这些补充的名称和程序段“字”的含义一样。名称必须是唯一的。同一个名称不可以用于不同的对象。

模块

模块是指编程和程序执行时所需要的所有文件。

倾斜加工

通过倾斜加工可以方便地进行无法在机床坐标系中进行的钻削和铣削加工。

清零

在清零时，→ CPU 中以下的存储器将被清零：

- → 工作存储器
- → 装载存储器的读写区

- → 系统存储器
- → 备份存储器

驱动

属于 CNC 的组件，它根据 NC 设定执行转速控制和转矩控制。

曲率

轮廓的曲率 k 是轮廓点上该段圆弧半径 r 的倒数 (k = 1/r)。

软件限位开关

软件限位开关用于限制轴的移动范围，阻止滑块驶出硬件限位开关。 每个轴可以给定 2 组数值，它们可以由 → PLC 分别激活。

软键

在在屏幕上显示的按键，可以动态地根据当前的操作变化。 这些功能键（软键）可以任意指定由软件规定的功能。

设定数据

设定数据确定机床的性能，按照系统软件定义的方法在系统中设定。

输入/输出模块

用于建立 CPU 和过程之间的联系。

输入/输出模块是：

- → 数字量输入/输出模块
- → 模拟量输入/输出模块
- → 模拟器模块

数据传输程序 PCIN

PCIN 是一种辅助程序，通过串行接口发送和接收 CNC 用户数据，如零件程序、刀具补偿等等。PCIN 程序可以在标准工业 PC 中的 MSDOS 下运行。

数据块

1. → PLC 的数据单元，可以访问 → HIGHSTEP 程序。
2. →NC 的数据单元：数据块包含全局用户数据的数据定义。数据可以在定义时直接初始化。

数据字

一个 → 数据块中两个字节大小的数据单位。

丝杠螺距误差补偿

控制系统根据保存的测量值补偿作进给运动的滚珠丝杠的机械误差。

速度控制

如果每个程序段轴的移动量非常小，可以使用预读功能(→ Look Ahead) 预读多个程序段来获得合理的移动速度。

通道

通道可以单独处理一个 → 零件程序，而不受其他通道的影响。一个通道仅控制指定给它的进给轴和主轴。不同通道的零件程序执行过程可以通过 →同步功能进行协调。

同步

→ 零件程序中的指令，用于协调同一加工地点时不同→ 通道中的加工过程。

同步动作

1. 辅助功能输出
   在工件加工期间，可以把工艺功能（→ 辅助功能）从 CNC 程序中输出到 PLC 中。 通过这些辅助功能可以控制机床的附加设备，比如顶尖套筒、机械手、卡盘等等。
2. 快速辅助功能输出
   如果开关功能对时间要求较高，可以缩短 → 辅助功能的应答时间，避免加工过程不必要的停顿。

同步轴

同步轴需要获知其行程。而几何轴需要获知其轨迹行程。

同步轴

同步轴为→ 龙门轴，其设定位置始终由主动轴导出，因此两者同步运行。对于操作员和编程者而言，同步轴是“不存在的”。

外部零点偏移

由 → PLC 给定的零点偏移。

网络

网络指通过 → 连接电缆相连的多个 S7-300 和其它终端设备，比如一台编程器。相连设备通过网络进行数据交换。

文本编辑器

参见 → 编辑器

系统变量

无需 → 零件程序编程员的工作，已经存在的变量。 它由数据类型和变量名称定义，变量名称前有符号**$**。 参见 → 用户自定义变量。

系统存储器

系统存储器是 CPU 中的一个存储器，其中保存有：

- 操作系统所需要的数据
- 操作数：定时器、计数器和标志位

线性插补

刀具以直线运行到目标，同时进行工件的加工。

线性轴

与回转轴相反，线性轴指按直线运行的轴。

象限误差补偿

由导轨上不断变化的摩擦阻力导致的过象限轮廓误差可以最大程度地用象限误差补偿消除。象限误差补偿的参数可以通过圆弧测试确定。

信息

零件程序中编入的所有信息以及系统检测出的 → 报警均在操作面板上显示，带日期和时间，并有相应的清除条件符号。报警和信息单独显示。

旋转

→ 框架的一个分量，定义坐标系的旋转角度。

循环

受保护的子程序，用于执行 → 工件上反复出现的加工过程。

样条插补

通过样条插补，控制系统只需少数几个目标轮廓控制点便可生成一条光滑的曲线。

钥匙开关

→ 机床控制面板上的钥匙开关占据 4 个位置，它们由控制系统的操作系统指定相应的功能。钥匙开关有三把不同颜色的钥匙，钥匙可以在指定位置插拔。

异步子程序

指可以通过一个中断信号（比如信号“高速 NC 输入”）启动的、与当前程序状态异步执行（即无关）的零件程序。

引导启动

上电后装载系统程序。

英制单位

单位制，距离采用“英寸”及其下级小数单位。

用户程序

可编程控制器 S7-300 中用 STEP 7 语言编写的的用户程序。用户程序为模块化结构，由各个模块构成。

基本的模块类型有：

- 代码模块
  该模块含有 STEP 7 指令。
- 数据模块
  该模块包含有用于 STEP 7 程序的常量和变量。

用户存储器

包含了所有的程序和数据，比如零件程序、子程序、注释、刀具补偿、零点偏移、框架以及通道和程序用户数据。

用户自定义变量

用户可以在 → 零件程序或者数据块（全局用户数据）中自由使用自定义变量。一个定义通常含有数据类型和变量名称。参见 → 系统变量。

预读功能

利用功能**预读** 可以预读一定数量的程序段而优化加工速度。

预符合

如果轨迹行程几乎快接近设定三角形的终端位置，则进行程序段转换。

圆弧插补

在轮廓上两个固定点之间，→ 刀具以给定的进给率沿圆弧运行，加工工件。

运行到固定点

机床中有一些固定点，比如换刀点、装料点、托盘更换点等等。这些点的坐标在控制系统中定义。控制系统使对应轴移动到这些固定点，如果可能，→ 快速移动轴。

运行范围

线性轴最大允许的运行范围可以达到±9 位。 绝对值取决于所选择的输入精度、位置控制精度以及单位制（英制或者公制）。

运行方式

SINUMERIK 控制系统的运行控制方式。定义的运行方式有 → Jog, → MDA, → 自动。

运行方式组

加工工艺上关联的进给轴和主轴可以归为一个工作方式组（BAG）。一个 BAG 的进给轴和主轴可以由一个或多个 → 通道控制。同一个工作方式组中的通道均有相同的 → 工作方式。

增量坐标

以增量数（步数）指定行程。 步数可以作为 → 设定数据保存或者通过相应的按键（10、100、1000 和 10000）选取。

增量坐标

也称为相对坐标：指出轴相当于起点需要移动的行程以及方向。参见 → 绝对坐标

诊断

1. 系统操作区
2. 控制系统不仅有自诊断程序，还可以为维修提供辅助测试。状态、报警和服务信息

中断程序

中断程序是一种特殊 → 子程序，由程序执行过程中的外部事件（外部信号）启动。正在执行的程序段终止，系统自动保存轴的中断位置。

中间程序段

带 → 刀具补偿（G41/G42）的运动可以由一定数量的中间程序段（位于补偿级的程序段，没有轴运动）中断，此时刀具补偿仍能进行正确计算。 控制系统能预读的最大中间程序段数量可以通过系统参数设定。

轴

数控系统中的轴根据其功能可以分为：

- 轴：进行插补的轨迹轴
- 辅助轴：不进行插补的进给轴和定位轴，具有单轴进给率。辅助轴不参与加工，而是用于执行辅助运动，比如移动机械手、转动刀库等。

轴地址

参见 → 轴名称

轴名称

为了明确标识控制系统的所有通道轴和 → 机床轴，每根轴有一个通道范围内或系统范围内唯一的名称。→ 几何轴名为 X 轴、Y 轴和 Z 轴。 围绕几何轴旋转的回转轴名为 A 轴、B 轴和 C 轴。

主程序

之前零件程序固定划分为主程序和 → 子程序，因此有“主程序”的概念。在如今的 SINUMERIK NC 语言中，这种固定划分已不再存在。原则上，每个零件程序都可以在通道中选择和启动。其在 → 程序级 0（主程序级）上运行。在主程序中，其他的零件程序或 → 循环都可以作为子程序来调用。

主程序段

通过“:”引导的程序段，包含在零件程序中启动工作流程所需要的所有数据。

主动轴

主动轴为 → 龙门轴，对于操作员和编程者而言，它可以象普通轴一样控制。

装载存储器

在 → PLC 的 CPU314 中，装载存储器就是 → 工作存储器。

准停

使轴准确地移动到指定位置，或者以非常慢的速度移动到指定位置。为缩短移动时间，可以为快速移动运动和进给运动定义 → 准停界限。

准停界限

如果所有的轨迹轴均到达准停界限，则控制系统会认为轴已经精确到达目标。随后 → 零件程序会切换程序段。

子程序

之前零件程序固定划分为主程序和子程序，因此有“子程序”的概念。 在如今的 SINUMERIK NC 语言中，这种固定划分已不再存在。 原则上，每个零件程序或 → 循环在另一个零件程序中都可作为子程序调用。 其在下一个 → 程序级（x+1）（子程序级（x +1））上运行。

自动方式

控制系统的运行方式（程序段连续运行，符合 DIN 标准）：控制系统中的运行方式，这种方式下选择 → 零件程序并连续执行程序。

坐标系

参见 → 机床坐标系, → 工件坐标系

坐标转换

轴的附加或者绝对零点偏移。

索引

$

$AA_ATOL, 594
$AA_COUP_ACT
使用耦合轴时, 600
轴向引道值耦合时, 625
$AA_ESR_ENABLE, 723
$AA_LEAD_SP, 625
$AA_LEAD_SV, 625
$AC_ACT_PROG_NET_TIME, 710
$AC_ACTUAL_PARTS, 714
$AC_AXCTSWA, 694
$AC_AXCTSWE, 694
$AC_CTOL, 594
$AC_CUT_INV, 512
$AC_CUTMOD, 511
$AC_CUTMOD_ANG, 511
$AC_CUTMODK, 511
$AC_CUTTING_TIME, 710
$AC_CYCLE_TIME, 710
$AC_DELAYFST, 575
$AC_ESR_TRIGGER, 723
$AC_OLD_PROG_NET_TIME, 710
$AC_OLD_PROG_NET_TIME_COUNT, 710
$AC_OPERATING_TIME, 710
$AC_OTOL, 594
$AC_PROG_NET_TIME_TRIGGER, 711
$AC_REPOS_PATH_MODE, 584
$AC_REQUIRED_PARTS, 713
$AC_SMAXVELO, 589
$AC_SMAXVELO_INFO, 589
$AC_SPECIAL_PARTS, 714
$AC_TOTAL_PARTS, 713
$AC_TRAFO_CORR_ELEM_P, 440
$AC_TRAFO_CORR_ELEM_T, 440
$AC_TRAFO_ORIAX_LOC, 440
$AN_AXCTAS, 694
$AN_AXCTSWA, 694
$AN_ESR_TRIGGER, 723
$AN_LANGUAGE_ON_HMI, 1038
$AN_POWERON_TIME, 709
$AN_SETUP_TIME, 709
$NT_CLOSE_CHAIN_T, 440
$NT_CNTRL, 440
$NT_CORR_ELEM_P, 439
$NT_CORR_ELEM_T, 439
$NT_NAME, 431
$NT_ROT_AX_NAME, 508
$NT_TRAFO_INDEX, 431
$P_ACTBFRAME, 341
$P_AD, 511
$P_BFRAME, 340
$P_CHBFRAME, 340
$P_CHBFRMASK, 341
$P_CTOL, 594
$P_CUT_INV, 512
$P_CUTMOD, 511
$P_CUTMOD_ANG, 511
$P_CUTMOD_ERR, 513
$P_CUTMODK, 511
$P_DELAYFST, 575
$P_IFRAME, 342
$P_IS_EES_PATH, 243
$P_NCBFRAME, 340
$P_NCBFRMASK, 341
$P_ORI_DIFF, 503
$P_ORI_POS, 503
$P_ORI_SOL, 504
$P_ORI_STAT, 507
$P_OTOL, 594
$P_PATH, 242
$P_PFRAME, 342
$P_PROG, 242
$P_PROGPATH, 242
$P_SIM, 308
$P_STACK, 242
$P_SUBPAR, 177
$P_TOOLENV, 523
$P_TOOLENVN, 522
$PA_ATOL, 594
$SA_LEAD_TYPE, 624
$SC_CONTPREC, 568
$SC_MINFEED, 568
$SC_PA_ACTIV_IMMED, 255
$SN_PA_ACTIV_IMMED, 255
$TC_CARR1...14, 487
$TC_CARR18...23, 488
$TC_CARR18[m], 491
$TC_DP1 ... 25, 441
$TC_ECPxy, 446
$TC_SCPxy, 446

*

*（计算功能）, 81

/

/（计算功能）, 81

+

+（计算功能）, 81

<

<（比较运算符）, 83
<< (链接运算), 93
<=（比较运算符）, 83
<>（比较运算符）, 83

=

==（比较运算符）, 83

>

>（比较运算符）, 83
>=（比较运算符）, 83

0

0 字符, 91

A

A 样条, 278
ABS, 81
ACCLIMA, 563
ACOS, 81
ACTBLOCNO, 189
ACTFRAME, 315
ADISPOSA, 310
ALF
　用于从轮廓快速退刀, 144
AND, 83
APR, 49
APRB, 49
APRP, 49
APW, 49
APWB, 49
APWP, 49
AS, 227
ASIN, 81
ASPLINE, 271
ATAN2, 81
ATOL, 591
AV, 633
AX, 685
AXCTSWE, 693
AXCTSWEC, 693
AXCTSWED, 693
AXIS, 31
AXNAME, 92
AXSTRING, 685
AXTOCHAN, 153
AXTOSPI, 685

B

B 样条, 279
B_AND, 83
B_NOT, 83
B_OR, 83
B_XOR, 83
BAUTO, 271
BFRAME, 315
BLOCK, 215
BLSYNC, 138
BNAT, 271
BOOL, 31
BOUND, 87
BRISK, 561
BRISKA, 561
BSPLINE, 271
BTAN, 271

C

C 样条, 280
CAC, 269
CACN, 269
CACP, 269
CALL, 214
CALLPATH, 217
CASE, 114
CDC, 269
CFINE, 326
CHAN, 31
CHANDATA, 244
CHAR, 31
CHKDNO, 483
CIC, 269
CLEARM, 130
CLRINT, 140

COARSE, 633
COARSEA, 310
COLLPAIR, 426
COMPCAD, 285
COMPCURV, 285
COMPLETE, 244
COMPOF, 285
COMPON, 285
COMPSURF, 285
CONTDCON, 747
CONTPRON, 740
CORRTRAFO, 433
COS, 81
COUPDEF, 633
COUPDEL, 633
COUPOF, 633
COUPOFS, 633
COUPON, 633
COUPONC, 633
COUPRES, 633
CP, 403
CPBC, 645
CPDEF, 644
CPDEL, 644
CPFMOF, 647
CPFMON, 647
CPFMSON, 646
CPFPOS + CPOF, 647
CPFPOS + CPON, 645
CPFRS, 645
CPLA, 644
CPLCTID, 645
CPLDEF, 644
CPLDEL, 644
CPLDEN, 645
CPLINSC, 649
CPLINTR, 649
CPLNUM, 645
CPLOF, 645
CPLON, 645
CPLOUTSC, 649
CPLOUTTR, 649
CPLPOS, 645
CPLSETVAL, 645
CPMALARM, 650
CPMBRAKE, 650
CPMPRT, 649
CPMRESET, 648
CPMSTART, 649
CPMVDI, 650
CPOF, 644
CPON, 644
CPRECOF, 568
CPRECON, 568
CPROT, 252
CPROTDEF, 247
CPSETTYPE, 650
CPSYNCOP, 649
CPSYNCOP2, 650
CPSYNCOV, 650
CPSYNFIP, 649
CPSYNFIP2, 650
CPSYNFIV, 650
CSPLINE, 271
CTAB, 613
CTABDEF, 602
CTABDEL, 609
CTABEND, 602
CTABEXISTS, 608
CTABFNO, 618
CTABFPOL, 618
CTABFSEG, 618
CTABID, 612
CTABINV, 613
CTABISLOCK, 612
CTABLOCK, 610
CTABMEMTYP, 612
CTABMPOL, 618
CTABMSEG, 618
CTABNO, 618
CTABNOMEM, 618
CTABPERIOD, 612
CTABPOL, 618
CTABPOLID, 618
CTABSEG, 618
CTABSEGID, 618
CTABSEV, 613
CTABSSV, 613
CTABTEP, 613
CTABTEV, 613
CTABTMAX, 613
CTABTMIN, 613
CTABTSP, 613
CTABTSV, 613
CTABUNLOCK, 610
CTOL, 591
CTRANS, 326
CUT3DC, 462
CUT3DCC, 472
CUT3DCCD, 471
CUT3DCD, 462
CUT3DF, 466
CUT3DFD, 466
CUT3DFF, 466

CUT3DFS, 466
CUTMOD, 508
CUTMODK, 508
CYCLE4071
外部编程, 889
CYCLE4072
外部编程, 891
CYCLE4073
外部编程, 894
CYCLE4074
外部编程, 895
CYCLE4075
外部编程, 899
CYCLE4077
外部编程, 901
CYCLE4078
外部编程, 904
CYCLE4079
外部编程, 906
CYCLE435 - 设置修整器坐标系
外部编程, 841
CYCLE495 - 成型
外部编程, 842
CYCLE60 - 雕刻
外部编程, 780
CYCLE61- 平面铣削
外部编程, 783
CYCLE62- 轮廓调用
外部编程, 786
CYCLE63 - 轮廓腔铣削
外部编程, 787
CYCLE64 - 预钻轮廓腔
外部编程, 791
CYCLE70 - 螺纹铣削
外部编程, 792
CYCLE72 - 轨迹铣削
外部编程, 795
CYCLE76 - 矩形凸台
外部编程, 799
CYCLE77 - 圆形凸台
外部编程, 802
CYCLE78 - 螺纹铣削
外部编程, 805
CYCLE79 - 多边形
外部编程, 808
CYCLE800 – 回转
外部编程, 844
CYCLE801 - 栅格/框架位置模式
外部编程, 849
CYCLE802 - 任意位置
外部编程, 851
CYCLE81 - 钻中心孔
外部编程, 811
CYCLE82 - 钻削
外部编程, 812
CYCLE83 - 深孔钻削
外部编程, 815
CYCLE830 - 深孔钻削 2
外部编程, 855
CYCLE832 - 快速设定
外部编程, 861
CYCLE84 – 刚性攻丝
外部编程, 819
CYCLE840 – 攻丝，带补偿夹具
外部编程, 865
CYCLE85 - 铰孔
外部编程, 823
CYCLE86 - 镗孔
外部编程, 825
CYCLE899 - 铣削敞开槽
外部编程, 869
CYCLE92 - 切断
外部编程, 826
CYCLE930 - 凹槽
外部编程, 872
CYCLE940- 退刀槽
外部编程, 876
CYCLE95 - 切削轮廓
外部编程, 828
CYCLE951- 切削
外部编程, 879
CYCLE952 - 槽式车削
外部编程, 882
CYCLE98 - 螺纹链
外部编程, 830
CYCLE99 - 螺纹车削
外部编程, 836

D

D 号码
检查, 483
任意赋值, 483
重命名, 484
DCI, 54
DCM, 54
DCU, 54
DEF, 31
DEFAULT, 114
DEFINE ... AS, 227
DELAYFSTOF, 573
DELAYFSTON, 573
DELDL, 447

DELETE, 161
DELOBJ, 417
DELTOOLENV, 520
DIN 子程序名称, 240
DISABLE, 140
DISPLOF, 189
DISPLON, 189
DISPR, 578
DIV, 81
DL, 445
DO, 655
DRIVE, 561
DRIVEA, 561
DV, 633
DYNFINISH, 565
DYNNORM, 565
DYNPOS, 565
DYNROUGH, 565
DYNSEMIFIN, 565

E

Easy XML, 703
EAUTO, 271
EES, 234
EES 标识符, 236
EG
    电子齿轮, 626
EGDEF, 626
EGDEL, 631
EGOFC, 631
EGOFS, 631
EGON, 627
EGONSYN, 627
EGONSYNE, 627
ELSE, 123
ENABLE, 140
ENAT, 271
ENDFOR, 125
ENDIF, 123
ENDLABEL, 116
ENDLOOP, 125
ENDWHILE, 127
ESR, 722
ESRR, 729
ESRS, 728
ETAN, 271
EVERY, 655
EXECSTRING, 78
EXECTAB, 753
EXECUTE, 756
EXP, 81
EXTCALL
    用于 SINUMERIK 828D, 223
    用于 SINUMERIK 840D sl, 219
EXTCLOSE, 715
EXTERN, 208
EXTOPEN, 715

F

FALSE, 31
FCTDEF, 457
FCUB, 556
FENDNORM, 309
FFWOF, 567
FFWON, 567
FIFOCTRL, 570
FILEDATE, 168
FILEINFO, 168
FILESIZE, 168
FILESTAT, 168
FILETIME, 168
FINE, 633
FINEA, 310
FLIN, 556
FNORM, 556
FOR, 125
FPO, 556
FRAME, 31
FROM, 655
FTOCOF, 460
FTOCON, 460

G

G 代码
    间接编程, 74
G 功能组
    工艺, 565
G290, 736
G291, 736
G5, 398
G62, 309
G621, 309
G7, 398
G810...G819, 308
G820...G829, 308
GEOAX, 688
GET, 148
GETACTTD, 485
GETD, 148
GETDNO, 484

GETTCOR, 523
GETTENV, 521
GETVARAP, 66
GETVARDFT, 69
GETVARDIM, 68
GETVARLIM, 68
GETVARPHU, 66
GETVARTYP, 71
GOTO, 111
GOTOB, 111
GOTOC, 111
GOTOF, 111
GOTOS, 110
GP, 75
GROUP_ADDEND - 附加运行结束
    外部编程, 910
GROUP_BEGIN - 程序块开始
    外部编程, 909
GROUP_END - 程序块结束
    外部编程, 910
GUD, 32

H

HOLES1 - 直线位置模式
    外部编程, 760
HOLES2 -位置模式“圆弧”
    外部编程, 760

I

ID, 655
IDS, 655
IF, 123
IFRAME, 315
INDEX, 96
INICF, 31
INIPO, 31
INIRE, 31
INIT, 130
INITIAL, 244
INITIAL_INI, 244
INT, 31
INTERSEC, 751
IPOBRKA, 310
IPOENDA, 310
IPOSTOP, 633
IPTRLOCK, 576
IPTRUNLOCK, 576
ISAXIS, 685
ISFILE, 166
ISNUMBER, 92
ISOCALL, 216
ISVAR, 64

J

JERKLIM, 587
JERKLIMA, 563

L

L..., 206
LEAD, 359
LEADOF, 620
LEADON, 620
LENTOAX, 545
LIFTFAST, 142
Link
    变量, 29
LLI, 45
LN, 81
LONGHOLE - 长孔
    外部编程, 777
LOOP, 125
LUD, 32

M

M17, 193
M30, 193
MASLDEF, 651
MASLDEL, 651
MASLOF, 651
MASLOFS, 651
MASLON, 651
MATCH, 96
MAXVAL, 87
MCALL, 212
MD10010, 130
MD10280, 130
MD15800, 28
MD18104, 519
MD18116, 520
MD18156, 28
MD20360, 528
MD24558, 530
MD24658, 530
MEAC, 297
MEAFRAME, 334
MEAS, 294
MEASA, 297

MEAW, 294
MEAWA, 297
MINDEX, 96
MINVAL, 87
MMC, 703
MOD, 81
MODAXVAL, 685
MPF, 232

N

NAMETOINT, 422
NCK, 31
NCK 标识符, 236
NEWCONF, 155
NOC, 633
NOT, 83
NPROT, 252
NPROTDEF, 247
NUMBER, 92
NUT, 371

O

OEM 函数, 308
OEMIPO1/2, 308
OEM 地址, 308
OMA1 ... OMA5, 308
OR, 83
ORIAXES, 368
ORIC, 477
ORICONCCW, 371
ORICONCW, 371
ORICONIO, 371
ORICONTO, 371
ORICURVE, 374
ORID, 477
ORIEULER, 368
ORIMKS, 366
ORIPATH, 383
ORIPATHS, 383
ORIPLANE, 371
ORIRESET(A, B, C), 357
ORIROTA, 379
ORIROTC
    刀具定向旋转时, 379
    刀具旋转插补时, 385
ORIROTR, 379
ORIROTT, 379
ORIRPY, 368
ORIRPY2, 368
ORIS, 477
ORISOF, 392
ORISOLH, 499
ORISON, 392
ORIVECT, 368
ORIVIRT1, 368
ORIVIRT2, 368
ORIWKS, 366
OS, 657
OSB, 657
OSC, 477
OSCILL, 663
OSCTRL, 657
OSD, 477
OSE, 657
OSNSC, 657
OSOF, 477
OSP1, 657
OSP2, 657
OSS, 477
OSSE, 477
OST, 477
OST1, 657
OST2, 657
OTOL, 591

P

P..., 210
P_ACTFRAME, 342
PCALL, 217
PDELAYOF, 671
PDELAYON, 671
PFRAME, 315
PHI
    定向多项式, 378
    沿锥面定向时, 371
PHU, 46
PL
    多项式插补时, 286
    样条插补时, 271
PO, 286
PO[PHI]
    刀具定向旋转时, 383
    定向多项式, 378
    沿锥面定向时, 371
PO[PSI]
    刀具定向旋转时, 383
    定向多项式, 378
    沿锥面定向时, 371

PO[THT]
刀具定向旋转时, 383
定向多项式, 378
PO[XH]
定向多项式, 378
双触点定向设定时, 374
PO[YH]
定向多项式, 378
双触点定向设定时, 374
PO[ZH]
定向多项式, 378
双触点定向设定时, 374
POCKET3 - 矩形腔
外部编程, 763
POCKET4 - 圆形腔
外部编程, 767
POLF
NC 控制的退回, 723
POLFA, 723
POLFMASK
NC 控制的退回, 723
POLFMLIN
NC 控制的退回, 723
POLY, 286
POLYPATH, 286
PON, 680
PONS, 671
POSFS, 633
POT, 81
PREPRO, 193
PRESETON, 330
PRESETONS, 332
PRIO, 138
PRLOC, 31
Process DataShare, 715
PROTA, 427
PROTD, 429
PROTS, 428
PSI
定向多项式, 378
沿锥面定向时, 371
PTP, 403
PTPG0, 403
PTPWOC, 403
PUD, 32
PUNCHACC, 671
PUTFTOC, 460
PUTFTOCF, 459
PW, 271

R

READ, 163
REAL, 31
REDEF, 38
RELEASE, 148
REP, 56
REPEAT, 116
REPEATB, 116
REPOSA, 578
REPOSH, 578
REPOSHA, 578
REPOSL, 578
REPOSQ, 578
REPOSQA, 578
RET, 194
RET（可设定）, 196
RETB（可设定）, 202
RG, 27
RINDEX, 96
RMBBL, 578
RMEBL, 578
RMIBL, 578
RMNBL, 578
ROUND, 81
ROUNDUP, 171
RPY 角, 362
Run MyScreens, 703

S

SAVE, 183
SBLOF, 184
SBLON, 184
SCPARA, 697
SD, 271
SD41610, 440
SD41611, 440
SD42475, 390
SD42476, 390
SD42477, 390
SD42900, 451
SD42910, 451
SD42920, 451
SD42930, 452
SD42935, 454
SD42940, 455
SD42984, 510
SET, 56
SETAL, 720

SETDNO, 484
SETINT, 138
SETM, 130
SETTCOR, 530
SIN, 81
SLOT1- 纵向槽
    外部编程, 771
SLOT2 - 圆弧槽
    外部编程, 774
SOFT, 561
SOFTA, 561
SON, 671
SONS, 671
SPF, 232
SPI, 685
SPIF1, 671
SPIF2, 671
SPLINEPATH, 283
SPN, 677
SPOF, 671
SPP, 677
SPRINT, 100
SQRT, 81
START, 130
STARTFIFO, 570
STAT, 404
STOPFIFO, 570
STOPRE, 570
STRING, 31
STRINGIS, 699
STRLEN, 95
SUBSTR, 97
SYNR, 31
SYNRW, 31
SYNW, 31

T

TAN, 81
TANG, 549
TANGDEL, 554
TANGOF, 553
TANGON, 552
TCARR, 493
TCOABS, 493
TCOFR, 493
TCOFRX, 493
TCOFRY, 493
TCOFRZ, 493
THETA
    刀具定向旋转时, 379
    刀具旋转插补时, 385
TILT, 359
TLIFT, 551
TMOF, 683
TMON, 683
TOFFOF, 496
TOFFON, 496
TOLOWER, 95
TOOLENV, 517
TOUPPER, 95
TOWBCS, 452
TOWKCS, 452
TOWMCS, 452
TOWSTD, 452
TOWTCS, 452
TOWWCS, 452
TRAANG
    可编程角度, 397
TRACON, 401
TRACYL, 394
TRAFOOF, 416
TRAFOON, 431
TRAILOF, 597
TRAILON, 597
TRANSMIT, 394
TRAORI, 356
TRUE, 31
TRUNC, 81
TU, 408

U

ULI, 45
UNTIL, 128

V

VELOLIM, 588
VELOLIMA, 563

W

WAITC, 633
WAITE, 130
WAITENC, 695
WAITM, 130
WAITMC, 130
WHEN, 655
WHEN-DO, 666
WHENEVER, 655
WHENEVER-DO, 666
WHILE, 127

WORKPIECE, 731
WRITE, 156

X

XOR, 83

摆

摆动
    分度横向进给, 665
    通过同步动作控制, 663
    同步摆动, 663
    异步, 657
    异步摆动, 657
摆动运动
    反向点, 665
    回转范围, 665
    在换向点处的横向进给, 667

保

保护区, 247

报

报警
    在 NC 程序中设置, 720

比

比较运算符, 83

变

变量
    类型转换, 90, 92
    用户自定义, 31

标

标签, 116

补

补偿存储器, 441

不

不区分大小写, 235

步

步冲, 671

采

采集和查找不可查找的区域, 577

参

参数
    传递，在子程序调用时, 176
    -传递，在子程序调用时, 208
    刀具, 441
    实际, 176
    形式, 175

槽

槽式车削 - CYCLE952
    外部编程, 882

测

测量任务状态, 305

插

插入深度, 465

查

查找路径
    对于子程序调用, 241
    可编程的查找路径, 218

成

成型 - CYCLE495
    外部编程, 842

程

程序
    初始化, 244
    存储器, 233
    定址, 236
    分支, 114
    跳转, 111
    运行时间, 709
    重复, 210
程序部分
    重复, 116
程序部分重复
    带间接编程 CALL, 215
程序存储器
    标准目录, 232
    文件类型, 232
程序段显示
    抑制, 189
程序块结束 - GROUP_END
    外部编程, 910
程序块开始 - GROUP_BEGIN, 909
程序循环
    IF 循环, 123
    REPEAT 循环, 128
    WHILE 循环, 127
    计数循环, 125
    结束循环, 125

冲

冲压, 671

初

初始化
    数组, 56
初始化程序, 244

除

除数多项式, 290

触

触发事件
    测量时, 301

存

存储器
    程序, 231
    工作, 244
    缓冲, 570

带

带可回转的线性轴的转换, 355

单

单程序段
    抑制, 184
单位置, 367

刀

刀架
    可定向, 493
    -运动, 487
刀具
    参数, 441
    定向, 477
    定向，在框架更换时, 495
刀具半径补偿
    角部减速, 309
刀具补偿
    补偿存储器, 441
    附加, 445
    用于磨损值的坐标系, 452
刀具定向
    轨迹相对, 382
刀具定向 ORIRESET 的初始位置, 358
刀具链, 438

点

点对点运行, 403

电

电子齿轮, 626

雕

雕刻 - CYCLE60
    外部编程, 780

定

定向编程, 368
定向矢量 THETA, 379
定向轴, 368
定向转换 TRAORI
    定向编程, 357
    定向编程变量, 357
    定向运行, 347
    机床运动, 348
    生成 5/6 轴转换, 349
定址, 236

端

端面铣削, 365

多

多边形 - CYCLE79
    外部编程, 808
多项式插补, 286
多项式系数, 287

方

方向矢量, 362

附

附加运行结束 - GROUP_ADDEND
    外部编程, 910

刚

刚性攻丝 – CYCLE84
    外部编程, 819

跟

跟随轴
    轴向引道值耦合时, 620

工

工件
    计数器, 713
    目录, 233
    主目录, 232
工件链, 438
工作存储器, 244

攻

攻丝，带补偿夹具 – CYCLE840
    外部编程, 865

轨

轨迹铣削 - CYCLE72
    外部编程, 795

含

含角度偏移的位置同步, 633

宏

宏, 227

环

环形槽 - SLOT2
    外部编程, 774

缓

缓冲
    存储器, 570

回

回转 – CYCLE800
    外部编程, 844

极

极坐标转换, 349

急

急动
-补偿, 587
限值, 561

几

几何轴
切换, 688

计

计数循环, 125
计算参数
全球, 27
通道专用, 26

加

加工时间, 710
加速模式, 561

间

间接编程
G 代码, 74
地址, 72
零件程序行的, 78
位置属性的, 75

铰

铰孔 - CYCLE85
外部编程, 823

矩

矩形腔 - POCKET3
外部编程, 763
矩形凸台 - CYCLE76
外部编程, 799

可

可定向刀架, 487
可用性
系统方面, 5
可转换的几何轴, 688

控

控制
结构, 122

快

快速离开工件轮廓, 142
快速设定 - CYCLE832
外部编程, 861

框

框架
NCU 全局, 338
赋值, 324
级联, 344
框架级联, 324
调用, 323
通道专用, 339
系统, 339
框架变量
赋值, 320
调用坐标转换, 313
预定义框架变量, 315
框架部件
FI, 322
MI, 322
RT, 322
SC, 322
TR, 322

链

链接
由字符串, 93

零

零点偏移
外部, 328

路

路径说明, 237

轮

轮廓
    编码, 747
    -表格, 740
    预处理, 740
    重定位, 578
轮廓精度
    可编程, 568
轮廓腔铣削 - CYCLE63
    外部编程, 787
轮廓调用 - CYCLE62
    外部编程, 786
轮廓预处理
    报警应答, 756
轮廓元素
    退回, 753

逻

逻辑运算符, 83

螺

螺纹车削 - CYCLE99
    外部编程, 836
螺纹链 - CYCLE98
    外部编程, 830
螺纹铣削 - CYCLE70
    外部编程, 792
螺纹铣削 - CYCLE78
    外部编程, 805

毛

毛坯定义, 731

磨

磨损量, 446

目

目录路径, 238

内

内角处拐角延迟, 309

欧

欧拉角, 362

耦

耦合
    同类, 644
耦合系数, 597
耦合运动, 597
耦合轴, 599
耦合状态
    使用耦合轴时, 600
    轴向引道值耦合时, 625
耦合组合, 597

平

平滑
    定向曲线, 392
平面铣削 - CYCLE61
    外部编程, 783

嵌

嵌套深度
    控制结构的, 122

切

切槽 - CYCLE930
    外部编程, 872
切断 - CYCLE92
    外部编程, 826
切削
    辅助功能, 739
切削 - CYCLE951
    外部编程, 879
切削轮廓 - CYCLE95
    外部编程, 828
切削刃编号, 483

驱

驱动器名称, 237

全

全局零件程序存储器 (GDIR), 234

任

任意位置 - CYCLE802
    外部编程, 851

设

设定值同步, 636
设置修整器坐标系 - CYCLE435
    外部编程, 841
设置值, 446

深

深孔钻削 - CYCLE83
    外部编程, 815
深孔钻削 2 - CYCLE830
    外部编程, 855

剩

剩余行程删除, 303
剩余时间
    工件, 712

实

实际值同步, 636

输

输出
    外部设备/文件上, 715

数

数据级, 54
数组, 56
    -定义, 56
    元素, 56
数组索引, 59

速

速度同步, 636

所

所有拐角处都减速, 309

镗

镗孔 - CYCLE86
    外部编程, 825

调

调用带路径说明和参数的子程序, 217

跳

跳转
    回到程序开始, 110
    至跳转标记, 111
跳转标记
    程序部分重复时, 116
    程序跳转时, 112

停

停止
    NC 控制的, 727
    驱动自控, 728
停止程序段, 577

同

同步摆动
    IPO 周期分析, 668
    进刀运动, 667
    配置摆动轴和进给轴：, 665
    确定进给, 666
    同步动作, 666
    下一个分度横向进给, 669
    在换向区的横向进给。, 667
同步方式, 636
同步运行
    粗, 636
    精, 636

同步主轴
成对定义, 639
耦合, 633

退

退刀槽 - CYCLE940
外部编程, 876
退回
NC 控制的, 723
驱动自控, 729

外

外部编程, 909
外部零点偏移, 328

位

位置模式“方阵/框架” - CYCLE801
外部编程, 849
位置模式“圆弧” - HOLES2
外部编程, 760
位置模式“直线” - HOLES1
外部编程, 760
位置属性
间接编程, 75
位置同步, 633

文

文件
信息, 168
文件名, 239

无

无限循环, 125

铣

铣刀刀尖, 465
铣刀辅助点, 465
铣刀加工点, 465
铣削开口槽 - CYCLE899
外部编程, 869

系

系统
方面可用性, 5
系统变量
测量探头限制, 305
探头状态, 305
系统框架, 339

向

向上舍入, 171

斜

斜角转换 (TRAANG)
可编程角度, 397
斜向切入式磨削, 398

旋

旋转
定向矢量的, 379
旋转矢量的插补, 379
旋转轴
方向矢量, 487
距离矢量, 487
扭转角, 487

循

循环报警, 720

样

样条
插补, 271
类型, 278
样条组合, 283

异

异步摆动, 657

引

引导值模拟, 624

引导值耦合
实际值和设定值耦合, 623
引导轴和跟随轴同步, 623
引导轴
轴向引道值耦合时, 620

用

用户 XML, 703

与

与轨迹相对的刀具定向, 382

语

语言模式, 736

预

预钻轮廓腔 - CYCLE64
外部编程, 791

圆

圆弧数据
计算, 754
圆形腔 - POCKET4
外部编程, 767
圆形凸台 - CYCLE77
外部编程, 802

运

运动
分解, 491
运动关系类型, 491
运动结束条件
可编程, 310
运行模式
测量时, 302
运行时间
控制结构的属性, 122

在

在线－刀具长度补偿, 496

长

长孔 - LONGHOLE
外部编程, 777

中

中断程序
关闭/接通, 140
后退运行, 144
可编程的运动方向, 144
快速离开工件轮廓, 142
删除, 140
重新赋值, 139

轴

轴
耦合, 599
取轴, 148
轴向引导值耦合, 620

主

主轴
取轴, 148

柱

柱面转换, 349

转

转换
刀具定向的初始位置与运动无关, 346
方位转换, 345
级联, 401
级联的转换, 347
三轴/四轴和五轴转换, 356
运动转换, 346
转换方式
一般功能, 345
转换时的边界条件, 415
转速同步, 633

子

子程序
可编程的查找路径, 218
-名称, 173
调用，不带参数传递, 206
-调用，带参数传递, 208
-调用，间接, 214
-调用，模态, 212
跳转，可设定（RET ...）, 196
跳转，可设定（RETB...）, 202
应用, 172
重复, 210

自

自动的中断指示, 577
自动位移划分, 677

字

字符串
格式化, 100
链接, 93
运算符, 91
长度, 95

纵

纵向槽 - SLOT1
外部编程, 771

钻

钻削 - CYCLE82
外部编程, 812
钻中心孔 - CYCLE81
外部编程, 811

坐

坐标 PTP 运动, 403